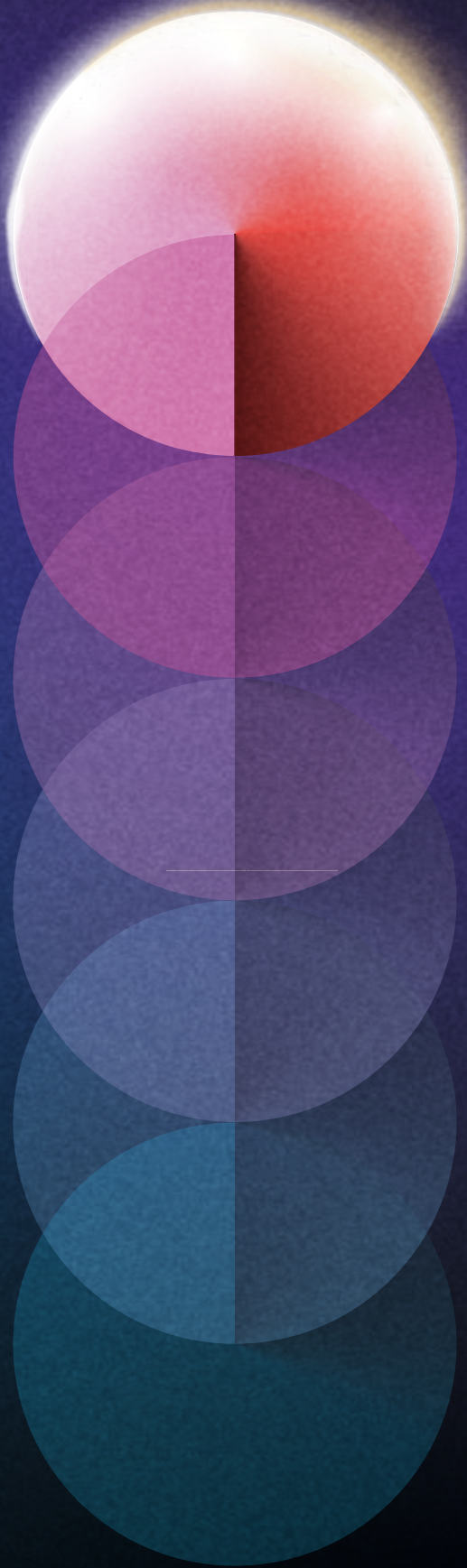
Agile Practice Guide

Second Edition



Project Management Institute.

Agile Alliance

Agile Practice Guide

Second Edition

Library of Congress Cataloging-in-Publication Data

Names: Project Management Institute issuing body | Agile Alliance contributor
Title: Agile practice guide / [Project Management Institute, Agile Alliance].
Description: Second edition. | Newtown Square, Pennsylvania, USA : Project Management Institute, [2026] | Includes index.
Identifiers: LCCN 2026004073 (print) | LCCN 2026004074 (ebook) | ISBN 9781628258318 paperback | ISBN 9781628258325 ebook
Subjects: LCSH: Project management | Project management--Methodology | Project management--Standards
Classification: LCC HD69.P75 A397 2026 (print) | LCC HD69.P75 (ebook)
LC record available at https://lccn.loc.gov/2026004073
LC ebook record available at https://lccn.loc.gov/2026004074

Published by:
Project Management Institute, Inc., 18 Campus Blvd., Ste. 150, Newtown Square, Pennsylvania 19073-3299 USA
Phone: +1 610 356 4600 | PMI.org | Online Support: www.pmi.org/about/contact

This Practice Guide was jointly funded with Agile Alliance® and was developed in collaboration with members of the Agile Alliance®.

The paper used in this book complies with the Permanent Paper Standard issued by the National Information Standards Organization (Z39.48—1992).

This publication has been designed with sustainability in mind, using practices that minimize paper usage and reduce environmental impact.

1 2 3 4 5 6 7 8 9 10

.

Notice

The Project Management Institute, Inc. (PMI) standards and guideline publications, of which the document contained herein is one, are developed through a voluntary consensus standards development process. This process brings together volunteers and/or seeks out the views of persons who have an interest in the topic covered by this publication. While PMI administers the process and establishes rules to promote fairness in the development of consensus, it does not write the document and it does not independently test, evaluate, or verify the accuracy or completeness of any information or the soundness of any judgments contained in its standards and guideline publications.

PMI disclaims liability for any personal injury, property or other damages of any nature whatsoever, whether special, indirect, consequential or compensatory, directly or indirectly resulting from the publication, use of application, or reliance on this document. PMI disclaims and makes no guaranty or warranty, expressed or implied, as to the accuracy or completeness of any information published herein, and disclaims and makes no warranty that the information in this document will fulfill any of your particular purposes or needs. PMI does not undertake to guarantee the performance of any individual manufacturer or seller's products or services by virtue of this standard or guide.

In publishing and making this document available, PMI is not undertaking to render professional or other services for or on behalf of any person or entity, nor is PMI undertaking to perform any duty owed by any person or entity to someone else. Anyone using this document should rely on his or her own independent judgment or, as appropriate, seek the advice of a competent professional in determining the exercise of reasonable care in any given circumstances. Information and other standards on the topic covered by this publication may be available from other sources, which the user may wish to consult for additional views or information not covered by this publication.

PMI has no power, nor does it undertake to police or enforce compliance with the contents of this document. PMI does not certify, test, or inspect products, designs, or installations for safety or health purposes. Any certification or other statement of compliance with any health or safety-related information in this document shall not be attributable to PMI and is solely the responsibility of the certifier or maker of the statement.

Preface

The Project Management Institute (PMI) and Agile Alliance® chartered this practice guide in 2017 to create a greater understanding of agile approaches in their communities. The vision for the second edition of this practice guide is to build on this foundation and provide a focus on enterprise agility as a key enabler for navigating business transformation and delivering results, moving beyond agility in software development.

Project teams are using agile approaches in a variety of industries beyond software development. This practice guide will help equip project teams with tools, situational guidelines, and an understanding of the available agile techniques and approaches to enable better results.

Both PMI and Agile Alliance realize that expansion has created a need for a common language, open-mindedness, and the willingness to be flexible in how products and deliverables are brought to market. In addition, both organizations realize there are multiple ways to achieve successful delivery. There is a broad range of tools, techniques, and frameworks—and teams have choices for approaches and practices to fit their projects and organizational cultures to achieve desired outcomes.

The *Agile Practice Guide*—Second Edition core committee members are from varying backgrounds within Agile Alliance and PMI and use various approaches. Some of the committee members are consultants and some work inside organizations. All have worked in agile ways for more than 10 years.

This icon marks notes on how generative artificial intelligence (GenAI) may support or challenge agile work. These are ideas, not endorsements or instructions. At the end of each section, you'll find examples of how GenAI could enhance efficiency, creativity, or decision-making, along with cautions where its use may introduce risk or harm.

Before applying any GenAI solution, always consider data sensitivity, consent, and ethics. Pilot in small steps, review outcomes with stakeholders, and pause when trust, accuracy, or context could be compromised.

Table of Contents

List of Figures and Tables

Figures

Tables

Section 1

Introduction

Welcome to the second edition of the *Agile Practice Guide*! Like the original guide, this edition is a collaborative effort by the Project Management Institute (PMI) and Agile Alliance. The core writing team, comprising volunteers from both organizations, developed this guide by drawing on subject matter expertise from a broad range of current practitioners and leaders from diverse backgrounds, beliefs, and cultures.

This practice guide provides practical guidance geared toward project and product development leaders and team members who intend to deploy an agile approach in planning and executing projects and organizational initiatives. New content on artificial intelligence (AI), sustainability, psychological safety, managing portfolios, and enterprise agility has been added to this second edition.

The core team adopted a more informal, relaxed writing style for this practice guide than is typical for PMI standards. Similar to the first edition of the *Agile Practice Guide*, this second edition incorporates elements, such as tips, sidebars, and case studies, to better illustrate key points and concepts. These elements are included to enhance the readability and user-friendliness of this practice guide.

Agile has expanded into manufacturing, education, healthcare, finance, marketing, government, engineering, construction, and other industries to varying degrees, going beyond its roots in software development. Agile is increasingly being used in research and development (R&D), innovation pipelines, and even fundamental scientific research. The scope of this new edition additionally introduces and acknowledges the importance of enterprise agility in supporting agile ways of working.

A Guide to the Project Management Body of Knowledge (PMBOK® Guide)—Seventh Edition [1][1] marked a shift from practice-based guidance to a principle-based structure. The seventh edition introduced a value delivery system that supports the full spectrum of delivery practices, ranging from agile to

[1] The numbers in brackets refer to the list of references at the end of this practice guide.

predictive, and their combinations. The *PMBOK® Guide*—Eighth Edition [2] builds on that foundation, extending the life cycle view to include both project and product delivery. This alignment with product management reflects how many agile teams now work.

1.1 Agile Practice Changes

The first edition of the *Agile Practice Guide* was released in 2017. Since then, the world of work has changed significantly. The COVID-19 pandemic and the rapid growth of new technologies, such as artificial intelligence (AI), have transformed how teams operate. The number of professionals now engaged in remote work has steadily increased, with many workers embracing the flexibility and autonomy it offers. At the same time, some organizations are calling for a return to the office [3]. This push and pull between distributed and centralized work continues to shape team dynamics.

Agile approaches have also evolved. Product management has gained widespread traction. Teams increasingly adopt product-focused roles, terminology, and frameworks. Agile-adjacent approaches such as Lean Startup and design thinking are now frequently integrated into agile delivery models.

Artificial intelligence (AI), particularly generative AI (GenAI) and agentic AI, has emerged as a collaborator. Teams can use it to assist with planning, writing, analysis, and decision support. Its impact continues to grow, reshaping how knowledge work is done. GenAI can also introduce errors, bias, and risk. The PMI guide, *AI Essentials for Project Professionals*, [4] provides a thorough understanding of AI, focusing on both foundational knowledge and practical applications in project and product management.

The agile ecosystem has expanded and, in some ways, contracted. Scaling frameworks like Scaled Agile Framework (SAFe®), Large Scale Scrum (LeSS), Nexus, and Scrum@Scale are now widely adopted. Toolkits such as PMI® Disciplined Agile® (DA®) offer tailoring and improvement guidelines.

This environment has led to a gap between the C-suite and their perception of the value of agile practices. As a result, there has been some disillusionment with teams that adopt agile practices without delivering meaningful outcomes. Regardless, agile approaches are widely used for high-uncertainty work and commonly taught in schools.

To help teams concentrate on delivering value-focused initiatives, outdated metrics are now being replaced with leaner, more meaningful indicators that focus on customer outcomes and flow efficiency.

1.1.1 Agile Terminology Changes

The terminology of agile has changed. The term *hybrid* is used to describe the combination of predictive and agile practices. Yet, because hybrid can describe many different combinations, this practice guide quickly reframes delivery as a continuum and invites practitioners to select their own relevant mix that best fits each situation. Terms once considered standard are being replaced with more inclusive and respectful alternatives.

Backlog grooming is more commonly referred to as backlog refinement, and daily coordination meetings, daily syncs, or daily scrums (in Scrum) have now replaced daily standups. Roles such as scrum master and concepts like servant leadership are being reassessed, although no universally accepted alternatives have been agreed upon yet. Inclusive practices and updated terminology reflect a broader cultural shift in how team members relate and communicate with one another.

1.2 Social Changes

The agile community has undergone significant social changes in recent years, driven by a growing emphasis on inclusivity, well-being, and environmental responsibility. These shifts are reflected in three key areas: evolving terminology to promote greater understanding and respect, prioritizing psychological safety to foster collaborative environments, and integrating sustainability principles to assist with long-term viability. These changes not only transform the way agile teams work but also amplify their social impact by fostering more inclusive, resilient, and sustainable organizational cultures.

1.2.1 Psychological Safety

Psychological safety is now a basic requirement for team performance. People do their best work when they feel safe to contribute, raise concerns, and share new ideas. Creating and supporting this kind of environment is now regarded as a key responsibility of leaders and peers alike. See Section 4.2 for an explanation of psychological safety.

The *PMBOK® Guide*—Eighth Edition [2] also introduced "Build an Empowered Culture" as a project management principle, reinforcing the importance of trust and safety in today's organizations.

1.2.2 Sustainability

Agile ways of working grew from a call for a sustainable pace, reminding practitioners that teams should create value without burning out. Today, the sustainability idea stretches further: Sustainable products safeguard people, the planet, and long-term prosperity. When discussing sustainability in an agile context, the focus should be on making decisions now that will not compromise tomorrow's options.

In the context of this practice guide, every agile choice, scope, schedule, cadence, funding consideration, etc., should factor in the long-term environmental, social, and economic effects rather than only near-term delivery.

1.3 Strategic Changes

This practice guide reflects the evolving landscape of agile approaches to project management, incorporating strategic changes that prioritize enterprise agility and visible C-suite sponsorship. This edition also focuses on delivering strategic value and driving meaningful outcomes.

1.3.1 Enterprise Agility

Due to increasing volatility, uncertainty, and the frequency of disruption in business environments, the need for enterprise agility is increasing. Enterprise agility requires visible and active sponsorship from the C-suite and encompasses an organization's strategic capability to adapt quickly to changing conditions such as shifts in the market, customer needs, or competition while maintaining strategic focus and delivering value, with middle-management empowerment as a critical enabler of enterprise agility. This approach goes beyond one team or department; it is a leadership mindset translated into an enterprise capability that aligns purpose, culture, and strategy execution to enable continuous reinvention and long-term growth despite disruption. It requires leaders who model an agile mindset, develop operating models that eliminate waste, and build teams that learn and evolve faster than the world around them.

For more information about enterprise agility, refer to the *Manifesto for Enterprise Agility* [5] developed in partnership by PMI and Agile Alliance.

1.3.2 Alignment With M.O.R.E.

Increasingly, organizations are looking beyond traditional measures of success to assess their delivery on enterprise project and product portfolios and strategic initiatives. The *PMBOK® Guide*—Eighth Edition [2] aligns with the M.O.R.E. principles—Manage perceptions, Own project success, Relentlessly reassess, Expand perspective—from PMI, which serve as a call to action for project professionals. Along with the new definition of project success—delivers value that is worth the effort and expense—this broader vision embraces a more proactive, value-driven approach to project success by encouraging strategic decision-making across the project life cycle.

The M.O.R.E. principles offer a comprehensive approach to achieving project success. Manage perceptions means ensuring key stakeholders are able to perceive the project's value; Own project success requires moving beyond execution to take accountability for delivering tangible value while minimizing waste; Relentlessly reassess project parameters involves continuously reassessing the perception of value and adjusting plans in response to change; and Expand perspective requires considering the project's broader impact on the business and beyond.

M.O.R.E. redefines project success by moving beyond traditional metrics such as scope, budget, and schedule to focus on delivering value—as perceived by stakeholders—in a complex environment characterized by hyper-competition, regulatory instability, and rapid innovation. By adopting this broader perspective, the project community can focus on achieving meaningful outcomes that benefit those they serve.

The second edition of the *Agile Practice Guide* also aligns with M.O.R.E. and demonstrates how teams deliver on strategy through their day-to-day practices in the following ways:

- **Manage perceptions.** The *Agile Practice Guide* puts a working product, transparency, and continuous feedback at the center of stakeholder engagement. Reviews show outcomes, not just outputs. Information radiators make progress and trade-offs visible. Measures tie increments to goals and constantly assess if the value obtained is worth the effort and expense invested. Agile teams run value-focused reviews, keep a value burnup chart visible, and link each increment to a small metric shift or retired risk.
- **Own project success beyond project management success.** This practice guide promotes a product mindset that extends beyond project delivery, with whole-team ownership and collaboration across delivery, operations, and business as key values. Agile teams map objectives to outcomes, treat value as the guiding star in planning, and remove impediments that block delivery, even if they sit outside classic project boundaries.
- **Relentlessly reassess project parameters.** This guide's iterative cadence of planning, reviews, and retrospectives creates regular checkpoints to adjust scope, priorities, and risk responses. Backlog refinement, hypothesis-driven iterations, flow metrics support, and root cause identification enable rapid course correction. Agile teams should reevaluate priorities with stakeholders each iteration, use cycle time and work-in-process (WIP) limits to expose delays, and replace fixed plans with rolling forecasts tied to evidence of value delivered.
- **Expand perspective.** The *Agile Practice Guide* includes life cycle selection, hybrid options, and systems thinking as key considerations. The guidance connects team practices to longer product life cycles, value streams, and the broader strategic business context. Agile teams should apply a product mindset, choose the life cycle that fits their project context, surface upstream and downstream impacts with value stream mapping, and make enterprise objectives visible in team goals.

M.O.R.E. promotes the mindset and values of leading organizations. The *Agile Practice Guide* supplies the patterns, events, and metrics that help teams act on them.

1.4 *Agile Practice Guide*—Second Edition Changes

Since the first edition, the pace of change has only increased. Environmental disruption, political shifts, and rapid technological advances are the new normal. In this context, agility is not optional, it is a necessity. This second edition reflects that reality with updated guidance, inclusive language, and practical tools for modern teams. This practice guide recognizes a wide range of delivery approaches and highlights the values and mindsets needed to navigate today's complexity. Additionally, this guide supports practitioners in applying agile ways of working that truly serve their teams, customers, and organizations.

The *Agile Practice Guide*—Second Edition is project-/product-focused and addresses life cycle selection, agile implementation, and organizational considerations for using agile approaches. Organizational change management (OCM) is essential for implementing or transforming practices, but since OCM is a discipline within itself, it is outside the scope of this practice guide. Those seeking guidance in OCM may refer to *Managing Change in Organizations: A Practice Guide* [6].

Additional items that are in scope and out of scope for this practice guide are listed in Table 1-1.

Table 1-1. In-Scope and Out-of-Scope Items

In Scope	Out of Scope
"What?" and "Why?" principles-based guidance for implementing agile approaches at the project/product or team level	Detailed "How-to" instructions for executing agile projects/product development
Coverage of most popular agile approaches, as listed in industry surveys	Coverage of niche approaches, company-specific methods, or incomplete life cycle techniques
Considerations for portfolio and program management and how these concepts map to their lean thinking product counterparts	Comprehensive coverage of strategy-to-delivery initiative selection and prioritization
Suitability factors to consider when choosing an agile approach and/or practice	Recommending or endorsing a particular approach/practice
Discussion on the use of agile beyond software development	Removal of software industry influence on agile approaches (Note that software is included in this practice guide even though the use of agile is growing in many other industries beyond software.)
Guidance, techniques, and approaches to consider when implementing agile in projects or organizations	Prescriptive, step-by-step instructions on how to implement agile in projects or organizations
Definitions of generally accepted terms	Industry or framework-specific terms and/or definitions

This practice guide is for teams and organizations that find themselves in the "messy middle" between predictive and agile approaches—those who are trying to address rapid innovation and complexity and are dedicated to continuous team improvement and empowerment. This practice guide provides useful guidance for successfully delivering business value to meet customers' evolving expectations and needs.

This practice guide is organized as follows:

- **Section 2—An Introduction to Agile Values and Principles.** This section contains values and principles based on the *Manifesto for Agile Software Development* [7]. It also covers the concepts of definable and high-uncertainty work, and the correlation between Lean, the Kanban Method, and agile approaches.
- **Section 3—Development Life Cycles and Approach Selection.** This section introduces the various life cycles discussed in this practice guide. It also addresses suitability filters, tailoring guidelines, and common combinations of approaches.
- **Section 4—Creating an Agile Environment.** This section discusses critical factors to consider when creating an agile environment such as supportive leadership and team composition.
- **Section 5—Delivering in an Agile Environment.** This section includes information on how to organize teams, and common practices teams can use for delivering value on a regular basis. It provides examples of empirical measurements for teams and for reporting status.
- **Section 6—Organizational Considerations for Agility.** This section explores enterprise factors that impact the use of agile approaches such as culture, readiness, business practices, and the role of a project management office (PMO).

The annexes, appendixes, references, bibliography, and glossary provide additional useful information and definitions:

- **Annexes.** Contain mandatory information that is too lengthy for inclusion in the main body of the practice guide.
- **Appendixes.** Contain nonmandatory information that supplements the main body of this practice guide.
- **References.** Identify where to locate standards and other publications that are cited in this practice guide.
- **Bibliography.** Lists additional publications by section that provide detailed information on topics covered in this practice guide.
- **Glossary.** Presents a list of terms and their definitions that are used in this practice guide.

Section 2

An Introduction to Agile Values and Principles

Embracing agile values and principles transforms the way organizations approach definable and high-uncertainty work, fostering a culture of empowerment, adaptability, and continuous improvement. By integrating lean thinking and leveraging GenAI insights, teams can enhance their responsiveness to change and drive innovation.

2.1 Definable Work and High-Uncertainty Work

Project work ranges from definable work to high-uncertainty work (knowledge work). Definable projects are characterized by clear procedures that have proved successful on similar projects in the past. The production of a car, electrical appliance, or home after the design task is complete are examples of definable work. The production domain and processes involved are usually well-understood and there are typically low levels of execution uncertainty and risk.

High-uncertainty or knowledge work projects involve novel design and problem-solving. These projects require subject matter experts to collaborate and solve problems to create a solution. Examples of people encountering high-uncertainty work include software system professionals, product designers, doctors, teachers, lawyers, and many problem-solving engineers. As more definable work is automated, project teams are undertaking more high-uncertainty projects such as GenAI projects, biotechnology, and climate change initiatives that require the techniques described in this practice guide.

High-uncertainty projects have high rates of change, complexity, and risk. These characteristics can present problems for traditional, predictive approaches that aim to determine the bulk of the requirements up front and control changes through a change request process. Instead, agile approaches were created to explore feasibility in short cycles and quickly adapt based on evaluation and feedback.

Value-based delivery guides teams to confirm that each outcome is worth the time, effort, and money invested. When work is highly definable, planners protect value through detailed scope, flexible change management, and cost control. When work is highly uncertain, teams use short cycles and rapid feedback to validate value early, adapt, and avoid investing in the wrong solution.

2.2 Agile Fundamentals

Commercial software development work in the 1990s and early 2000s was often encumbered by rigid methodologies, with projects frequently being delivered over budget, behind schedule, and failing to meet customer needs. This situation created a need to establish a new approach to these projects, implementing shorter feedback loops, iterative approaches, and a focus on customer value. To help formalize this new agile movement, software industry thought leaders published the *Manifesto for Agile Software Development* [7] in 2001 (see Figure 2-1).

Twelve clarifying principles flowed from these values as shown in Figure 2-2.

Although originating in the software industry, these values and principles have since spread to many other industries such as healthcare, education, and engineering. When doing so, some practitioners replace "Working software over comprehensive documentation" with "Working solutions over comprehensive documentation."

This embodiment of mindset, values, and principles defines what constitutes an agile approach. The various agile approaches in use today share common roots with the agile mindset, values, and principles. This relationship is shown in Figure 2-3.

We are uncovering better ways of developing software by doing it and helping others do it. Through this work we have come to value:

Individuals and interactions over processes and tools

Working software over comprehensive documentation

Customer collaboration over contract negotiation

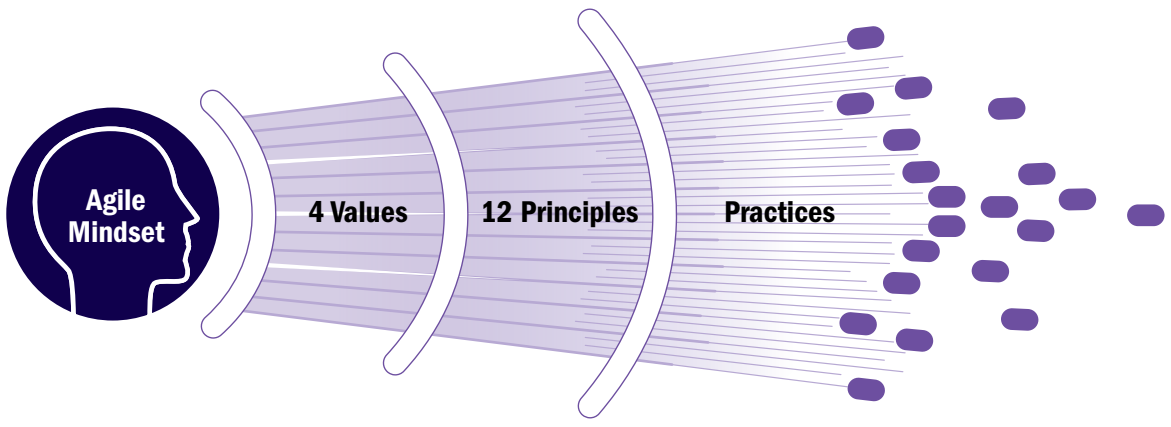
Responding to change over following a plan

That is, while there is value in the items on the right, we value the items on the left more.

Figure 2-1. The Four Values of the *Manifesto for Agile Software Development*

1. Our highest priority is to satisfy the customer through early and continuous delivery of valuable software.
2. Welcome changing requirements, even late in development. Agile processes harness change for the customer's competitive advantage.
3. Deliver working software frequently, from a couple of weeks to a couple of months, with a preference to the shorter timescale.
4. Business people and developers must work together daily throughout the project.
5. Build projects around motivated individuals. Give them the environment and support they need, and trust them to get the job done.
6. The most efficient and effective method of conveying information to and within a development team is face-to-face conversation.
7. Working software is the primary measure of progress.
8. Agile processes promote sustainable development. The sponsors, developers, and users should be able to maintain a constant pace indefinitely.
9. Continuous attention to technical excellence and good design enhances agility.
10. Simplicity—the art of maximizing the amount of work not done—is essential.
11. The best architectures, requirements, and designs emerge from self-organizing teams.
12. At regular intervals, the team reflects on how to become more effective, then tunes and adjusts its behavior accordingly.

Figure 2-2. The 12 Principles Behind the *Manifesto for Agile Software Development*



Agile is a mindset defined by four values and 12 principles, and manifested through many different practices. Agile practitioners select practices based on their needs.

Figure 2-3. The Relationship Between the *Agile Manifesto* Values, Principles, and Common Practices

As shown in Figure 2-3, the model, based on the work of Ahmed Sidky [8], articulates agile as a mindset defined by the *Agile Manifesto* values, guided by the *Agile Manifesto* principles, and enabled by various practices. It is worth noting that while the term *agile* became popularized after publication of the *Agile Manifesto*, the approaches and techniques being used by project teams today existed before the *Manifesto for Agile Software Development* [7] by many years and, in some cases, decades.

Agile approaches and agile methods are umbrella terms that cover a variety of frameworks and approaches. Figure 2-4 places agile in context and visualizes it as a blanket term, referring to any kind of approach, technique, framework, method, or practice that fulfills the values and principles of the *Agile Manifesto*. Figure 2-4 also shows agile and the Kanban Method as subsets of Lean. This is because they are named instances of Lean thinking that share Lean concepts such as: "focus on value," "small batch sizes," and "elimination of waste."

In general, there are two strategies to fulfill agile values and principles. The first is to adopt a formal agile approach, intentionally designed and proven to achieve desired results. Then take time to learn and understand the agile approaches before changing or tailoring them. Premature and haphazard tailoring can minimize the effects of the approach and thus limit benefits. See Appendix X2 for tailoring considerations.

The second strategy is to implement changes to project practices in a manner that fits the project context to achieve progress on a core value or principle. Use timeboxes to create features, or specific techniques to iteratively refine features. Consider dividing up one large project into several smaller deliverables, if that works for the specific project context. Implement changes that will help the project succeed—the changes do not need to be part of the organization's formal practices. The end goal is to deliver a continuous flow of value to customers and achieve better business outcomes.

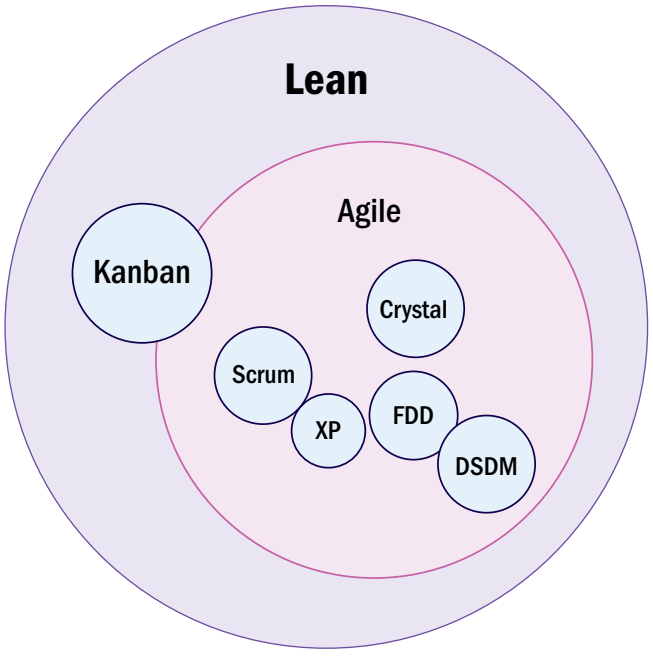


Figure 2-4. Agile Is a Blanket Term for Many Approaches

2.3 Lean Thinking Concepts

One way to think about the relationship between Lean, agile, and the Kanban Method is to consider agile and the Kanban Method as descendants of Lean thinking. In other words, lean thinking is a superset, sharing attributes with agile and Kanban.

This shared heritage is very similar and focuses on delivering value, respect for people, minimizing waste, being transparent, adapting to change, and continuously improving. Project teams sometimes find it useful to blend various methods—whatever works for the organization or team is what should be done regardless of its origin. The objective is the best outcomes regardless of the approach used.

The Kanban Method is inspired by the original lean manufacturing system and used specifically for knowledge work. The method emerged in the mid-2000s as an alternative to the agile approaches that were prevalent at the time.

The Kanban Method is less prescriptive than some agile approaches and less disruptive, as it is the original "start-where-you-are" approach. Project teams can begin applying the Kanban Method with relative ease and progress toward other agile approaches if that is what they deem necessary or appropriate. For more details on the Kanban Method, see Annex A3, Overview of Agile and Lean Frameworks.

USE CASE

There is, and probably always will be, a lot of discussion around the Kanban Method and whether it belongs to the lean or agile movement. Kanban was created in lean manufacturing even though Kanban for knowledge work was created by David J. Anderson and a collection of people from the Corbis team.

2.3.1 Design Thinking and Lean Startup

Design thinking and Lean Startup are both human-centered approaches to agile project and product development, but they differ in their focus and execution. Design thinking emphasizes understanding user needs through observation and empathy, whereas Lean Startup focuses on quickly building and testing a minimum viable product (MVP) to validate assumptions.

Design thinking and Lean Startup approaches expand an agile practitioner's toolbox for early product validation. These approaches can help teams clarify problems, test ideas, and ground backlogs in evidence. By examining these agile-adjacent practices, practitioners can gain a continuous flow from discovery to delivery, increasing the chance that increments address real customer and business needs.

2.3.2 Design Thinking Principles

Design thinking is a human-centered approach to problem-solving. The approach combines analytical rigor with creative exploration so teams can discover, frame, and address real customer needs. See Figure 2-5.

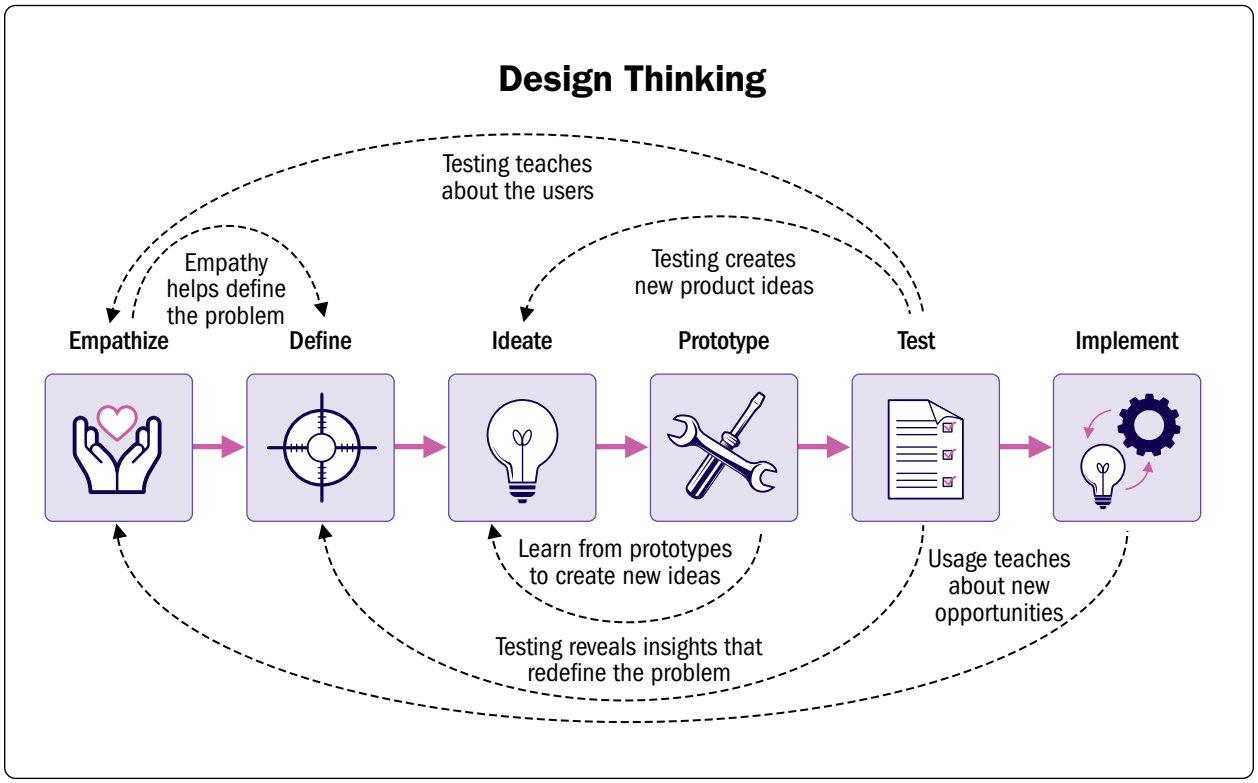


Figure 2-5. Design Thinking Elements

Project teams can apply design thinking's core characteristics when they want to do the following:

- **Empathize with users.** Teams investigate how people work, think, and feel to ground decisions in lived experience.
- **Define the problem.** Insights from research help articulate a concise problem statement that guides future work.
- **Generate ideas.** Diverse participants create multiple solution options, broadening the pool of possibilities before committing to one direction.
- **Prototype.** Move from abstract to concrete. Low-fidelity artifacts, such as sketches or models, invite rapid feedback and minimize the cost of change.
- **Test to reveal insights.** Iterate toward value. Feedback loops validate assumptions early, reducing the risk of building the wrong solution.
- **Deliver and iterate.** Deliver and learn about new opportunities.

For agile practitioners, understanding the concepts of design thinking can strengthen product development in several ways, such as the following:

- Empathy work and concise problem statements link backlog items to verified needs, so sprint/iteration goals align with genuine customer value.
- Early prototypes invite user feedback during each iteration, shortening learning loops and reducing downstream rework.

- Visual artifacts, such as journey maps, give developers, testers, designers, and business stakeholders a common language, improving collaboration and keeping technical discussions tethered to user outcomes.

Applying design thinking can help agile teams ground backlog items in evidence gathered through structured discovery, so effort aligns with real customer needs. The approach adds a deliberate cycle of empathy, problem framing, and rapid concept testing that feeds directly into iterative delivery. This effort helps reduce rework and sharpens the product focus. By integrating this discovery cadence with any agile approach, teams can gain clearer purpose and higher confidence that the increments they release will create real value.

2.3.3 Lean Management and Lean Startup Principles

Lean Startup emerged in the early 2000s. Lean Management is a management philosophy aimed at maximizing customer value while minimizing waste across all operations. Lean Startup popularized the concepts such as validated learning, innovation accounting, and perseverance versus pivot decisions (see Figure 2-6). Today, the model influences startups and large enterprises alike because it replaces intuition-based launches with evidence-driven steps toward product-market fit.

Lean Startup treats every new product idea as a testable hypothesis. Teams create the smallest testable version of the idea, often called a minimum viable product (MVP), and cycle through a build–measure–learn loop to collect real customer data. The results either confirm the team's assumptions or reveal the need to "pivot"; that is, change direction before investing further time and money.

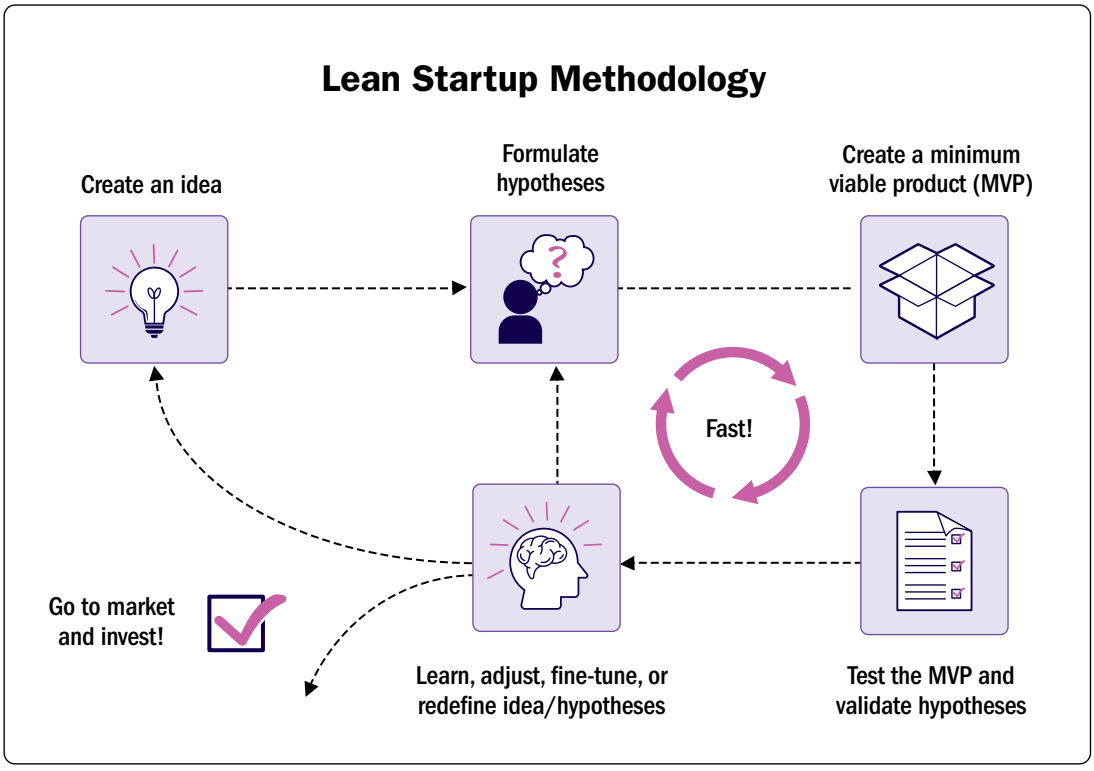


Figure 2-6. Lean Startup

Understanding Lean Startup benefits agile teams because it adds a rigorous learning cadence to their delivery rhythm. Instead of assuming that backlog items already reflect customer value, teams run targeted experiments and use hard data to shape the backlog itself. This practice cuts waste and reveals viable opportunities sooner. This approach can also increase the likelihood that each sprint/iteration delivers something customers truly want while still meeting business objectives.

2.3.4 Combining Approaches

Practitioners can combine design thinking, Lean Startup, and agile delivery as a powerful, integrated approach to product development. Design thinking focuses on deeply understanding user needs to shape the initial product concept. Lean Startup then rapidly tests and validates assumptions through MVPs and iterative experimentation. Agile delivery ensures continuous feedback and value delivery throughout the product life cycle. Together, these approaches help teams determine what to build, how to build it efficiently, and how to adapt over time to evolving market and customer needs.

GenAI Insights

GenAI offers significant potential to enhance agile practices by accelerating learning, improving decision-making, and increasing productivity across frameworks such as Lean, Kanban, and design thinking. GenAI can generate customized training materials and case studies that help teams better understand and apply agile values, principles, and mindsets. Within agile environments, GenAI can analyze workflows to distinguish between definable and high-uncertainty work, identify bottlenecks, and visualize process improvements. The technology also serves as a creative accelerator, helping teams ideate, test, and refine solutions more quickly regardless of the agile framework or approach used. In Lean, GenAI can identify sources of waste, propose alternative process flows, and simulate the impact of changes before implementation. Within Kanban, it can analyze work-in-process (WIP) data to recommend flow optimizations and WIP limits, or generate visual models that make work more transparent and adaptive. In design thinking, GenAI expands creativity by producing "what-if" scenarios, suggesting prototype variations, and synthesizing user insights from large, qualitative data sets to inspire new perspectives. When thoughtfully integrated, GenAI enhances collaboration, reduces administrative overhead, and amplifies innovation by giving teams the time and tools to explore ideas that might otherwise remain undiscovered.

At the same time, integrating AI into agile teams introduces ethical, cultural, and operational risks that can outweigh its benefits if left unchecked. Overreliance on GenAI may encourage a checklist mentality or distort the *Agile Manifesto*'s people-first intent by prioritizing speed and automation over collaboration and adaptability. AI can alter team roles, blur accountability, and embed bias in decision-making, leading to a loss of trust or poor governance. It can also expose sensitive data, generate misleading insights, or replace genuine empathy with artificial assumptions in design thinking and

Lean Startup contexts. To mitigate these risks, organizations should establish clear policies defining roles, responsibilities, consent, and data usage; set strict limits on what AI can access; and conduct regular audits to help ensure transparency, accountability, and alignment with agile's core values of trust, collaboration, and continuous learning. GenAI contributions are not free; they require a subject matter expert to check them for accuracy and value. This effort takes time and multiple iterations of refinement. However, the value they can deliver, even when these implementation costs are deducted, can still be significant.

Section 3

Development Life Cycles and Approach Selection

Projects come in many shapes and sizes, and there are a variety of ways to undertake them. Project teams should have awareness of the characteristics and options available to select the approach most likely to be successful for the situation.

This practice guide refers to three types of life cycles, defined as follows:

- **Predictive life cycle.** Predictive life cycles take a more traditional approach, with the bulk of planning occurring up front, then executing in a single pass—a sequential process.
- **Continuum of life cycles.** Often referred to as hybrid life cycles, this continuum (or spectrum) of life cycle approaches mixes components of predictive approaches and elements of iterative/incremental agile approaches.
- **Agile life cycle.** An agile life cycle is an approach that is both iterative and incremental to refine work items and deliver frequently.

3.1 What Should Non-Agile Approaches Be Called?

There is no single term that is universally used to describe non-agile approaches. Initially, this practice guide used the term *plan-driven* to describe the emphasis on an up-front plan and the subsequent execution of that plan. Some people use the terms *traditional*, *waterfall*, or *serial* to reflect common nomenclature. This second edition now uses the term *predictive* since it is used in the *PMBOK® Guide*—Eighth Edition [2]. Many organizations do not experience either of these extremes and instead occupy some middle ground, which is common. However, project professionals still require a way to talk about both ends of the spectrum. If agile (adaptive) is at one end, the other end is predictive.

3.2 Product Management in Agile Environments

Products are one of the primary vehicles through which strategy meets customer needs and teams create lasting value. According to the *PMBOK® Guide*—Eighth Edition [2], a *product* is an artifact that is produced, is quantifiable, and can be either an end item in itself or a component item. For example, an end item can be tangible, such as a laptop computer that is complete, or intangible, such as a digital platform or subscription service. An end item may also be a component item that is a key internal product used in a laptop.

More broadly, a product is a solution designed to meet customer or user needs and is expected to evolve through continuous feedback, innovation, and adaptation. A product may also be a digital solution (e.g., software application), an ongoing service (e.g., customer support, training), or a hybrid offering that combines physical and intangible components. In modern agile environments, many organizations deliver service-centric or software-driven products that evolve continuously with minimal physical components. While projects are temporary initiatives in a unique context undertaken to create value, products are enduring, evolving entities without a defined end date.

Key characteristics of products include the following:

- Long-term investments,
- Focus on continuous value delivery,
- Ongoing alignment with customer and market expectations,
- Built-in feedback and learning cadences that shorten the idea-to-value cycle, and
- Value and market fit with differentiation and positioning.

3.3 Product Life Cycle Phases

Most products evolve through the following stages (Figure 3-1):

- **Introduction.** Ideation and early validation.
- **Growth.** Gaining traction and expanding capabilities.
- **Maturity.** Stabilizing, optimizing, and differentiating.
- **Decline.** Reduction in demand.
- **Retirement.** End-of-life planning and withdrawal from the market.

Throughout this life cycle, various projects may be initiated to introduce new features, drive marketing campaigns, or manage migration and retirement activities (see Figure 3-2).

USE CASE

Gaming consoles launch with hype (conception), see rapid sales and developer adoption (growth), hit peak saturation (maturity), and eventually see waning support and new systems emerge (decline and evolve to other types of products).

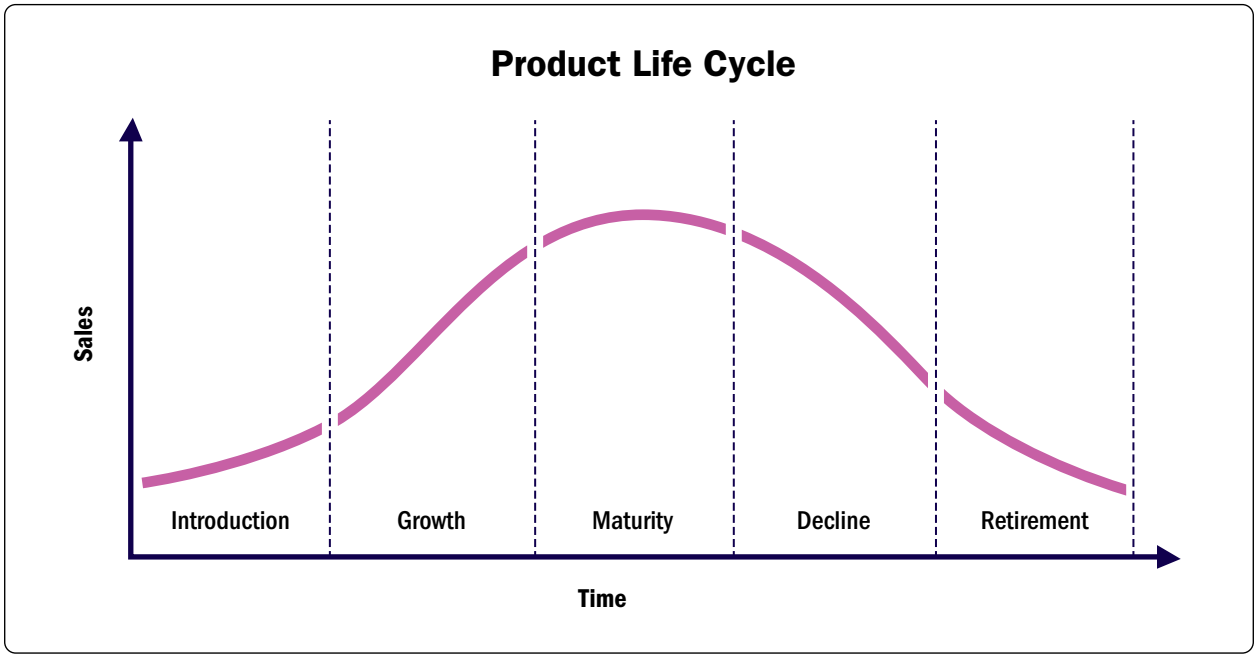


Figure 3-1. Product Life Cycle Phases

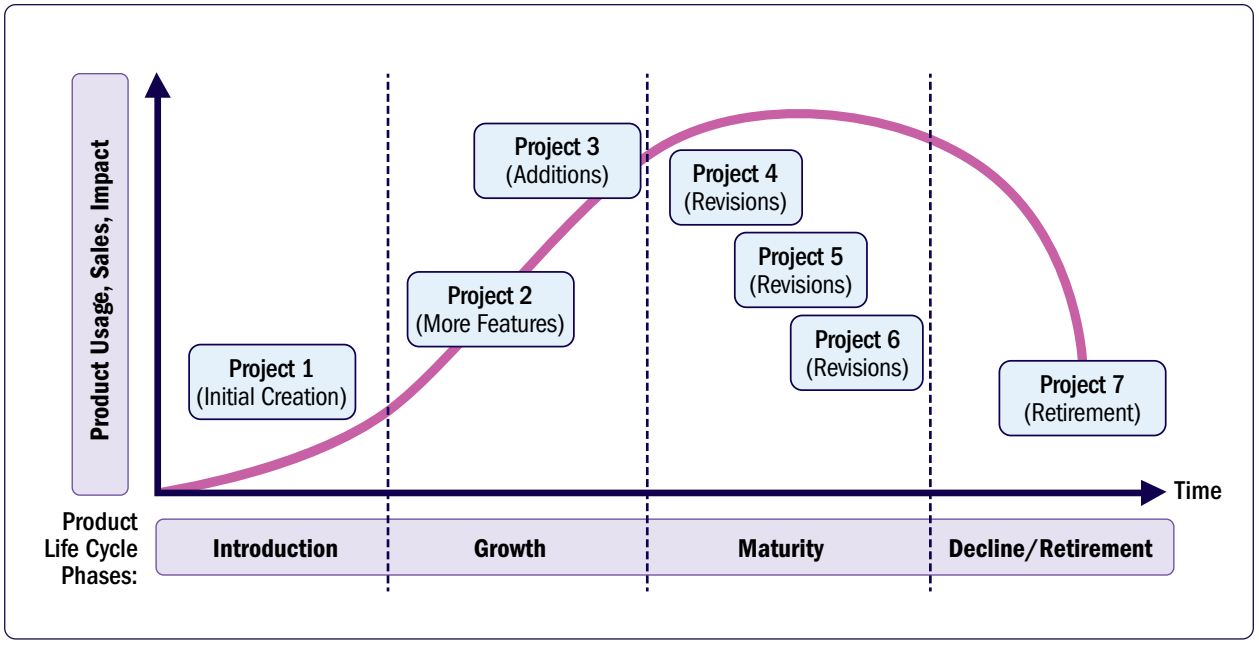


Figure 3-2. New Feature Development Throughout the Product Life Cycle

3.4 Differences Between Product and Project Teams

Agile teams can be organized into different structures, including product teams and project teams, which have distinct characteristics and responsibilities, as follows:

- **Product teams.** Continuous teams are funded based on outcomes and are responsible for long-term value delivery.
- **Project teams.** Temporary teams are scoped around delivering a specific outcome, often with fixed time and resource constraints.

While both require cross-functional collaboration, product teams are often more stable and strategic in nature.

3.5 Comparison of Product and Projects

Understanding the distinctions between product and project work is essential for organizational alignment. A side-by-side comparison highlights key differences (see Table 3-1).

3.5.1 What Is Product Management?

Product management integrates people, data, processes, and systems to create, maintain, and evolve a product or service throughout its life. Product management exists as both a function and a role within organizations.

3.5.1.1 Product Management as a Function

The function of product management encompasses organizational processes and governance that support the product's success, including the following:

- Defining strategic goals and direction,
- Coordinating cross-functional collaboration,

Table 3-1. Product and Project Comparison

Characteristic	Product	Project
Time horizon	Ongoing, long-term	Temporary, defined start and end
Funding model	Continuous, value-based	Fixed, based on deliverables, outcomes, or milestones
Team structure	Ongoing, market-driven investment	Timeboxed, budget constrained
Change handling	Iterative, responds to feedback and market needs	Change control/backlog/by baseline
Examples	Mobile app, software platform	New feature release, marketing campaign

- Managing product roadmaps and life cycle planning, and
- Aligning product decisions with business outcomes.

3.5.1.2 Product Management as a Role

The role is commonly held by a product manager or product owner who is responsible for maximizing value delivery. Responsibilities often include the following:

- Defining the product vision and success metrics;
- Conducting market and user research;
- Prioritizing work based on value and feedback;
- Collaborating with engineering, design, marketing, and operations teams; and
- Monitoring performance and market demands, then iterating based on insights.

3.5.1.3 Role Collaboration Across the Life Cycle

In agile environments, product and project teams should collaborate across different phases of the product life cycle as follows:

- During **introduction,** product managers define the vision, conduct research, and explore feasibility. Agile practitioners may facilitate planning workshops or backlog creation.
- During **growth**, agile practitioners help manage delivery cadence and dependencies, whereas product managers respond to market feedback and adjust priorities.
- In **maturity**, coordination around optimization efforts, scaling, pivoting, or reinvigorating becomes key to all roles.
- In **decline** and **retirement**, project managers may lead sunset initiatives or data migration efforts, whereas product managers assess long-term strategy or product evolution.

While some organizations blend project and product management roles, having clearly defined responsibilities and shared communication channels can reduce overlap, improve decision-making, and increase organizational agility.

3.5.1.4 Product Management in Agile Practice

Agile product management embraces uncertainty and feedback. Work is delivered incrementally through short cycles (iterations or sprints) that allow continuous validation and adjustment.

Product managers in agile environments use practices such as the following:

- **Minimum marketable feature (MMF).** The smallest piece of functionality that delivers significant value to the user and can be marketed and sold on its own.
- **Minimum viable product (MVP).** The smallest implementation that lets practitioners test a key assumption about a problem, solution, or market with real users and gather evidence to decide whether to proceed, pivot, or stop.
- **A/B testing.** Test alternative versions to understand user preferences.
- **Beta testing.** Release to a small subset of users for feedback.
- **Phased rollouts.** Gradually launch features to reduce risk.
- **Customer interviews.** Gain qualitative insight directly from users.

- **Feedback loops.** Use insights to reprioritize and evolve the product.
- **Design thinking and Lean Startup concepts.** These concepts can help to better empathize with customer needs and determine when to pivot, if necessary.

USE CASE

Food and beverage companies often release limited-edition flavors (e.g., BBQ, dill pickle, kimchi) to test consumer interest. Sales data, social media reactions, and customer surveys inform whether to continue, modify, or discontinue the products. These experiments mirror agile product management at scale.

3.6 Product Portfolio Management

Many organizations manage more than one product, often structured into a product portfolio. Product portfolio management (PPM) helps to balance the following:

- Strategic investments across new and existing products;
- Talent and funding allocation and prioritization; and
- Risk, value, regulatory, and sustainability alignment.

Portfolio-level visibility allows executives and teams to do the following:

- Evaluate product performance across business lines;
- Decide when to invest, sustain, or retire offerings; and
- Align roadmaps to broader enterprise goals.

USE CASE

A technology company may maintain a portfolio that includes core enterprise platforms, innovation pilots, and regional applications. PPM enables leadership to optimize investments, prevent duplication, and support a coherent customer experience.

Strategy and investment funding sets guardrails. Agile portfolio operations manage flow and WIP, and lean governance keeps outcomes and compliance visible. Together they form the lean portfolio operating model. See Section 6.5.2 on lean portfolio management for additional information.

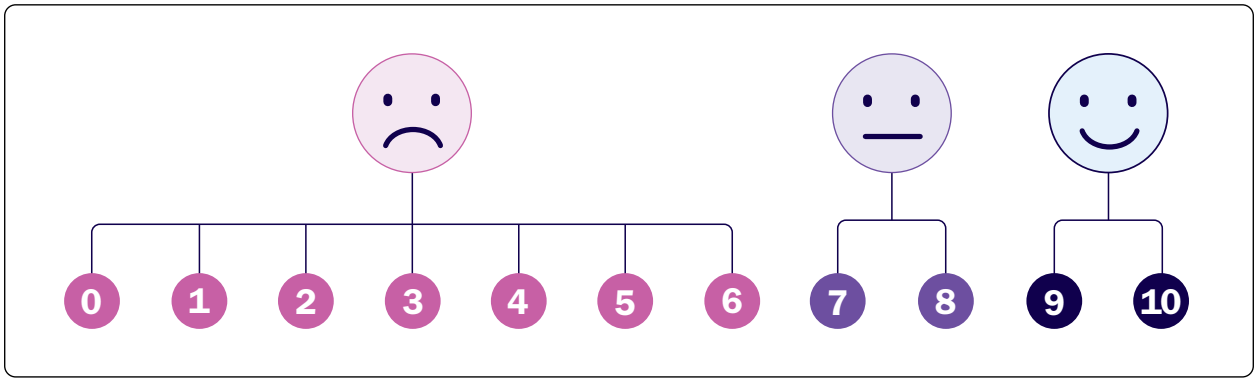


Figure 3-3. Customer Experience Metrics

3.7 Customer Experience and Product Metrics

Customer experience (CX) is a core focus of modern product management that helps gauge product performance in the market. Common product metrics include the following (see Figure 3-3):

- **Customer acquisition cost (CAC).** Cost to gain a customer.
- **Retention rate.** Percentage of users who return.
- **Churn rate.** Percentage of users lost over time.
- **Net Promoter Score℠ (NPS®).**[2] Willingness to recommend.
- **Strategic-fit metrics.** Objective and key results (OKR) progress and cycle time from insight to decision.
- **Average revenue per user (ARPU).** Revenue generated per user.

USE CASE

NPS® kiosks in airports or public restrooms offer real-time input on user experience. These quick-response systems, often placed at exits or near service areas, ask visitors to rate their satisfaction with a simple button-press. The results provide immediate visibility into how people feel about the cleanliness, service, or efficiency of a location.

3.8 Sustainability and Ethical Design

Product teams are increasingly considering the broader impact of their work as it affects both society and the environment.

[2] Net Promoter®, NPS®, NPS Prism®, and the NPS-related emoticons are registered trademarks of Bain & Company, Inc., NICE Systems, Inc., and Fred Reichheld. Net Promoter Score℠ and Net Promoter System℠ are service marks of Bain & Company, Inc., NICE Systems, Inc., and Fred Reichheld.

Sustainability practices help reduce negative environmental impacts through design, energy usage, and life cycle management. Ethical design helps products to do the following:

- Protect user privacy,
- Promote accessibility, and
- Avoid manipulation or harm.

USE CASE

Fairphone is a Dutch company that designs smartphones with sustainability and ethical sourcing at the forefront. Fairphone emphasizes modular design to extend device lifespan, uses responsibly sourced materials such as fair trade gold, and partners with factories that meet high labor standards. This approach reduces electronic waste, supports fair labor practices, and raises consumer awareness around the environmental and social costs of electronics.

Product managers are stewards of responsible innovation. Many practitioners now track ethical outcomes alongside business key performance indicators (KPIs) to help ensure that product growth does not come at the expense of user trust or planetary health. This approach aligns with the *PMBOK® Guide*—Eighth Edition principle of Integrate Sustainability Within All Project Areas [2].

3.9 Characteristics of Project and Product Life Cycles

Projects and products have unique characteristics, and the life cycle selected can further distinguish management practices and development approaches. Table 3-2 summarizes the characteristics of predictive, agile, and a continuum of approaches that are covered in this practice guide.

Table 3-2. Characteristics of Life Cycles

Characteristics				
Approach	**Requirements**	**Activities**	**Delivery**	**Goal**
Predictive	Controlled or change control	Performed once for the entire project	Single/phased delivery	Manage cost, time, and meet objectives
Agile	Dynamic	Repeated until correct	Frequent small deliveries	Customer value via frequent deliveries and feedback
A continuum of approaches	Mixed	Performed through the entire project	Multiple deliveries	Customer value via delivery of elements that are well-known and elements that evolve during the project

TIP

About the Term *Continuum of Approaches*

A hybrid approach combines elements of two or more different approaches. Often, this is a combination of predictive and agile principles and practices. Because approaches can be combined in many different ways, simply using the term *hybrid* does not provide any insight into the ways of working selected. For instance, one hybrid could be largely predictive with just a few agile elements such as a weekly retrospective. Another hybrid could be largely agile with a predictive component, such as up-front analysis to determine the likely scope, followed by predominantly agile execution. So, while the term *hybrid* may be helpful shorthand for labeling mixed ways of working, a more accurate term is a *continuum* or *spectrum* of practices, with more detailed explanations of the components selected.

It is important to note that all projects have these characteristics and no project is completely devoid of considerations around requirements, delivery, change, and goals. A project's inherent characteristics determine which approach is the best fit for that project.

Another way to understand how project approaches vary is by using a continuum ranging from predictive approaches on one end, to agile approaches on the other end, with more iterative or incremental approaches in the middle.

Figure X3-1 in Appendix X3 in the *PMBOK® Guide*—Eighth Edition [2] displays the continuum as a flat line. This view emphasizes the shifting of project characteristics from one end to the other. Another way to visualize the continuum is a spectrum, as shown in Figure 3-4.

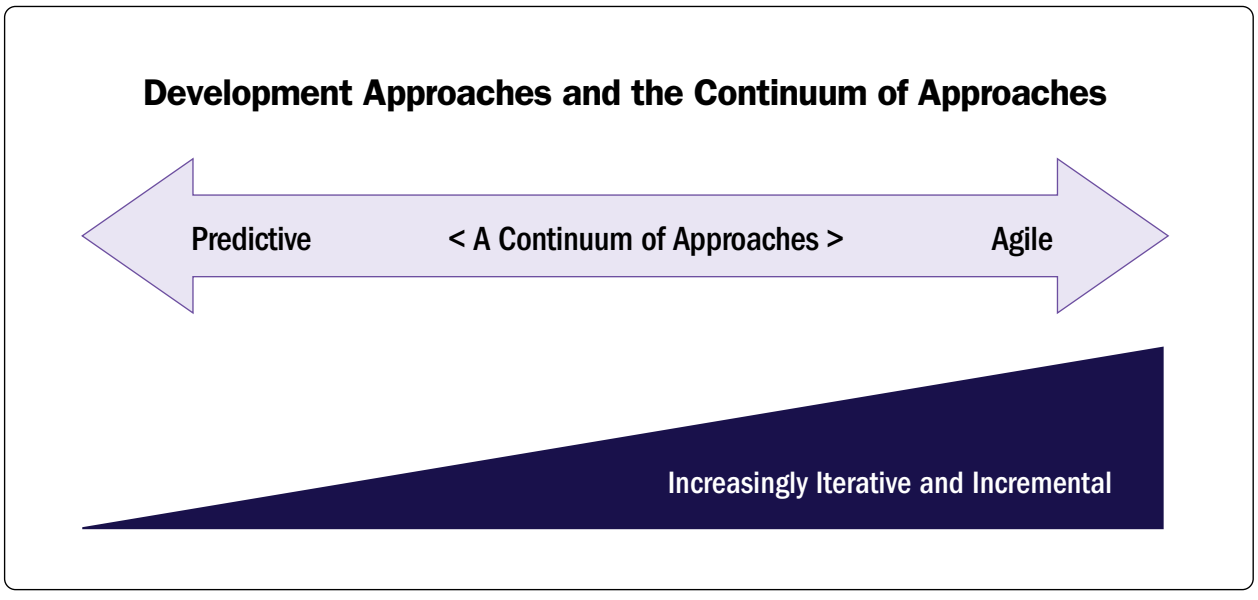


Figure 3-4. The Continuum of Approaches

No approach can be perfect for all projects. Instead, each project finds a spot on the continuum that provides an optimum balance of characteristics for its context, such as the following:

- **Predictive approaches.** Predictive approaches can take advantage of things that are known and proven. This reduced uncertainty and complexity can allow teams to segment work into a sequence of predictable groupings.
- **Iterative and/or incremental approaches.** Iterations allow feedback on partially completed or unfinished work to improve and modify that work. Increments provide finished deliverables that the customer may be able to use immediately.
- **Agile approaches.** Agile approaches leverage both the aspects of iterative and incremental characteristics. When teams use agile approaches, they iterate over the product to create finished deliverables. The team can gain early feedback and provide customer visibility, confidence, and control of the product. Because the team can release earlier, the project may provide an earlier return on investment because the team delivers the highest value work first.

3.9.1 Characteristics of Predictive Approaches

Predictive approaches can help practitioners to take advantage of high certainty around firm requirements, stable teams, and low risk. As a result, project activities often execute in a serial manner, as the example shows in Figure 3-5.

To apply this approach, the team requires detailed plans to know what to deliver and how. These projects succeed when other potential changes are restricted (e.g., requirements change; project team members change what the team delivers). Team leaders should aim to minimize change for predictive projects.

When the team creates detailed requirements and plans at the beginning of the project, they can articulate the constraints. The team can then use those constraints to manage risk and cost. As the team progresses through the detailed plan, they monitor and control changes that might affect the scope, schedule, or budget.

By emphasizing a departmentally efficient, serialized sequence of work, predictive projects do not typically deliver business value until the end of the project. If a predictive project encounters changes or disagreements with the requirements, or if the technological solution is no longer straightforward, the predictive project will incur unanticipated costs.

3.9.1.1 Planning Is Always There

A key thing to remember about life cycles is that each of them shares the element of planning. What differentiates a life cycle is not whether planning is done, but rather how much planning is done and when.

Figure 3-5. Predictive Approach

At the predictive end of the continuum, the plan drives the work. As much planning as possible is performed up front. Requirements are identified in as much detail as possible. The team estimates when they can deliver which deliverables and performs comprehensive procurement activities.

In iterative approaches, prototypes and proofs are also planned, but the outputs are intended to modify the plans created in the beginning. Earlier reviews of unfinished work help inform future project work.

Meanwhile, teams plan incremental activities to deliver successive subsets of the overall project. They may plan several successive deliveries in advance or only one at a time. The deliveries inform future project work.

Agile project teams also plan. The key difference is that the team plans and replans as more information becomes available from reviewing frequent deliveries. Regardless of the project life cycle, all projects require planning.

3.9.2 Characteristics of Agile Life Cycles

Agile life cycles are those that fulfill the four values and 12 principles of the *Agile Manifesto*. In particular, customer satisfaction increases with early and continuous delivery of valuable products. Moreover, an incremental deliverable that is functional and provides value is the primary measure of progress. Agile life cycles combine both iterative and incremental approaches to help projects adapt to high degrees of change and deliver value more often. Kanban uses a continuous pull model without iterations, but still focuses on frequent delivery.

Some agile approaches use iterations that are generally 1 to 4 weeks in duration, with a demonstration of the accomplishments at the end of each iteration. The entire project team should be involved in the planning at various levels. In the iteration, the team members should determine the scope they can achieve based on the prioritized requirements in the form of the product backlog, estimate the amount of work, and work collaboratively to develop the scope during the iteration (see Figure 3-6).

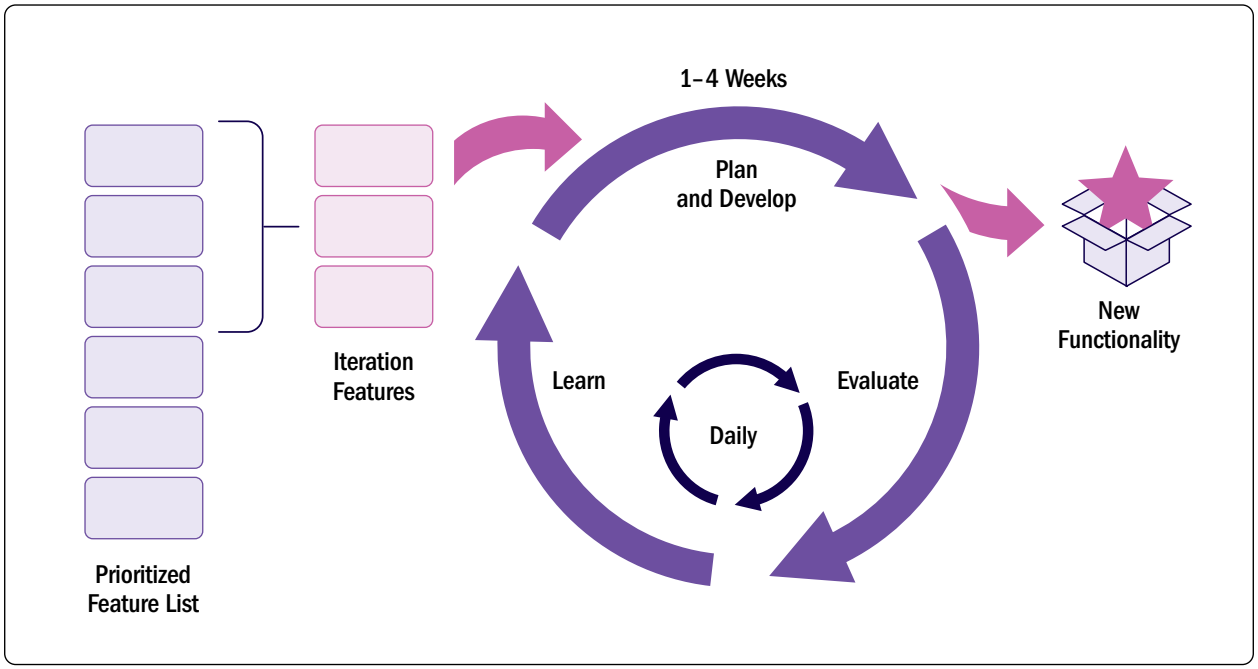


Figure 3-6. Agile Life Cycle

3.10 Selecting Life Cycles

To increase the likelihood of success and positive outcomes, teams should use the life cycle most appropriate for the type of work they are undertaking.

For defined, repeatable projects with little uncertainty or likelihood of change, a predictive approach is appropriate. Here, a "plan-the-work, work-the-plan" approach is efficient and allows for clear plans and schedules to be shared with all stakeholders.

Difficulties arise when teams try to use these approaches in intangible, unfamiliar, and new environments. Differences in understanding frequently occur when practitioners lack physical reference points such as, "I want a wooden door like this one, but a foot taller." These differences result in increases in change requests, reported defects, uncertainties, and risks.

In novel, intangible environments like software development or filmmaking, things rarely progress predictably enough to follow the "plan-the-work, work-the-plan" mantra of industrial projects. New technology evolution accelerates the rates of change. Demands to deliver faster can worsen the situation. Many of today's projects fit this new breed of project that were christened "knowledge work" projects by Peter Drucker in *The Landmarks of Tomorrow* (1959) [9]. Drucker later coined the term "knowledge worker" in *The Effective Executive* [10] in 1966. Later still, in 1999, Drucker suggested that "the most valuable asset of a 21st-century institution, whether business or non-business, will be its knowledge workers and their productivity." [11]

Additionally, many traditional industrial projects have been automated. This practice leaves a higher proportion of new projects developing largely invisible, intangible, and difficult-to-reference products and services under the category of knowledge work.

3.10.1 Understanding Complexity and Uncertainty

Project environments vary in complexity. Some are straightforward and predictable. Others are ambiguous, dynamic, and subject to frequent change. Complexity may also arise from social, cultural, and political factors. Choosing a delivery approach that aligns with the level of complexity helps teams reduce risk, improve outcomes, and adapt when conditions shift.

Many modern projects operate within what are known as complex adaptive systems (CAS). These are systems where individual components interact in nonlinear and often unpredictable ways. The work of one team can influence the direction of others. Stakeholder expectations evolve as new information becomes available. In CAS environments, cause and effect are difficult to trace, and simple planning approaches may not be sufficient. Instead, teams benefit from life cycles that support learning, experimentation, and incremental progress.

Two models commonly used to help teams assess complexity are the Stacey matrix and the Cynefin Framework. Both tools provide a way to understand the nature of the work and guide life cycle selection accordingly.

The Stacey matrix plots two dimensions: the level of agreement on what is needed (requirements uncertainty), and the level of certainty on how to deliver it (technical uncertainty).

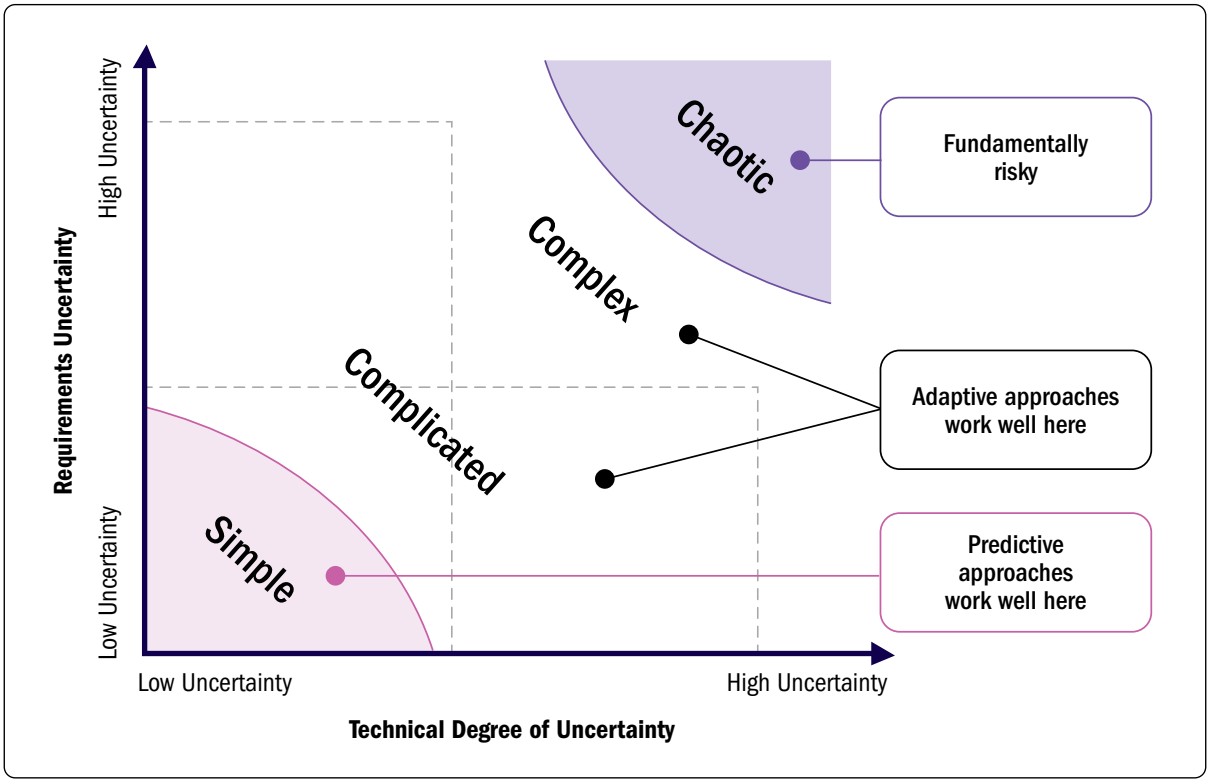


Figure 3-7. Stacey Matrix

The Stacey matrix in Figure 3-7 demonstrates the following:

- In the lower-left corner of the matrix, where both the requirements and technology are certain, traditional predictive approaches work well.
- As one or both dimensions become less certain, when requirements are unclear, or when multiple technical options are available, projects enter the complicated, complex, or chaotic regions.
- In complicated or complex areas, adaptive or exploratory life cycles offer better support, allowing teams to discover solutions iteratively through feedback and validation.

TIP

What do simple, complicated, and complex projects mean? Consider the Sydney Opera House construction project. On the surface, the project seemed straightforward: Design and build a world-class performance venue in Sydney's harbor. The original plan projected 4 years of construction at a cost of AUD$7 million. There was an ambitious, unprecedented design and low uncertainty about how the project would proceed until construction began.

Continued on page 30

As often happens with large projects, unexpected challenges emerged: radical design changes, engineering hurdles, political pressures, and escalating costs. The project ultimately took 14 years to complete and cost AUD$102 million.

When a team works on a project with little opportunity for interim deliverables or prototyping, it will most likely use a predictive life cycle to manage it. The team can adapt to what it discovers, but will not be able to rely on agile approaches to manage the iterative discovery of requirements or deliverables for feedback.

The Sydney Opera House project was not simple by any means. However, many projects that start out in the lower-left part of the Stacey matrix have no real way to shift to other approaches midstream. Assess the project, both in its requirements and in its means of delivery, to discuss and then determine the best-fit approach for its life cycle.

The Cynefin Framework offers another lens, categorizing contexts into five domains:

1. **Simple**, where cause and effect are obvious and good practices apply.
2. **Complicated**, where expert analysis is required but predictable outcomes are possible.
3. **Complex**, where outcomes emerge through experimentation and learning.
4. **Chaotic**, where immediate action is required to stabilize the situation.
5. **Disorder** (sometimes labeled "aporia" or "confused"), when the situation is unclear or miscategorized.

In the complex domain, the recommended approach is to probe, sense, and respond. This approach aligns with agile practices that favor short feedback cycles, iterative delivery, and stakeholder collaboration. In the complicated domain, a more analytical approach may work, possibly with a hybrid life cycle that includes planning phases supported by iterative execution.

Both the Stacey matrix and Cynefin Framework can help teams assess where their work falls on the spectrum from predictable to emergent. This in turn supports better alignment between project context and delivery approach. These models remind teams that life cycle selection is not just a matter of preference; it is a response to the nature of the problem being solved.

3.10.2 Facilitating Life Cycle Discussions

Selecting a delivery approach is not only a technical decision, but also a social one. The team may have a clear understanding of the environment's complexity, but if key stakeholders do not share that understanding, alignment can suffer. To be successful, life cycle decisions should be understood, accepted, and supported by those who fund, govern, undertake, and receive the work.

Tools like the Agile Suitability Filter, available at https://suitabilityfilter.com, can facilitate these discussions. The filter provides a structured way to assess a project's alignment with agile delivery characteristics, including factors such as clarity of requirements, stakeholder availability, level of innovation, and tolerance for change. Each item is scored on a sliding scale, and the results are presented as a visual spectrum showing how well-suited a project may be to agile approaches.

By using the filter collaboratively, teams and stakeholders can compare perceptions, explore areas of agreement or concern, and discuss the reasoning behind their inputs. This collaboration promotes transparency and builds shared understanding. Rather than debating whether to "be agile," the discussion shifts toward identifying the conditions needed for adaptive delivery to succeed and exploring where traditional methods may provide better structure.

The Agile Suitability Filter is particularly useful in mixed environments such as organizations with formal governance processes or cross-functional dependencies. The filter supports context-based tailoring and can serve as a bridge between delivery teams and oversight functions. The results can also be revisited later in the project life cycle, offering a repeatable way to evaluate whether the chosen approach is still appropriate.

Facilitating stakeholder alignment early increases confidence in the selected life cycle and helps create the conditions necessary for it to succeed. Figure 3-8 shows a sample plot from the Agile Suitability Filter.

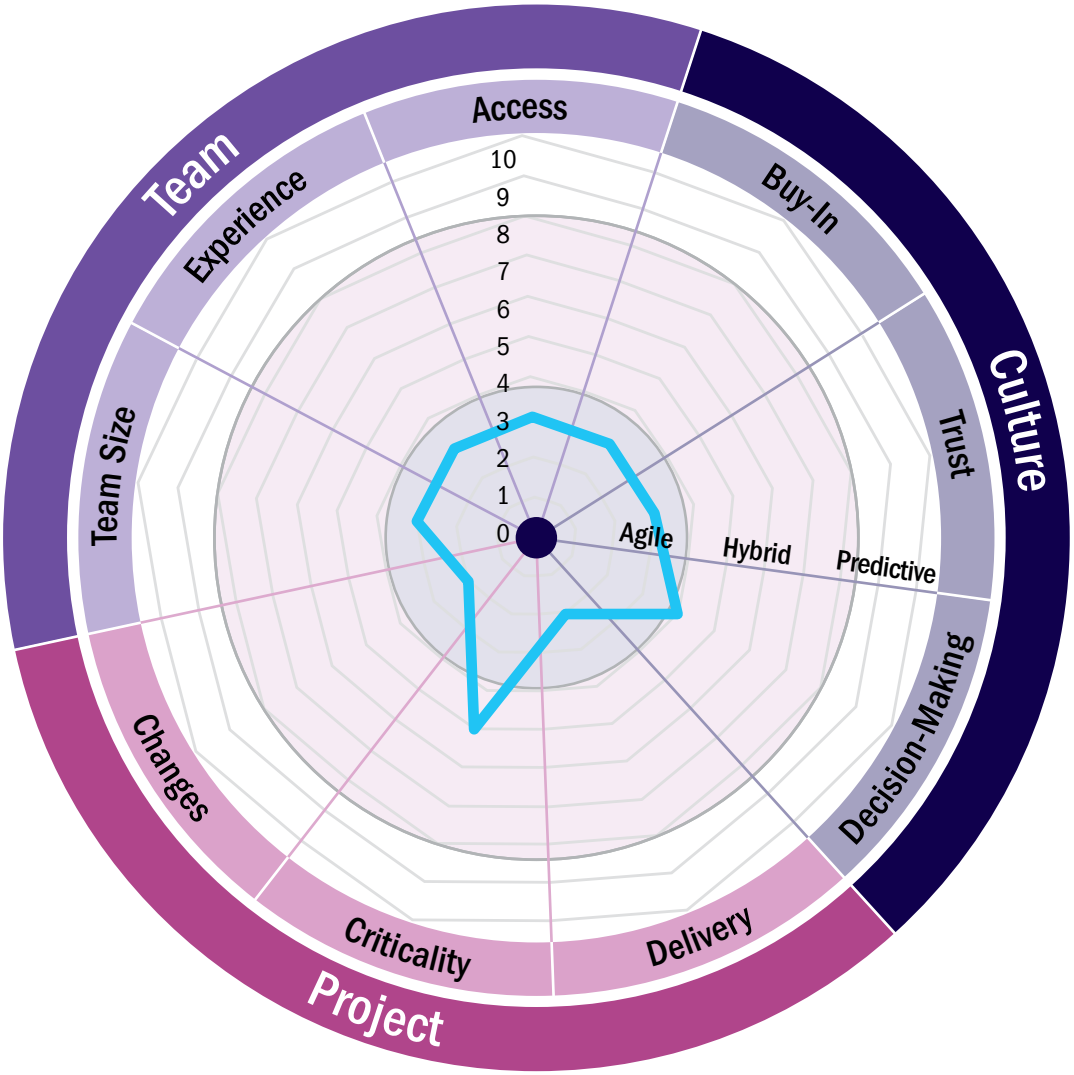


Figure 3-8. Agile Suitability Filter Sample Plot

Figure 3-9. Agile Development Followed by a Predictive Rollout

3.10.3 Combining Approaches

It is not necessary to use a single approach for an entire project. Projects often combine elements of different life cycles to achieve certain goals. Figure 3-9 depicts an approach that combines agile and predictive approaches. The early processes utilize an agile development life cycle, which is then followed by a predictive rollout phase. This approach can be used when there is uncertainty, complexity, and risk in the development portion of the project that would benefit from an agile approach, followed by a defined, repeatable rollout phase that is appropriate to undertake in a predictive way, perhaps by a different team. An example of this approach is the development of a new high-tech product followed by rollout and training for thousands of users.

USE CASE

Combining Approaches

A pharmaceutical company had a time-consuming U.S. Food and Drug Administration (FDA) approval process tagged onto the end of its development process, and its entire life cycle looked like Figure 3-10. While project teams undertook drug trials in an agile fashion, they had to present the drugs to an external group to perform the FDA approval process. A consultant helped to integrate the FDA approval process portion into the agile development process to create a more streamlined hybrid approach.

FDA approval is required to be completed at the end of the development process or repeated after any change (this includes even after very minor changes), so the process had to remain at the end as a separate phase. Integration using the iterative process was unsuccessful. However, the consultant created some useful quick-start guides and testing protocols that shortened the final FDA approval process.

3.10.4 Combined Agile and Predictive Approaches

Another approach is to use a combination of agile and predictive approaches throughout the life cycle.

In Figure 3-10, a combination of both predictive and agile approaches is used in the same project. Perhaps the team is incrementally transitioning to agile and using some approaches like short iterations, daily coordination meetings, and retrospectives, but other aspects of the project, such as up-front estimation, work assignment, and progress tracking, are still following predictive approaches.

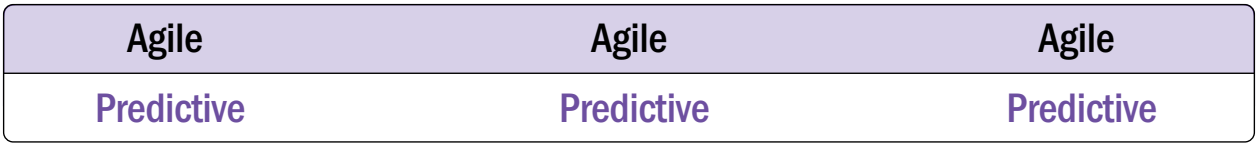


Figure 3-10. A Combined Agile and Predictive Approach Used Simultaneously

Using both predictive and agile approaches is a common scenario. It would be misleading to call the approach agile since it clearly does not fully embody the agile mindset, values, and principles. However, it would also be inaccurate to call it predictive since it is a hybrid approach (a point on the continuum of approaches).

3.10.5 A Predominantly Predictive Approach With Some Agile Components

Figure 3-11 shows a small agile element within a chiefly predictive project. In this case, a portion of the project with uncertainty, complexity, or opportunity for scope creep is being tackled in an agile way, but the remainder of the project is being managed using predictive approaches. An example of this approach would be an engineering firm that is building a facility with a new component.

While the majority of the project may be routine and predictable, like many other facility projects the organization has undertaken before, this project incorporates a new roofing material. The contractor may plan for some small-scale installation trials on the ground first to determine the most appropriate installation method and to uncover issues early while there is plenty of time to solve them and incrementally improve processes through experimentation and adaptation.

3.10.6 A Largely Agile Approach With a Predictive Component

Figure 3-12 depicts a largely agile approach with a predictive component. This approach might be used when a particular element is nonnegotiable or not executable using an agile approach. Examples include integrating an external component developed by a different vendor that cannot—or will not—partner in a collaborative or incremental way. A single integration is required after the component is delivered.

3.10.7 Combined Life Cycles as Fit-For-Purpose

Project teams may design a combined life cycle based on project risks. For example, a campus construction project may have multiple buildings to improve and build. An incremental approach would focus resources on completing some buildings earlier than others, accelerating the return on investment. Each individual delivery may be sufficiently understood to benefit from a predictive approach for that building alone.

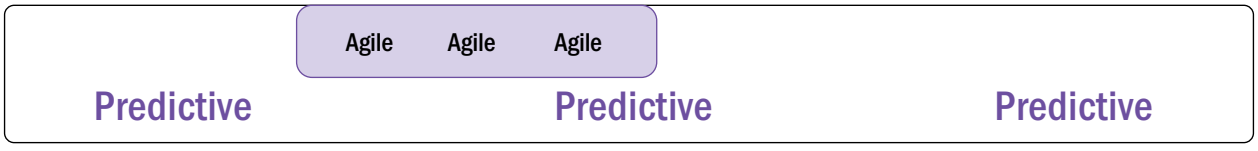


Figure 3-11. A Largely Predictive Approach with Agile Components

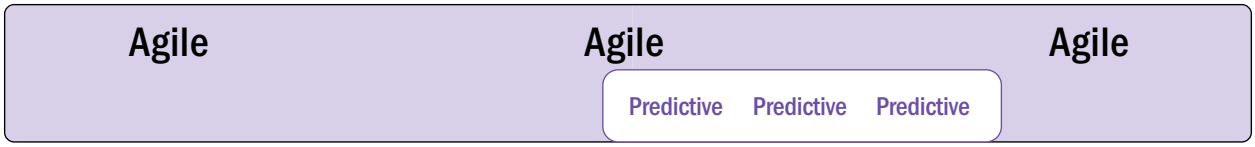


Figure 3-12. A Largely Agile Approach with a Predictive Component

The goal of project management is to produce business value in the best possible way given the current environment. It does not matter if that way is agile or predictive. The questions to ask are:

- How can we be most successful?
- How can we deliver value in short cycles?
- Is feedback needed as the team produces value? If so, increments will help.
- Is it necessary to manage risks as ideas are explored? If so, iterations or agile will help.

When the organization cannot deliver intermediate value, agile approaches may not be useful. Elect a life cycle or a combination of life cycles that work for the project, the risks, and the culture. The decision to use combined life cycles should be periodically revisited during the project, considering changes in context, risks, and delivery needs.

Agile is about customer-based delivery on a frequent basis. That delivery creates feedback for the team. The team uses that feedback to plan and replan the next segment of work.

USE CASE

A government department had a credit insurance application development project. The multiyear project aimed to replace its aging underwriting system with a new, more responsive user interface and system integrations. The bulk of the project was undertaken using an agile approach with continual business input.

The premium rate calculations were handed down from the Organisation for Economic Co-operation and Development (OECD) as a 200-page specification. The steps were very clearly explained with little opportunity for confusion (or interim result confirmation by the business) and were coded by a separate team working its way through the calculation steps. The two teams collaborated on the input variables required for the calculation and how to consume and display the output values, but beyond that, the calculation team worked in a largely predictive manner.

When the calculation team's portion was complete, the outputs from the premium rate calculations were displayed on the screens and in the reports. Then the business users provided feedback on the appearance and use of the information. The two teams ran concurrently, but had little need for interaction. Having them physically close to each other made it easier to check in on development progress, but largely they were separate subprojects.

3.10.8 Combined Life Cycles as a Transition Strategy

Many teams are not able to make the switch to agile ways of working overnight. Agile techniques look and feel very different to those who are accustomed to and have been successful in a predictive environment. The larger the organization and the more moving parts, the longer it can take to transition. For that reason, it makes sense to plan a gradual transition. Transition should also consider the cultural aspects of the organization. Change takes time, and people need to adapt and learn new ways of working.

A gradual transition involves adding more iterative techniques to improve learning and alignment among teams and stakeholders. Later, consider adding more incremental techniques to accelerate value and return on investment to sponsors. This combination of various approaches is considered a continuum of life cycles approach.

Try these new techniques on a less risky project with a medium-to-low degree of uncertainty. Then, when the organization is successful with a continuum approach, try more complex projects that require more of those techniques to be added. This is a way to tailor the progressive transition to the organization's situation and specific risks and the team's readiness to adapt and embrace the changes.

3.11 Mixing Agile Approaches

Agile teams rarely limit their practices to one agile approach. Each project context has its own peculiarities, such as the varied mix of team member skills and backgrounds; the various components of the product under development; and the age, scale, criticality, complexity, and regulatory constraints of the environment in which the work takes place.

Agile frameworks are not customized for the team. The team may need to tailor practices to deliver value on a regular basis. Often, teams practice their own special blend of agile, even if they use a particular framework as a starting point.

USE CASE

Blending Approaches

As an example of tailoring agile frameworks, one of the most common blends in widespread use involves a coordinated use of the Scrum framework, the Kanban Method, and elements of the Extreme Programming (XP) method. Scrum provides guidance on the use of a product backlog, product owner, scrum master, and cross-functional development team, including sprint planning, daily scrums, sprint reviews, and sprint retrospective sessions. A kanban board helps the team to further improve its effectiveness by visualizing the flow of work, making impediments easily visible, and allowing flow to be managed by adjusting work-in-process (WIP) limits. In addition, XP-inspired engineering practices, such as use of story cards, continuous integration, refactoring, automated testing, and test-driven development, further increase the effectiveness of the agile team. In summary, the blend of practices from these various sources produces a synergistic result of higher performance than each individual component in isolation.

3.12 Project Factors That Influence Tailoring

Sometimes project attributes require tailoring an approach for a better fit. Table 3-3 identifies some project factors and tailoring options to consider.

Table 3-3. Tailoring Options to Improve Fit

Project Factor	Tailoring Options
Stakeholder availability or engagement is inconsistent	Build more frequent, lightweight check-ins (like async check-ins or short video updates) and document key decisions transparently to help ensure all stakeholders, even those who miss meetings, can stay informed and engaged.
Customer needs or solution requirements are highly ambiguous or evolving	Use short, experiment-driven discovery cycles (such as rapid prototyping, user testing, and ongoing backlog refinement) to validate assumptions and quickly pivot as the team's understanding improves.
Demand pattern: steady or sporadic	Many teams find that using a cadence (in the form of a regular timebox) helps them demo, retrospect, and take in new work. In addition, some teams need more flexibility in their acceptance of more work. Teams can use flow-based agile with a cadence to incorporate the best of both worlds.
Rate of process improvement required by the level of team experience	Retrospect more often and select improvements.
Flow of work is often interrupted by various delays or impediments	Consider making work visible using kanban boards and experimenting with limits for the various areas of the work process to improve flow.
Quality of the product increments is poor	Consider using the various test-driven development practices. This mistake-proofing discipline makes it difficult for defects to remain undetected.
More than one team is needed to build a product	To scale from one to several agile teams with minimal disruption, first learn about agile program management or formal scaling frameworks. Then, craft an approach that fits the project context.
Project team members are inexperienced in the use of agile approaches	Consider starting by training team members in the fundamentals of the agile mindset and principles. If the team decides to use a specific approach such as Scrum or Kanban, provide a workshop on that approach so the team members can learn how to use it.

Teams should select and test these tailoring options. If things get better, continue them; if things get worse, stop, reexamine, and try it differently or try something else.

For additional guidance on factors that influence tailoring, see Appendix X2, Attributes That Influence Tailoring.

GenAI Insights

GenAI is becoming a valuable asset in product and project management, enhancing decision-making, identifying patterns, and streamlining tasks across development life cycles. Artificial intelligence can analyze large volumes of historical project data to help classify work as definable or high uncertainty, supporting life cycle selection, opportunities for tailoring, and providing stakeholders with clear trade-off recommendations. By leveraging issue logs, change requests, and customer satisfaction (CSAT) data, GenAI can recognize project characteristics and suggest appropriate delivery approaches and decision factors. In product management, AI augments judgment by surfacing insights from customer feedback, market signals, and performance metrics, helping teams prioritize backlogs, shape roadmaps, and align product strategy with organizational goals. GenAI can also support requirement design by assisting with scope definition, identifying dependencies, and generating use-case scenarios that anticipate potential risks or rework. The technology can assess complexity, model "what-if" scenarios, and simulate outcomes to inform portfolio management and resource allocation decisions. When applied thoughtfully, it can improve planning accuracy, accelerate execution, and empower leaders to tailor life cycle approaches to fit context, complexity, and stakeholder needs.

At the same time, AI's growing role in life cycle and product management introduces challenges related to bias, transparency, and overreliance on automation. Also, because large language models (LLMs) have been trained on books, articles, and other online materials, which deliberately shorten project artifacts for brevity and reading ease, they tend to generate similarly abbreviated outputs that might not be detailed enough for real-world use. The outputs should instead be used as starting points for local subject matter expert expansion, not used as is. Likewise, teams using GenAI for product features and ideas may risk mirroring other product features and competing in a "red ocean" of similarity unless they use carefully crafted prompts instructing models to look beyond typical markets for more "blue ocean" opportunities. While GenAI accelerates analysis and execution, it should complement—not replace—the critical thinking, empathy, and ethical judgment of product managers. Overdependence on AI-generated insights can lead to blind trust in data patterns that reinforce outdated methods or overlook emerging risks. Predictive recommendations may oversimplify nuanced trade-offs between innovation, speed, and sustainability, or create a false sense of certainty in dynamic environments.

Section 4

Creating an Agile Environment

Creating an agile project environment requires careful consideration of several key factors, including fostering an agile mindset, adopting lightweight approaches to work, and establishing effective team structures. This section explores these elements in detail, providing guidance for teams and organizations to thrive in an agile landscape.

4.1 Start With an Agile Mindset

The agile mindset is a way of thinking that is not about simply doing agile, but about being agile. It means embracing the values and principles of the *Agile Manifesto* that empower teams to solve problems, adapt quickly, and deliver real value.

Managing a project using an agile approach requires project teams to adopt an agile mindset. The answers to the following reflection questions can help to develop an implementation strategy:

- How can the project team act in an agile manner, focused on delivering customer value?
- How are we continuously learning and adapting based on real feedback and results?
- What can the team deliver quickly and obtain early feedback on to benefit the next delivery cycle?
- How can the team act in a transparent manner?
- What work can be avoided, delayed, or reprioritized to focus on high-priority items?
- How can a supportive-leadership approach benefit the achievement of the team's goals?

4.2 Psychological Safety

Psychological safety refers to a shared belief among team members that the environment is safe for interpersonal risk-taking. This safety includes speaking up, asking questions, sharing ideas, admitting mistakes, and suggesting changes without fear of embarrassment or retaliation. The concept of psychological safety has roots in earlier scholarship and was operationalized at the team level and popularized by Amy Edmondson [12].

Psychological safety creates conditions where individuals feel respected and accepted. When present, it enables open communication, increases engagement, and supports continuous improvement and learning. In contrast, a lack of psychological safety can lead to silence, unresolved problems, and reduced collaboration.

USE CASE

Google's Project Aristotle

Google conducted a multiyear study known as Project Aristotle to identify the factors that make teams effective. After analyzing more than 180 teams, the study concluded that psychological safety was the most critical attribute.

Teams that exhibited high psychological safety were more likely to share ideas, acknowledge errors, and take creative risks. These behaviors led to stronger performance, higher innovation, and better adaptability. The study demonstrated that how teams interact matters more than who is on the team.

Project Aristotle reinforced the importance of team climate and helped shape Google's team development practices.

4.2.1 Success, Learning, and Risk Reduction

Psychological safety supports project success by improving team communication, knowledge sharing, collaboration, risk identification, innovation, engagement, and decision-making. Teams with high psychological safety are more likely to identify problems early, discuss them openly, and take corrective action. This communication reduces the likelihood of project delays, quality issues, or costly rework.

Psychological safety also enhances innovation. By encouraging experimentation and constructive challenges, teams are better able to adapt to changing conditions and deliver creative solutions.

4.2.2 The Four Stages of Psychological Safety

Dr. Timothy R. Clark's Four Stages of Psychological Safety™ model [13] describes psychological safety as progressing through four developmental stages. Each stage builds upon the previous stage, reflecting a growing level of inclusion, engagement, and team maturity.

The four development stages are as follows:

1. **Inclusion safety.** Individuals feel accepted as members of the team. They are recognized for who they are and granted a sense of belonging.
2. **Learner safety.** Team members feel safe to ask questions, experiment, and learn through trial and error. This environment encourages curiosity and active participation.

3. **Contributor safety.** Individuals are empowered to contribute ideas and participate in decision-making. Feedback is welcomed and used constructively.
4. **Challenger safety.** Team members can challenge assumptions and suggest changes without fear of negative consequences. This safety enables innovation and continuous improvement.

Psychological safety increases across these stages. High-performing teams develop the ability to reflect on their work, adjust processes, and engage in difficult conversations constructively.

USE CASE

Retrospectives and Challenger Safety

Retrospectives provide a structured opportunity for teams to reflect and improve. When psychological safety is present, team members are more likely to suggest changes, raise concerns, and identify lessons learned.

Teams often experiment with new practices for a defined period. Those practices that prove valuable are retained, while others are discontinued after reflection. This approach builds a mindset of learning through experimentation. Clark's model [13] can be used as an organizational diagnostic tool and guide for team development actions.

4.2.3 Psychological Safety Across Agile Project Activities

Psychological safety is vital throughout an agile project. Table 4-1 lists aspects of agile work that are positively affected when individuals feel protected and able to speak freely.

Across these cadences, psychological safety converts transparency into inspection and inspection into adaptation—reducing rework, improving flow and quality, and increasing engagement and innovation.

4.2.4 Common Threats to Psychological Safety

Several conditions can limit psychological safety in teams. Any behaviors that shame or punish team members for speaking up discourage open communication. This includes public criticism or assigning blame rather than focusing on solutions.

Hierarchical or cultural barriers may also reduce safety. When team members believe their input will be dismissed due to status, role, or background, they are less likely to contribute.

In high-pressure environments, deadlines or fear of reputational damage may lead individuals to withhold concerns. This fear can prevent early detection of risks and reduce opportunities to improve outcomes.

When teams focus only on execution and neglect reflection, learning shrinks and trust erodes. Agile teams that drop retrospectives are especially at risk. Balance execution with adaptability by making regular space for reflection and innovation. This helps prevent burnout, supports creativity, and strengthens a resilient, high-performing team culture. Remote team environments can undermine psychological safety by diluting informal check-ins, masking tone, and amplifying bystander silence.

Table 4-1. Effects of Psychological Safety on Agile Work

Agile Element	Effect(s) of Psychological Safety
Backlog refinement	Psychological safety enables people to ask clarifying or "naive" questions, challenge assumptions, and slice work into smaller, testable items without embarrassment. A clear definition of ready (DoR) and team working agreements protect inquiry and respectful dissent, reducing ambiguity and hidden risk before planning.
Sprint planning	Safety allows the team to voice capacity limits, technical concerns, and delivery risks early; negotiate scope openly; and agree on a realistic sprint goal based on evidence rather than pressure. Trade-offs are surfaced transparently, creating shared ownership of the plan.
Daily work and daily scrum	Team members raise impediments immediately, ask for help, and swarm on blocked work without fear of blame. A visible board keeps progress and blockers transparent, shortening the time to remove impediments.
Sprint review	The team invites candid stakeholder feedback, treats surprises as data, and adjusts the product backlog openly. Meeting the definition of done (DoD) enhances confidence in the increment, reducing fear of release and supporting constructive challenge.
Sprint retrospective	The team reflects candidly on behaviors and system constraints, updates working agreements, and selects a small number of experiments to run during the next sprint. Learning is prioritized over fault finding, turning insights into concrete changes that the team owns.

4.2.5 The Role of the Leader

Leaders play a central role in creating a psychologically safe environment. Leaders strengthen psychological safety by modeling authentic vulnerability, admitting uncertainties, sharing mistakes, and asking for help. This safety begins by modeling curiosity and acknowledging that they do not have all the answers. Admitting fallibility and responding openly to feedback encourages others to do the same.

Leaders can support psychological safety by establishing inclusive team norms. These norms help guide respectful interaction and create space for diverse perspectives.

Amplifying quieter voices and making space for underrepresented views helps ensure all team members feel heard, with their input acknowledged, recognized, and—ideally—carried forward. This effort also improves decision quality by increasing the range of input.

The way leaders respond to challenges or disagreement sets the tone for the team. When responses are constructive and nondefensive, it signals that learning and improvement are valued over being right.

4.2.6 The Role of the Team

Teams build psychological safety through daily habits. They start with inclusion by treating others as they would like to be treated and by respecting differences in background, working style, and pace. In doing this, team members listen fully and make space for every voice. This involves acting in cooperation, with professionalism and respect for other team members.

Learning comes next. Team members should not be afraid to ask for help, share what they do not know, or treat mistakes as data. Work is not school; helping a teammate helps everyone.

Contribution follows. Team members should see different strengths and weaknesses as an advantage that makes the group stronger than any one person. This involves matching tasks to strengths, pairing team members to cover gaps, and sharing credit.

Finally, team members should challenge how work is done. It is healthy to question the status quo and suggest new approaches when done respectfully and at the right time. Effective teams create small experiments with clear outcomes, review the results together, and then decide and move as one.

4.2.7 Practices to Build Psychological Safety

Psychological safety is cultivated through daily interactions and intentional practices. One effective approach is to define team working agreements collaboratively. These agreements clarify expectations for respectful dialogue, shared accountability, and inclusion. Teams can use pairing/peer feedback practices as additional mechanisms for building trust and mutual learning.

Retrospectives should be conducted in a way that promotes reflection rather than blame. When team members feel safe to speak openly, retrospectives become valuable opportunities for learning and adaptation.

When concerns are not being voiced, anonymous surveys or suggestion tools can provide an alternative channel for feedback. These tools surface insights that may otherwise go unspoken.

Recognizing questions, experimentation, and lessons learned reinforces the importance of continuous improvement. Teams are more likely to innovate when they know their input is valued and failure is treated as a source of insight.

In remote environments, additional attention is needed to build connection. Practices such as regular check-ins and inclusive facilitation help maintain engagement and help ensure all team members are included in discussions.

4.2.8 Diagnosing and Measuring Psychological Safety

Psychological safety can be assessed using both surveys and team observations. One common method is the use of pulse surveys such as Amy Edmondson's seven-question diagnostic [12]. This tool offers a brief but effective way to gauge whether team members feel safe to speak up, admit mistakes, and offer ideas.

Observation of team behavior also provides insight. Signs such as prolonged silence, hesitation to raise concerns, or consistent deferral to authority may indicate limited psychological safety.

Teams can also reflect on their current environment by mapping their position along the four stages of psychological safety. These structured exercises can prompt meaningful conversations and identify areas for growth.

Together, surveys, observations, and reflection provide a more complete picture of team dynamics. These tools also support continuous development of a psychologically safe environment.

USE CASE

Diagnostic Tool: Edmondson's Seven-Item Survey

Amy Edmondson's survey [12] can help assess whether individuals feel safe to take interpersonal risks. However, surveys can be prone to response bias and need to be combined with qualitative methods (check-ins, interviews, focus groups). In the survey, participants respond to statements such as:

- If you make a mistake on this team, it is not held against you.
- It is safe to take a risk on this team.
- Members of this team are able to bring up problems and tough issues.

This diagnostic tool is often used by team leads or facilitators to identify improvement opportunities and support open discussion.

4.2.9 Sustaining Psychological Safety

Psychological safety is an ongoing commitment in which metrics or observable signals—meeting participation rates, open feedback frequency, turnover frequency—play an important role. These signals help teams indicate the presence or absence of psychological safety over time. Project managers, team leaders, and product owners play a dual role. In addition to leading work, they shape the team environment along with the team. This effort includes creating space for open dialogue, encouraging reflection, and shifting the focus from control to support.

Psychological safety aligns with other key leadership concepts. Systems thinking recognizes the influence of structures and culture. Supportive leadership focuses on removing obstacles and enabling team success. A growth mindset emphasizes development over perfection.

Sustaining psychological safety is an ongoing commitment. When it is present, teams are better equipped to navigate complexity, solve problems collaboratively, and deliver lasting value.

4.3 Supportive Leaders Empower the Team

Agile approaches emphasize supportive leadership as a way to empower teams. Supportive leadership is the practice of leading through service to the team, by focusing on understanding and addressing the needs and development of team members to enable the highest possible team performance.

The role of a supportive leader is to facilitate the team's discovery and definition of agile. Supportive leaders practice and radiate agile, approaching project work in the following order:

- **Purpose.** Work with the team to define the "why" or purpose so they can engage and coalesce around the goal for the project. The entire team optimizes at the project level, not the person level.
- **People.** Once the purpose is established, encourage the team to create an environment where everyone can succeed. Ask each team member to contribute to the project work.
- **Process.** Do not plan on following the "perfect" agile process, but instead look for results. When a cross-functional team delivers finished value often and reflects on the product and process, the teams are agile.

The following characteristics of supportive leadership enable project leaders to become more agile and facilitate the team's success:

- Promoting self-awareness;
- Listening;
- Serving those on the team;
- Helping people grow;
- Coaching versus controlling;
- Promoting safety, respect, and trust; and
- Promoting the energy and intelligence of others.

Supportive leadership is not unique to agile. However, once having practiced it, supportive leaders can usually see how well supportive leadership integrates into the agile mindset and value.

When leaders develop their supportive leadership or facilitative skills, they are more likely to become agile. As a result, supportive leaders can help their teams collaborate to deliver value faster.

Successful agile teams embrace the growth mindset, where people believe they can learn new skills. When the team and the leaders believe they can all learn, everyone becomes more capable. In addition to OKRs and goal question metrics (GQMs), KPIs provide a clear view of performance and help connect aspirational outcomes with tangible performance metrics.

4.3.1 Leader Responsibilities

Supportive leaders manage relationships to build communication and coordination within the team and across the organization. These relationships help the leaders navigate the organization to support the team. This kind of support helps to remove impediments and facilitates the team to streamline its processes. Because supportive leaders understand agile and practice a specific approach to agile, they can assist in fulfilling the team's needs.

4.3.1.1 Leaders Facilitate

When project managers act as supportive leaders, the emphasis shifts from managing coordination to facilitating collaboration. Facilitators help everyone do their best thinking and work. Facilitators encourage the team's participation, understanding, and shared responsibility for the team's output. Facilitators also help the team create acceptable solutions.

Supportive leaders promote collaboration and conversation within the team and between teams. For example, a leader helps to expose and communicate bottlenecks inside and between teams. Then the teams resolve those bottlenecks.

Additionally, a facilitator encourages collaboration through interactive meetings, informal dialogue, and knowledge sharing. Leaders do this by becoming impartial bridge builders and coaches rather than by making decisions for which others should be responsible.

4.3.1.2 Leaders Remove Organizational Impediments

The first value of the *Agile Manifesto* is prioritizing individuals and interactions over processes and tools. What better responsibility for a supportive leader to take on than to take a hard look at processes that are impeding team or organizational agility and work to streamline them?

For example, if a department requires extensive documentation, the role of the leader could be to work with that department to review required documentation, assist with creating a shared understanding of how agile deliverables meet those requirements, and evaluate the amount of documentation required so teams are spending more time delivering a valuable product instead of producing exhaustive documentation.

Supportive leaders should also look at other processes that are lengthy, causing bottlenecks, and impeding team or organizational agility. Examples of processes or departments that may need to be addressed include finance, change control boards, or audits. Leaders can partner and work with others to challenge them to review their processes to support agile teams and leaders.

For example, what good is it for the team to deliver a working product every 2 weeks only to have the product fall into a queue or process that could take 6 or more weeks to release due to lengthy release processes? Far too many organizations have these bottleneck processes that prevent teams from quickly delivering valuable products or services. The supportive leader has the ability to change or remove these organizational impediments to support delivery teams.

4.3.1.3 Leaders Pave the Way for Others' Contributions

In agile, the team manages its work process and its work product. That self-management and self-organization applies to everyone supporting the organization and project. Supportive leaders work to fulfill the needs of the teams, projects, and the organization.

Leaders may work with facilities for a team space, work with management to enable the team to focus on one project at a time, or work with the product owner to develop stories with the team. Some leaders work with auditors to refine the processes needed in regulatory environments, and some leaders work with the finance department to transition the organization into incremental budgeting.

Effective leaders focus on paving the way for the team to do its best work. The supportive leader influences projects and encourages the organization to think differently.

TIP

Interpersonal Skills Versus Technical Skills

In addition to supportive leadership, team members should emphasize their interpersonal and emotional intelligence skills, not just their technical skills. Everyone on the team should strive to demonstrate initiative, integrity, emotional intelligence, honesty, collaboration, humility, and a willingness to communicate effectively in various ways, enabling the entire team to work together well.

The team needs these skills so they can respond well to shifts in project direction and technical product changes. When everyone can adapt to the work and one another, the entire team is more likely to succeed.

4.3.1.4 Consider These Leader Responsibilities

Supportive leaders can have many possible titles, but what is most important is what they do. The following are some examples of the responsibilities a supportive leader may have:

- **Educate stakeholders around why and how to be agile.** Explain the benefits of business value based on prioritization, greater accountability and productivity of empowered teams, and improved quality from more frequent reviews, among other benefits.
- **Build the team capabilities through mentoring, encouragement, and support.** Advocate for team member training and career development. The seemingly contradictory quote, "We lead teams by standing behind them," speaks to the role of the leader in developing their team members. Through support, encouragement, and professional development, team members can gain confidence, take on larger roles, and contribute at higher levels within their organizations. A key role of the leader is to nurture and develop team members through and beyond their current roles, even if that means losing them from the team.
- **Leaders should support team members by identifying the root causes of recurrent problems.** Leaders can do this by using fast solution approach techniques to help teams be more efficient and reduce frustration.
- **Help the team with technical project management activities like quantitative risk analysis.** Sometimes, team members may not have knowledge or experience in roles or functions. Leaders who may have more exposure or training in techniques can support the team by providing training or undertaking these activities.
- **Celebrate team successes, examples of team member support, and productive bridge-building activities with external groups.** Create upward spirals of appreciation and goodwill for increased collaboration.
- **Practice emotional intelligence to guide interactions and decisions.** Model self-awareness, active listening, and curiosity. Use nondefensive responses, paraphrasing, and check-ins to surface concerns early. Name emotions and trade-offs in tense moments to deescalate conflicts. Invite dissenting views, acknowledge uncertainty, and repair trust quickly after missteps so collaboration stays productive.

4.3.2 The Role of the Project Manager

The traditional project manager role often needs to be redefined in an agile context because many agile frameworks and approaches do not explicitly describe it. However, pragmatic agile practitioners and organizations realize that project managers can add significant value in many situations. In some agile environments, the project manager becomes more of a coach, helping teams focus on value, adapt to change, and maintain clear communication with internal and external stakeholders.

Agile projects frequently face complexities that extend beyond day-to-day team facilitation, such as handling cross-team dependencies, managing external stakeholder expectations, navigating organizational constraints, and ensuring regulatory or contractual compliance. A project manager may bring coordination and big-picture oversight, helping unblock teams by resolving issues that are outside the direct reach of team leads, facilitators, or product owners. Project managers may act as connectors and enablers: They orchestrate collaboration across multiple teams or vendors, align project goals with strategic business outcomes, secure resources, manage risks, and drive sustained communication with sponsors and executives. In large-scale agile transformations or regulated industries, project managers may help maintain delivery discipline, help ensure transparency, and create stability, making agile principles more sustainable as organizations and projects grow in complexity.

4.3.3 Supportive Leadership

The *PMBOK® Guide*—Eighth Edition [2] defines the project manager as "the person assigned by the performing organization to lead the team that is responsible for achieving the project objectives."

Many project managers are accustomed to being at the center of coordination for the project, often responsible for tracking and representing the team's status to the rest of the organization. This approach worked well when projects were decomposed into siloed functions.

However, in high-change projects, there is more complexity than one person can manage. Instead, cross-functional teams coordinate their own work and collaborate with the business representative (the product owner).

When working on an agile project, project managers shift from being at the center to supporting the team and leadership. In an agile environment, project managers are supportive leaders, changing their emphasis to coaching people who want help, fostering greater collaboration on the team, and aligning stakeholder needs. As a supportive leader, project managers encourage the distribution of responsibility to the team—the people who have the knowledge to get work done.

TIP

In larger agile-delivery settings, project managers may often move into more external-facing, coordination-type roles. They may be responsible for aligning or coordinating across multiple teams and dependencies, escalating and removing organization-level impediments, and/or guiding teams through enterprise processes and governance.

4.4 Team Composition

A core tenet in both the values and the principles of the *Agile Manifesto* is the importance of individuals and interactions. Agile approaches optimize the flow of value, emphasizing rapid feature delivery to the customer rather than focusing on keeping people busy.

TIP

Agile Principle #5

Build projects around motivated individuals. Give them the environment and support they need and trust them to get the job done.

When teams think about how to optimize the flow of value, the following benefits become apparent:

- People are more likely to collaborate.
- Teams finish valuable work faster.
- Teams waste much less time because they do not multitask or need to reestablish context.

4.4.1 Agile Teams

Agile teams focus on delivering smaller increments of value in shorter timeframes so they can obtain customer feedback and pivot/respond accordingly to that feedback. In practice, the most effective agile teams tend to range in size from three to nine members to keep communication channels manageable without formal structures. Ideally, agile teams are given the tools to enable face-to-face collaboration, whether it be through colocation or through the use of videoconferencing tools. Team members should be 100% dedicated to the teams. Agile encourages self-managing teams, where team members decide who will perform the work within the next period's defined scope. Agile teams thrive with supportive leadership. The leaders assist the team's approach to their work.

Cross-functional agile teams produce functional product increments frequently. That is because the teams collectively own the work and together have all the necessary skills to deliver completed work.

Regardless of the overall agile approach, the more a team limits its work in process (WIP), the more likely its members can collaborate to expedite work across the board. Team members in successful agile teams collaborate in various ways, such as pairing, swarming, and mobbing, to avoid falling into the trap of mini-waterfalls instead of collaborative work. *Mini-waterfalls* occur when the team first addresses all of the requirements in a given period, then attempts to do all of the design, and then moves on to do all of the building. Using this scenario, at some point in the building or testing following the building, the team may realize it had assumptions that are no longer valid. In this case, the team wasted time addressing all of the requirements. Instead, when team members collaborate to produce a small number of features across the board, they learn as they proceed and deliver smaller finished features.

Table 4-2. Attributes of Successful Agile Teams

Attribute	Goal
Dedicated people	• Increased focus and productivity • Small teams (fewer than 10 people)
Cross-functional team members	• Develop and deliver often • Deliver finished value as an independent team • Integrate all work activities to deliver finished work • Provide feedback from inside the team and from others such as the product owner
Face-to-face collaboration	• Better communication • Improved team dynamics • Knowledge sharing • Reduced cost of learning • Able to commit to working with one another
Mixed team of generalists and specialists	• Specialists provide dedicated expertise and generalists provide flexibility of who does what • Team members bring their specialist capabilities and often become generalizing specialists, with a focused specialty plus breadth of experience across multiple skills
Stable work environment	• Depend on one another to deliver • Agreed-upon approach to the work • Simplified team cost calculations (run rate) • Preservation and expansion of intellectual capital

Agile projects benefit from project team structures that improve collaboration within and among the teams. Table 4-2 shows how collaborative team members boost productivity and facilitate innovative problem-solving.

4.4.2 Agile Roles

In agile approaches, three common roles are used (see Table 4-3):

- Cross-functional team members,
- Product owner, and
- Team facilitator/supportive leader.

Table 4-3. Agile Team Roles

Role	Description
Cross-functional and self-organizing team members	Cross-functional teams have all the skills necessary to produce a working product. Cross-functional teams consist of professionals who deliver a potentially releasable product at a regular cadence. Cross-functional teams are critical because they can deliver finished work in the shortest possible time, with higher quality and without external dependencies. They are also self-organizing. Team members decide how to do the work, share ownership of outcomes, and adjust roles as needed. They align on clear goals and simple guardrails, then manage tasks and quality practices. They inspect results often, learn quickly, and adapt plans so the product improves each cycle.
Product owner	The product owner is responsible for guiding the direction of the product. Product owners rank the work based on its business value. Product owners work with their teams daily by providing product feedback and setting direction on the next piece of functionality to be developed/delivered. That means the specific work is small, often small enough to be described on one index card. The product owner works with stakeholders, customers, and the teams to define the product direction. Typically, a product owner can be the customer business owner and bring deep subject matter expertise to decisions. Sometimes, the product owner requests help from people with deep domain expertise, such as architects, or deep customer expertise such as product managers. Product owners need training on how to organize and manage the flow of work through the team. In agile, product owners create the backlog for and with the team. The backlog helps the teams see how to deliver the highest value without creating waste. A critical success factor for agile teams is strong product ownership. Without paying attention to the highest value for the customer, the agile team may create features that are not appreciated or otherwise insufficiently valuable, thus a waste of effort.
Team facilitator	The third role typically seen on agile teams is of a team facilitator—a supportive leader. This role may be called a project manager, scrum master, project team lead, team coach, or team facilitator. All agile teams need supportive leadership on the team. People need time to build their leadership skills of facilitation, coaching, and impediment removal.

TIP

Organizations may invite external agile coaches to help them when their internal coaching capability is not yet fully developed.

External coaches have the advantage of experience but the disadvantage of weak relationships in the client organization. Internal coaches, conversely, have strong relationships in their organization but may lack the breadth of experience that comes from exposure to more agile project configurations.

4.4.3 Clarifying Responsibilities

With the introduction of agile practices and new ways of working, there is often a need for focused effort on clarifying roles and responsibilities within a team or organization. When teams understand contribution patterns for their delivery activities, they make faster decisions, reduce handoffs, and deliver value more effectively. However, achieving this clarity while maintaining agile flexibility presents a common challenge for organizations.

The problem is not that agile teams need more structure; it is that they need the right kind of clarity to support collaboration and speed. Traditional approaches often miss this distinction, leading organizations to swing between two problematic extremes.

Organizations transitioning to agile practices frequently oscillate between approaches that undermine effective delivery with the following extremes:

- **Full autonomy (self-organizing).** In pursuit of agile flexibility, some organizations eliminate all role definitions, expecting cross-functional teams to self-organize completely. While this supports team ownership, it often creates:
  - Diffusion of responsibility, where critical work falls through the cracks,
  - Bloated meetings where everyone feels they must be involved in everything,
  - Unclear decision rights that slow down progress, and
  - Team member uncertainty about expected contributions.
- **Prescriptive frameworks (rigid role definitions).** At the opposite extreme, organizations implement detailed frameworks with explicit role definitions, often layering new structures on top of existing ones without reconciliation. This approach typically results in:
  - Bureaucratic handoffs that slow delivery,
  - Overlapping responsibilities across departments,
  - Renamed roles without redefined purpose, and
  - Accountability gaps, where multiple people own the same outcome.

Both extremes fail because they focus on organizational structure rather than work clarity. The first extreme provides no guidance for contribution; the second constrains contribution through rigid boundaries.

The introduction of scaling frameworks has sometimes amplified these problems. Organizations adopt new roles and responsibilities without addressing existing overlaps, creating layers of complexity rather than clarity. Leadership changes and evolving delivery models further complicate the landscape, contributing to redundant roles and inconsistent expectations across teams.

Without deliberate coordination, teams experience confusion about individual contributions, responsibilities, levels of autonomy, and decision-making authority when traditional role definitions fail to align with agile constructs.

4.5 A Lightweight RAF Model

Agile organizations benefit from simple, transparent approaches to aligning responsibilities, especially as work spans functions, roles, and delivery layers. Rather than rewriting job descriptions or creating complex organizational charts, teams can clarify who contributes what to specific activities using lightweight, work-focused models.

The responsible, accountable, facilitator (RAF) model (see Table 4-4) offers a practical alternative to traditional RACI (responsible, accountable, consulted, informed) matrices. The RAF model is a practical, lightweight alternative to a RACI when teams and organizations need clear guidance in execution and delivery. In some teams and organizations, we find more time and energy spent worrying about the "consulted" and "informed" aspects; in reality, it's more important that we focus on ownership and accountability.

This model fits well in agile settings, where collaboration and speed are critical. The RAF model focuses on activities and outcomes rather than job titles and can be applied selectively where confusion exists. Teams can use a lightweight RAF model (see an example in Table 4-5) to clarify recurring work such as the following:

- Defining and validating requirements,
- Setting or updating technical standards,
- Managing cross-team dependencies, and
- Leading planning and coordination.

Table 4-4. The Responsible, Accountable, Facilitator (RAF) Model

Role	Definition	When to Use
Responsible	Who actively performs the work	When multiple people could do the work but someone needs to drive it
Accountable	Who owns the outcome or decision	When decision rights are unclear or when multiple stakeholders claim ownership
Facilitator	Who supports the process, resolves blockers, and maintains momentum	Helpful for complex work requiring coordination across teams or functions

Table 4-5. Example of a Lightweight Responsible, Accountable, Facilitator (RAF) Model

	Product Owner	Team Facilitator	Business Analyst	Developers	Quality Assurance
Product roadmap	A, R	F			
Requirements/user stories	A		R		
Team sync		F		A, R	A, R
Develop and execute test cases					A, R
Stakeholder review	A	F		R	R

Note: R = responsible, A = accountable, F = facilitator

TIP

Start with activities causing delays or confusion. Apply an RAF model only where clarity is missing, not as a universal requirement.

4.5.1 The Benefits of Role Clarity

When responsibilities are clear for specific work, teams experience multiple benefits such as the following:

- **Reduced duplicated effort** through clear contribution boundaries,
- **Faster decision-making** with defined decision rights,
- **Improved delivery speed** by eliminating unnecessary handoffs,
- **Psychological safety** as team members understand their expected contributions, and
- **Better collaboration** through shared understanding of who does what.

This clarity supports rather than constrains agile flexibility. Teams can adapt their approach to different types of work while maintaining a shared understanding of contribution patterns.

4.5.2 Implementation Guidelines

Successful responsibility clarification in agile environments follows several key practices, including the following:

- **Focus on activities, not titles.** Define who does what for specific work rather than creating rigid job descriptions.
- **Apply selectively.** Use lightweight models only where confusion exists, not as universal organizational requirements.
- **Adapt to context.** Allow teams to adjust their responsibility patterns according to the work at hand and their team's capabilities.
- **Emphasize outcomes.** Help ensure clarity supports faster delivery and better decisions, not bureaucratic compliance.
- **Iterate and improve.** Review and adjust responsibility patterns during retrospectives or team reviews.
- **Retire stale definitions.** Stop using responsibility models that no longer influence decisions or resolve confusion.

By following these practices, organizations can achieve the clarity needed for effective agile delivery without sacrificing the flexibility that makes agile practices valuable.

USE CASE

Financial Services Transformation

A U.S.-based financial services organization experienced frequent shifts in executive leadership over 4 years, with each new leader introducing different frameworks, vendors, roles, and operating models. However, little effort was made to streamline or retire legacy structures, creating a highly complex software development life cycle with multiple approval points, overlapping roles, and fragmented accountability.

The organization was overly complex and had multiple leaders involved in governance and sign-offs due to a lack of trust, basically requiring everyone to be consulted and informed. This arrangement seriously slowed down the teams as well as delivery of value to customers. To address this, teams applied a lightweight RAF delivery model to clarify responsibilities for recurring activities such as scope definition, architectural review, and release coordination. This model not only helped the teams to focus on their roles and activities but also required the organization to help ensure high-performing individuals were in all the roles, built trust into the process, and streamlined delivery. This approach helped reduce unnecessary handoffs, define decision rights, and create consistent delivery experiences across teams and departments.

4.6 Generalizing Specialists

Agile teams are cross-functional even though their members often do not start off that way. However, many successful agile teams are made up of generalizing specialists, or individuals with T-shaped skills.

T-shaped skills refer to a focused specialty plus a breadth of experience across multiple skills rather than a single specialization. Agile team members work to develop such characteristics as a result of intense collaboration and self-organization to swarm and get work done quickly, which requires them to routinely help one another.

A single person's throughput is not relevant. Focusing on a single person's throughput may even be harmful if it creates a bottleneck for the rest of the team. The goal is for the team to optimize the delivery of finished work to get feedback.

If the customer desires great results, such as rapid feature delivery with excellent quality, the team cannot be structured just with specialist roles in an attempt to maximize resource efficiency. The team's objective is flow efficiency, optimizing the throughput of the entire team. Small batch sizes promote working together as a team. The product owner's job is to make sure the team works on the highest value work.

4.6.1 I-Shaped Skills and T-Shaped Skills

Some people have deep specializations in one domain but rarely contribute outside of that domain. These people are known in agile communities as *I-shaped people* as a result of their I-shaped skills. By contrast, *T-shaped people* supplement their expertise in one area with supporting—but less-developed—competencies in associated areas and good collaboration skills. As an example, a person who can test some areas of the product and also develop different areas of the product is considered a T-shaped person.

A T-shaped person has a defined, recognized specialization and primary role as well as the skills, versatility, and aptitude for collaboration to help other people when and where necessary. This collaboration reduces handoffs and the constraints of having only one person who is able to perform the work.

4.7 Team Structures

Teams have adopted agile principles and practices across many industries. These industries organize people into cross-functional teams to iteratively develop working products.

USE CASE

The core team that assembled to write this practice guide had varied backgrounds—some represented PMI and some represented Agile Alliance. The team was self-organizing and worked in increments to complete the work. PMI assembled a group of subject matter experts to inspect the work, which allowed the team to incorporate feedback and improve the product as it was developed.

Some organizations have been able to create colocated, cross-functional teams; other organizations function differently. Instead of having all team members colocated, some organizations have distributed or dispersed teams. *Distributed teams* employ cross-functional teams in different locations. *Dispersed teams* may have each team member working in a different location, either in an office or from home. While these arrangements are not ideal due to increased communication costs, they are a reality for many teams today.

In one large, U.S.-based financial institution, a program included team members based on the East Coast of the United States as well as in several locations throughout India. When the team first started, it was one large, dispersed team (user experience [UX] roles, analysts, developers, and testers) doing an "around the clock" development practice, where some working time overlapped across team members to do warm handoffs with the work. Team members conducted daily coordination meetings and used videoconferencing tools to include all team members. Key roles (analysts, product owners, UX designers, and development leads) in the United States would come in early to answer any questions from their India-based team members and help resolve impediments.

As the product started growing and more funding came through, the team decided to break into five smaller teams. To do this, they built colocated, distributed teams in various locations. The team also decided to build cross-functional teams in each of these locations consisting of colocated developers and testers.

The team had a core set of analysts as well based in the two U.S. locations, who worked with their U.S.-based product manager and product owners and then collaborated with each of the teams, respectively. Although some structure was in place where product reviews were conducted as an entire program, most of the other activities were conducted at a team level based on what worked best for each team. This arrangement allowed the teams to self-organize.

4.7.1 Dedicated Team Members

What happens when the team members' time is not 100% dedicated to the team? While this condition is not ideal, it unfortunately cannot be avoided in all situations.

The key problem with having someone invest only a capacity of 25% or 50% to the team is that team members will have to multitask and task-switch. Multitasking reduces the throughput of the team's work and impacts the team's ability to predict delivery consistently.

TIP

Multitasking slows the progress of the entire team because team members waste time on task-switching and/or waiting for one another to finish other work. When people are 100% dedicated to the team, the team has the fastest possible throughput.

Teams experience productivity losses somewhere between 20% and 40% when task-switching. This loss increases exponentially as the number of tasks increases.

When a person multitasks between two projects, that person is not 50% on each project. Instead, due to the cost of task-switching, the person is somewhere between 20% and 40% dedicated to each project.

People are more likely to make mistakes when they multitask. Task-switching consumes working memory, and people are less likely to recall context when they multitask.

When everyone on the team is 100% allocated to one project, they can continuously collaborate as a team, making everyone's work more effective.

USE CASE

Since the core team members who developed this practice guide could not dedicate 100% of their capacity to the team's efforts, their throughput was substantially lower than what it might have been if they could have afforded to colocate and invest their full-time attention to the project. However, it was not feasible to colocate and focus at full capacity. Therefore, the team identified their dispersion as a potential risk. The team tracked and monitored the progress of their work using collaborative tools and adjusted assignments based on individual capacity accordingly.

Annex A1 of this practice guide provides details on the application of agile within the project management performance domains presented in the *PMBOK® Guide*—Eighth Edition [2]. Refer to the annex for more tips for teams working in agile environments, specifically relating to the processes in the Resources performance domain.

TIP

Not all teams have all the roles that they need. For example, some teams need support from external areas such as security, database administrators, or research analysts. When a team has temporarily assigned specialists, it is important to help ensure that everyone has the same set of expectations. Is this specialist 100% allocated to the team and for how long? Set expectations with everyone (the specialist and the team) to clarify the level of commitment so the team can deliver. Part-time assignments can create risks for the project.

4.7.2 Remote and Hybrid-Location Teams

Remote and hybrid-location work models are now widely adopted across many industries. What began as a temporary solution during the COVID-19 pandemic has evolved into a long-term approach for organizing work. Remote collaboration is no longer considered a short-term fix. It is now a strategic decision that helps organizations access talent, reduce costs, and improve flexibility.

A remote team consists of individuals working from separate locations rather than a shared office. These locations may span cities, time zones, or countries. A hybrid-location team includes both on-site and remote members who collaborate using a mix of physical and virtual methods. Both models depend on digital tools, asynchronous communication, and shared work rhythms.

Before the pandemic, most companies operated in colocated, office-based environments. Remote work was limited to specific roles. When full remote work became necessary, teams quickly adapted.

Many roles previously thought to require an on-site presence, including call center agents, product managers, and teachers, shifted to remote work. This change forced a reexamination of how and where work gets done.

As conditions stabilized, organizations developed new work strategies. Some eliminated offices entirely and adopted a distributed model. Others downsized to focus on collaborative spaces rather than individual workstations. Return-to-office (RTO) policies vary, ranging from set in-office days to full flexibility. These changes continue to shape organizational culture, satisfaction, and retention.

Reactions to these models differ. Some team members value hybrid models for their work-life balance. Others prefer full remote work. In some cases, RTO policies have led to voluntary attrition.

4.7.3 Benefits and Challenges of Remote Work

Remote and hybrid models offer real benefits but also introduce new complexities. Organizations should balance both models to develop a sustainable model. The benefits and challenges of each model may include:

Benefits:

- Improved productivity due to fewer workplace distractions,
- Lower costs for both organizations and team members,
- Wider talent pools through the removal of geographic limits,
- Increased flexibility and reduced stress,
- Reduced environmental impact from less commuting, and
- More diverse perspectives, leading to broader innovation.

Challenges:

- Proximity bias may impact recognition and advancement for remote team members,
- Lower cohesion and trust without in-person interaction,
- Greater security risks requiring strong cybersecurity,
- Difficulty separating work from personal time,
- Fatigue from tool use and digital overload,
- Need for managers to shift leadership and engagement approaches,
- Difficulty onboarding and mentoring remote team members, and
- Time zone barriers and challenges with async communication.

Agile organizations that apply thoughtful, adaptive practices can address these trade-offs and build strong teams across all environments.

4.7.4 Key Success Factors for Remote Teams

Remote and hybrid teams require new practices. Agile principles still apply, but their implementation changes in virtual settings.

TIP

Remote work shifts the focus. It is no longer about tracking hours or physical presence. Teams are measured by the outcomes they deliver. Trust replaces in-office observation as the foundation of team leadership. Connection is built through shared goals and regular communication, not proximity.

Successful remote teams typically implement the following core practices:

- Build trust through purpose, accountability, and ownership.
- Set clear communication norms about availability and tool use.
- Choose collaboration tools that support various types of work.
- Use video in meetings when possible to support human connection.
- Allow for individual focus and flexible schedules.
- Create team rituals like check-ins or virtual coffee chats.
- Celebrate team and individual progress.
- Support work-life boundaries to avoid burnout.
- Promote inclusion by rotating time zones, using accessible formats, and inviting all voices.

These practices help remote and hybrid teams stay aligned and productive without relying on a shared workspace.

USE CASE

A global organization operating across seven time zones adopted a distributed model as part of its core strategy. The organization closed its main office and offered stipends for home offices or coworking spaces. To maintain connection, the company hosts quarterly in-person gatherings. This blend of flexibility and engagement supports alignment and innovation.

4.7.5 Agile Practices for Remote and Hybrid-Location Teams

Agile practices work well in remote settings with adjustments. Teams should account for time zones, fewer in-person interactions, and heavy use of digital tools. The following practices can help teams to create ways of working that are adaptable to their needs:

- **Working agreements.** Teams should cocreate and revisit agreements that guide communication, availability, and tool use. These agreements reduce friction and clarify expectations.
- **Daily coordination meetings.** Keep these short and focused. Use videoconferencing tools for connection. In highly distributed teams, consider asynchronous communication using chat or documents.
- **Sprint/iteration planning.** Use digital boards for shared visibility. Send goals or context in advance. Capture key decisions in shared documents.
- **Backlog refinement.** Hold frequent, lightweight refinement sessions. Use collaboration tools to update or clarify items between meetings.
- **Retrospectives.** Create a safe space using tools like whiteboards or polls. Rotate facilitation. Enable anonymous feedback if needed.
- **Reviews and demos.** Encourage video participation and Q&A. Record sessions for those who cannot attend. In hybrid settings, help ensure equal participation for remote team members.
- **Information radiators.** Use digital kanban boards and dashboards to replace physical boards. Keep them current and use them in meetings.
- **Pairing and swarming.** Use screen sharing and chat tools for virtual pairing. Swarm using short work blocks and team check-ins.
- **Async communication and updates.** Use written updates, recorded videos, or message threads to reduce meetings and keep everyone informed.

USE CASE

Workplace Systems International Ltd, a United Kingdom-based software provider, adapted agile practices to fit a hybrid model. All events became virtual to support equal participation. Facilitators rotated to build shared ownership. Visual collaboration tools helped teams help ensure alignment. These adjustments helped the company maintain agility while working in a hybrid environment.

Adapting agile for remote work is not about reinventing practices. It is about preserving core values like collaboration, transparency, and continuous delivery.

GenAI Insights

GenAI can play a valuable role in strengthening agile practices by streamlining operations, revealing hidden insights, and improving collaboration. For remote and hybrid teams, GenAI can summarize chats, draft updates, and coordinate handoffs across time zones. AI tools can also convert meeting minutes, task descriptions, and complex documents into different languages. This makes cross-language collaboration easier and can reduce miscommunication, although individuals should still confirm critical information. GenAI cuts meeting time, reduces miscommunication, and speeds shared understanding. The tool can read team or kanban boards and chat messages to create daily briefs, flag blockers, suggest next steps, and propose schedules that work across multiple time zones and locations. GenAI also can support psychological safety and inclusion by analyzing sentiment in chat platforms to detect early signs of frustration or burnout, gathering anonymous input during retrospectives so everyone has a voice, and reviewing transcripts or team meetings to flag unconscious bias in language.

These same capabilities can cause harm when they are applied without care. Sentiment analysis and monitoring may feel more like surveillance than support, prompting people to hold back rather than speak openly—or engage in "sentiment stuffing," where people deliberately speak up or make comments just for the sentiment analysis tools. This feigning of agreement, or support, erodes the purpose of open discussion. Anonymous input tools can erode trust if employees doubt their confidentiality, whereas bias detection may misinterpret tone or cultural nuance, flagging issues that may not actually exist. Overautomation of agile practices can stifle learning and dialogue, reducing team collaboration to an "AI says so" exercise. In these instances, GenAI should be used as an augmenter: a way to streamline operations, reveal hidden insights, and improve collaboration without replacing empathy, adaptability, and critical thinking.

Section 5

Delivering in an Agile Environment

This section explores key agile practices and team structures that underpin successful agile approaches to project management, including the composition and dynamics of agile teams, the implementation of agile approaches, and common challenges that can affect project delivery.

5.1 Charter the Project and the Team

Every project should have a project charter so the project team knows why this project matters, where the team is headed, and what the project objective is. However, the project charter itself may not be enough for the team. Agile teams require team norms and an understanding of how to work together. As a result, teams should consider creating a team charter.

The chartering process helps a team learn how to work together and coalesce around the project.

At a minimum, agile project teams should establish the project vision or purpose and a clear set of working agreements for their agile projects.

5.1.1 Agile Project Charter

An agile project charter answers the following questions:

- **Why are we doing this project?** This is the project vision/objectives.
- **Who benefits and how?** This may be part of the project vision and/or project purpose.
- **What does "done" mean for the project?** These are the project's release criteria.
- **How are we going to work together?** This explains the intended flow of work.

A supportive leader may facilitate the chartering process. A team can coalesce by working together, and the project charter is a great way to start working. In addition, team members may want to collaborate to understand how they will perform this work together.

Agile project charters should be lightweight, and teams may not need a formal process for chartering as long as team members understand how to work together. Some teams benefit from a team-chartering process.

5.1.2 Team Charter

The following are some chartering ideas for team members to use as a basis for their social contract:

- Team values such as sustainable pace and core hours;
- Working agreements such as what "ready" means so the team can take in work, what "done" means so the team can judge completeness consistently, respecting the timebox, or the use of WIP limits;
- Ground rules such as one person talking in a meeting; and
- Group norms such as how the team treats meeting times.

The leader, together with the team, may decide to address other behaviors. Remember that the team's social contract—its team charter—is how the team members interact with one another. The goal of the team charter is to create an agile environment in which team members can work to the best of their ability as a team. See PMI online digital systems for examples of team canvases, templates, and agile project charters.

5.2 Common Agile Practices

Sections 5.2.1 through 5.2.8 describe some of the most common agile project practices.

5.2.1 Backlog Planning

The backlog is the ordered list of all the work for a team. There is no need to create all of the stories/tasks for the entire project before the work starts—only enough to understand the first release in broad brushstrokes and then sufficient items for the next iteration. Common backlog ordering (prioritization) schemes include those found in Table 5-1.

Table 5-1. Common Backlog Prioritization Techniques

Project Factor	Tailoring Options
MoSCoW (must have, should have, could have, and will not have)	Bucket by necessity. Fast for workshops; coarse-grained and subjective.
Stack rank/forced ranking	One ordered list (no ties). Brutally clarifying; sparks debate; needs a single decision owner.
Value versus effort matrix (2 × 2)	Plots impact against effort. Good for visuals and quick cuts; depends on rough estimates.
Cost of delay (including CD3 = CoD ÷ Duration)	Prioritizes by time-sensitivity and economic impact.
WSJF (weighted shortest job first)	(User business value + time criticality + risk reduction) ÷ size Scales well in backlogs; still relies on relative scoring.
Kano	Classifies features as basic, performance, or delighter. Guides outcome mix; not a queueing method.
Buy-a-feature/$100 test	Stakeholders "spend" budget on items. Surfaces true preferences; can be gamed if scope is unclear.

Product owners (or a product owner value team that includes the product manager and all relevant product owners for that area of the product) might produce a product roadmap to show the anticipated sequence of deliverables over time. The product owner replans the roadmap based on what the team produces and emerging information. Backlog planning not only structures the team's work but also creates transparency and alignment, helping everyone focus on delivering value continuously.

5.2.2 Backlog Refinement

The product owner should work with the team to prepare user stories for the upcoming iteration. These preparation meetings are typically called *refinement sessions* and are conducted in the middle of the iteration to get ready for upcoming iterations. When individuals are unavailable for whatever reason, the product owner and team discuss openly, confirm that the team has reduced capacity, and plan accordingly. The purpose of these sessions is to refine enough stories so the team understands what the stories are and how large the stories are in relation to one another. Having a DoR enhances collaboration between agile teams and end stakeholders to help ensure that requirements are gathered consistently across user stories. This also helps with larger working agreements among agile teams, stakeholder groups, leadership, and PMOs to arrive at standards/guidelines for user stories and prioritization.

There is no consensus on how long a refinement should be. A continuum of details exists that may include the following:

- **Just-in-time (JIT) refinement for flow-based agile.** The team takes the next card off the prioritized to-do column and discusses it.
- **A timeboxed, 1-hour discussion midway through a 2-week iteration.** The team selects an iteration duration that provides them frequent-enough feedback.
- **Multiple refinement discussions for iteration-based agile teams.** Teams can use this when they are new to the product, product area, or problem domain.

TIP

Consider using impact mapping to see how the product fits together. Under typical circumstances, the product owner leads this work. A supportive leader can facilitate any necessary meetings as a way of helping the project.

Refinement meetings allow the product owner to present story ideas to the team and for the team to learn about potential challenges or problems in the stories. If the product owner is unsure of the dependencies, the product owner can request the team to spike the feature to help understand the risks.

There are many ways for the product owner to conduct backlog preparation and refinement meetings, including the following:

- Encourage the team to work as triads of developer, tester, and business analyst/product owner to discuss and write the story.
- Present the overall story concept to the team. The team should discuss and refine it into as many stories as required.
- Work with the team to find various ways to explore and write the stories together, making sure all of the stories are small enough so the team can produce a steady flow of completed work. Consider working toward the goal of completing a story at least once a day.

Teams often have a goal of spending not more than 1 hour per week refining stories for the next batch of work. Teams want to maximize the time spent doing work as opposed to planning work. If the team needs to spend more than 1 hour per week refining stories, the product owner could be overpreparing, or the team may be lacking some critical skills needed to evaluate and refine the work.

5.2.3 Planning for Iteration-Based Agile

Each team's capacity is different. Each product owner's typical story size is different. Teams consider their story size so they do not try to commit to more stories than there is team capacity to complete within one iteration.

When people are unavailable (e.g., holidays, vacations, or anything that prevents people from participating in the next set of work), the team will not be able to finish the same amount of work as it finished in the previous time period. When teams have a reduced capacity, they will only plan for work that meets that capacity.

Teams estimate what they can complete, which is a measure of capacity (see Section 5.5 for examples). Teams cannot predict with 100% certainty what they can deliver since they cannot know the unexpected. When product owners make the stories smaller and teams see progress in the form of a finished product, teams learn what they are able to do for the future.

Agile teams do not plan just once in a single chunk. Instead, agile teams plan a little, deliver, learn, and then replan a little more in an ongoing cycle.

TIP

Draw the team's attention to the anti-pattern (common problem/issue) and help the team discover how to improve its daily coordination meeting.

5.2.4 Daily Coordination Meetings

Teams use daily coordination meetings to communicate, microcommit to one another, uncover problems, and help ensure the work flows smoothly through the team.

Timebox the event to no longer than 15 minutes. During the daily coordination meeting, the team reviews its task or kanban board, and anyone from the team can facilitate the meeting.

In iteration-based agile, everyone answers the following questions in a round-robin fashion:

- What have I completed since the last meeting?
- What am I planning to complete between now and the next meeting?
- What are my impediments (or risks or problems)?

Questions like these generate answers that allow the team to self-organize and hold one another accountable for completing the work they committed to the day before and throughout the iteration.

Flow-based agile has a different approach to these meetings, focusing on the team's throughput. The team assesses the board from right to left. Questions typically include the following:

- What do we need to do to advance this piece of work?
- Is anyone working on anything that is not on the board?
- What do we need to finish as a team?
- Are there any bottlenecks or blockers to the flow of work?

One of the antipatterns typically seen is that daily coordination meetings become status meetings. Teams that have traditionally worked in a predictive environment may tend to fall into this antipattern since they are used to providing a status update.

Another antipattern typically seen in daily coordination meetings is that the team begins to solve problems as they become apparent. Daily coordination meetings are for realizing there are problems, not for solving them. Add the issues to a "parking lot," create another meeting (that might immediately follow the daily coordination meeting), and solve problems there.

Teams run their own daily coordination meetings. When run well, these meetings can be very useful—provided the nature of the team's work requires intense collaboration. Make a conscious decision about when the team can effectively use daily coordination meetings.

TIP

Encourage any team member to facilitate the daily coordination meeting—instead of a team facilitator or leader—to help ensure it does not turn into a status meeting. The team should use this event to self-organize and make commitments to one another.

5.2.5 Working Collaboratively

Agile teams create value fastest when they collaborate in real time on small slices of work that yield useful feedback. A cross-functional group that includes product, development, and any other needed skills should share a single DoD, work in close proximity (or stay connected through video and shared digital whiteboards for distributed contexts), and swarm the highest priority story until it is complete.

Common techniques include pairing or mobbing, which includes rotating roles so every person on the team is involved with both driving and reviewing work, and using visible artifacts such as story maps or cumulative flow diagrams to expose progress and bottlenecks. By limiting WIP and focusing everyone on the same goal, the team shortens decision paths, uncovers knowledge gaps early, and builds collective ownership of quality.

5.2.6 Demonstrations/Reviews

As the team completes the features, usually in the form of user stories, it periodically demonstrates the working product. The product owner sees the demonstration and accepts or declines stories.

In iteration-based agile, the team demonstrates all completed work items at the end of the iteration. In flow-based agile, the team demonstrates completed work when it is time to do so, usually when enough features have accumulated into a set that is coherent. Teams, including the product owner, need input to decide how early to ask for product feedback.

As a general guideline, demonstrate whatever the team has as a working product at least once every 2 weeks. That frequency is enough for most team members to obtain feedback that prevents them from heading in the wrong direction. Two weeks is also frequent enough for teams to keep the product development clean enough to build a complete product as often as they want or need to.

A fundamental part of what makes a project agile is the frequent delivery of a working product. A team that does not demonstrate or release cannot learn fast enough and is likely not adopting agile techniques. The team may require additional coaching to enable frequent delivery.

5.2.7 Retrospectives

An important agile practice, if not the most important agile practice, is the retrospective because it allows the team to learn about, improve, and adapt its process.

Retrospectives help the team learn from its previous work on the product and its process. One of the principles behind the *Agile Manifesto* is: "At regular intervals, the team reflects on how to become more effective, then tunes and adjusts its behavior accordingly."

Many teams use iterations—especially 2-week iterations—because the iteration prompts a demonstration and a retrospective at the end. However, the team does not need iterations to retrospect. Team members may decide to retrospect at the following key times:

- When the team completes a release or ships something (The release does not have to be a monumental increment. It can be any release, no matter how small.);
- When more than a few weeks have passed since the previous retrospective;
- When the team appears to be stuck and completed work is not flowing through the team; or
- When the team reaches any other milestone.

Teams benefit from allocating enough time to learn, either from an interim retrospective or an end-of-the-project retrospective. Teams should learn about their work product and work process. For example, some teams have trouble finishing work. When teams plan enough time, they can structure their retrospective to gather data, process that data, and decide what to try later as an experiment.

First and foremost, a retrospective is not about blame; the retrospective is a time for the team to learn from previous work and make small improvements. Create a psychologically safe space where every member can speak candidly, share feedback, and do so without fear of blame or retaliation.

The retrospective is about looking at the qualitative (people's feelings) and quantitative (measurements) data, then using that data to find root causes, design countermeasures, and develop action plans. The project team may end up with many action items to remove impediments.

Consider limiting the number of action items to the team's capacity to address improvement in the upcoming iteration or work period. Trying to improve too many things at once and not finishing any of them is much worse than planning to complete fewer items and successfully completing all of them. Then, when time allows, the team can work on the next improvement opportunity on the list. When the team selects the improvements, they decide how to measure the outcomes. Then, in the next time period, measure the outcomes to validate the successes or failures of each improvement.

A facilitator from the team leads them through an activity to rank the importance of each improvement item. Once the team ranks the improvement items, they choose the appropriate number to work on for the next iteration (or add work to the flow if using flow-based agile).

GenAI tools can read notes and boards, anonymize text, suggest small experiments, and track actions over time.

5.2.8 Shipping Increments and Delivering Value

Working products, even in the smallest increments, are the primary measure of progress in agile projects. Teams plan and refine backlog items so that a slice of work can move from vision to value. Delivering the increment and gathering feedback on its real-world use is what turns outputs into outcomes.

Shipping matters for the following reasons:

- **It confirms value.** Only a released increment allows customers and stakeholders to decide whether the outcome is worth the investment.
- **It surfaces hidden work.** Packaging, deployment, documentation, compliance checks, and operational handoffs all become visible when the team ships often.
- **It reduces risk.** Frequent delivery limits the cost of getting requirements wrong and reveals integration or environment issues early.

A clear DoD gives the team a shared view of what "ready to ship" means. A DoD typically states that the increment is:

- Integrated with related work and passing relevant tests or reviews;
- Compliant with quality, safety, or regulatory standards;
- Documented and supported with any required training or instructions; and
- Accepted by the designated value representative or product owner.

Once the DoD is met, the increment can be released. Some teams deploy every accepted item; others batch several items into a timed release. The value representative or product owner decides which option balances customer benefit, risk, and release overhead.

Practical steps for frequent shipping include the following:

1. **Build in deliverability.** Integrate, test, and review work continuously so each change can flow smoothly toward release.
2. **Keep work small and independent.** Try to break backlog items into slices that provide value without relying on unseen dependencies.
3. **Use lightweight governance.** Apply concise checklists, templates, or automated scans where possible to satisfy required controls without delay.
4. **Release on a cadence or on demand.**
    - Cadence releases (for example, every 2 weeks) create a predictable rhythm for marketing, training, and support.
    - On-demand releases allow the team to push increments whenever the benefit warrants it.
5. **Collect outcome data.** After each shipment, capture usage insights, stakeholder feedback, and operational metrics. Feed these observations into backlog refinement and planning.

Team responsibilities for shipping increments should include the following:

- **Product owner or value representative.** Decide when to release, balancing customer needs and technical readiness.
- **Makers and subject matter experts.** Do the work, maintain the workflow, perform quality checks, and keep the DoD achievable.
- **Leader or facilitator.** Remove impediments to smooth flow such as access to tools or cross-team coordination.

Shipping increments is not simply another agile event; it is the purpose of the work. Regular delivery transforms backlog items into validated value and enables continuous inspection, adaptation, and informed investment decisions.

5.2.9 Execution Practices That Help Teams Deliver Value

If the team does not pay attention to quality, it will soon become impossible to release anything rapidly.

The following technical practices may help the team to deliver at their maximum speed:

- **Continuous integration.** Perform frequent incorporation of work into the whole no matter what the product is, and then retest to determine that the entire product still works as intended where applicable. Pair continuous integration with continuous delivery.
- **Testing at all levels.** Employ system-level testing for end-to-end information and unit testing for the building blocks. In between, understand if there is a need for integration testing and where. Teams find smoke testing—testing the most basic, critical features—helpful as a first look at whether the work product is any good. Teams often find that deciding when to run regression tests—and which ones—can help them maintain product quality with good build performance. Agile teams have a strong preference for automated tests so they can build and maintain a momentum of delivery.
- **Spikes (timeboxed research or experiments).** Spikes are useful for learning and may be used in circumstances such as estimation risk management, acceptance criteria definition, and understanding the flow of a user's action through the product. Spikes are helpful when the team needs to learn some critical technical or functional element.

5.3 Multiteam Coordination and Dependencies (Scaling)

Many projects incur dependencies (technical and organizational), even when they are not managed within a given program. For this reason, it is necessary to have an understanding of how agile works within an existing portfolio and program management context.

5.3.1 Frameworks

The guidance of the most widespread agile approaches, such as Scrum and Extreme Programming (XP), focus on the activities of a single, small, cross-functional team. While this is very useful for efforts that require a single team, it may provide insufficient guidance for initiatives that require the collaboration of multiple agile teams in a portfolio or program.

A range of frameworks (e.g., Scaled Agile Framework [SAFe®], Large Scale Scrum [LeSS], Nexus, and PMI® Disciplined Agile®) and approaches (e.g., Scrum of Scrums) have emerged to cater to just such circumstances. More details on these can be found in Annex A3.

5.3.2 Considerations

There is more than one way to scale work. The team might need to scale the work of several agile projects into a single agile program. Alternatively, the organization can design a structure that supports agile approaches across the entire portfolio.

For example, it is helpful to start small and learn as rapidly as possible about what works well in the organizational context. Teams can achieve successful outcomes even when everything is not completely transformed into an agile approach.

Regardless of the approach, a critical success factor is the healthy agile team. If using an agile approach for a single team is not successful, avoid scaling up toward using it more broadly; instead, address the organizational impediments that prevent teams from working in an agile way.

The goal of large-scale agile projects is to coordinate the efforts of different teams to bring value to customers. There is more than one way to do that. Teams may use a formal framework or apply agile thinking to adjust existing program management practices.

5.4 Troubleshooting Agile Challenges

Agile approaches were born out of the need to solve issues associated with high rates of change, uncertainty, and complexity. Due to these origins, agile approaches contain a variety of tools and techniques for dealing with issues that may present problems in predictive approaches. Refer to Table 5-2.

TIP

Teams should demonstrate often to solicit feedback and show progress. Encourage the project management office (PMO) and other interested parties to watch demos so the people deciding on the project portfolio can see the actual progress.

As organizations scale their adoption of agile, they may encounter other problems. Refer to Table 5-3.

Table 5-2. Agile Pain Points and Troubleshooting Possibilities

Pain Point	Troubleshooting Possibilities
Clarity and Alignment (Direction, Rules, Priorities)	
Unclear purpose or mission	Create a project brief or team charter that states the vision, mission, and simple success tests; review the vision regularly.
Unclear working agreements	Build a team charter that lists shared values, meeting etiquette, decision rules, and definitions of "ready" and "done" (DoR, DoD); post the charter where the team can see it.
Unclear team context or boundaries	Clarify product boundaries, available assets, key stakeholders, and major risks during chartering.
Unclear or changing requirements	Cocreate a product vision and lightweight roadmap using techniques like story mapping; break large ideas into small, testable backlog items.
Inefficiently ordered backlog	Prioritize by value and risk; use simple ordering rules (e.g., cost of delay) and revisit often.
Impossible or unrealistic stakeholder demands	Make capacity and trade-offs visible; offer options with impacts; negotiate scope, not dates.
Governance or compliance friction	Involve compliance early; agree on lightweight controls; automate evidence where practical.
Metrics misuse (velocity used as a threat)	Reframe metrics as learning signals; use a small set of outcomes and flow indicators; avoid target gaming.
Flow and Forecasting (Throughput, Predictability, Handoffs)	
Schedule prediction problems	Use flow metrics (throughput, cycle time) and ranges; forecast with historical data and simple models.
Unexpected or unforeseen delays	Visualize blockers; run "delay diagnostics"; limit work in process (WIP); shorten feedback loops.
Dependency overload between teams	Map dependencies; reduce coupling; create integration cadences; use interface contracts.
Work delays from poorly refined items	Strengthen refinement; add acceptance criteria; split stories; involve the right roles early.
Rush/wait flow or uneven workload	Plan within the team's capacity; set WIP limits; use pairing or swarming; reduce multitasking.
Team struggles with obstacles	Empower impediment escalation; timebox spikes; swarm on systemic blockers.

Table 5-2. *(Continued)*

Pain Point	Troubleshooting Possibilities
Too much up-front work leading to rework	Favor thin slices and early validation; prototype risky parts; defer details until the last responsible moment.
Stalled continuous improvement	Pick no more than three improvement actions per retrospective; make progress visible; review the impact next time.
False starts and wasted effort	Engage the product owner or value representative daily; test assumptions; validate big assumptions early through small releases or experiments.
People and Team Health (Structure, Safety, Sustainability)	
Siloed teams	Move work to teams, not teams to work; increase cross-functionality; establish shared goals.
Lack of psychological safety	Set team working agreements; model curiosity and blameless postmortems; make it safe to raise risks.
Unsustainable pace or burnout	Right-size commitments; enforce slack and focus time; rotate on-call/critical duties.
Quality and Value Realization (Building the Right Thing and Building It Right)	
Poor or missing customer feedback	Tighten feedback loops; schedule frequent demos with real users; use lightweight experiments.
Poor customer engagement	Identify value proxies; cocreate with customers; reduce ask size and increase frequency.
Defects and quality problems	Strengthen DoD; automate tests; integrate continuously; use root cause analysis.
Technical debt	Make debt visible; budget capacity for refactoring; set guardrails for introducing new debt.
Poor user experience	Test with users early; use simple user experience (UX) heuristics; make UX acceptance criteria explicit.
Too much product complexity	Favor simplicity; remove or defer low-value features; revisit must-haves with data.

Table 5-3. Scaling Pain Points and Troubleshooting Possibilities

Pain Point	Troubleshooting Possibilities
Fragmented product vision	Run a joint vision workshop with leaders, product owners, and team representatives; create a single vision statement and lightweight roadmap referenced by all teams; keep a shared product goal visible in quarterly or release planning; confirm every backlog item links back to the vision.
Dependency traffic jams	Map technical and timing dependencies on a coordination board and review it at least weekly; invite affected teams to short sessions to resequence work or split items; decouple architecture or workflows so teams can progress in parallel; set clear integration-ready criteria and hold frequent integration checkpoints.
Unaligned cadences	Agree on one cycle for major events such as planning, reviews, and retrospectives; if iteration lengths differ, set common calendar dates for cross-team syncs and demos; use one release or integration schedule, even when teams deliver internally at other times; publish the cadence calendar for leaders and stakeholders.
Governance overload	Bring audit, security, and compliance roles into backlog reviews early; replace large-gate reviews with small, frequent check-ins that focus on completed increments; use concise checklists and automate evidence capture where practical; keep governance artifacts lightweight and visible to all teams.
Portfolio-level staffing challenges	Create long-lived, cross-functional teams dedicated to one value stream; limit work in progress at the portfolio level so teams are not pulled onto multiple initiatives; visualize team allocations on a portfolio board and discuss trade-offs before moving people; prioritize outcomes over utilization statistics.

5.5 Measurements in Agile Projects and Products

Transitioning to agile means using different measurements. Implementing agile means looking at new metrics that matter to the team and management. These metrics matter because they focus on customer value.

One problem with status reporting is that it hinges on the team's ability to predict completion or use traffic light status to describe the project. For instance, project leaders may describe the project as "90% done." At that point in time, however, the team might attempt to integrate the various pieces into a product. The team may then discover missing requirements or surprises, or find that the product doesn't integrate the way they thought it would.

The project is really only partly done, and the traffic light status reporting does not reflect the true state. Too often, the project team realizes it will need additional or just as much time to complete the remainder of the project. For too many projects, the team realizes they are—at most—10% done because of issues the team discovered.

The problem with predictive measurements is that they often do not reflect reality. It often happens that a project status light is green up until 1 month before the release date; this is sometimes referred to as a *watermelon project* (green on the outside, red on the inside). Often, project status lights turn red with seemingly no warning because there is no empirical data about the project until 1 month before the release date.

Metrics for agile projects contain meaningful information that can provide a historical track record because agile projects deliver value (finished work) on a regular basis. Project teams can use such data for improved forecasts and decision-making.

Surrogate measurements such as percent done are less useful than empirical measurements like finished features. Agile helps teams see problems and issues so the team can diagnose and address them.

In addition to quantitative measures, the team can consider collecting qualitative measures. Some of these qualitative measures may focus on practices the team has chosen and aim to assess how well the team uses those practices, for example, the business satisfaction with delivered features, the morale of the team, and anything else the team wants to track as a qualitative measure.

5.5.1 Flow Metrics

Agile and other adaptive approaches rely on frequent inspection of empirical data to steer toward the best possible outcomes. The goal of measurement is therefore actionable insight—information that helps teams, sponsors, and stakeholders decide what to do next, not metrics used to judge or coerce the people doing the work.

Historically, many organizations defaulted to velocity (the sum of story point estimates completed within an iteration) as the primary measure of progress. Velocity was intended only as a lightweight, team-specific planning aid, but over the last decade, it has frequently been misapplied in the following ways:

- **Misuse (weaponization) of velocity.** Using a team's velocity to compare, rank, or threaten people creates fear and the loss of transparency.
- **False precision.** Treating story points as interchangeable across teams ignores differences in context, sizing scales, and definitions of done.
- **Perverse incentives.** When velocity becomes a target, teams may inflate estimates, split work unnaturally, or sacrifice quality to meet a target.

The originators of the metric now advise against making velocity a management performance indicator. Consequently, the *Agile Practice Guide*—Second Edition no longer recommends tracking story point velocity outside the team that produced it or using it to assess performance, forecast at scale, or set delivery targets.

5.5.2 Alternative Ways to Track Progress and Predictability

Flow metrics are quantitative indicators of how work items move through the system from arrival to completion. They reveal speed (time), volume (count), and WIP, helping teams spot bottlenecks and forecast delivery. See Table 5-4 for some alternative ways to track progress.

Table 5-4. Alternative Metrics for Tracking Flow

Metric	What It Shows	Why It Matters
Throughput	Count of work items (e.g., stories, features, minimum business increments [MBIs]) finished per time period	Stable throughput lets teams forecast with statistics and spot sudden drops in productivity.
Cycle time	Elapsed time from when a work item starts to when it is finished	Shorter, predictable cycle times correlate with faster feedback and earlier value delivery.
Lead time	Elapsed time from when a request is made to when it is delivered	Lead times can highlight system-wide delays (e.g., waiting for prioritization, reviews, deployments) and support service-level expectations.
Work in process (WIP)	Number of items actively being worked on at any moment	Limiting WIP exposes bottlenecks, reduces context-switching, accelerates flow, and improves throughput.
Escaped defects and rework	Issues found after release and effort spent fixing them	Identified defects can be a direct indicator of product quality and sustainability of pace.
Queue length	Number of items waiting in each workflow state (e.g., "Ready," "Design," "Quality Assurance")	Growing queues signal bottlenecks and hidden delays; trimming queues shortens lead time and improves predictability.

Plotting these metrics on a cumulative flow diagram (CFD) or simple run charts can make trends and bottlenecks obvious at a glance (see Figure 5-1).

5.5.3 Metrics That Capture Value and Outcomes

While flow metrics reveal *how* efficiently the team delivers, stakeholders ultimately care about *what* is delivered and its impact. Consider linking delivery data to one or more of the following outcome-oriented measures:

- **Customer satisfaction/Net Promoter Score℠(NPS®)[3]**—Frequent pulse surveys of end users or internal customers.
- **Business value delivered**—Features mapped to OKRs or GQMs, benefits hypothesis tracking, or revenue/profit contribution where measurable.

[3] Net Promoter®, NPS®, NPS Prism®, and the NPS-related emoticons are registered trademarks of Bain & Company, Inc., NICE Systems, Inc., and Fred Reichheld. Net Promoter Score℠ and Net Promoter System℠ are service marks of Bain & Company, Inc., NICE Systems, Inc., and Fred Reichheld.

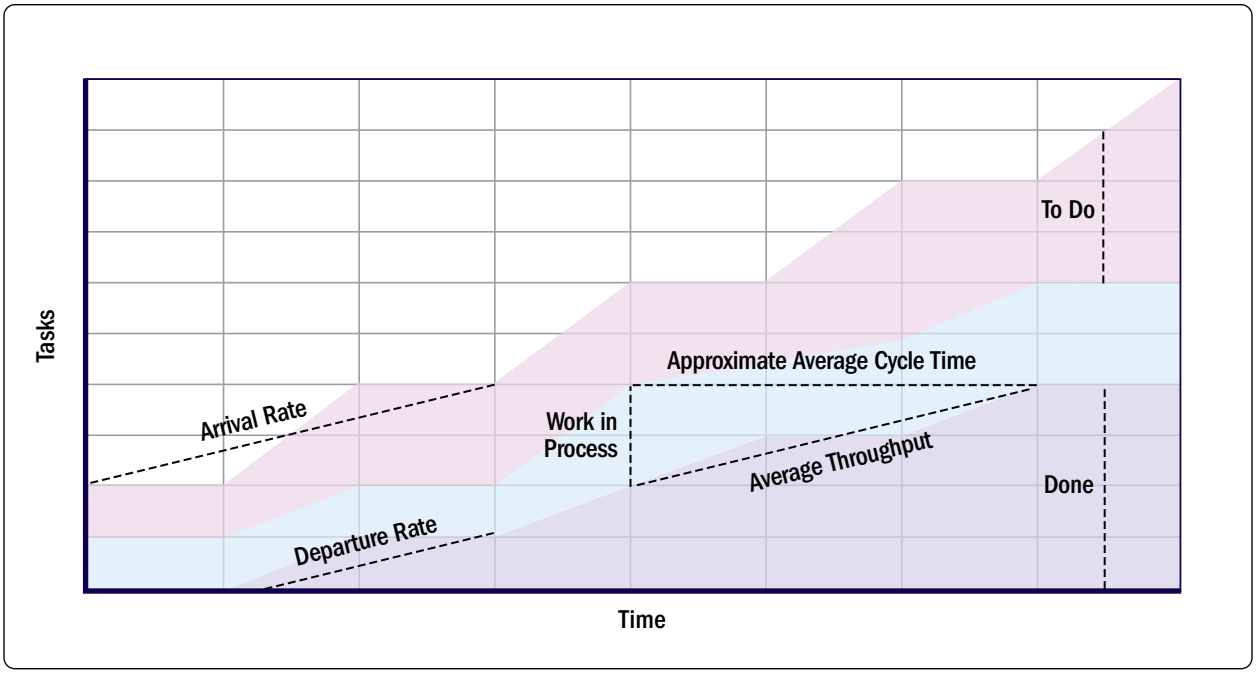


Figure 5-1. Cumulative Flow Diagram

- **Adoption and usage analytics**—Real-time telemetry showing whether released capabilities are actually used (e.g., churn reduction and NPS®[4]/CSAT gains).
- **Time-to-learn**—How quickly the team validates assumptions with real customer feedback.

5.5.4 Measurement Guidelines

To establish robust measurements, consider the following guidelines that promote effective data collection, visualization, and utilization:

1. **Start small and evolve.** Choose a minimal set of metrics that reveal constraints without increasing reporting overhead.
2. **Visualize openly.** Make data visible to everyone on information radiators or dashboards; encourage questions rather than judgments.
3. **Review and act.** Inspect metrics during retrospectives or flow-review cadences; prioritize and agree on one experiment at a time to improve the system.
4. **Decommission low-value metrics.** Stop collecting data that no longer influences decisions.

5.5.5 Reporting to Sponsors

Sponsors still need to know when significant portions of scope might be finished and how confidently the team can make that call. Combining stable throughput with an up-to-date item count allows

[4] Net Promoter®, NPS®, NPS Prism®, and the NPS-related emoticons are registered trademarks of Bain & Company, Inc., NICE Systems, Inc., and Fred Reichheld. Net Promoter Score℠ and Net Promoter System℠ are service marks of Bain & Company, Inc., NICE Systems, Inc., and Fred Reichheld.

for probabilistic forecasts (e.g., “There is an 85% chance we deliver the remaining 45 stories within 10 iterations.”). Teams should present forecasts as ranges or confidence curves—never as single-point commitments.

5.5.6 Strategic Metrics

Both GQMs and OKRs offer structured frameworks that help agile teams align daily work with strategic intent. While OKRs emphasize aspirational outcomes and team alignment, GQMs provide a disciplined approach to defining and measuring goals through targeted questions and metrics, making both approaches valuable in modern agile environments.

5.5.6.1 Goal Question Metric (GQM)

The GQM approach offers a structured way for project teams to connect the information they collect with the decisions they need to make. This can be at the project, product, or portfolio level. GQM starts with a clear goal. The team then frames questions that reveal progress toward that goal. Next, it selects metrics that answer those questions. This step-by-step link ties measurement directly to value delivery. See Table 5-5 for key GQM concepts.

GQM is iterative; goals may evolve as more is learned, prompting new questions and updated metrics.

5.5.6.1.1 Why Agile Practitioners Should Care

Agile practitioners should understand the value of GQM as it provides a structured approach for helping ensure metrics are aligned with specific objectives and ultimately drive meaningful outcomes such as the following:

- **Focus on value.** GQM discourages collecting data for its own sake. Instead, each metric traces back to a business or product goal, helping teams avoid vanity reporting.
- **Adaptability.** Because the goals are clearly stated, teams can review them whenever conditions change. This practice fits naturally with adaptive planning cycles.
- **Transparency and alignment.** Questions translate strategic intent into language the delivery team understands, improving stakeholder dialogue and clarifying expectations.
- **Risk reduction.** Early signals from well-chosen metrics alert the team to emerging issues, enabling timely inspection and adaptation.

Table 5-5. Key GQM Concepts

Element	Description	Practical Illustration
Goal	The outcome the team hopes to achieve, framed in business or product terms	Improve cycle time through the deployment pipeline to shorten feedback loops.
Question	A line of inquiry that clarifies whether progress toward the goal is occurring	How long does a typical change take from commit to production?
Metric	A specific, observable data point that answers the question	Average lead time per change in hours

5.5.6.1.2 Applying GQM in an Agile Context

To effectively apply GQM in an agile environment, follow these key steps that help guide teams through defining, measuring, and adapting their project goals and metrics:

1. **Define or refine the goal** with the product owner and key stakeholders during a planning or strategy workshop.
2. **Elicit questions** in a facilitated session with the whole team, converting the goal into concrete uncertainties the team should explore.
3. **Select lightweight metrics** that can be gathered continuously from existing tooling (e.g., value flow diagrams, defect counts, or customer satisfaction pulse surveys).
4. **Inspect and adapt** by reviewing trends at the end of each iteration or release boundary and adjusting goals, questions, or metrics as learning occurs.

USE CASE

A warehouse improvement team sets the goal of reducing picking errors by 30%. They ask the question, "What proportion of orders leave the warehouse with at least one item picked incorrectly?" and track the error-rate metric taken from daily quality-check logs. Reviewing that single number in each retrospective shows whether changes, such as clearer shelf labels or shorter pick paths, are moving the team toward the goal.

5.5.6.1.3 Common Pitfalls

When implementing GQM, teams should be aware of several common pitfalls that can undermine its effectiveness, including the following:

- **Overmeasurement.** Tracking every available number dilutes focus. Retain only metrics that directly answer agreed-upon questions.
- **Metric drift.** When questions change, refresh or retire the related metrics rather than extend them beyond their relevance window.
- **Hidden assumptions.** Surface the rationale behind each question to uncover biases and refine the measurement system.

GQM helps agile teams sharpen their measurement discipline, linking data to decisions and checking that inspection and adaptation revolve around outcomes rather than outputs. When metrics link back to clear goals and questions, teams gain actionable insight. This approach builds stakeholder confidence and speeds value delivery.

5.5.6.2 Objectives and Key Results (OKRs)

Objectives and key results (OKRs) provide a lightweight framework for aligning day-to-day work with strategic intent (Table 5-6). An objective states a qualitative, inspirational outcome; each key result defines a measurable signal that progress toward that objective is happening.

Table 5-6. Key Concepts of Objectives and Key Results (OKRs)

Element	Description	Practical Illustration
Objective	A concise, motivational statement describing a desired end state	Delight customers with a faster checkout.
Key Result	A quantitative indicator that shows whether the objective is being achieved	Reduce average cart-to-payment time from 90 seconds to 45 seconds.

Regular inspection keeps effort focused on outcomes that matter rather than task completion alone.

A small number of key results (typically three to four) keeps attention on the signals that truly matter.

USE CASE

Using OKRs

eCommerce Return Reduction Objective (by Q2 2028): Lower online shopping return rates to protect the margin and customer trust (see Table 5-7).

Table 5-7. OKR Example Table: eCommerce Return Reduction

Key Result	Baseline	Target
Overall return rate (all products)	8%	≤ 5%
Apparel return rate after new sizing guide	12%	≤ 8%
Repeat-purchase rate within 90 days	22%	≥ 28%
Customer-initiated, live-chat sessions about sizing	1,400/month	≤ 900/month

5.5.6.2.1 Why Agile Practitioners Should Care

Agile practitioners should understand the value of using OKRs because they align with agile principles, ultimately driving teams toward meaningful customer and organizational impact with the following benefits:

- **Outcome orientation.** OKRs pull teams toward customer or business impact, complementing backlog items that describe outputs.
- **Transparent alignment.** Shared objectives link team-level work to portfolio or enterprise goals, aiding coordination without heavy governance.

- **Built-in adaptation.** Quarterly (or shorter) cycles invite frequent recalibration, matching the cadence of iterative planning and review.
- **Motivation.** Aspirational objectives energize teams while clear metrics show tangible progress, supporting intrinsic motivation theories.

5.5.6.2.2 Applying OKRs in an Agile Context

To effectively integrate OKRs into agile workflows, the following key steps can help guide teams through setting, measuring, and achieving meaningful objectives:

1. **Draft objectives.** Collaborate with product owners and stakeholders early in the planning horizon, limiting each team to one or two high-value themes.
2. **Define key results.** Implement measures that are unambiguous, timeboxed, and directly within the team's influence.
3. **Integrate with iteration events.** Review key result trends in sprint reviews or system demos; adjust backlog items to close gaps.
4. **Score and reflect.** At the cycle's end, rate each key result (e.g., 0–10 scale). Celebrate learning and refine the next set of OKRs accordingly.

5.5.6.2.3 Common Pitfalls

When implementing OKRs in agile environments, teams may encounter some common pitfalls that can hinder their effectiveness, including the following:

- **Metric overload.** Too many key results can dilute focus; prioritize the handful of measures that will provide the clearest signal.
- **Vanity targets.** Selecting numbers that improve easily without real customer benefit undermines credibility. For example, metrics such as "training hours completed" that measure seat time, not skill use, would be better replaced by "behavior change observed" or "error reduction after training." Additionally, "social likes on release posts," which is a popularity proxy and easy to spike, would be better switched to metrics such as "active users retained," "conversion," or "churn reduction."
- **Set-and-forget metrics.** Neglecting midcycle reviews can turn OKRs into static documentation rather than a dynamic guide.
- **Misaligned horizons.** Objectives that span multiple years may conflict with agile planning rhythms; break them into shorter steps.

Using OKRs can help agile teams bridge the gap between strategic aspirations and everyday delivery. By articulating compelling objectives and pairing them with concrete key results, teams gain a clear line of sight to the value they aim to create, enabling transparent alignment, rapid feedback, and purposeful adaptation.

GenAI Insights

GenAI has the potential to streamline many aspects of agile work, from team startup to scaled delivery. In chartering projects and teams, it can synthesize diverse inputs into a draft vision, roles, and working agreements, reducing startup time and helping ensure all perspectives are captured. For delivery, GenAI can recommend WIP limits and service policies on kanban boards, fitting models to throughput and cycle time to stabilize flow and reduce delays. GenAI can also help prioritize backlogs, providing insights on business value and ways to speed up delivery. At scale, it can scan dependencies across multiple backlogs, surface risks early, suggest sequencing options, and analyze product metrics to highlight bottlenecks or trends. Even after allowing for the time of having a human in the loop to check suggestions and add local expertise, these AI uses save time, reduce bias, and provide sharper visibility into work, creating more space for teams to focus on problem-solving and value delivery.

The same capabilities can also create harm if applied without care. GenAI-generated charters may shortcut the difficult but essential conversations that build trust, alignment, and ownership, leaving teams less committed to outcomes. Automated WIP or measurement recommendations risk becoming rigid rules rather than insights for discussion, undermining adaptability, conversation, and collaboration. At the scaling level, AI-driven dependency or performance data may be misinterpreted or overweighted, shifting accountability from people to algorithms. If teams rely too heavily on these outputs, they may stifle dialogue, collaboration, and ownership. The idea is to use GenAI as a supportive tool that increases awareness and eases administrative tasks, not as a substitute for the human judgment, negotiation, and shared accountability that make agile ways of working successful.

Section 6

Organizational Considerations for Agility

Most projects exist in an organizational context. Cultures, structures, and policies can influence both the direction and outcomes of any project or product. These dynamics can challenge project leaders.

While project leaders may not have the ability to change organizational dynamics as they see fit, they are expected to navigate those dynamics skillfully.

This section explores the way the organization—and in some circumstances, the project context—can influence projects. Leaders should explore options for change to increase the chance of project success.

Agility is more effective and sustained as the organization adjusts to support it. True enterprise agility requires a systemic approach that spans strategic planning, governance, culture, leadership, and operational infrastructure.

To enable enterprise-level agility, the organization should evolve its mindset, structures, and supporting functions in parallel with delivery practices. This enables all functions to operate as an integrated network, capable of coordinated adaptation in response to shifting conditions.

6.1 Organizational Change Management

Organizational change management is the holistic, structured approach for transitioning individuals and organizations to achieve intended business benefits that support agility.

The PMI publication, *Managing Change in Organizations: A Practice Guide* [6], describes a comprehensive and holistic approach for successfully introducing meaningful change. The recommendations offered in the practice guide include the following:

- Models for describing change dynamics,
- A framework for achieving change, and
- Application of change management practices at the portfolio, program, and project levels.

Sections 6.1.1 and 6.1.2 explore the considerations of change management specific to an agile context. Figure 6-1 shows the relationship between these two topics.

6.1.1 Drivers for Change Management

All projects create change. In agile environments, the pace of delivery and the way teams work can increase that change. Two internal drivers are common:

- **Accelerated delivery.** Frequent increments can outpace the receiving organization's readiness. Without preparation, adoption lags and value is delayed.
- **Shifts in ways of working.** Greater collaboration, more handoffs, and iterative prototypes can feel disruptive and invite resistance.

Change management should also address strategic drivers that originate outside the team:

- **Market and competitive pressure**—New offerings, pricing shifts, or customer expectations.
- **Compliance and policy mandates**—Regulatory deadlines and audit requirements.

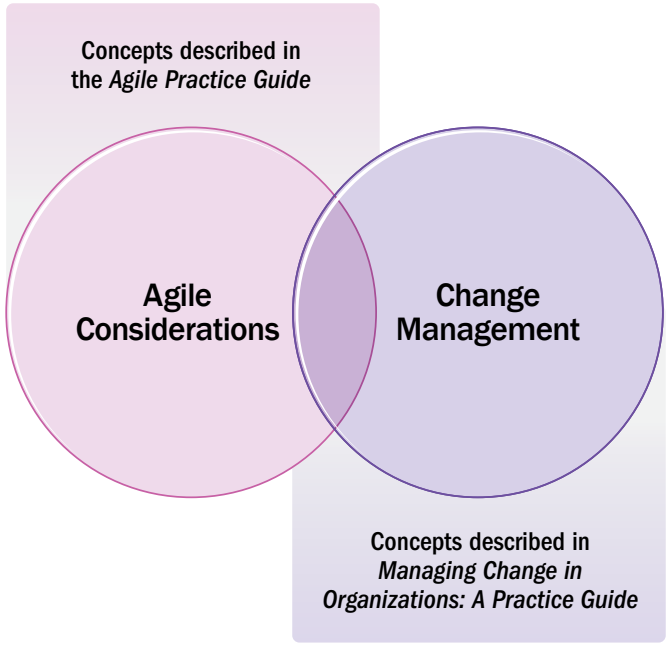


Figure 6-1. The Relationship Between Change Management and Agile Approaches

- **Technology and operating model shifts**—Platform changes, automation, and reorganizations.
- **Closing value gaps**—Strategic goals that current processes cannot meet.

Position change management as a value enabler, not just a friction reducer. Positioning change management connects delivery to results by driving:

- **Adoption.** People use the new process or product as intended with the skills and support they need.
- **Outcomes.** Desired effects are achieved such as faster cycle times, higher safety, or better customer experience.
- **Benefits realization.** Outcomes persist and translate into measurable benefits like revenue uplift, cost reduction, churn reduction, or compliance confidence.

This framing aligns with agile's concepts of "focus on value," "enable change," and "systems thinking." Organizations should start change management with strategic planning, continue it through delivery, and sustain it into operations. Treat it as a core, enterprise-level capability that makes frequent delivery meaningful and turns outputs into durable results.

6.1.2 Organizational Agility

Organizations beginning to use agile approaches should understand the relative compatibility of those methods with their current approaches. Some organizations will have characteristics that more easily support agile principles of cross-department collaboration, continuous learning, and evolving internal processes. Examples of these change-friendly characteristics include the following:

- Executive management's willingness to change;
- The organization's willingness to shift the way it views, reviews, and assesses employees;
- Centralization or decentralization of portfolio, program, project, or product management functions;
- A focus on short-term budgeting and metrics versus long-term goals; and
- Talent management maturity and capabilities.

Conversely, there are other institutional characteristics that may be roadblocks to achieving the changes associated with organizational agility. Examples of these include the following:

- Work is decomposed into departmental silos, creating dependencies that prevent accelerated delivery instead of building cross-functional teams with guidance from centers of competencies.
- Procurement strategies are based on short-term pricing strategies rather than long-term competencies.
- Leaders are rewarded for local efficiencies rather than end-to-end flow of project delivery or optimizing the whole (in regard to the organization).
- Employees are specialized contributors with limited tools or incentives to diversify their skills instead of building generalizing specialists.
- Decentralized portfolios pull employees simultaneously onto too many projects at once instead of keeping them focused on one project at a time.

The degree to which an organization is willing to review and modify these practices will determine how quickly and effectively agile approaches can be adopted. However, in response to these organizational impediments to agility, leaders can try various approaches to accelerate a cultural compatibility for the following:

- Visible and active executive sponsorship;
- Change management practices, including communication and coaching;
- Progressively pacing the adoption of agile practices;
- Incremental introduction of agile practices to the team; and
- Leading by example by using agile techniques and practices where possible.

6.1.3 Agile Change Management

When addressing an individual challenge area or implementing a new hybrid or agile approach, it is recommended to undertake the work incrementally. A common practice is to treat the change process as an agile project with its own backlog of changes that could be introduced and prioritized by the team, based on perceived value or other considerations. Each of the changes can be treated as an experiment, which is tested for a short period of time to determine suitability as is, or the need for further refinement/consideration.

Use kanban boards to track progress, showing the new approaches already in use as “done,” those being tried as “in progress,” and those still waiting to be introduced as “to do.” See Figure 6-2 for the initial board with a ranked backlog. Figure 6-3 shows an example of what a board might display as work progresses.

Using these tools to organize and manage the change implementation provides visibility into progress and also models the approaches being implemented. Organizational change can also use common Kanban metrics to help monitor the rate of change such as change lead time, adoption rate, throughput, and so forth. Rolling out changes in a transparent and appealing way improves the likelihood of their success.

6.2 Organizational Culture

An organization’s culture is its DNA—its core identity. Culture will always influence the use of agile approaches. Organizational culture runs along a continuum, from highly predictive plans to Lean Startup approaches, where everything is an experiment. Although agile approaches fit well with the Lean Startup culture, a highly predictive organization can encourage empirical measurements, small experiments, and learning so they can move toward agility.

6.2.1 Assessing Culture

Every project must navigate a complex landscape of competing demands and priorities. How can the team go fast without compromising quality? How can the team preserve flexibility while also hitting a firm date? Most importantly, how does the team satisfy and meet the requirements of the customer?

Project leaders may feel their job is to meet every expectation of every stakeholder, but when compelled to make a choice, there is often a priority depending on the culture and requirements of

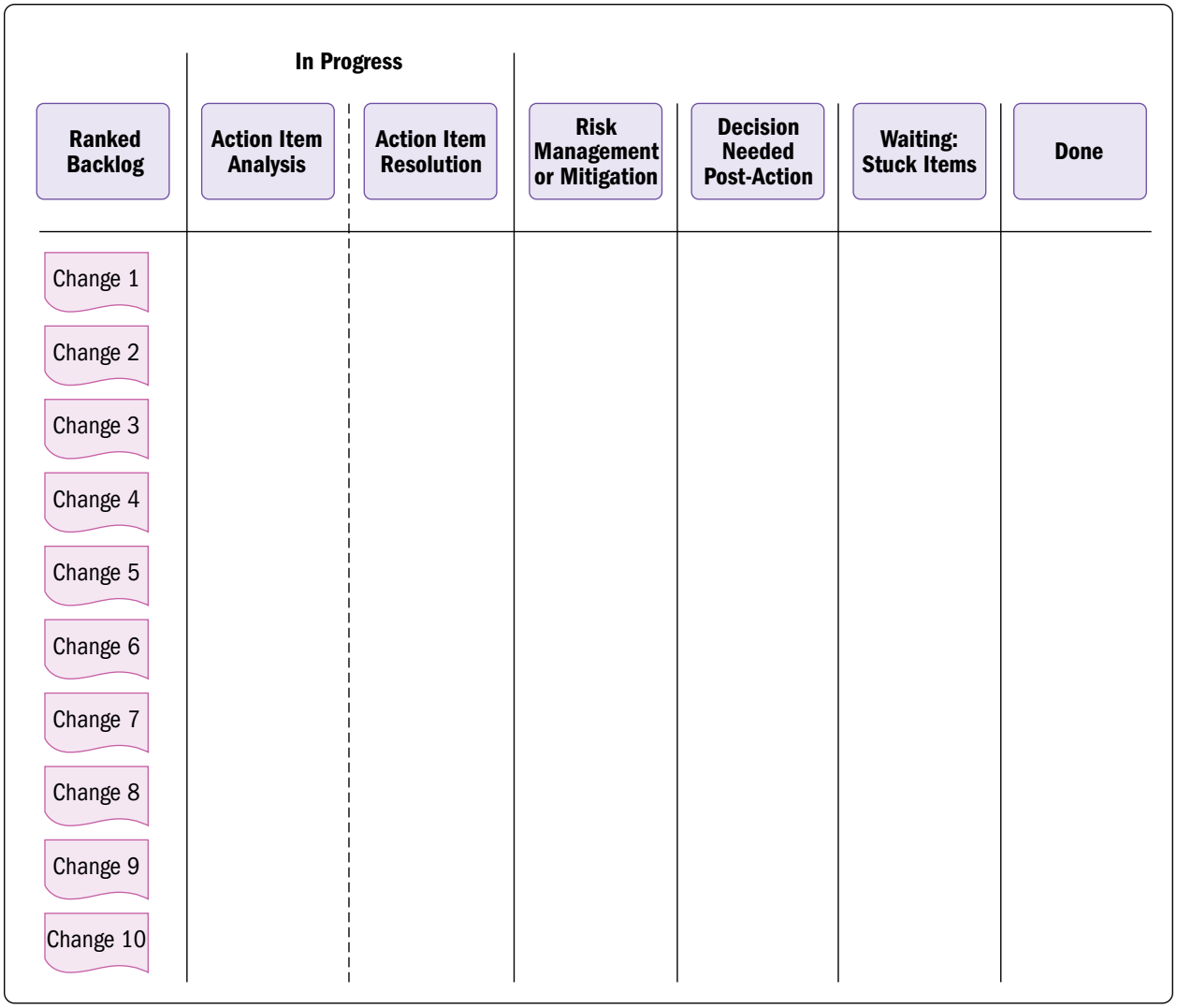


Figure 6-2. Initial Ranked Backlog for Changes

the organization's business environment. For example, a digital banking project has a greater bias for speed, whereas a government program may have a greater bias for generalization and stability.

This statement stresses the importance of people's commitment and passion for a cause. No matter what strategy or plan you implement with your team, its success is going to be governed by the people implementing the plan. If the people who are driving the strategy are not passionate about the change, or worse, are apathetic about their job and their organization, then you stand little chance of implementing the change.

To navigate these dynamics, project leaders should take the time to assess where emphasis is most often applied in the organization. Figure 6-4 illustrates what an assessment might look like. In this example, a project leader initiates a conversation about organizational priorities with stakeholders, team members, and senior management. Those priorities are then recorded as positions on a sliding scale between two extremes. The results are then used to find agile techniques that best fit with those priorities.

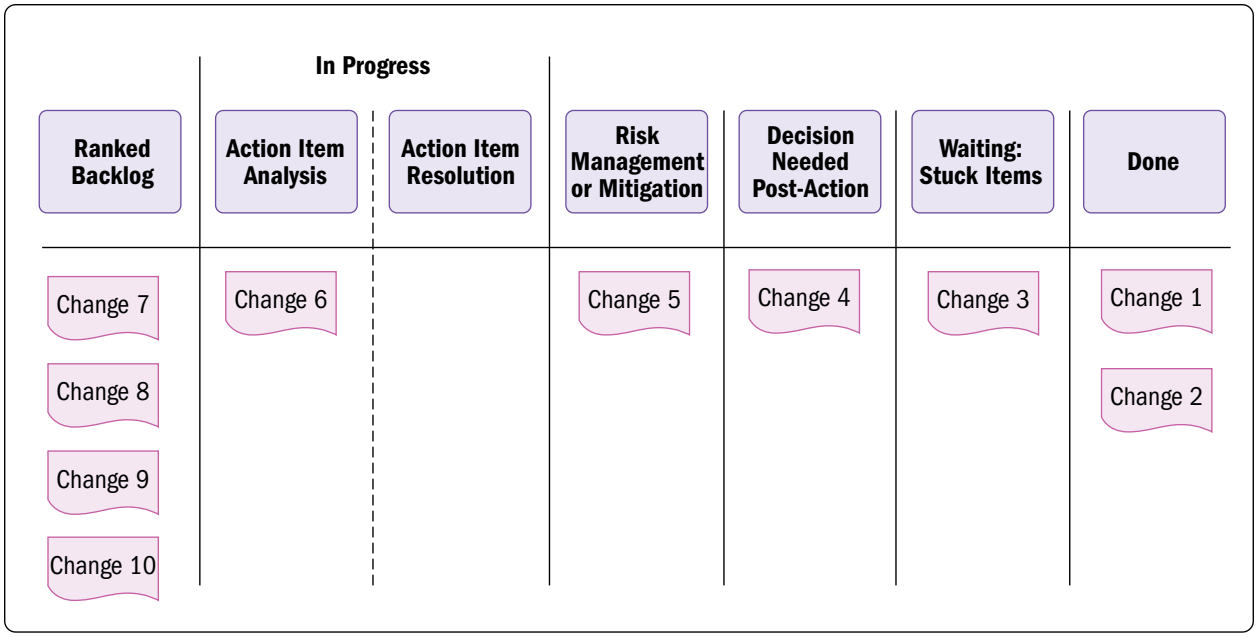


Figure 6-3. Using Backlogs and Kanban Boards to Organize and Track Change Work

Several models exist for assessing such dynamics; however, the model or method used is not that important. It is more critical that project leaders invest the effort to understand the forces that shape their context. Understanding the organization and the industry requirements that an organization needs to satisfy allows for choosing the right conversations, the right trade-offs, and, especially, the right techniques.

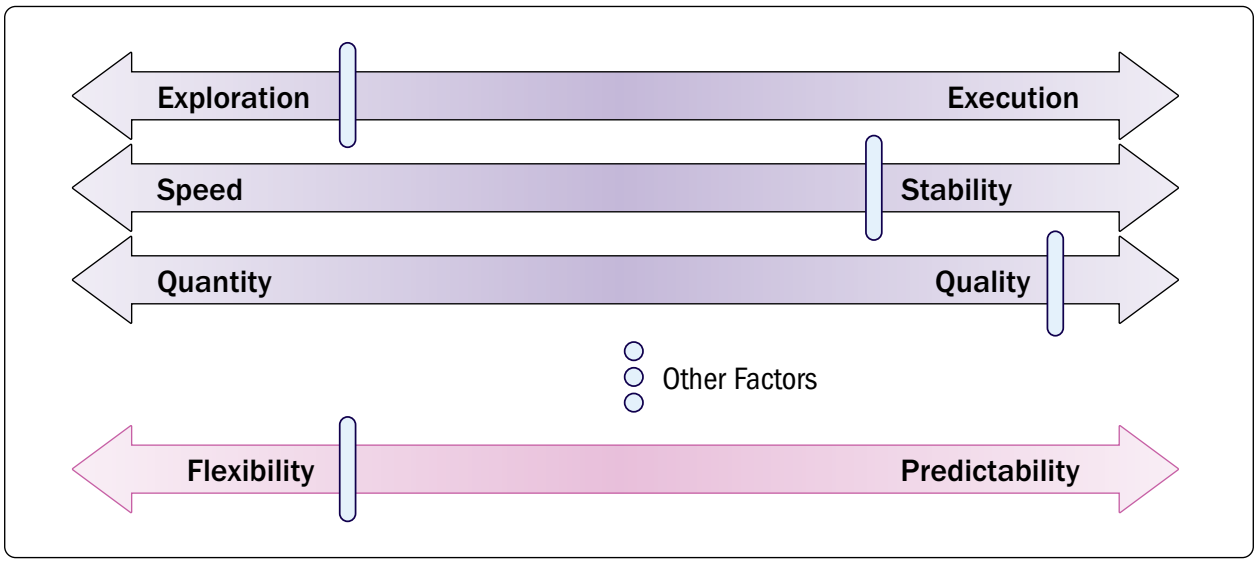


Figure 6-4. Example of Assessing Organizational Culture

6.3 Team Structures

Team structures refer to how teams are organized and how they interact to support collaboration, delivery, and performance. Team structure directly influences how work flows, how decisions are made, and how effectively teams can deliver value. Agile teams benefit from structures that promote cross-functional collaboration, reduce friction, and support alignment with customer outcomes.

Agile principles emphasize the importance of team-level empowerment, shared ownership, and sustainable delivery. Examples include the following:

- Daily collaboration between business and technical team members;
- Motivated individuals supported with the tools and autonomy they need;
- Self-organizing teams that define how to accomplish their goals;
- Frequent reflection to adapt team practices; and
- Small team sizes, typically five to nine members, with the skills needed to deliver value without external dependencies.

6.3.1 Common Team Structures

Organizations use a variety of models to organize teams (see Figure 6-5). Each has benefits and trade-offs depending on size, complexity, and goals.

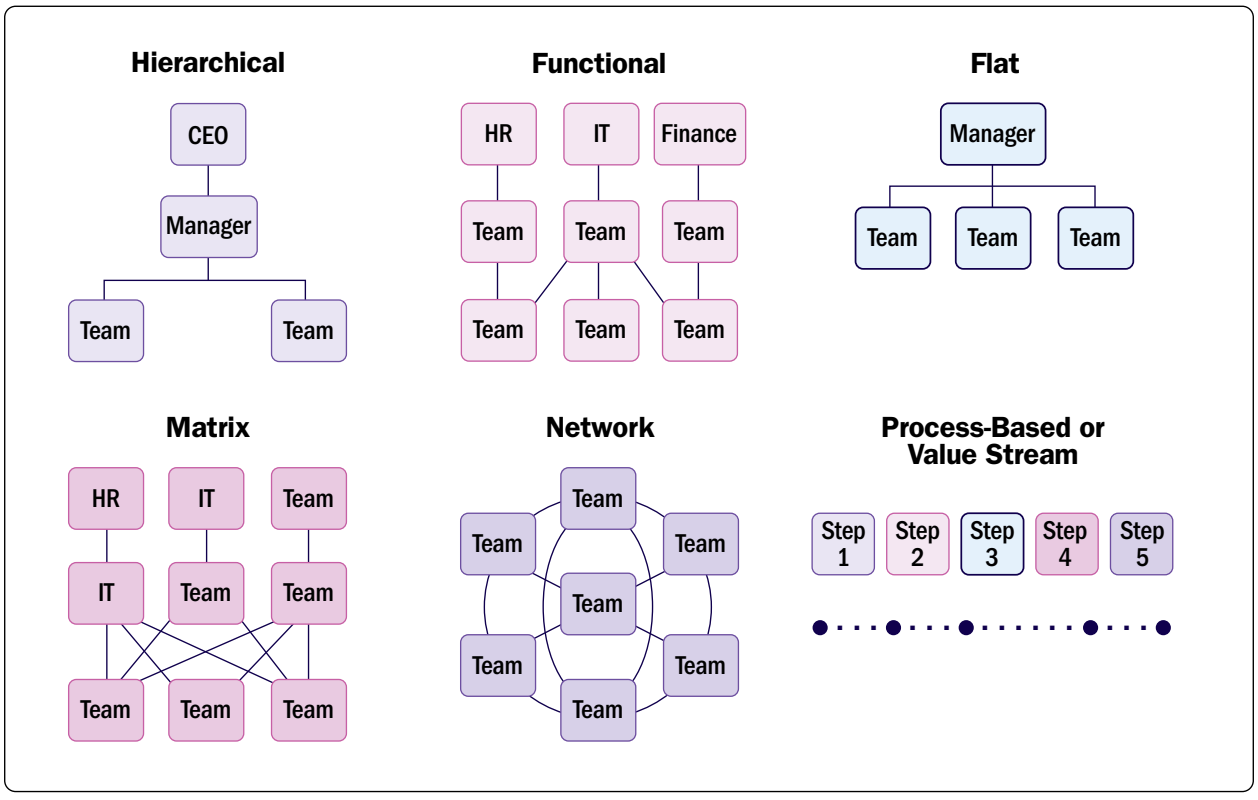


Figure 6-5. Common Team Structures

Teams are typically organized into one of the following structures:

- **Hierarchical.** Clear authority flows from the top. Useful for repeatable processes, though often slower to adapt.
- **Functional.** Teams are grouped by expertise (e.g., IT, finance). This structure encourages specialization but may create silos within a department or organization.
- **Flat.** Fewer layers of management exist in this structure. This arrangement increases speed and ownership, though role clarity may suffer as scale increases.
- **Matrix.** Team members report to both functional and project leaders. This arrangement promotes collaboration but decision-making may become unclear.
- **Network.** This arrangement uses a more fluid structure based on relationships. This structure enables innovation, though consistency can be difficult to maintain.
- **Process-based or value stream.** Teams are organized around delivering end-to-end customer value. This structure reduces handoffs and increases ownership.

Many agile organizations are shifting toward value stream aligned structures. These arrangements enable teams to manage work from intake to delivery with fewer dependencies. For example, in a healthcare setting, the journey from patient admission through discharge can be managed by cross-functional teams aligned to that workflow.

USE CASE

Airport Ground Operations Realignment

One international airport restructured its ground operations to reduce delays and improve the passenger experience. Instead of organizing staff by department—baggage, fueling, boarding—teams were formed around flights. Each team included representatives from all service areas involved in a flight. This arrangement allowed the group to coordinate in real time, respond quickly to changes, and reduce delays. The structure emphasized shared responsibility for outcomes.

6.3.2 Factors That Influence Team Structure

Selecting an appropriate team topology involves understanding the nature of the work and the surrounding environment. Factors to consider include the following:

- **Type of work.** Stable and repetitive work may benefit from functional structures, whereas dynamic environments call for cross-functional teams.
- **Team size.** Smaller teams may work best in flatter structures.
- **Specialization.** If deep expertise is required, functional or specialist teams may be necessary.
- **Geographic distribution.** Network and matrix models often support distributed teams.
- **Decision-making speed.** Flatter models may reduce delays.
- **Customer alignment.** Value-stream structures promote proximity to customer outcomes.

- **Coordination complexity.** Team-of-teams or matrix models may help manage interdependencies in large programs.
- **Organizational and cultural maturity.** Select structures the culture can sustain; assess norms, psychological safety, leadership behaviors, and readiness for empowerment; then pace adoption accordingly.

USE CASE

Global Software Platform Integration

A global technology company initially organized its engineering teams by geography and business unit. Each group developed software aligned with regional needs, which led to fragmentation. When the company began integrating its systems into a unified global platform, it faced significant architectural inconsistencies. To address this, the company reorganized its teams into cross-functional, globally aligned groups based on product lines. This supported more cohesive delivery and a scalable architecture.

Agile organizations may also use the following hybrid team compositions:

- **Generalist teams**—Broadly skilled teams that adapt quickly.
- **Specialist teams**—Deeply focused teams that support complex or regulated domains.
- **Hybrid teams**—Groups that combine flexibility with expertise across disciplines.
- **Tiger teams or SWAT teams**—Short-term, cross-functional groups created to tackle high-priority issues or opportunities.

Well-structured teams support agility by reducing handoffs, shortening feedback loops, and maintaining alignment with organizational goals. As organizational systems become more complex, the way teams are structured increasingly influences the success of the solutions they create.

Different team structures, such as dedicated teams, cross-functional squads, tiger teams, or SWAT teams, serve distinct purposes and are best suited for specific organizational contexts and objectives. Each team type carries potential trade-offs and should therefore be applied with deliberate consideration of its suitability and associated risks.

USE CASE

Volvo Cars—Agile Reorganization in Automotive Research and Development

Volvo Cars carried out a large-scale agile transformation several years ago when it reorganized its R&D department. In this transformation, engineers and project managers were reshaped from traditional departments into cross-functional agile teams. These teams operated within a team-of-teams model, collaborating toward shared objectives rather than working in silos. Although management structures shifted,

Continued on page 94

most teams remained intact, preserving earlier gains in team maturity. As a result, higher-maturity teams maintained their performance even through change, while less mature teams experienced the most disruption. The reorganization demonstrated that resilience in agile team structures supports stability even amid organizational change.

6.3.3 Reteaming

Reteaming is a team-formation concept that challenges the traditional agile principle of keeping teams stable over time to build predictability and performance. While long-standing teams can develop strong cohesion and delivery rhythm, research suggests that teams that remain unchanged for too long may become stagnant, limiting learning and innovation.

Heidi Helfand, author of *Dynamic Reteaming* [14], argues that team changes are not only inevitable due to hiring, attrition, or shifting business needs but can also be beneficial. Her framework outlines five common patterns of reteaming:

- **One by one.** A single team member is added or removed.
- **Grow and split.** A team grows too large and divides into two or more smaller teams.
- **Merging.** Two or more teams are combined into a single unit.
- **Switching.** Team members rotate between teams, exchanging roles and perspectives.
- **Isolation.** A smaller team is formed from members extracted from one or more existing teams.
- **Change frequency and clarity.** Frequent reteaming or structure shifts without clear purpose and guardrails can erode trust, predictability, and cadence; make changes deliberately and review impacts.

Even small changes, such as adding or removing a single person, can significantly influence a team's dynamic. A new team member may introduce fresh ideas, different work practices, and cultural shifts. Conversely, removing a team member may impact knowledge continuity and morale.

Reteaming may be a useful approach when teams are no longer learning, delivering effectively, or maintaining a healthy dynamic. It can help reset collaboration, introduce diverse viewpoints, and create new energy.

However, reteaming should be approached thoughtfully. Teams benefit when changes are planned, communicated clearly, and followed with additional time for adjustment. Leaders should actively observe and support the team after changes are made, helping members reestablish working agreements, clarify roles, and reconnect to shared goals as they move through the Tuckman ladder team stages.

6.4 Business Practices to Enable Agility

The willingness and ability to create new competencies within an organization when the need arises is a mark of organizational agility. These changes do not have to be significant and could be less

disruptive in an organization that is focused on agility and the results it provides. Transparency and open collaboration are absolutely key.

As cross-functional teams deliver value, the teams and individuals might encounter problems with various support functions in the organization.

As a team delivers value on a regular basis, the finance department may have the opportunity to capitalize on the product differently. If the team has contracts with other organizations, the procurement department may need to change those contracts to help the other organizations deliver value frequently and synchronize with the team.

Once teams start to work in a cohesive and cooperative manner, they will challenge internal management policies. Human resources may notice that individual incentives make less sense, and managers may struggle with the performance appraisals of self-organizing employees. In each case, these are opportunities to review the degree to which existing practices support agile ways of working.

Some of the typically impacted functions and how they can adopt agile practices are shown in Table 6-1.

As organizations progress to greater agility, there will be obvious needs for additional business units to change the way they interact and perform their responsibilities. The changes that have benefited other areas of the organization should then be embraced so the effectiveness of the entire organization can be realized.

6.5 Lean Portfolio Management

Lean portfolio management is the application of agile and Lean principles to how an organization manages its portfolio of work. A Lean portfolio includes initiatives that align with strategic goals and focus on delivering value to customers. This approach helps avoid investing in work that no longer meets business needs.

Enterprise agility is the strategic capability to dynamically prioritize, reprioritize, or stop initiatives based on changing business conditions.

Table 6-1. Functions and Agile Practices

Function	Agile Practices
Human resources	Agile recruitment, continuous learning, flexible work arrangements, and iterative performance management
Compliance	Adaptive processes, risk-based monitoring, real-time reporting, and cross-functional collaboration
Finance	Iterative budgeting and forecasting, value-driven decisions, and cost transparency
Governance	Agile frameworks, collaborative decision-making, stakeholder engagement, and dynamic models

The goal is to align strategy, execution, and benefits realization. This goal is reached through frequent evaluation, prioritization, and adjustment of initiatives. Lean portfolio management supports flexible funding and iterative planning so organizations can shift funding, people, and resources to higher-priority work as needed.

Lean portfolio management consists of three main dimensions:

- **Strategy and investment funding.** Align the portfolio with strategic goals and allocate funding to the highest value work.
- **Agile portfolio operations.** Enable teams to deliver iteratively and respond to change quickly.
- **Lean governance.** Provide financial and operational oversight through audit-friendly processes and forecasting.

Unlike traditional portfolio management, Lean portfolio management focuses on frequent delivery and value review. Initiatives are assessed regularly to determine if they are still aligned with customer needs and strategic priorities. If not, the work is paused or stopped. Funding, people, and resources are then redirected to higher-impact efforts. High-performing lean portfolio management groups will limit the portfolio WIP by capping the number of epics actively being worked on. This cap helps focus capacity efforts and shorten the time to value delivery.

Lean portfolio management builds on core Lean principles:

- **Identify value.** Understand what products or services deliver value to customers.
- **Map the value stream.** Visualize the workflow to identify delays or inefficiencies.
- **Create flow.** Improve the flow of work by removing waste and handoffs.
- **Establish pull.** Let work move through the system based on demand.
- **Continue to improve.** Review and refine processes regularly.

Traditional portfolio models often rely on annual planning cycles. Once approved, teams work on initiatives for extended periods, even if the value decreases. This model can result in wasted funding, people, and resources as well as missed opportunities. In contrast, lean portfolio management enables continuous planning, shorter feedback loops, and faster decision-making.

6.5.1 Benefits of Lean Portfolio Management

Lean portfolio management offers numerous benefits that help organizations optimize their project portfolios and achieve strategic objectives, including the following:

- **Maximized value.** Teams align work to customer needs and business goals.
- **Faster time to market.** Quick delivery enables faster feedback and learning.
- **Improved value delivery.** Focusing on high-impact work increases customer benefits.
- **Greater efficiency.** Funding, people, and resources are able to shift as priorities change, reducing waste and delays.

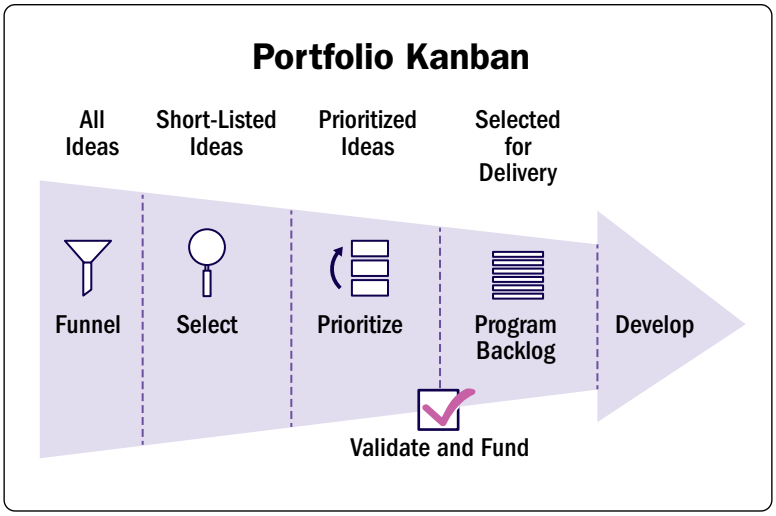


Figure 6-6. Portfolio Kanban

6.5.2 Prioritization in Lean Portfolio Management

Prioritization in lean portfolio management helps determine which initiatives to pursue. Teams assess work based on constraints, capabilities, potential impacts, and alignment with strategic goals. This process is continuous, allowing organizations to respond to change. See Figure 6-6 for a common tool used for prioritization in lean portfolio management: Portfolio Kanban.

TIP

A common tool used in lean portfolio management is the Kanban Method funnel. This funnel captures ideas and potential initiatives before detailed planning begins. The funnel acts as a backlog of possibilities, including market shifts, new technologies, or customer needs. At this stage, there are no limits on how many ideas can be submitted. Once in the funnel, ideas move through review and refinement. Those ideas with the highest strategic alignment or potential impact advance for further analysis and development. Ideally, objective measures are put in place for prioritization to eliminate subjective decision-making—which may have been used in the past in the form of authority, influence, or possibly even the personal preferences of the most senior leader.

Key aspects of prioritization include:

- **Clear criteria.** Use consistent factors like ROI, strategic alignment, or market demand.
- **Prioritization matrix.** Compare initiatives based on impact and effort.
- **Focus on value.** Choose work that delivers strong outcomes with minimal funding, people, and resources.
- **Ongoing review.** Prioritization is continuous. Teams adjust based on new insights.

TIP

Using Impact Versus Effort for Decision-Making

Teams often assess initiatives using a simple matrix that compares impact to effort. This matrix helps teams focus on work that offers the greatest return for the least investment.

Impact is the potential benefit or value of a project. It can be measured through outcomes like customer satisfaction, revenue growth, or strategic advantage.

Effort includes the time, funding, people, resources, or complexity involved in delivering the work. This assessment may involve development time, team size, or process challenges.

By comparing both factors, teams can identify high-impact, low-effort work that drives strong results. This approach improves decision-making and helps teams spend time where it matters most.

Participatory budgeting is a funding approach that helps organizations reallocate resources more quickly. This approach brings together business leaders and technical teams to decide how to distribute capacity across value streams. These decisions are reviewed and adjusted quarterly, making it easier to shift priorities when needed. Lean portfolio management supports agility at scale, enabling better alignment, faster learning, and improved delivery of value across the organization.

6.5.3 Portfolio Sync and Planning Cadence

Lean portfolio management relies on regular, cadence-based events to maintain alignment and support timely decision-making. One common approach is to hold a type of portfolio sync or alignment session, typically every 6 to 12 weeks. This session brings together key stakeholders to review progress, assess priorities, and reallocate funding, people, and resources as needed. This approach also provides a forum to raise risks, dependencies, or shifts in strategic direction.

Larger planning sessions, such as quarterly portfolio planning sessions, allow for broader reflection and adjustments across the portfolio. These events often include updates on strategic goals, retrospective findings, and forecasts for the next cycle of initiatives.

Cadence events create consistent feedback loops and make it easier to respond to change, validate outcomes, and align efforts across teams and business units.

USE CASE

National Financial Services Firm

A financial services firm adopted lean portfolio principles to improve alignment between its technology and business portfolios. The firm introduced portfolio sync meetings every 6 weeks and used an impact-effort matrix to prioritize initiatives. Work that no longer aligned with evolving regulations or customer expectations was paused or stopped. The firm saw greater transparency, faster decision-making, and improved risk management.

6.5.4 Feedback Loops and Portfolio Metrics

To continuously improve portfolio performance, organizations benefit from establishing feedback loops and tracking KPIs. These feedback mechanisms help teams and leaders evaluate if current initiatives are delivering value and meeting strategic goals.

Metrics might include the following:

- **Delivery lead time**—How quickly work moves from idea to deployment.
- **Return on investment (ROI)**—A measure of the value generated relative to cost.
- **Team throughput**—The number of completed items in a given period.
- **Customer satisfaction**—Feedback through surveys, Net Promoter Score℠ (NPS®)[5], or usage patterns.

Regular reviews of these metrics can help organizations adapt quickly, validate if the portfolio is delivering on outcomes, and guide decisions about where to shift funding, people, and resources.

USE CASE

Global Consumer Goods Company

A global consumer goods organization applied lean portfolio management to accelerate product innovation. Instead of approving projects annually, the company shifted to rolling wave planning and continuous funding cycles. Product teams were given more autonomy and reviewed quarterly by a cross-functional leadership team. As a result, time to market improved by 30%, and fewer people were tied up in low-impact initiatives.

6.6 Agile and the Project Management Office (PMO)

In most organizations, the project management office (PMO) exists to shepherd business value throughout the organization. The PMO might do this by helping projects achieve their goals. Sometimes, the PMO educates teams (or arranges for training) and supports projects. The PMO may also advise management about the relative business value for a given project or set of projects.

Because agile creates cultural change over a period of time, the organization might also need to change, including the PMO. For example, managers make decisions about which projects to fund and when, and teams decide what they need for training or advice.

6.6.1 An Agile PMO Is Value-Driven

Any project should deliver the right value to the right audience at the right time. The PMO's objective is to facilitate and enable this goal. An agile-based PMO approach is based on a

[5] Net Promoter®, NPS®, NPS Prism®, and the NPS-related emoticons are registered trademarks of Bain & Company, Inc., NICE Systems, Inc., and Fred Reichheld. Net Promoter Score℠ and Net Promoter System℠ are service marks of Bain & Company, Inc., NICE Systems, Inc., and Fred Reichheld.

customer-collaboration mindset and is present in all PMO programs. In many cases, this means the PMO operates as if it were a consulting business, tailoring its efforts to meet the specific needs of a given project. Some projects may need tools and templates, whereas others may benefit from executive coaching. The PMO should strive to deliver what is needed and keep the pulse on its customers to help ensure it knows and is able to adapt to their needs. This intrapreneur approach focuses on the PMO activities that are perceived as most valuable to the projects it supports.

6.6.2 An Agile PMO Is Invitation-Oriented

To accelerate progress on a value-based charter, a PMO may be tempted to mandate certain solutions or approaches (e.g., making everyone do it the same way to get some quick wins). However, a more deliberate perspective incorporates the desire for employee engagement. This engagement is achieved by inviting only those interested in collaborating with PMO services. Higher engagement with PMO practices makes it easier for those practices to remain. If the PMO is delivering value to its clients, it is more likely that clients will request its services and adopt its practices.

6.7 Agile and xMOs

The organizational unit once called a PMO now appears in many guises: value management office (VMO), agile transformation office (ATO), strategy realization office (SRO), portfolio enablement office (PEO), and others. In this practice guide, we use the neutral label "xMO" to cover them all. An xMO helps the enterprise deliver the right outcomes at the right pace while guarding flow, risk, and strategic alignment.

TIP

Use the following question prompts to test whether the xMO is delivering real, visible value:

1. Which portfolio metric triggered the last genuine debate?
2. Where does work wait the longest between teams or functions?
3. How often did environmental, social, and governance (ESG); regulatory; or adoption insight change the backlog priority this quarter?

Section 6.7.1 explains how an xMO can support enterprise agility through adaptive planning, dynamic funding, and lean governance across the portfolio.

6.7.1 Outcome- and Flow-Focused

Traditional PMOs track time and cost. An xMO focuses on outcomes and the smooth flow of value across value streams. The xMO replaces stage gate checkpoints with continuous sensing and feedback. Table 6-2 explains the shift in metrics tracking.

Table 6-2. A Shift Toward Outcome-Driven Metrics

Practice Shift	Legacy Focus	Modern Focus
Success lens	On time, on budget	Outcome OKRs; Net Promoter Score℠ (NPS®)[6]; environmental, social, and governance (ESG) KPIs
Funding model	Annual capital parcels	Incremental, outcome-based tranches
Control points	Earned value (EV) reports	Flow metrics (lead time, throughput, cost of delay)
Review cadence	Quarterly portfolio boards	Monthly value stream reviews
Organizational structure	Functional project teams	Stable value stream teams funded for the life cycle

6.7.1.1 Metrics Spotlight

Pick a small set of leading and lagging signals such as lead time, flow efficiency, cost of delay, outcome OKRs, and carbon per feature. Display them side-by-side on a program kanban board or value stream management (VSM) dashboard so leaders can see trade-offs at a glance. Update the set during each review cycle and drop any metric that no longer drives decision-making.

6.7.2 Enablement, Transformation, and Community

Early agile advocates asked PMOs to "invite, not mandate." Remote and hybrid work now make that stance essential. The xMO serves as an enabler, community hub, and agile transformation office, when required.

The xMO can do the following:

- Curate lightweight playbooks, templates, and learning paths;
- Facilitate communities of practice that meet on-site and online;
- Host lean-portfolio-canvas and participatory budgeting workshops;
- Guide transformation waves;
- Pilot new ways of working;
- Measure adoption;
- Measure impact to the organization;

[6] Net Promoter®, NPS®, NPS Prism®, and the NPS-related emoticons are registered trademarks of Bain & Company, Inc., NICE Systems, Inc., and Fred Reichheld. Net Promoter Score℠ and Net Promoter System℠ are service marks of Bain & Company, Inc., NICE Systems, Inc., and Fred Reichheld.

- Offer agile portfolio management;
- Promote agile approaches to project management;
- Keep a visible change backlog; and
- Provide stewardship to functional leaders once practices are embedded.

The xMO avoids:

- Owning day-to-day delivery,
- Imposing one "true" framework, and
- Measuring teams on process adherence instead of customer impact.

TIP

Distributed teams working remotely need digital radiators, asynchronous coaching hours, and virtual "serendipity" forums that replicate corridor conversations.

6.7.3 Multidisciplinary and Data-Informed

Complex portfolios need more than schedulers and cost controllers. Modern xMOs blend expertise in analytics, change, sustainability, and technology. See Table 6-3 for xMO emerging roles and contributions.

New digital tooling trends apply VSM platforms that can combine code, ticket, and deployment events and data to visualize flow. AI layers add forecasts for metrics such as schedule slippage, escaped defects, and carbon hotspots. Start with an opt-in pilot and an ethics checklist.

Table 6-3. xMO Emerging Roles and Contributions

Emerging Role	Contribution
Value stream analyst	Maps flow, spots bottlenecks, suggests systemic fixes
Data scientist/AI coach	Builds predictive forecasts and early warning risk models
ESG advisor	Aligns portfolio scoring with environmental, social, and governance objectives
Transformation lead	Orchestrates change roadmaps, manages learning backlog
Data product owner	Maintains clean metric pipelines and privacy compliance

6.7.4 Governance and Reporting Patterns

Lean portfolio management frameworks replace heavyweight stage gates with light, frequent alignment ceremonies such as the following:

- **Quarterly portfolio syncs** confirm strategy, adjust guardrails, and reset weightings on OKRs.
- **Monthly value stream reviews** inspect flow data, check progress on outcome KPIs, and remove systemic blockers.
- **On-demand guardrail checks** trigger when spending, risk, or ESG variance exceeds a predefined threshold.
- **Transformation pulse checks** track adoption health, capability uplift, and cultural signals, sharing findings in the same cadence review.
- **Backlog rankings** use weighted shortest job first (WSJF) to prioritize minimum marketable products (MMPs) so feedback arrives sooner.

Include ESG, adoption, and flow metrics on one dashboard so leaders can see trade-offs clearly.

6.7.5 Dynamic Funding

Dynamic funding treats money as a flow to outcomes. Value streams receive a baseline to keep teams stable. Additional tranches are released against clear hypotheses, leading indicators, and outcome targets. When marginal benefit drops, funding is reduced or redirected. Reviews happen on a fixed cadence, typically quarterly. Use simple guardrails: capped tranche size, explicit exit criteria, and a small set of KPIs tied to customer or mission results.

Keep governance lightweight and auditable. Record assumptions, decisions, and evidence for each tranche. Track outcome progress, learning gained, and unit economics, not just spending. Use staged paths such as discover, pilot, scale, and run to guide risk and investment. Create triggers to pause or pivot when signals degrade such as flat KPIs, rising risk, or low adoption. This approach supports transparency, regulatory needs, and timely reallocation while keeping the focus on value.

6.7.6 Roles at a Glance

A few well-defined roles keep strategy, flow, and change connected across the portfolio. See Table 6-4.

6.7.7 Common Antipatterns With Corrective Actions

Watch for the following pitfalls that can drain time, data, and goodwill from an xMO:

- **Vanity dashboards.** Charts abound but no decision changes are made. Fix: Tie each chart to a specific decision, owner, review cadence, and threshold. Add a short decision log that records what changed after each review.
- **Shadow governance.** Teams report twice—once to the xMO and again to legacy gates. Fix: Map all reporting asks, remove duplicates, and publish a single evidence list. Make the xMO the first point of call for data. Pull information from source systems to avoid duplication.
- **Framework monoculture.** One-size prescription ignores local context.
- **Data overload.** Metrics are collected without action.

Table 6-4. Key xMO Roles

Who	Key Contribution
xMO lead	Aligns portfolio guardrails with strategy and culture
Transformation lead (agile transformation office)	Coaches executives, shapes the change backlog, and transfers ownership to line teams
Epic/initiative owner/product manager	Defines outcome KPIs and stewards benefits realization
Team coach	Embeds flow practices, mentors metrics, and champions behavior
Finance partner	Links rolling forecasts to incremental funding dates

TIP

Tips for Getting Started

The following steps provide a practical path to launch—or relaunch—an xMO:

1. Identify a priority customer journey and map its value stream.
2. Establish a cross-functional team for the value stream and fund it for one quarter.
3. Create a backlog of change enablers and timebox the first wave to 90 days.
4. Visualize the flow and outcome OKRs on a portfolio kanban board.
5. Apply WSJF prioritization, deliver the first MMP, and inspect results.
6. Expand to the next value stream when adoption health scores reach 80% or higher.

6.8 Maintaining Agility Across the Organization

As organizations adopt agile ways of working, they often struggle to keep practices consistent at scale. Enterprise agility rests on structural enablers. These enablers make rapid, effective adaptation possible across many teams. Cross-functional teams, roles, and units improve collaboration and knowledge flow across boundaries.

Modular design and aligned decision authority support quick responses without losing strategic coherence. Scalability depends on decentralized governance and well-integrated shared services. A synchronized planning cadence keeps diverse initiatives moving together. Reengineered pathways for strategy, decisions, and learning connect the enterprise.

These pathways allow governance, coordination, and decision-making to work across boundaries. As a result, agility becomes an enterprise operating principle, not just a team capability.

Communities and groups sustain this architecture. Centers of excellence (CoEs), communities of practice (CoPs), and special interest groups (SIGs) help teams share methods, compare results, and build skills. Chapters, guilds, and networks can play the same role with different levels of formality. These communities connect pilots to the wider organization, reduce drift, and maintain momentum after initial success. Together, structural enablers and communities create the conditions for adaptation, learning, and reliable delivery at scale.

6.8.1 Centers of Excellence

A center of excellence (CoE) is the most formal structure organizations adopt to promote agile standards and delivery excellence. A CoE is usually a group of individuals appointed by the organization to support and evolve a specific domain of knowledge—in this case, agile. Some CoEs operate as stand-alone departments with dedicated leadership and full-time staff members. Others function as virtual or matrixed teams composed of agile coaches, scrum masters, product managers, or tooling specialists embedded across the organization. Agile transformation offices (ATOs), value management offices (VMOs), or other xMO structures are often variations of agile or delivery CoEs. These groups share similar functions but go by different names based on an organizational or leadership preference.

The CoE's responsibilities often include developing and maintaining agile role definitions, setting and refining delivery standards, supporting enterprise tooling, providing training and onboarding, and offering coaching or advisory services. As agile matures across an organization, CoEs also facilitate feedback loops between delivery teams and leadership, promoting continuous improvement of agile practices.

USE CASE

A global financial services company established an agile CoE as part of its digital transformation. The CoE developed playbooks, defined agile team structures, and provided coaching support across multiple business units. The CoE did not act as an enforcer of rules but rather served as a central service group to support delivery teams, maintain alignment, and scale agile in a consistent yet adaptable way.

While CoEs bring valuable structure, organizations should be careful that these groups do not become disconnected from teams or act as audit bodies. The risk of becoming an "ivory tower" increases when CoEs focus too heavily on governance without supporting on-the-ground delivery. To prevent CoEs from becoming compliance bodies, organizations should make sure that they operate as enablers, not enforcers. Recommended practices include rotating active practitioners through these groups, limiting tenure, and using lightweight governance such as quarterly "inspect and adapt" sessions to align learning with delivery outcomes. A CoE's effectiveness depends on its abilities to listen, adapt, and serve as a partner to agile teams.

Here are some metrics that may be beneficial to measure a CoE's effectiveness:

- **Agile adoption rate**—Percentage of teams or business units actively practicing agile at or above the target maturity level.
- **Business value delivered**—Total measurable business outcomes (revenue, cost savings, OKR progress) enabled by CoE-supported initiatives.
- **Cycle-time improvement**—Reduction in average time from idea to delivery across agile teams.
- **Agile capability growth**—Improvement in team or individual agile competency scores from assessments or surveys.
- **Stakeholder satisfaction**—Average satisfaction or NPS®[7] score from teams and leaders engaging with the CoE.

6.8.2 Communities of Practice

A community of practice (CoP) is typically a semiformal group of individuals who share a passion or expertise in a particular discipline. CoPs may form around agile roles, such as scrum masters or product owners; specific practices, such as facilitation or story writing; or broader areas, such as business analysis or design thinking. Participation in a CoP is usually voluntary and self-selected, which contributes to their energy and effectiveness.

CoPs operate as peer-learning spaces, where participants exchange ideas, explore techniques, and collaborate on ways to improve their craft. While some organizations formally support CoPs by allocating time or budget, many CoPs arise organically based on interest and commitment. The CoP members may meet regularly to discuss challenges, test new tools, or cocreate job aids and frameworks.

USE CASE

Agile business analysis CoPs have become increasingly common. While many agile frameworks do not explicitly include the business analyst role, the underlying need to understand and define business problems remains critical. These CoPs help analysts adapt their skills to agile environments, explore ways to collaborate with agile teams, and spread good practices across the organization.

CoPs can be highly effective when tied to organizational learning goals, but they may lose momentum if participants lack time or if their work is not visibly adopted. Because they do not typically have decision-making authority, CoPs rely on influence rather than enforcement.

[7] Net Promoter®, NPS®, NPS Prism®, and the NPS-related emoticons are registered trademarks of Bain & Company, Inc., NICE Systems, Inc., and Fred Reichheld. Net Promoter Score℠ and Net Promoter System℠ are service marks of Bain & Company, Inc., NICE Systems, Inc., and Fred Reichheld.

6.8.3 Special Interest Groups

A special interest group (SIG) is a loosely structured community formed around a specific topic, trend, or challenge. SIGs are often the most informal of the organizational communities and typically emerge when individuals want to explore a particular subject more deeply, advocate for a change, or connect with others who share common interests. Participation is open, and the focus is often on learning, awareness building, and influence rather than delivery or standardization.

While SIGs are not typically established to set enterprise policy, they can raise important questions or propose changes that shape culture and practice over time. For example, an SIG might explore new technical practices, raise awareness about diversity in agile roles, or research emerging trends like agile hardware development. One widely recognized example is the Women in Agile organization, which began as a grassroots initiative and has grown into a global movement supporting the advancement and visibility of women in agile communities. Many organizations support employee participation in this external group to promote leadership development and inclusion.

Some SIGs form entirely within an organization and focus on business or delivery outcomes.

USE CASE

An enterprise human resources (HR) department created an SIG focused on agile HR and people experience. The group was formed to explore how agile could improve employee onboarding, internal mobility, and leadership development. Although not part of a formal HR transformation initiative, the SIG influenced how agile practices were used to improve internal employee services and HR delivery.

Because SIGs are usually not tied to specific delivery objectives or formal structures, they rely on member engagement to remain active. Their influence is often cultural or advisory. While they may not drive enterprise-level decisions directly, they help create a more connected, thoughtful, and inclusive organization by providing space for exploration, shared interest, and advocacy.

GenAI Insights

AI can support agility across the organization when used thoughtfully with human checks and balances. In change management, AI can analyze surveys, communications, and adoption metrics to spot resistance and suggest ways to address it. For culture, AI can scan feedback and collaboration data to highlight trends and track progress on initiatives. In shaping team topology, the tool can identify skill gaps, overlaps, and workload imbalances to recommend stronger structures. Business practices

Continued on page 108

also benefit when AI reviews processes to spot inefficiencies, suggest simpler approaches, and automate routine tasks like reporting or compliance. In lean portfolio management, predictive analytics, prioritization models, workflow optimization, and real-time dashboards can improve planning and visibility. In the PMO, AI can reduce manual reporting by pulling data from team tools, creating dashboards, and showing how work connects to strategic goals, helping shift the PMO from compliance to value delivery.

These same uses can cause harm if applied poorly. Sentiment tracking or feedback analysis may feel like surveillance, eroding trust. Dashboards and automated recommendations can oversimplify cultural dynamics, reduce people to data points, or turn human judgment into rigid rules. In portfolio management, overautomation can produce plans that follow data but miss new trends and opportunities. In the PMO, constant monitoring can slip into command-and-control behavior, using metrics to police rather than support. At the enterprise level, excessive reliance on automated insights can overwhelm leaders or centralize decision-making, weakening cross-functional collaboration, accountability, and ownership.

AI works best as a firewalled input step, providing relevant data for human decisions, not a replacement for established practices. Its real value is in reducing noise, surfacing patterns, and freeing teams to focus on high-value work. When outputs are accepted without question, agility becomes mechanical. By keeping people at the center and using AI to inform rather than dictate, organizations can capture its benefits while preserving judgment, adaptability, and trust across the organization.

Section 7

Ongoing Evolution of Agile

Since the first *Agile Practice Guide* was published, the world of work has transformed in ways few could have predicted. The COVID-19 pandemic and advancements in technologies such as AI have accelerated the adoption of remote work models, removing team member location as a barrier to collaboration. Teams across industries have embraced virtual tools and practices, enabling them to connect, deliver, and thrive regardless of geography.

The joined forces of Agile Alliance and PMI represent a powerful opportunity to continue uniting communities, extending reach, sharing knowledge, and advancing the practice of agility on a global scale. Together, these organizations can support practitioners in evolving their skills, exploring new approaches, and pushing boundaries.

Agile practices—once considered for only software development team use—have now become mainstream. Agile ways of working are the norm across a wide range of industries, functions, and organization sizes. This shift reflects a broader understanding that adaptability, transparency, and value delivery are essential for long-term success in a rapidly changing world.

The rapid rise of AI is now reshaping how teams work, make decisions, and deliver value. AI tools are automating routine tasks, providing deeper data insights, and allowing for faster, more informed decision-making. AI is unlocking new possibilities while also challenging us to address ethical considerations, transparency, and human-to-AI partnerships.

Global connectivity is also expanding opportunities for collaboration and inclusion, allowing diverse teams from around the world to contribute unique perspectives. Organizations are increasingly focused on sustainability, resilience, and social responsibility, areas where agile practices can help adapt strategies and deliver measurable impact.

PMI and Agile Alliance invite the community to share experiences, insights, and lessons learned, including how AI, remote work, and other innovations are shaping organizations and agile practices across industries, functions, and organizations. Feedback and participation in the broader agile community will help shape future editions, helping ensure the *Agile Practice Guide* continues to evolve alongside the people and organizations it serves. Look for representatives from both PMI and Agile Alliance at conferences and meetings, and engage them in discussion. Please feel welcome to provide feedback and engage in conversation regarding the content of this practice guide at standards.department@pmi.org.

Annex A1
PMBOK® Guide—Eighth Edition Mapping

This annex realigns the *Agile Practice Guide*—Second Edition with the structure of *A Guide to the Project Management Body of Knowledge (PMBOK® Guide)*—Eighth Edition [2]. Where the first edition of the *Agile Practice Guide* mapped to the Process Groups and Knowledge Areas of the sixth edition of the *PMBOK® Guide*, this second edition now maps to the *PMBOK® Guide*—Eighth Edition [2], which is framed around six principles, seven project management performance domains, and five Project Management Focus Areas.

Section A1.1 provides a crosswalk to assist readers in locating related guidance between the two publications and, where appropriate, highlights agile and hybrid considerations.

A1.1 Narrative Guidance by Project Management Performance Domain

Additional insight into how agile principles can be applied across various aspects of each of the project management performance domains is as follows:

- **Governance.** Agile governance relies on visible work and short feedback loops. Rather than gate reviews, product demonstrations and metrics dashboards enable evidence-based steering. When formal governance is mandated, teams can timebox increment reviews to satisfy oversight without stalling flow.
- **Scope (including quality)**. Evolving requirements are welcomed; backlog refinement and DoR checkpoints maintain clarity. Quality is embedded via acceptance-test-driven development, continuous integration, and retrospective-informed improvements.
- **Schedule.** Forecasting moves from Gantt charts to probabilistic delivery ranges. Cumulative flow diagrams and throughput trends reveal capacity; defined WIP limits help smooth variability.
- **Finance.** Adaptive projects align funding to validated learning. Guardrail metrics (e.g., cost per story point delivered, cumulative value score) support incremental investment decisions without heavy, up-front estimates.
- **Stakeholders.** Direct, frequent interaction (e.g., sprint reviews, story mapping) replaces layers of proxy reporting. Transparency encourages early value realization and trust.

- **Resources.** Teams are stable and multidisciplinary. Skill matrices inform upskilling; blockers to flow are removed collaboratively rather than escalated.
- **Risk.** Risk reviews are embedded in iteration planning, reviews, and retrospectives; mitigation actions are added to the backlog. Real-time dashboards track leading indicators such as increased work-item aging, higher percentages of blocked work, and growth in the technical debt backlog.

Annex A2
Software-Centric Approaches and Metrics for Agile Teams

Many agile principles are broadly applicable across industries, but some practices are specific to software delivery. This annex introduces several approaches and concepts relevant to project teams working in software-intensive environments. Annex A2 also extends Section 3, on life cycle selection, by explaining how cloud-native and microservices contexts can lead teams to adopt continuous delivery life cycles. This annex also details the supporting tool chains and flow-based metrics used to monitor performance and reliability.

The following technical practices, many of which come from Extreme Programming (XP), may help the team to deliver at their maximum speed:

- **Acceptance-test-driven development (ATDD).** In ATDD, the entire team meets and discusses the acceptance criteria for a work product. The team then creates the tests, which allows the team to write just enough code and automated tests to meet the criteria. For nonsoftware projects, consider how to test the work as the team completes chunks of value.
- **Test-driven development (TDD) and behavior-driven development (BDD).** Writing automated tests before writing or creating the product actually helps people design and mistake-proof the product. For nonsoftware projects, consider how to test the team's designs. Hardware and mechanical projects often use simulations for interim tests of their designs.

A2.1 DevOps: Bridging Development and Operations

DevOps is a cultural and technical movement that emphasizes collaboration between software development and IT operations. DevOps aims to shorten the time between writing code and delivering it to customers while improving quality and reliability.

Key DevOps principles include the following:

- Automating the software delivery pipeline,
- Continuously integrating and deploying code (CI/CD),
- Monitoring systems and responding quickly to incidents, and
- Collaborating across traditionally siloed teams.

DevOps is often supported by a delivery pipeline, which is a set of automated steps that move a change from development to production. These pipelines typically include the following:

- **Version control** to manage changes to source code and configuration;
- **Automated build and test processes** to help ensure code quality and detect regressions early;
- **Package management systems** to store compiled code and dependencies;
- **Configuration and infrastructure automation**, sometimes referred to as infrastructure as code, to consistently set up test and production environments;
- **Automated deployment mechanisms** that deliver changes to users with minimal manual intervention; and
- **Monitoring and observability tools** that provide real-time feedback on system health and usage.

The major components of DevOps include automating the software delivery process, using continuous integration and continuous deployment (CI/CD), monitoring systems and increased collaboration, and promoting collaboration across development and operations, as depicted in Figure A2-1.

Each of these capabilities contributes to shortening feedback loops and improving the overall stability and responsiveness of software systems. DevOps practices can be built into an agile development approach by visually tracking specific steps in DevOps processes with a kanban board or commonly used visual-tracking tools.

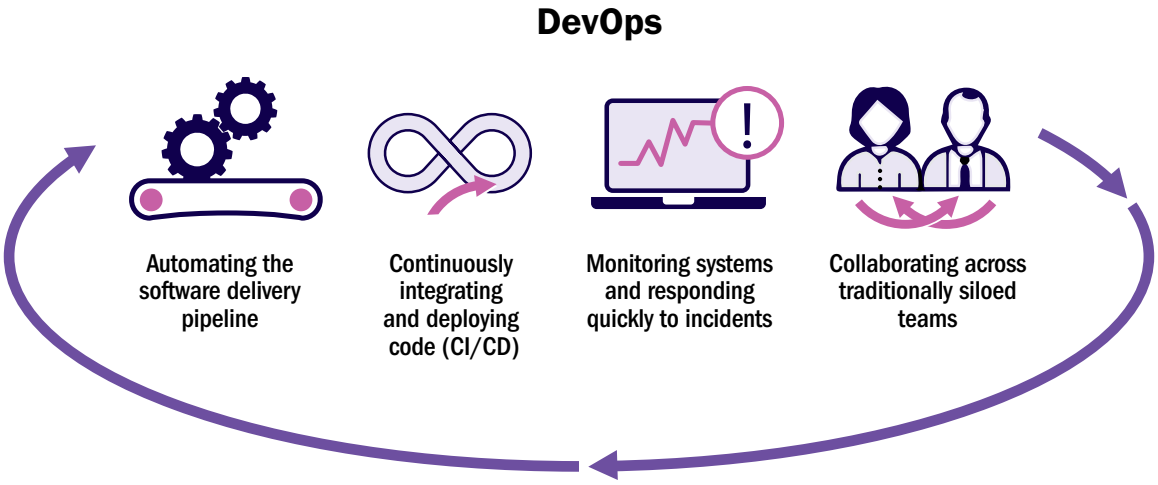


Figure A2-1. Main Components of DevOps

USE CASE

DevOps in a Government Services Project

A digital services team within a government agency was tasked with modernizing the public permit application system. Historically, new features have been released quarterly. Planning, coordination, and manual testing have frequently consumed significant time and effort, and production issues often further delayed releases. These challenges led to low levels of satisfaction among both internal stakeholders and the public.

To improve delivery, the team adopted agile practices supported by a DevOps approach. They introduced version control for all code changes, automated their testing and integration steps, and standardized the setup of test and staging environments using configuration scripts. An internal platform team created self-service deployment capabilities, allowing teams to deliver updates to the testing and production environments with minimal coordination.

As a result, the team moved from quarterly releases to releasing updates multiple times per week. Changes are smaller, easier to test, and less risky to deploy. Stakeholders are able to see progress in near-real time, and feedback cycles are shortened from weeks to days. Over time, the rate of production issues has dropped, and confidence in the system has improved—both within the organization and among end users.

This case illustrates how DevOps practices can enhance agility by improving flow, reducing manual effort, and increasing confidence in delivering working solutions.

A2.2 DevSecOps: Integrating Security Into Agile Delivery

DevSecOps extends DevOps by embedding security into every stage of the software development and delivery pipeline. The goal is to build secure systems as they are developed, rather than treating security as a separate phase or external responsibility.

Traditional models often delay security reviews until just before release. In contrast, DevSecOps integrates security thinking and tooling from the earliest stages of design and coding. This integration improves both security posture and delivery speed—two goals that are often perceived to be in conflict.

DevSecOps supports agility by enabling the following:

- Faster detection and remediation of vulnerabilities,
- Shift-left testing (early and frequent security scans),
- Collaborative threat modeling and secure design, and
- Increased confidence in the quality and safety of each small release.

DevSecOps practices are enabled by a range of tools and capabilities integrated into delivery pipelines. While the specific tools vary by organization, their functions are consistent and typically include the following:

- **Static application security testing (SAST)**—Analyzes source code or binaries for common coding vulnerabilities before deployment.
- **Dynamic application security testing (DAST)**—Scans a running application to identify security issues in real time such as input validation or session-handling weaknesses.
- **Software composition analysis (SCA)**—Identifies known vulnerabilities in third-party libraries or open-source components.
- **Secrets detection**—Helps ensure credentials and application programming interface (API) keys are not accidentally stored in source control.
- **Infrastructure and configuration scanning**—Validates that cloud or on-premises infrastructure is securely configured (e.g., firewall rules, access controls).
- **Container security**—Scans containers for vulnerabilities and policy violations.

In a DevSecOps pipeline, these scans are run automatically and continuously, often as part of the same integration and deployment processes that run tests and build the software. This effort reduces the cost and delay of fixing issues found late in the development process.

DevSecOps can be considered as DevOps wrapped in a suite of tools and techniques to plan for and implement comprehensive security measures, as depicted in Figure A2-2.

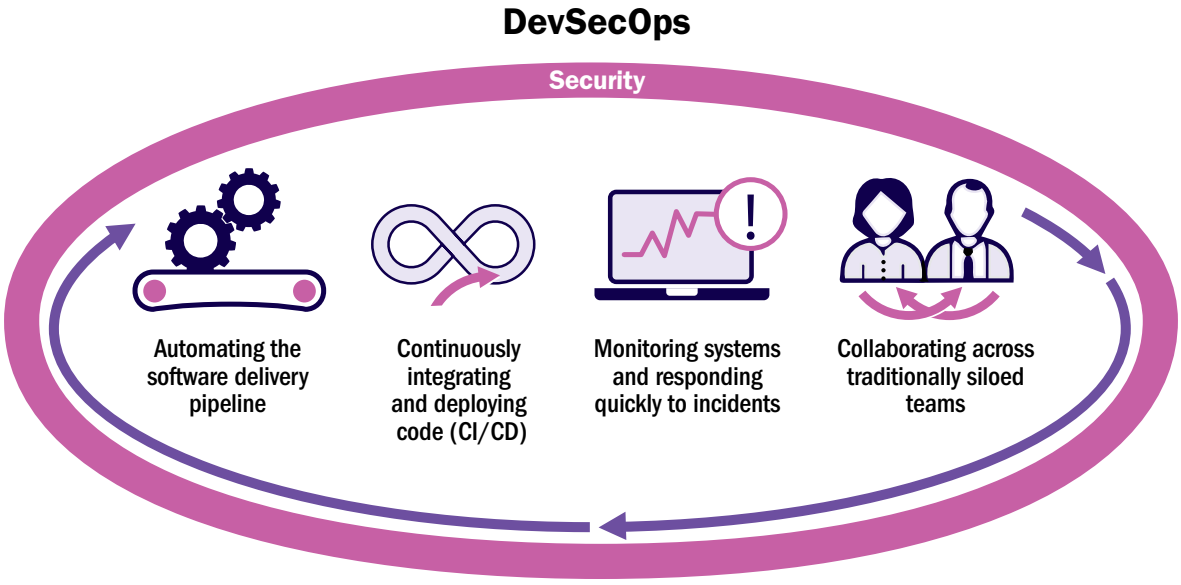


Figure A2-2. DevSecOps as a Comprehensive Security Wrapper to DevOps

USE CASE

Securing an Agile eGovernment Portal

A national revenue agency embarked on a multiyear modernization effort to replace an outdated tax portal with a mobile-responsive, user-friendly system. The initial project relied on manual security reviews late in the release cycle, which caused delays when issues were found just before launch.

After a security audit identified systemic risks—including open-source libraries with known vulnerabilities and inconsistent server configurations—the team decided to adopt DevSecOps practices.

They began by doing the following:

- Adding automated code scans for common security issues to every pull request,
- Integrating open-source dependency scanners into the build pipeline, and
- Establishing a security "champion" on each team to facilitate awareness and communication among developers and security professionals.

Security checks were implemented in parallel with functional testing, and infrastructure-as-code templates were enhanced to enforce secure configurations. Most importantly, the security team partnered with developers during design and refinement, not just during release approval.

As a result, the team identified and resolved issues earlier in the development cycle. Releases became more frequent and reliable, and leadership gained greater confidence in the overall system posture without slowing down delivery.

This experience demonstrates that integrating security into everyday development work increases both speed and safety. This integration is an essential outcome for modern agile teams working in regulated or sensitive environments.

A2.3 Low-Code/No-Code Platforms: Accelerating Delivery With Visual Tools

Low-code/no-code (LCNC) platforms are visual development environments that enable users to build software applications with minimal or no hand-coding. They are designed to increase the speed of development and reduce reliance on specialized programming skills.

While these platforms vary in complexity and capability, most include drag-and-drop interfaces, prebuilt components, and automated integration with databases or external services. Some platforms are designed for use by business users ("citizen developers"), whereas others are geared toward professional developers seeking to accelerate routine tasks.

Low-code platforms typically provide the following:

- Visual modeling tools for defining data models, workflows, and interfaces;
- Reusable templates or components that speed up form creation and logic;
- Built-in integration connectors for common enterprise tools and APIs;
- One-click deployment to test production environments; and
- Role-based access controls for managing permissions and governance.

No-code platforms go a step further by abstracting logic into rule-builders or natural language interfaces, allowing people with no programming background to configure applications.

Using LCNC platforms can support agile principles through the following:

- **Shortening feedback loops.** Working software can be shown to stakeholders within days or even hours.
- **Encouraging iteration.** Components can be quickly changed or repurposed.
- **Enabling empowered teams.** Cross-functional teams, including business subject matter experts, can collaborate directly in the tools selected.

However, LCNC projects still require good product thinking, stakeholder engagement, and governance. Without alignment and oversight, teams risk creating siloed tools, duplicated logic, or security and scalability issues.

USE CASE

Rapid Development of an Internal Tool Using a Low-Code Platform

A large insurance provider needed to digitize and automate a manual claims-intake process for internal staff. Historically, claims were submitted via spreadsheets and routed manually for approvals, creating delays and inconsistency.

Rather than waiting for limited developer capacity, a cross-functional team of business analysts and operations staff used a low-code platform to build a web-based intake form and workflow.

Key elements of the solution included the following:

- A drag-and-drop form builder to collect data from staff,
- Business rule engines to validate claim data and trigger routing,
- Email notifications and dashboards for visibility, and
- Integration with an internal database for case tracking.

The initial version was usable within 3 weeks and continuously improved based on feedback from staff. A senior software developer later reviewed the system for security and scalability before broader rollout.

As a result, the time to submit and route claims dropped by 60%, and process errors declined significantly. The business team felt ownership of the solution, and IT appreciated that core developers remained focused on more complex priorities.

This case shows how low-code tools, used thoughtfully, can support agile delivery by empowering the people closest to the problem to help shape and iterate on the solution.

A2.4 DORA Metrics: Measuring Software Delivery Performance

DevOps research and assessment (DORA) metrics provide a way to evaluate the effectiveness of software delivery in agile and DevOps environments. These metrics help teams measure how efficiently and reliably they deliver working software.

The goal is not to measure individual performance or enforce rigid benchmarks but rather to monitor system-level outcomes and support continuous improvement. These metrics are particularly useful in organizations adopting agile and DevOps at scale, where visibility into delivery health can guide prioritization, resourcing, and team support.

A2.4.1 The Four DORA Metrics

The DORA framework identifies four metrics that have shown strong correlation with organizational performance across industries and team structures:

1. **Deployment frequency**—How often a team successfully releases to production.

   What it tells us: Whether work is being delivered in small, incremental batches.

2. **Lead time for changes**—The time from code committed to code successfully running in production.

   What it tells us: How long it takes to turn an idea into customer-visible value.

3. **Change failure rate**—The percentage of deployments that result in degraded service or require remediation.

   What it tells us: How stable and reliable the delivery process is.

4. **Time to restore service**—The average time it takes to restore service after a production incident.

   What it tells us: How effectively teams respond to and recover from issues.

DORA metrics measure both delivery speed and delivery quality, as depicted in Figure A2-3.

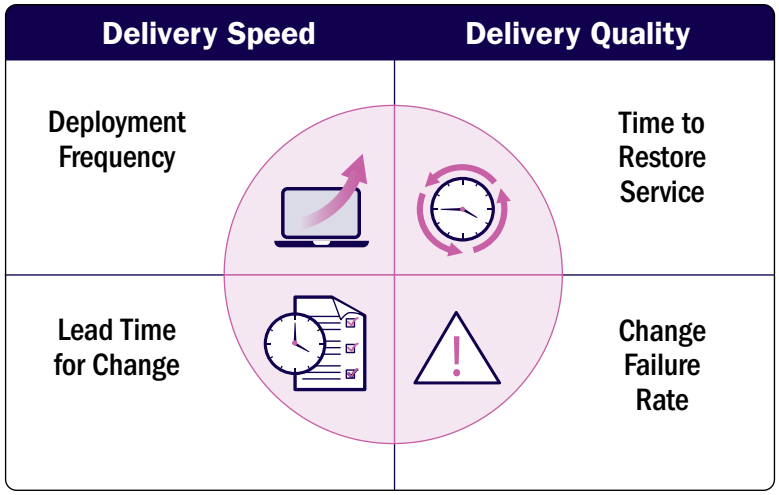


Figure A2-3. DORA Metrics Divided Into Delivery Speed and Delivery Quality Groups

A2.4.2 Interpreting the Metrics

DORA metrics are most valuable when they are:

- Tracked consistently over time for trends;
- Used to compare teams to themselves, not to each other;
- Combined with qualitative insights (e.g., team retrospectives, user feedback); and
- Framed as part of a continuous improvement conversation, not a performance evaluation.

Metrics can be visualized using dashboards or simple reports, depending on the organization's tooling. Many teams start by collecting these metrics manually before automating them.

USE CASE

Using DORA Metrics to Support Agile Team Growth

A product development group within a national health organization began transitioning to agile and DevOps practices. While they introduced automated testing and CI/CD pipelines, senior leadership lacked a clear view of delivery performance and impact.

The group piloted DORA-metrics collection across three teams. Initially, the data was inconsistent, but after establishing shared definitions and automating collection within their pipelines, trends began to emerge.

They discovered the following:

- One team deployed frequently but had a high change-failure rate.
- Another team deployed less often but had a low change-failure rate and faster recovery times.

Rather than push for uniformity, leadership used the data to open conversations. The high-failure team was given time to improve testing coverage and post-deployment monitoring. The team deployed less often identified deployment automation as a constraint and was supported in addressing it.

Over several quarters, all three teams improved in their own ways—and team leads reported feeling supported, not judged.

This experience highlights how DORA metrics, when used constructively, can guide adaptive improvement while preserving autonomy and trust.

A2.5 Summary

DevOps, DevSecOps, LCNC tools, and DORA metrics are responses to the growing need for speed, security, and measurable outcomes in software delivery. While not directly applicable to all types of projects, they can significantly improve flow and feedback in software contexts. Agile project teams working in these environments are encouraged to learn about these tools and approaches and apply them thoughtfully in their unique context.

A2.6 Annex 2 References

Sections A2.6.1 through A2.6.6 list the sources used in the development of this annex.

A2.6.1 DevOps Case Study References

The case study was based on the U.S. General Services Administration (GSA) 18F team's work modernizing legacy systems for federal agencies.

Primary references:

- **18F blog and case reports** on DevOps and CI/CD adoption in government IT
  - Specifically:
    - "From Idea to Production: Our CI/CD Pipeline"
    - "Modernizing Legacy Systems With DevOps Principles"

Project characteristics matched in the case study:

- Transition from quarterly to weekly releases,
- Implementation of automated CI pipelines and infrastructure provisioning, and
- Outcome metrics including reduced lead time and fewer production incidents.

Why selected:

- Publicly documented, noncommercial, and aligns with the neutral tone of the *Agile Practice Guide*, and
- Avoids endorsement while still offering a realistic, instructive example.

How adapted:

- Team and agency names removed,
- Specific tooling abstracted, and
- Outcomes generalized for use in standards-aligned guidance.

A2.6.2 Core Sources for DevOps Concepts

The following core sources were used to inform the DevOps concepts discussed:

1. **State of DevOps Reports** (Puppet; later, Google/DORA)
   - Foundation for DevOps pipeline capabilities and cultural practices
   - See the "2021 State of DevOps Report": https://dora.dev/research/2021/dora-report/
2. ***Accelerate: The Science of Lean Software and DevOps***
   - Forsgren, N., Humble, J., & Kim, G. (2018). *Accelerate: The Science of Lean Software and DevOps*. IT Revolution Press. ISBN: 978-1942788331
   - Defines core DevOps practices and metrics
   - Widely cited in both industry and academia
3. ***The DevOps Handbook***
   - Kim, G., Humble, J., Debois, P., & Willis, J. (2016). *The DevOps Handbook: How to Create World-Class Agility, Reliability, and Security in Technology Operations*. IT Revolution Press. ISBN: 978-1942788003
   - Explains DevOps principles, toolchain components, and real-world patterns

A2.6.3 DevSecOps References

The DevSecOps case study is an anonymized composite based primarily on publicly documented work from the Canada Revenue Agency (CRA) and supplemented by practices observed in the following:

1. U.S. Internal Revenue Service (IRS) modernization efforts
2. UK Government Digital Service (GDS) DevSecOps practices
3. U.S. Air Force Platform One DevSecOps initiative (for process modeling)

A2.6.4 Core Sources for the DevSecOps Case Study

The following sources were used to inform the DevSecOps case study:

- **Canada Revenue Agency (CRA)—Digital Transformation Strategy.** CRA has been public about its move toward agile and secure-by-design practices. Their published "Digital Innovation Strategy" discusses integrating cybersecurity in early delivery phases. Source: https://www.canada.ca/en/revenue-agency
- **National Institute of Standards and Technology (NIST) DevSecOps Framework and Guidance—NIST SP 800-204 Series.** "Security Strategies for Microservices-based Application Systems": https://csrc.nist.gov/pubs/sp/800/204/final
- **U.S. Department of Homeland Security—Continuous Diagnostics and Mitigation (CDM) Program.** Government case studies and standards related to secure development pipelines: https://www.cisa.gov/cdm
- **OWASP DevSecOps Maturity Model**
  - Defines key practices and levels of maturity for integrating security into DevOps: https://owasp.org/www-project-devsecops-maturity-model/
- **DevSecOps: A Leader's Guide to Producing Secure Software at Speed** (U.S. Department of Defense Enterprise DevSecOps Initiative)
  - Contains real-world examples and frameworks for integrating security early: https://software.af.mil
- **Agile Alliance DevOps Working Group Papers**
  - Provides noncommercial perspectives on DevOps and DevSecOps in agile contexts: https://www.agilealliance.org/devops
- **Google Cloud and DORA Reports (for organizational patterns)**
  - Emphasizes secure delivery metrics and shift-left approaches: https://dora.dev/research/2021/dora-report/

A2.6.5 Low-Code/No-Code Platform References

The following sources were used to inform the LCNC section and are suitable for documentation in audit trails:

1. **Forrester and Gartner LCNC Market Reports (2021–2023)**
   - Describe capabilities, trends, and categories of low-code platforms, e.g., "Magic Quadrant for Enterprise Low-Code Application Platforms"
2. **Harvard Business Review—"Citizen Developers: Powering the Next Wave of Innovation"**
   - HBR.org, July 2021
3. **Microsoft Power Platform and Salesforce Lightning Documentation (for feature baselines)**
   - Used to inform platform-agnostic capability descriptions
   - Sources not cited directly in publication text but inform general understanding

4. **IEEE Software (2022)—"Low-Code Development: What It Is and Why It Matters"**
    - Peer-reviewed technical exploration of LCNC risks and opportunities
5. **Agile Alliance Industry Reports on Agile in Non-IT Contexts**
    - Emphasize LCNC's role in enabling agility in business-led initiatives
    - https://www.agilealliance.org
6. **McKinsey & Company—"Tech Modernization for the Next Normal"**
    - Discusses LCNC's role in digital transformation and agile at scale: https://www.mckinsey.com
7. **Case inspiration drawn from published examples by Liberty Mutual, Munich Re, and various regional healthcare networks** (LCNC case studies from conferences like LowCodeCon and TechEd)

A2.6.6 Reference Sources for DORA Metrics Section

The following sources were used to inform the DORA metrics section:

1. ***Accelerate: The Science of Lean Software and DevOps***
    - Forsgren, N., Humble, J., & Kim, G. (2018). *Accelerate: The Science of Lean Software and DevOps*. IT Revolution Press. ISBN: 978-1942788331
    - Foundation for DORA metrics research
2. **2021 State of DevOps Report** (Google Cloud/DORA)
    - Methodology and statistical basis for DORA metrics: https://cloud.google.com/devops/state-of-devops
3. **Research papers and DevOps talks: Nicole Forsgren, PhD**
    - Primary researcher behind DORA metrics
    - "The ROI of DevOps" presentations and GitHub Engineering blog contributions
4. **ThoughtWorks Technology Radar**
    - Commentary on implementation of DORA in enterprise settings: https://www.thoughtworks.com/radar
5. **Case study inspired by implementations at**:
    - UK NHS Digital (agile metrics for service delivery)
    - Public sector DORA adoption at Shared Services Canada
    - Healthcare DevOps transformation reports (e.g., from conferences like DevOps Enterprise Summit)

Annex A3
Overview of Agile and Lean Frameworks

This annex describes some commonly used agile approaches. These approaches can be used "as is" or combined to adapt to what works best for a given environment or situation. It is not necessary to use any of these; an agile approach can be developed from scratch as long as it adheres to the mindset, values, and principles of the *Agile Manifesto.*

If the agile principles are followed to deliver value at a sustainable pace and the developed approach promotes collaboration with the customer, a specific approach is not required. A link to more information regarding each approach can be found in the Bibliography section of this practice guide.

A3.1 Selection Criteria for the *Agile Practice Guide*—Second Edition

There are too many agile approaches, techniques, and tool kits to be explicitly included in this practice guide. PMI® Disciplined Agile® (DA®) is designed to be a tool kit. Figure A3-1 provides examples of additional approaches and tool kits that are noteworthy based on the depth of guidance and breadth of their life cycles. The specific approaches selected for discussion are popular examples that are:

- **Designed for holistic use.** Some agile approaches are centered on a single project activity such as estimation or reflection. The listed examples include only the more holistic agile frameworks. Some are more full-featured than others, but all of the selected approaches are those intended to guide a broad set of project activities.
- **Formalized for common use.** Some agile frameworks are proprietary in nature and designed for specific use by a single organization or within a single context. The frameworks described in Sections A3.2 through A3.8 focus on those intended for common use in a variety of contexts.
- **Popular in modern use.** Some agile frameworks are holistically designed and well formalized but are simply not commonly used in most projects or organizations. The agile frameworks described in this annex have been adopted by a significant number of industries, as measured by a collection of recent industry surveys.

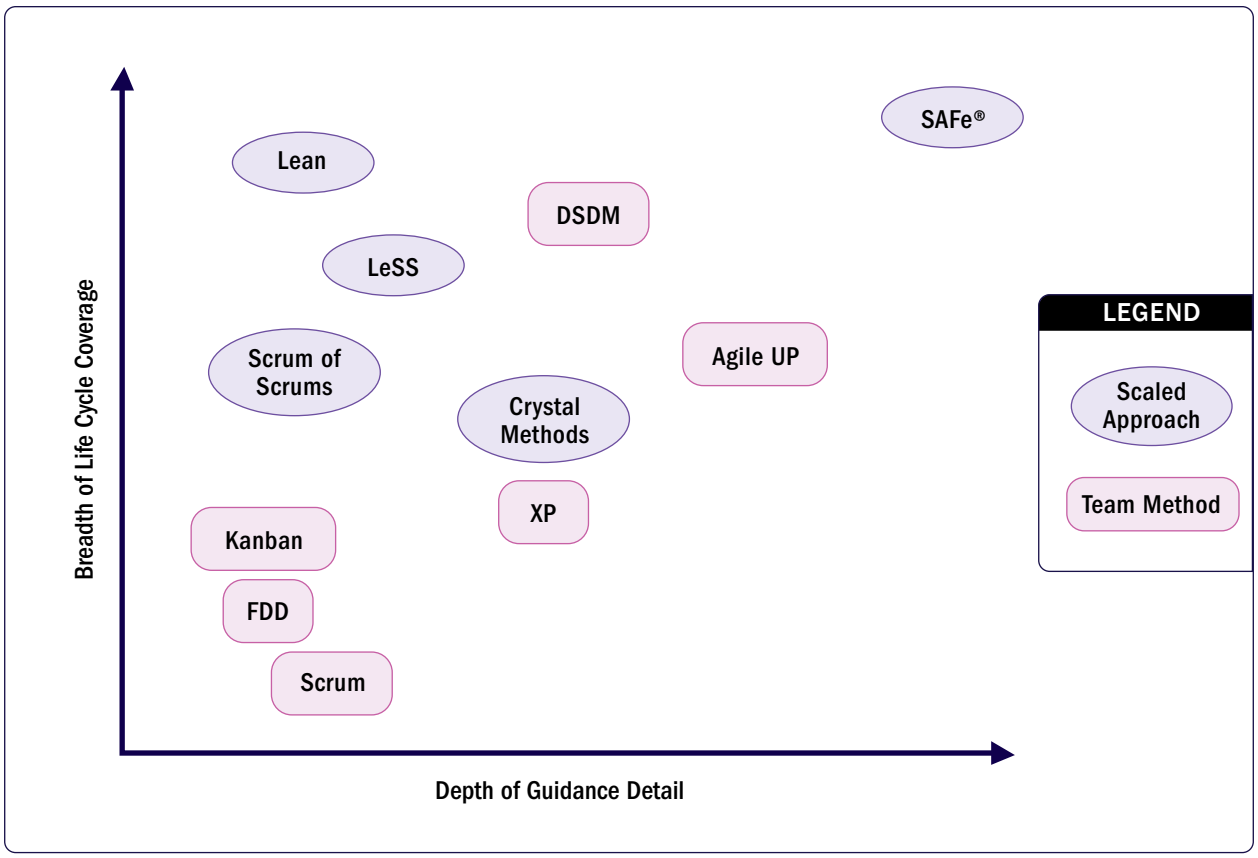


Figure A3-1. Agile Approaches Plotted by Breadth and Detail

A3.2 Scrum

Scrum is a single-team process framework used to manage product development. The framework consists of Scrum roles, events, artifacts, and rules, using an iterative approach to deliver working products. Scrum is run on timeboxes of 1 month or less with consistent durations called sprints, where a potentially releasable increment of a product is produced. Table A3-1 lists Scrum events and artifacts utilized for project execution.

The Scrum team consists of a product owner, development team, and scrum master:

- The product owner is responsible for maximizing the value of the product.
- The development team is a cross-functional, self-organizing team consisting of team members who have everything they need within the team to deliver the working product without depending on others outside of the team.
- The scrum master is responsible for helping ensure the Scrum process is upheld and works to help ensure the Scrum team adheres to the practices and rules in addition to coaching the team on removing impediments.

Scrum not only organizes work into roles, events, and artifacts but also drives continuous improvement through short cycles that enable the team to inspect, adapt, and deliver value frequently.

Table A3-1. Scrum Events and Artifacts

Events	Artifacts
Sprint Sprint planning Daily scrum Sprint review Sprint retrospective	Product backlog Sprint backlog Increments

A3.3 Extreme Programming (XP)

Extreme Programming (XP) is a software development method based on frequent cycles. The name is based on the philosophy of distilling a given good practice to its purest, simplest form and applying that practice continuously throughout the project.

XP is most known for popularizing a holistic set of practices intended to improve the results of software projects. The method was first formalized as a set of 12 primary practices but then gradually evolved to adopt several other corollary practices. These practices are listed in Table A3-2.

This evolution is the result of designing and adopting techniques through the filter of core values (communication, simplicity, feedback, courage, respect), and is informed by key principles (humanity, economics, mutual benefit, self-similarity, improvement, diversity, reflection, flow, opportunity, redundancy, failure, quality, small steps, and accepted responsibility).

Table A3-2. Practices of Extreme Programming (XP)

XP Practice Area	Primary	Secondary
Organizational	• Sit together • Whole team • Informative workspace	• Real customer involvement • Team continuity • Sustainable pace
Technical	• Pair programming • Test-first programming • Incremental design	• Shared code/collective ownership • Documentation from code and tests • Refactoring
Planning	• User stories • Weekly cycle • Quarterly cycle • Slack	• Root cause analysis • Shrinking teams • Pay per use • Negotiated scope contract • Daily coordination meetings
Integration	• 10-minute build • Continuous integration • Test-first development	• Single code base • Incremental deployment • Daily deployment

A3.4 Kanban Method

Kanban, the lean manufacturing concept, and the Kanban Method, a Lean inspired approach for knowledge work, are often confused.

Kanban in Lean manufacturing is a system for scheduling inventory control and replenishment. This process of just-in-time (JIT) inventory replenishment was originally seen in grocery stores when shelves were restocked based on the gaps in the shelves and not supplier inventory. Inspired by these JIT inventory systems, Taiichi Ohno developed Kanban, and it was applied at the main Toyota manufacturing facility in 1953.

The word "kanban" is literally translated as "visual sign" or "card." Physical kanban boards with cards enable and promote the visualization and flow of the work through the system for everyone to see. This information radiator (large display) is made up of columns that represent the states the work needs to flow through to get to done. The simplest boards could have three columns (i.e., To Do, Doing, and Done) but are adaptable to whatever states are deemed needed.

David J. Anderson created the Kanban Method for knowledge work and software development based on his implementations at Microsoft (2004) and Corbis (2006–2007), later consolidating the method's principles and practices in his 2010 book, *Kanban: Successful Evolutionary Change for Your Technology Business*.

The Kanban Method is utilized and applicable in many settings and allows for a continuous flow of work and value to the customer. The Kanban Method is less prescriptive than some agile approaches and thus less disruptive to begin implementing as it is the original "start where you are" method. Organizations can begin applying the Kanban Method with relative ease, progressing toward fully implementing the method if that is what they deem appropriate.

Unlike most agile approaches, the Kanban Method does not prescribe the use of timeboxed iterations. Iterations can be used within the Kanban Method, but the principle of pulling single items through the process continuously and limiting WIP to optimize flow should always remain intact. The Kanban Method may be best used when a team or organization is trying to improve the following:

- **Flexibility.** Teams are typically not bound by timeboxes and will work on the highest priority item in the backlog of work.
- **Focus on continuous delivery.** Teams are focused on flowing work through the system to completion and not beginning new work until WIP is completed.
- **Increased productivity and quality.** Productivity and quality are increased by limiting WIP.
- **Increased efficiency.** Teams check each task for value-adding or non-value-adding activities and remove the non-value-adding activities.
- **Team member focus.** Limited WIP allows the team to focus on the current work.
- **Variability in the workload.** The Kanban Method is useful when there is unpredictability in the way that work arrives and it becomes impossible for teams to make predictable commitments, even for short periods of time.
- **Reduction of waste.** Transparency makes waste visible so it can be removed.

The Kanban Method is derived from Lean thinking principles. The defining principles and the core properties of the Kanban Method are listed in Table A3-3.

Table A3-3. Defining Principles and Properties of the Kanban Method

Defining Principles	Core Properties
• Start with the current state • Agree to pursue incremental, evolutionary change • Respect the current process, roles, responsibilities, and titles • Encourage acts of leadership at all levels	• Visualize the workflow • Limit WIP • Manage flow • Make process policies explicit • Implement feedback loops • Improve collaboratively

The Kanban Method is a holistic framework for an incremental, evolutionary process and systems change for organizations. The method uses a pull system to move the work through the process. When the team completes an item, the team can "pull" a new item into that spot.

Kanban boards, such as the one shown in Figure A3-2, are a low-tech, high-touch tool that may seem overly simplistic at first, but those using them soon realize their power. Utilizing policies for entry and exit to columns as well as constraints such as limiting WIP, kanban boards provide clear insight to workflow, bottlenecks, blockers, and overall status. Additionally, the board acts as an information radiator to anyone who sees it, providing up-to-date information on the status of the team's work.

In the Kanban Method, it is more important to complete work than it is to start new work. There is no value derived from work that is not completed, so the team works together to implement and adhere to the WIP limits and get each piece of work through the system to "done."

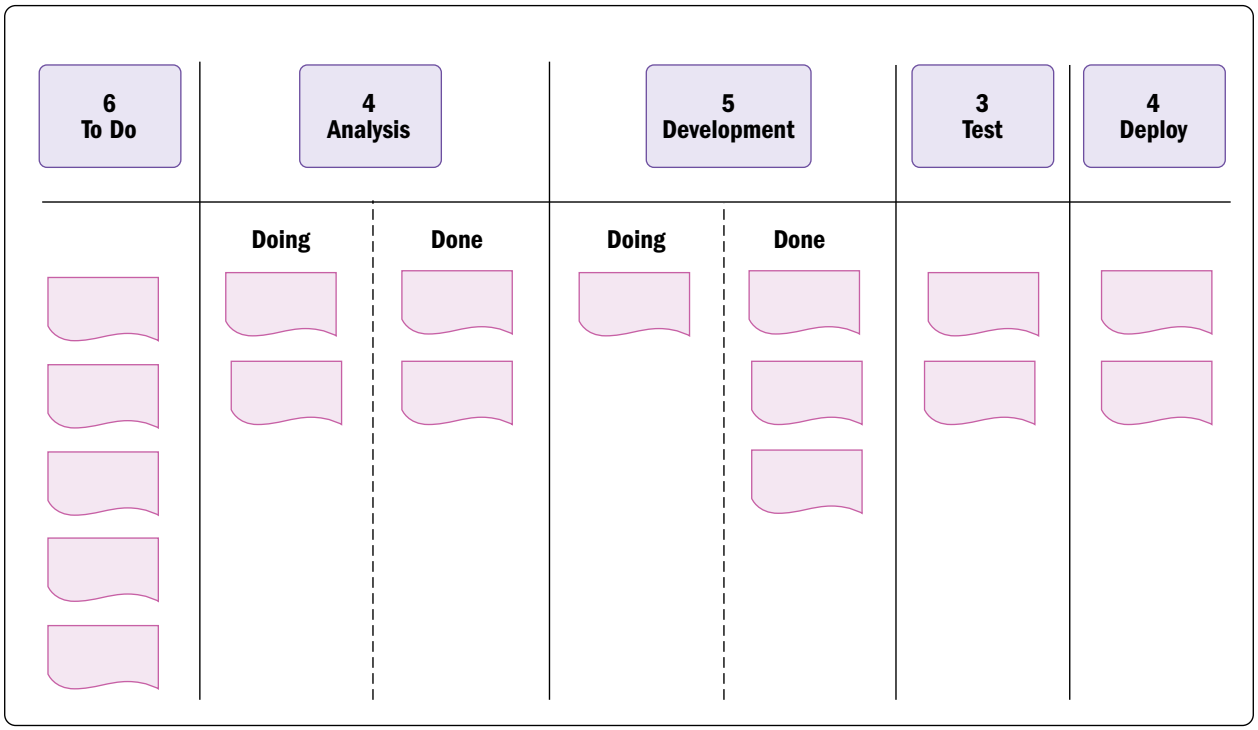


Figure A3-2. Kanban Board Demonstrating WIP Limits and a Pull System to Optimize the Flow of Work

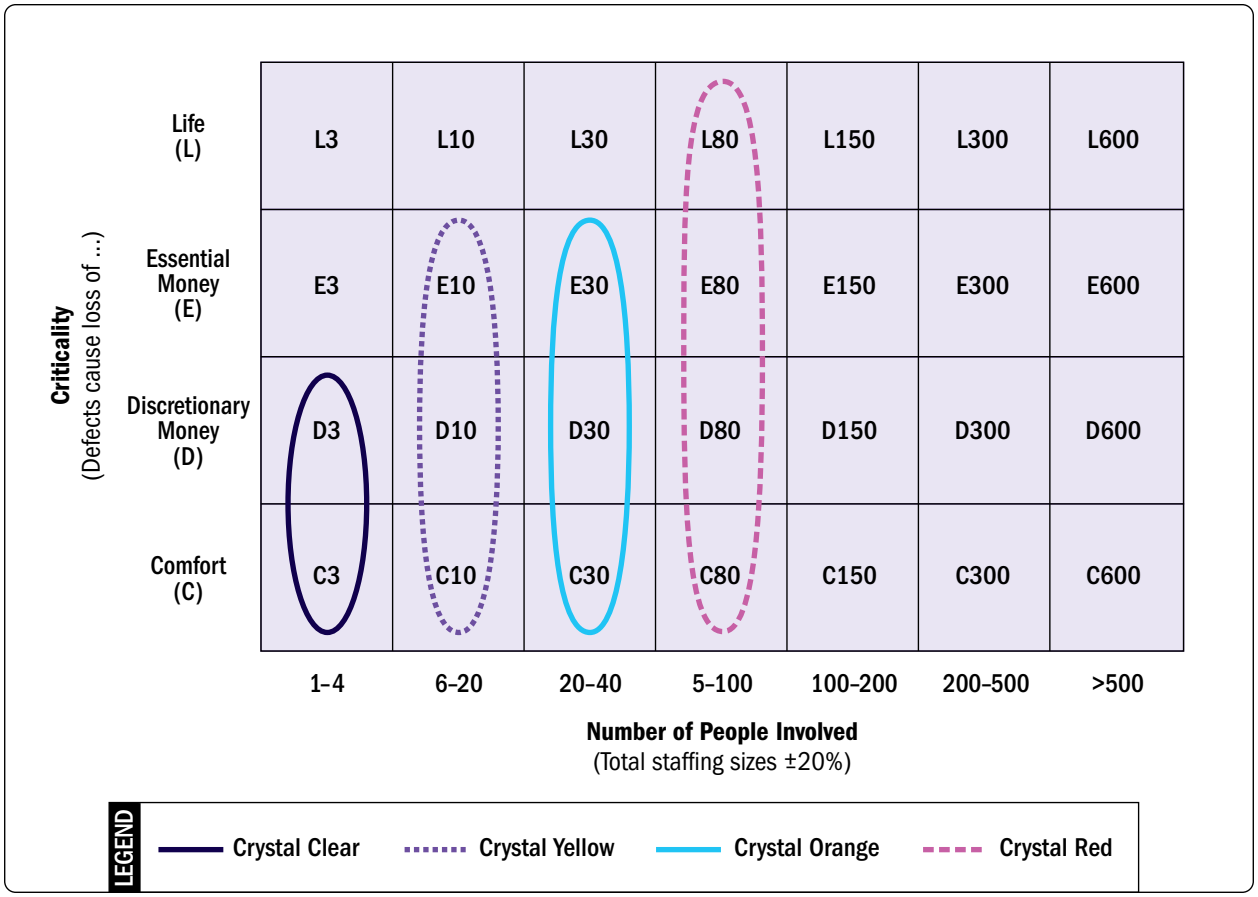


Figure A3-3. The Crystal Family of Methods

A3.5 Crystal Methods

Crystal is a family of methodologies. Crystal methodologies are designed to scale and provide a range of methodology rigor based on the project size (number of people involved in the project) and criticality of the project. See Figure A3-3.

Crystal methodology realizes that each project may require a slightly tailored set of policies, practices, and processes to meet the project's unique characteristics. The family of methodologies uses different colors based on "weight" to determine which methodology to use. The use of the word "crystal" comes from the gemstone, where the various faces represent underlying core principles and values. The "faces" are a representation of the techniques, tools, standards, and roles listed in Table A3-4.

A3.6 Feature-Driven Development

Feature-driven development (FDD) was developed to meet the specific needs of a large software development project. "Feature" refers to a small, business-value capability.

There are six primary roles on an FDD project, where individuals can take on one or more of the following roles:

- Project manager,
- Chief architect,

Table A3-4. The Core Values and Common Properties of Crystal

Core Values	Common Properties[1]
• People • Interactions • Community • Skills • Talents • Communications	• Frequent delivery • Reflective improvement • Close or osmotic communication • Personal safety • Focus • Easy access to expert users • Technical environment with automated tests, configuration management, and frequent integration

[1] The more these properties are in a project, the more likely it is to succeed.

- Development manager,
- Chief programmer,
- Class owner, and
- Domain expert.

An FDD project is organized around the following five processes or activities, which are performed iteratively:

- Develop an overall model,
- Build a features list,
- Plan by feature,
- Design by feature, and
- Build by features.

The life cycle flow and interaction of these five processes are illustrated in Figure A3-4.

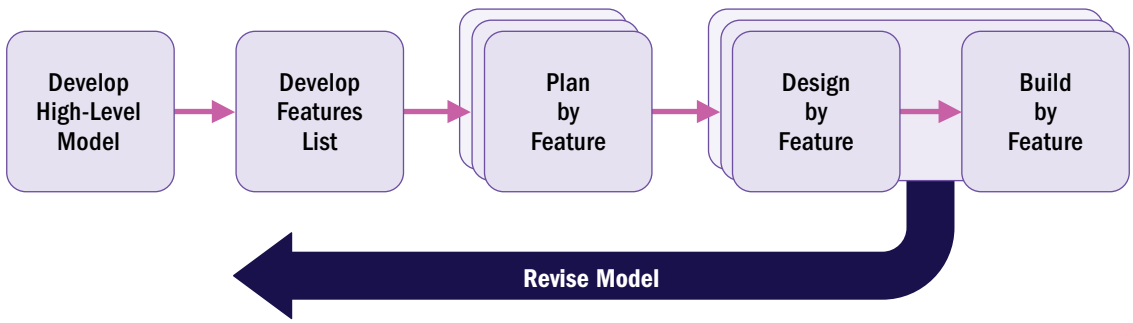


Figure A3-4. Feature-Driven Development Project Life Cycle

FDD activities are supported by a core set of software engineering good practices:

- Domain object modeling,
- Developing by feature,
- Individual class ownership,
- Feature teams,
- Inspections,
- Configuration management,
- Regular builds, and
- Visibility of progress and results.

A3.7 Dynamic Systems Development Method

Dynamic Systems Development Method (DSDM) is an agile project delivery framework initially designed to add more rigor to existing iterative methods popular in the 1990s. The method was developed as a noncommercial collaboration among industry leaders.

DSDM is best known for its emphasis on constraint-driven delivery. The framework will set cost, quality, and time at the outset and then use formalized prioritization of scope to meet those constraints, as shown in Figure A3-5.

Eight principles guide the use of the DSDM framework:

- Focus on the business need.
- Deliver on time.
- Collaborate.
- Never compromise quality.
- Build incrementally from firm foundations.
- Develop iteratively.
- Communicate continuously and clearly.
- Demonstrate control (using appropriate techniques).

A3.8 Scaling Frameworks

Sections A3.8.1 through A3.8.4 explore several prominent agile scaling frameworks, including Scrum of Scrums, Scaled Agile Framework (SAFe®), Large Scale Scrum (LeSS), and Scrum@Scale, which can help organizations implement agile practices at scale.

A3.8.1 Scrum of Scrums (SoS)

Scrum of Scrums (SoS), also known as “meta-Scrum,” is a technique used when two or more Scrum teams consisting of three to nine members each need to coordinate their work instead of one large Scrum team. A representative from each team attends a meeting with the other team representative(s), potentially daily but typically two to three times a week. The daily meeting is conducted similarly to the daily coordination meeting in Scrum, where the representative

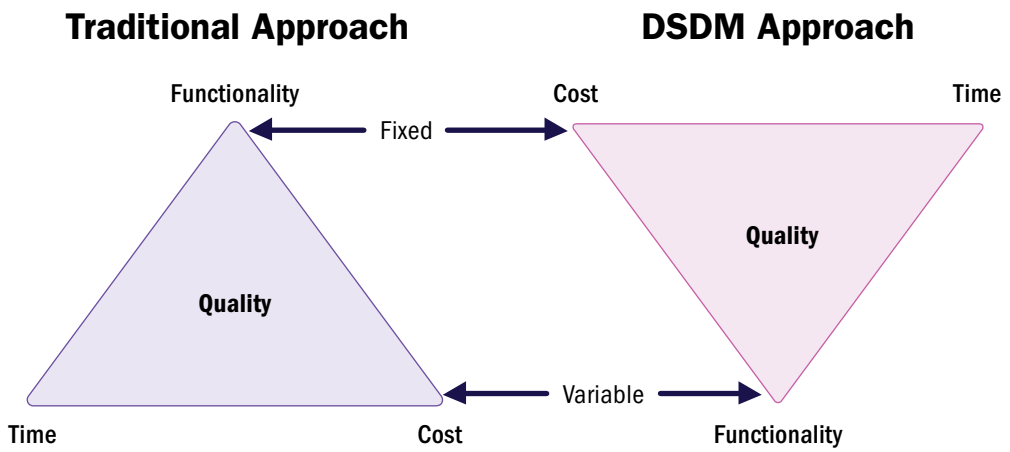


Figure A3-5. DSDM Approach to Constraint-Driven Agility

reports completed work, the next set of work, any current impediments, and potential upcoming impediments that might block the other team(s). The goal is to help ensure the teams are coordinating work and removing impediments to optimize the efficiency of all the teams.

Large projects with several teams may result in conducting a Scrum of Scrum of Scrums, which follows the same pattern as SoS, with a representative from each SoS reporting into a larger group of representatives, as shown in Figure A3-6.

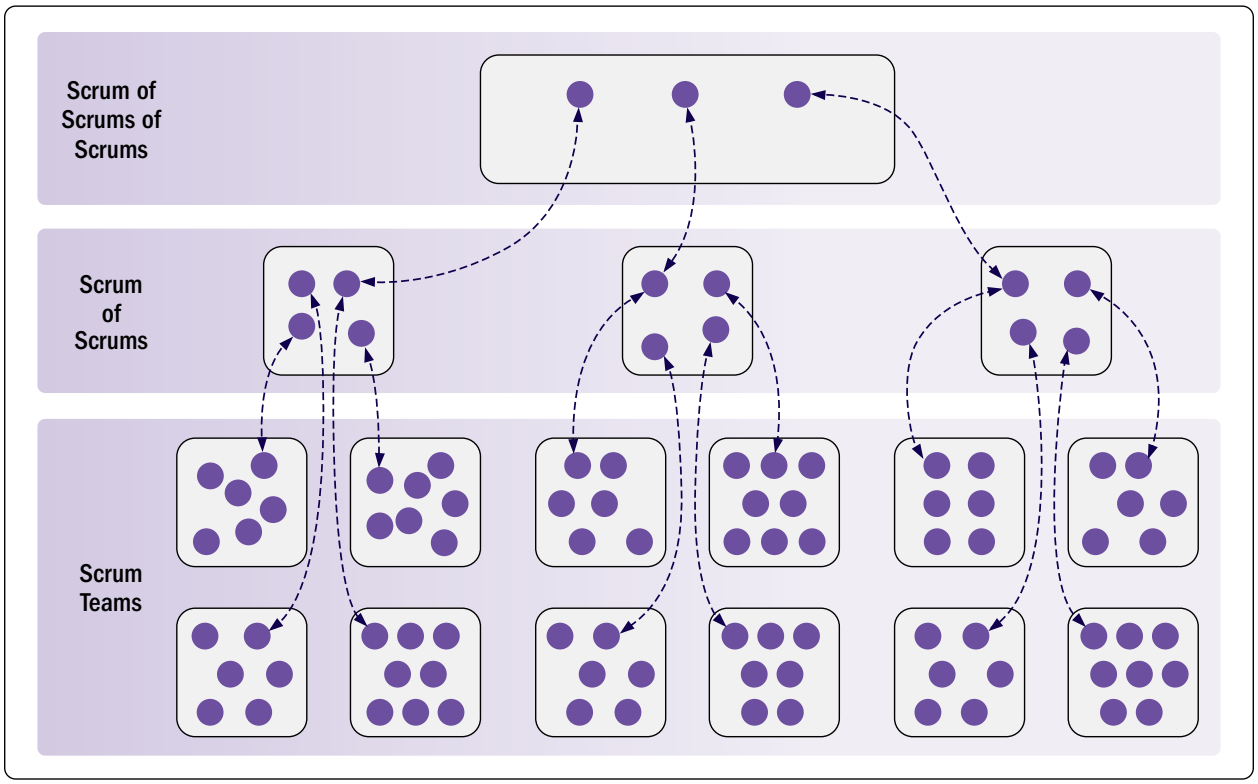


Figure A3-6. Representatives of Scrum Teams Participating in SoS Teams

A3.8.2 Scaled Agile Framework (SAFe®)

The Scaled Agile Framework (SAFe®) focuses on providing a knowledge base of patterns for scaling development work across all levels of the enterprise.

SAFe® is focused on the following principles:

- Take an economic view.
- Apply systems thinking.
- Assume variability; preserve options.
- Build incrementally with fast, integrated learning cycles.
- Base milestones on objective evaluation of working systems.
- Visualize and limit WIP, reduce batch sizes, and manage queue lengths.
- Apply cadence; synchronize with cross-domain planning.
- Unlock the intrinsic motivation of knowledge workers.
- Decentralize decision-making.
- Organize around value.

SAFe® focuses on detailing practices, roles, and activities at the portfolio, program, and team levels with an emphasis on organizing the enterprise around value streams that provide continuous value to the customer.

A3.8.3 Large Scale Scrum (LeSS)

Large Scale Scrum (LeSS) is a toolkit for organizing several development teams toward a common goal, extending the Scrum method shown in Figure A3-6. The core organizing principle is to retain as many elements of the conventional, single-team Scrum model as possible. This effort helps minimize any extensions to the model that could create unnecessary confusion or complexity. Table A3-5 shows a comparison of LeSS and Scrum.

Table A3-5. Comparison of LeSS and Scrum

Similarities of LeSS and Scrum	LeSS Techniques Added to Scrum
• One single product backlog • One DoD for all teams • One potentially shippable product increment at the end of each sprint • One product owner • Complete, cross-functional teams • One sprint	• Sprint planning is more formally divided into two parts: what and how • Organic, cross-team coordination • Overall cross-team refinement • Overall retrospective focused on cross-team improvements

To extend Scrum without losing its essence, LeSS promotes the use of certain discerning principles such as systems thinking, whole-product focus, and transparency.

A3.8.4 Scrum@Scale

Scrum, as first described in *The Scrum Guide*,[8] is a framework used by a single team to build, deliver, and maintain complex products. Over time, it has been applied more broadly, including to the development of products, processes, services, and systems that depend on the coordinated work of multiple teams.

Scrum@Scale was designed to coordinate this larger, multi-team environment efficiently. It does this by establishing a "minimum viable" layer of governance using a scale-free architecture, keeping oversight lightweight while enabling alignment across teams.

Scrum@Scale draws on core Scrum principles as well as complex adaptive systems thinking, game theory, and object-oriented concepts.

Scrum@Scale supports an organization in aligning several networks of Scrum teams around shared, prioritized objectives. It aims to achieve this by extending the operating patterns of a single Scrum team across a network, while keeping the management function within a deliberately lean, minimum viable bureaucracy.

A3.9 PMI® Disciplined Agile®

PMI® Disciplined Agile® is a process-decision tool kit that integrates several popular agile practices into a comprehensive model. PMI® Disciplined Agile® was designed to offer a balance between those popular methods deemed to be either too narrow in focus or too prescriptive in detail. To achieve that balance, the framework blends various agile techniques according to the following principles:

- **Delight customers.** Go beyond meeting expectations. Create experiences that win loyalty, or competitors will.
- **Be awesome.** Strive to improve people, teams, and organizations so work and results keep getting better.
- **Context counts.** Every situation is unique. Choose and evolve your way of working to fit current conditions.
- **Be pragmatic.** Aim for effectiveness. Use agile, lean, or traditional techniques when they best suit the work.
- **Choice is good.** Maintain options and understand trade-offs so you can select the best-fit practices.
- **Optimize flow.** Improve end-to-end value delivery, not just local team efficiency, to respond faster to needs.
- **Organize around products/services (new).** Structure teams around customer offerings and value streams.
- **Enterprise awareness.** Act with the whole organization in mind. Use and improve shared guidance, roadmaps, and assets.

[8] Schwaber, K., & Sutherland, J. (2020). *The Scrum guide—The definitive guide to Scrum: The rules of the game*. https://scrumguides.org/docs/scrumguide/v2020/2020-Scrum-Guide-US.pdf

Annex A4
Sustainability and Enduring Value Optimization

Agile ways of working grew from a call for a sustainable pace, reminding practitioners that teams should create value without burning out. Today, the sustainability idea stretches further: Sustainable products safeguard people, the planet, and long-term prosperity. When we talk about sustainability in an agile context, we are talking about making decisions now that will not mortgage tomorrow's options.

A4.1 Origin and Definition

Agile Alliance hosts the Agile Sustainability Initiative, which provides knowledge and inspiration to enable action on sustainability to the broader agile community.

The *PMBOK® Guide*—Eighth Edition [2] identifies "Integrate Sustainability Within All Project Areas" as a core principle, stating that a project should "meet the needs of the present without compromising the ability of future generations to meet their own needs."

In the context of this practice guide, every agile choice, scope, schedule, cadence, funding, and so forth should consider the long-term environmental, social, and economic effects rather than only near-term delivery.

A4.2 Why This Principle Matters

Sustainability has increasingly become a critical consideration in agile project management, influencing not only the environmental and social impacts of all types of projects but also their long-term success and viability, as highlighted by the following key factors:

- **Extended value horizon.** Stakeholders judge success beyond release day. Environmental and social impacts shape the license to operate, cost of capital, and brand trust.
- **Risk visibility.** Carbon pricing, resource scarcity, and new disclosure rules (e.g., Corporate Sustainability Reporting Directive [CSRD], International Financial Reporting Standard S2 [IFRS S2]) turn hidden externalities into near-term project risks.
- **Innovation driver.** Constraints around energy, materials, and equity often spark creative solutions, leaner code, circular supply loops, and inclusive features.

A4.3 Key Considerations for Agile Teams

The key considerations around products, people, and strategy for agile teams are described in .Figure A4-1.

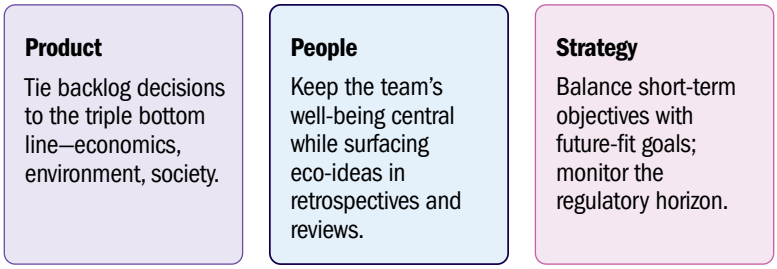


Figure A4-1. Key Considerations for Agile Teams

A4.3.1 In Practice: Examples and Roles/Responsibilities

Examples of incorporating sustainability practices in agile projects include the following:

- A fintech team rewrites batch jobs to run in off-peak, renewable windows, cutting cloud emissions by 30% and saving US$50,000 in operating expenses (OPEX) in Year 1.
- A construction company specifies recycled aggregate concrete, reducing embodied carbon by 25% while meeting budget requirements.

Roles and responsibilities for implementing sustainability practices on agile teams are detailed in Table A4-1.

The United Nations describes 17 sustainable development goals (SDGs) that most of the organizations worldwide try to align to (https://sdgs.un.org/goals).

TIP

Remember

Section 2 provides more information that reinforces the sustainability mindset, including Agile Principle 2, "focus on value," for balancing near-term ROI with life cycle impact and Agile Principle 4, "adaptation and feedback," for using iteration reviews to surface sustainability insights.

Table A4-1. Sustainability-Focused Roles and Responsibilities

Role	Focus
Product owner/product manager	Aligns backlog items with stated environmental and social goals
Team	Surfaces eco-improvement ideas, sizes impact, and tracks eco-debt
Team leader/agile coach	Keeps the principle visible in reviews and retrospectives; removes blockers to green choices
Leadership/sponsor	Provides guardrails and flexible funding so sustainable options remain viable

TIP

Try/Avoid: A Quick Reference

These paired bullets can help teams quickly steer toward productive habits—and away from common traps:

- **Try:** Adding an eco-impact bullet to the team's DoD

  Avoid: Labeling sustainability as "Phase II" work
- **Try:** Prioritizing backlog items that cut energy use ≥20%

  Avoid: Deferring eco-refactoring indefinitely
- **Try:** Applying double-materiality thinking during risk reviews

  Avoid: Tracking only internal cost and schedule metrics

A4.3.2 Self-Reflection Questions

Use these prompts to pause, inspect recent choices, and gauge whether they support enduring value:

- What long-term impacts, positive or negative, were created in this iteration?
- How did sustainability influence backlog prioritization during this sprint?
- Which trade-off weighed cost or schedule against carbon or social impact?

A4.4 Common Antipatterns

The following short warnings can help flag behaviors that repeatedly dilute sustainability outcomes and should be corrected early:

- **Greenwashing metrics.** Tracking vanity statistics that mask real impact.
- **Efficiency tunnel vision.** Maximizing sprint velocity while wasting resources.
- **Deferred accountability.** Pushing sustainability tasks outside the release cadence.

Practitioners should log sustainability risks and opportunities in the same backlog or risk register, reviewing them on the regular team cadence.

TIP

Regulatory Watchlist

Projects operating in the European Union should verify CSRD thresholds yearly. Similar disclosure rules (e.g., IFRS S2) may apply elsewhere.

This principle threads through life cycle selection, measurement, team practices, and governance. Section 6 provides details on measurement for teams looking for examples of leading/lagging indicators.

A4.5 Annex A4 References

The references for Annex A4 are captured in Table A4-2.

Table A4-2. Annex A4 References

#	Full Reference	Notes/Web Source
1	International Organization for Standardization. *ISO 14001:2015 Environmental Management Systems: Requirements with Guidance for Use*. Geneva: ISO, 2015.	Authoritative EMS specification, iso.org
2	World Resources Institute & World Business Council for Sustainable Development. *Greenhouse Gas Protocol: A Corporate Accounting and Reporting Standard* (Revised Ed.). WRI/WBCSD, 2004.	Widely adopted Scope 1–3 accounting framework, ghgprotocol.org
3	Science-Based Targets initiative. *SBTi Corporate Net-Zero Standard, Version 1.2*. SBTi, March 2024 (first issued 2021).	Science-aligned pathway for corporate net-zero targets files, sciencebasedtargets.org
4	European Union. *Directive (EU) 2022/2464 of 14 Dec 2022 (Corporate Sustainability Reporting Directive)*. *Official Journal of the European Union* L 322/15, 16 December 2022.	Mandatory sustainability disclosure rules for large EU-based companies, eur-lex.europa.eu
5	International Sustainability Standards Board. *IFRS S2 Climate-Related Disclosures*. IFRS Foundation, June 2023. Effective for periods beginning 1 January 2024.	Global baseline for climate disclosure in financial reports, ifrs.org
6	Taskforce on Nature-related Financial Disclosures. *Recommendations of the TNFD*. TNFD, September 2023.	Emerging nature-risk disclosure framework complementing TCFD, tnfd.global
7	Project Management Institute (PMI). (2025). *A Guide to the Project Management Body of Knowledge (PMBOK® Guide)—Eighth Edition*. PMI.	Source of the principle "Integrate Sustainability Within All Project Areas"
8	Beck, K., Beedle, M., van Bennekum, A., Cockburn, A., Cunningham, W., Fowler, M., Grenning, J., Highsmith, J., Hunt, A., Jeffries, R., Kern, J., Marick, B., Martin, R. C., Mellor, S., Schwaber, K., Sutherland, J., & Thomas, D. *Manifesto for Agile Software Development*. February 2001.	Defines "sustainable pace" principle, https://agilemanifesto.org/principles.html
9	*The Green Project Management (GPM)® Sustainability Competence Standard*	Provides strategies for sustainability in projects, https://www.pmi.org/standards/the-gpm-sustainability-competence-standard
10	*GPM® P5™ Standard for Sustainability in Project Management v3.1*	Addresses sustainable project management practices, https://www.gpm.org/images/PDF/P5/The_GPM_P5_Standard_for_Sustainability_in_Project_Management_v3.1_6.pdf

Note: *All URLs were live and documents accessible at the time of compilation (10 December 2025).*

Appendix X1
Agile Procurement and Contracts

As mentioned earlier in this practice guide, the *Agile Manifesto* values "customer collaboration over contract negotiation." Many project failures stem from breakdowns in the customer–supplier relationship. Projects incur more risk when those involved in the contract take the perspective of winners versus losers. A collaborative approach is one that pursues a shared risk–reward relationship, where all sides win. Some contracting techniques that can formalize this dynamic include the following:

- **Multitiered structure.** Rather than formalizing an entire contracting relationship in a single document, project parties can achieve more flexibility by describing different aspects in different documents. Mostly fixed items (e.g., warranties, arbitration) can be locked in a master agreement. Meanwhile, all parties list other items subject to change (e.g., service rates, product descriptions) in a schedule of services. The contract can reference them in the master services agreement. Finally, more dynamic items such as scope, schedule, and budget can be formalized in a lightweight statement of work. Isolating the contract elements that change more often into a single document simplifies modifications and thus flexibility.
- **Emphasize value delivered.** Many vendor relationships are governed by fixed milestones or phase gates focused on intermediate artifacts rather than a full deliverable of incremental business value. Often, these controls limit the use of feedback to improve the product. Instead, milestones and payment terms can be structured based on value-driven deliverables to enhance the project's agility.
- **Fixed-price increments.** Rather than lock an entire project scope and budget into a single agreement, a project can decompose the scope into fixed-price microdeliverables such as user stories. For the customer, this gives more control over how the money is spent. For the supplier, these fixed prices limit the financial risk of overcommitment to a single feature or deliverable.

- **Not-to-exceed time and materials (T&M).** Customers incur unwanted risk from a traditional T&M approach. One alternative is to limit the overall budget to a fixed amount. This option allows the customer to incorporate new ideas and innovations into the project that were not originally planned. When customers want to incorporate any new ideas, they have to manage to a given capacity, replacing original work with new work. Work should be closely monitored as the allocated hours reach their limit. Also, additional contingency hours can be planned into the maximum budget if considered helpful.
- **Graduated T&M.** Another alternative is a shared financial risk approach. In agile, the quality criteria are part of what "done" means. Therefore, the supplier can be rewarded with a higher hourly rate when delivery is earlier than the contracted deadline. Conversely, the supplier would suffer a rate reduction for late delivery.
- **Early cancellation option.** When an agile supplier delivers sufficient value with only half of the scope completed, the customer should not be bound to pay the remaining half if the customer no longer needs it. Instead, a contract can invite the customer to buy the remainder of the project for a cancellation fee. The customer limits budget exposure, and the supplier earns positive revenue for services no longer required.
- **Dynamic scope option.** For those contracts with a fixed budget, a supplier may offer the customer the option to vary the project scope at specified points in the project. The customer can adjust features to fit the capacity. Then, the customer can leverage innovation opportunities while limiting the supplier's risk of overcommitment.
- **Team augmentation.** Arguably, the most collaborative contracting approach is to embed the supplier's services directly into the customer organization. Funding teams instead of a specific scope can preserve the customer's strategic discretion on what work should actually be done.
- **Favor full-service suppliers.** To diversify risk, customers may seek a multisupplier strategy. However, the temptation may be there to contract the work so that each supplier does only one thing, which creates a web of dependencies before any usable service or product emerges. Instead, place an emphasis on engagements that deliver full value (as in the idea of completed independent feature sets).

It is possible to create agile contracts. Agile is built on a synergy of collaboration and trust. The supplier can help by delivering value early and often. The customer can help by providing timely feedback.

TIP

Culture Versus Structure

Some people insist that new organizational structures be installed before any cultural shift can begin. Others maintain the opposite—those new organizational structures are only superficial adjustments until the collective culture moves in a meaningful direction. In reality, one cannot progress without the other. Project leaders wanting to achieve agility should consider the current and future states of both these aspects in their organizations.

Appendix X2
Attributes That Influence Tailoring

Tailoring refers to the process of adapting agile frameworks, approaches, and/or practices to suit the specific needs of a team, project, or organization. The purpose is not to disregard core agile values, such as collaboration, adaptability, and delivering customer value, but to apply these values effectively within a given context.

Tailoring should be approached with discipline and experience. Changes to agile approaches are most effective when teams first have success using the approach as designed. Once they understand the intent behind specific activities, they can begin making adjustments that align with their environment without undermining agile principles.

TIP

A helpful metaphor is the Shu-Ha-Ri model of learning. Teams begin by following practices as prescribed (Shu), then experiment and adapt them (Ha), and eventually develop their own informed practices (Ri). Tailoring belongs in the Ha or Ri stage and should not be used as a shortcut to avoid discomfort or effort.

X2.1 Intentional Tailoring Versus Avoidance

A common mistake is using tailoring to avoid difficult conversations or uncomfortable practices. For example, if retrospectives are unproductive or unpopular, it may seem easier to stop doing them. However, the underlying problem may lie in facilitation or team dynamics—not in the value of the practice itself.

Effective tailoring identifies what is not working, explores the reasons, and tests a better approach. Dropping a practice without understanding its purpose or impact typically leads to more dysfunction, not less.

X2.2 How to Apply Tailoring

When tailoring, the following factors should be considered:

- Begin with a shared understanding of the current approach, activity, framework, method, or deliverable.
- Engage those affected by the proposed change in the discussions.
- Pilot the change using a short experiment timebox.
- Reflect on the outcome of the experiment during a retrospective.
- Institutionalize only the changes that demonstrably improve performance.

Successful tailoring requires both experimentation and reflection. Teams should revisit these decisions regularly to adapt to new challenges or shifting conditions.

It is also recommended that guidelines from Section 3 on life cycles and their selection be followed, which describe adopting (or tailoring) new approaches and common tailoring patterns and antipatterns. The PMI® Disciplined Agile® tool kit also provides curated options for tailoring processes and a framework for continuous improvement.

X2.3 Tailoring Based on Maturity and Operating Constraints

Organizational maturity, risk tolerance, and industry context all influence how agile practices are implemented. Some teams may operate in regulated environments or with legacy systems. Others may be in high-growth or early transformation stages. Tailoring should evolve based on delivery context.

Consider tailoring when the following occurs:

- **Work involves high risk or regulation.** Add quality checks or approvals while preserving fast feedback.
- **Teams depend heavily on one another.** Coordinate planning and align deliverables across groups.
- **The environment is highly dynamic.** Use shorter planning cycles and update backlogs more frequently.
- **Stakeholder needs vary.** Build structured checkpoints to maintain shared understanding.
- **Agile experience is uneven.** Introduce practices incrementally to support adoption and learning. Continuously check the level of maturity and decide corrective actions.

Tailoring helps teams build momentum while balancing alignment, autonomy, and accountability.

X2.3.1 Use the Tailoring Guidelines Table

Table X2-1 provides some common circumstances that may indicate opportunities for tailoring.

Table X2-1. Tailoring Guidelines

Situation	Tailoring Recommendation
Very large project teams	• Restructure large projects as multiple smaller projects. • Break large teams into multiple smaller teams and use program management to synchronize and coordinate the larger effort. • Consider a scaled agile or lean framework such as PMI® Disciplined Agile®, SAFe®, Nexus, or LeSS. Each offers some useful ideas, and each carries implementation risks and process weight/cost.
Partially remote or remote-first teams	• Use round-robin check-ins to help ensure participation and consensus for decisions. • Encourage all participants to turn on their cameras to allow for face-to-face conversations. • Use asynchronous communication methods like team chat boards and digital boards to keep team members in sync, and conduct sync touchpoints only two to three times a week.
Additional documentation and conformance checks for regulatory or compliance reasons	• Consider using a continuum of practices approach (multiple agile approaches) to get the benefits of improved collaboration and communication brought by agile when additional rigor is required. • Documentation or rigorous testing results could be part of what the team delivers, along with finished features. The DoD may include all documentation or required regulatory compliance being complete.
Stable requirements or execution process	• If the project has high rates of change during certain phases, such as design and development, and other phases are more defined and repeatable, such as rolling it out to customers, a continuum of practices approach that uses the appropriate life cycle model for each project phase may make more sense.
Teams are in functional silos inside functional organizations	• Experiment with cross-functional project delivery teams while maintaining HR or organizational structures to initially show the benefits.
Many of the team members have little technical domain knowledge	• Additional help assigning and directing may be necessary until the team gains the required skills. • Create an intentional focus and opportunity to learn during sprints or iterations to help provide guidance and build domain knowledge.
Lack of executive buy-in	• Find common areas for improvement based on the organization's needs, and then use experiments and retrospectives to move forward. • Explain agile in terms of Lean thinking: short cycles, small batch sizes, frequent reviews, and retrospectives with small improvements to help with education and understanding.

Tailoring decisions should be reviewed regularly to help ensure they continue to support transparency, flow, and value delivery.

X2.4 Guidelines for Balanced Tailoring

Tailoring should be intentional and outcome-oriented. Balanced tailoring should reflect the following:

- **Align with purpose.** Tailor practices to improve transparency, collaboration, and delivery speed.
- **Keep it simple.** Avoid overengineering roles, events, or documentation.
- **Adapt to real-world constraints.** Consider time zones, tooling, experience levels, and product complexity.
- **Inspect and adjust.** Review tailoring decisions regularly to help ensure continued relevance and impact.

Tailoring is not about loosening discipline. It is about making thoughtful adjustments so teams can work effectively in their environment, deliver customer value, and improve continuously.

Using AI to Tailor Agile Practices

AI can help teams make more informed decisions when tailoring agile practices to their environment. Rather than replacing team judgment, these tools offer support by surfacing trends, reducing manual overhead, and highlighting areas where change may improve outcomes.

The following examples demonstrate how AI can support tailoring decisions:

- **Surface delivery trends and flow issues.** AI can analyze throughput, cycle time, and blocker patterns, helping teams identify when to adjust work-in-progress (WIP) limits, planning cadences, or team structure.
- **Automate low-value or repetitive tasks.** Automating reporting, ticket updates, or routine documentation frees teams to focus on experimentation and continuous improvement.
- **Synthesize feedback for retrospectives.** AI tools that gather and summarize feedback from channels, tools, or surveys can help teams uncover friction points and tailor practices to improve collaboration and morale.
- **Analyze product usage and customer behavior.** Teams can use insights about how features are used or highlight trends in reported defects to reprioritize work and focus on delivering higher customer value.
- **Support asynchronous collaboration.** For distributed or hybrid teams, tools that generate meeting summaries or summarize long threads can make it easier to reduce the number of meetings and shift to more flexible collaboration practices.

These applications can improve visibility, reduce overhead, and give teams more capacity and clarity to tailor their approach with confidence.

X2.5 Appendix X2 References

Cornerstone Agility Team. (2023, July 11). *How to Tailor Your Agile Approach to Fit Your Unique Organizational Context*. Cornerstone Agility. https://www.cornerstoneagility.com/how-to-tailor-your-agile-approach-to-fit-your-unique-organizational-context/

Griffiths, M. (2007, June 20). *Agile Suitability Filters.* Leading Answers. https://leadinganswers.com/agile_suitabili/

Jeffries, R. (2006, May 2). *We Tried Baseball and It Didn't Work.* RonJeffries.com. https://ronjeffries.com/xprog/articles/jatbaseball/

Rothman, J. (2014, September 23). *One Experimental Possibility: Self-Organization From Component Teams to Feature Teams*. Rothman Consulting Group, Inc. https://www.jrothman.com/mpd/agile/2014/09/one-experimental-possibility-self-organization-from-component-teams-to-feature-teams/

Appendix X3
Agile Suitability Filter Tools

Agile literature contains many agile suitability filter tools to help assess the circumstances where an agile approach is appropriate. In 1994, the Dynamic Systems Development Method (DSDM) developed an Agile Project Suitability Questionnaire and an Organizational Suitability Questionnaire to help gauge likely fit and potential problem areas.

The Crystal family of approaches also employs suitability criteria, ranking projects by team size and the criticality of the product or service being developed. Crystal recommends that smaller, less critical projects be undertaken with lighter controls and simpler approaches. Large mission- or life-critical projects are recommended to use more rigor and validation.

Since the development of these approaches, many additional models have been created to help determine where and when to employ agile approaches. Boehm and Turner[9] adopted some of the elements from DSDM and Crystal to develop a popular assessment model to help determine if projects should be undertaken with agile or more traditional approaches.

Based on these previous models, and expanded to consider the middle ground of the continuum of development approaches, the following model is proposed. The model represents a synthesis of several suitability filter attributes to help organizations assess and discuss whether projects should be undertaken using predictive, hybrid (continuum), or agile approaches.

X3.1 Overview of the Model

Organizational and project attributes are assessed under the following three main categories:

- **Culture.** Is there a supportive environment with buy-in for the approach and trust in the team?
- **Team.** Is the team of a suitable size to be successful in adopting agile? Do its members have the necessary experience and access to business representatives to be successful?
- **Project.** Are there high rates of change? Is incremental delivery possible? How critical is the project?

[9] Boehm, B., & Turner, R. (2003). *Balancing agility and discipline: A guide for the perplexed*. Addison-Wesley Professional.

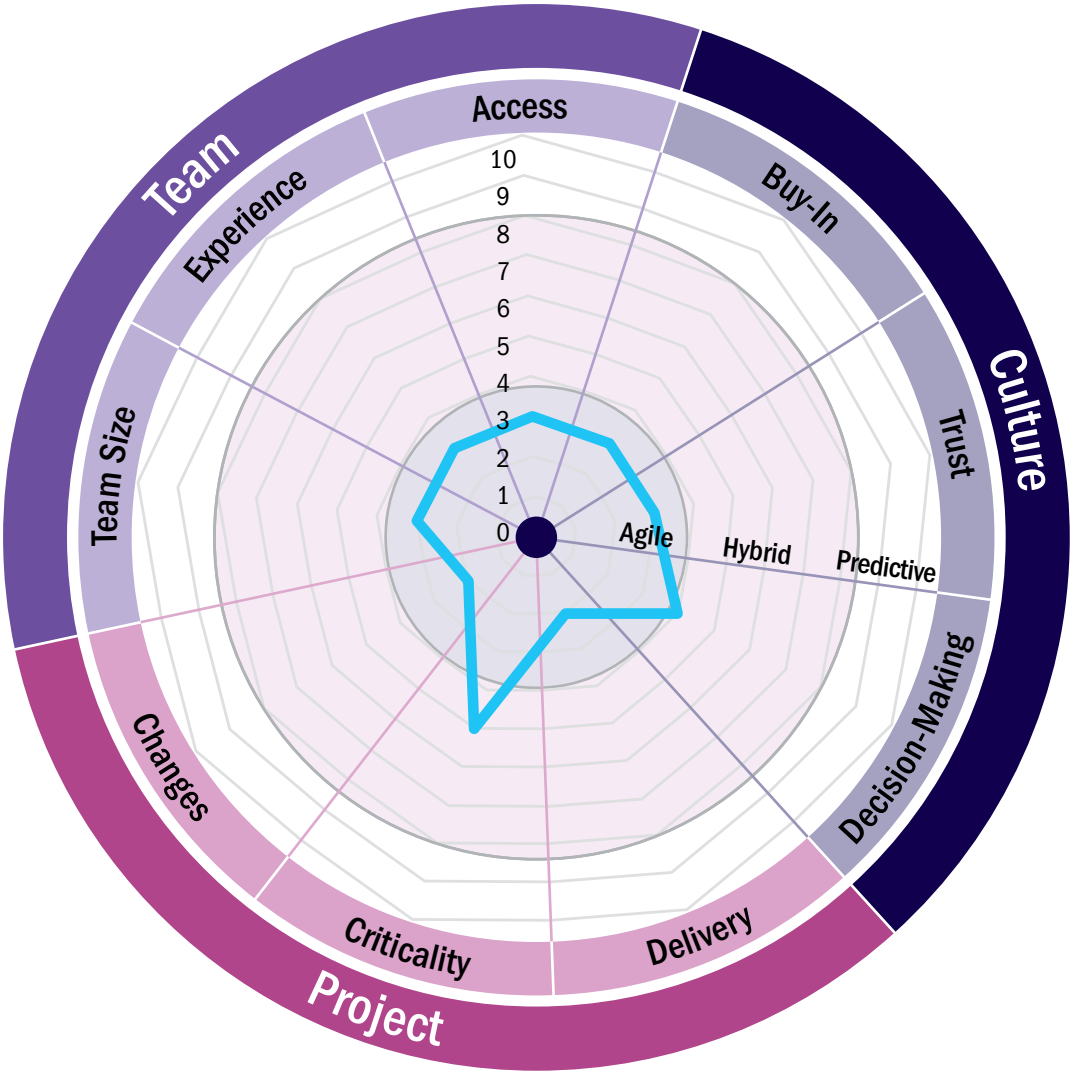


Figure X3-1. Model for Suitability of an Agile Approach

Questions in each of these categories are answered and the results plotted on a radar chart. Clusters of values around the center of the chart indicate a good fit for agile approaches. Results around the outside indicate a predictive approach may be more suitable. Values in the middle portion (between agile and predictive) indicate a hybrid (continuum) approach could work well. An example is shown in Figure X3-1.

X3.2 Instructions for Use

Access an online version of the tool at suitabilityfilter.com.

X3.2.1 Complete the Questionnaire as a Group

For small projects, this group may simply be the sponsor, technical lead, and a customer. For large projects, this may include representatives from the sponsoring group, project execution team,

impacted business group(s), project governance group(s), and the customer community. The idea is that just as multiple stakeholders should be involved in estimating and planning a project to avoid limited perspectives and personal biases, a diverse group of people should also assess the suitability of an approach to help ensure a comprehensive evaluation.

The value of the tool is the conversation it encourages with the invested parties of the project. Even if the results point to a hybrid approach but the stakeholders want to proceed with a largely agile or predictive approach, the team should follow the stakeholder consensus. This tool is a high-level diagnostic tool only; the final decision should rest with, and be supported by, the people involved.

X3.2.2 Score the Questions From 1 to 10

As a group, discuss and agree—or compromise—on a score that most accurately reflects the subjective evaluation of the question. While definitive options are only provided for the start, middle, and end points of the answer spectrum—representing scores of 1, 5, and 10—it is fine, even desirable, to use scores such as 2 for "almost a 1, but not quite," or 7 for "somewhere between a 5 and a 10." Again, the assessment is a discussion tool—views will be subjective, and shades of gray are to be expected.

When the group cannot agree on a score, discuss the issues openly and honestly. Before suggesting compromises, consider how successful the project is likely to be when the participants cannot agree on completing a simple assessment. When discussing the issues, if the differences of opinion can be identified, then it is working. At this point, everyone should be able to reach an agreement. Likewise, if the assessment indicates a predictive approach but everyone wants to try an agile approach (or vice versa), that is also fine; just understand the issues and discuss how impacts of the approach will be handled.

X3.2.3 Interpret the Results

Again, results clustered around the center in the agile zone indicate a good fit for a purely agile approach.

Results predominantly in the hybrid zone indicate that some combination of agile and predictive approaches might work best. However, it is also possible that an agile approach with some additional risk-reduction steps may suffice—such as extra education and training or extra validation and documentation rigor in the case of high-criticality projects. Alternatively, a predictive approach with some proof-of-concept work or extra processes may also work.

Results predominantly in the predictive zone indicate a good fit for a purely predictive approach. As mentioned in Section X3.2.2 (Score the Questions step), this diagnostic tool is aimed at starting meaningful conversations with the impacted parties about the most appropriate approach to use. If the approach suggested by the tool is not acceptable, a different approach is allowed. Use the results as inputs to the risk management process, since the tool indicates mismatches that may need to be managed.

X3.2.4 Case Studies

To illustrate how the radar chart works, two examples are provided, using the model to score very different types of projects. The first is an example of an online drugstore project (see Figure X3-2),

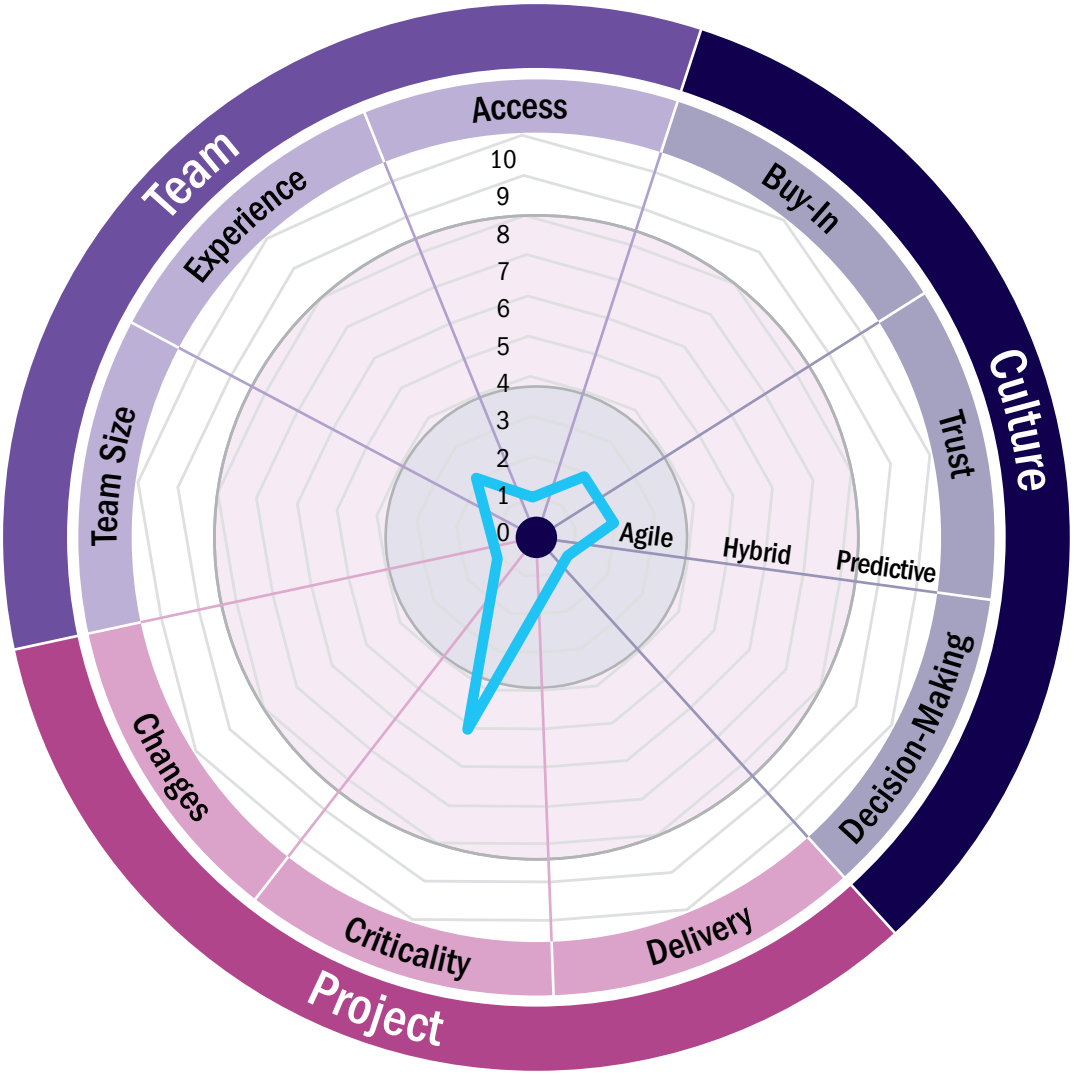


Figure X3-2. Drugstore Project

and the second is an example of a military messaging system (Figure X3-3). These two case studies illustrate some of the variances seen on projects. Central clustering indicates a good fit for agile approaches, whereas peripheral scores indicate that predictive approaches might be more suitable. Some projects are centered around the middle but then spike out on one or two axes. These projects may be best solved with a hybrid approach.

X3.2.4.1 Drugstore Example

The example project was initiated to develop an online drugstore to sell cheaper Canadian prescription drugs to primarily U.S. customers. The sale of these drugs is a contentious subject in Canada as well as the United States, and as a result, the industry is characterized by swift regulation changes and fierce competition. The project faced extremely volatile requirements with major changes week after week. The project also used very short (2-day) iterations and weekly releases to tackle the high rates of change.

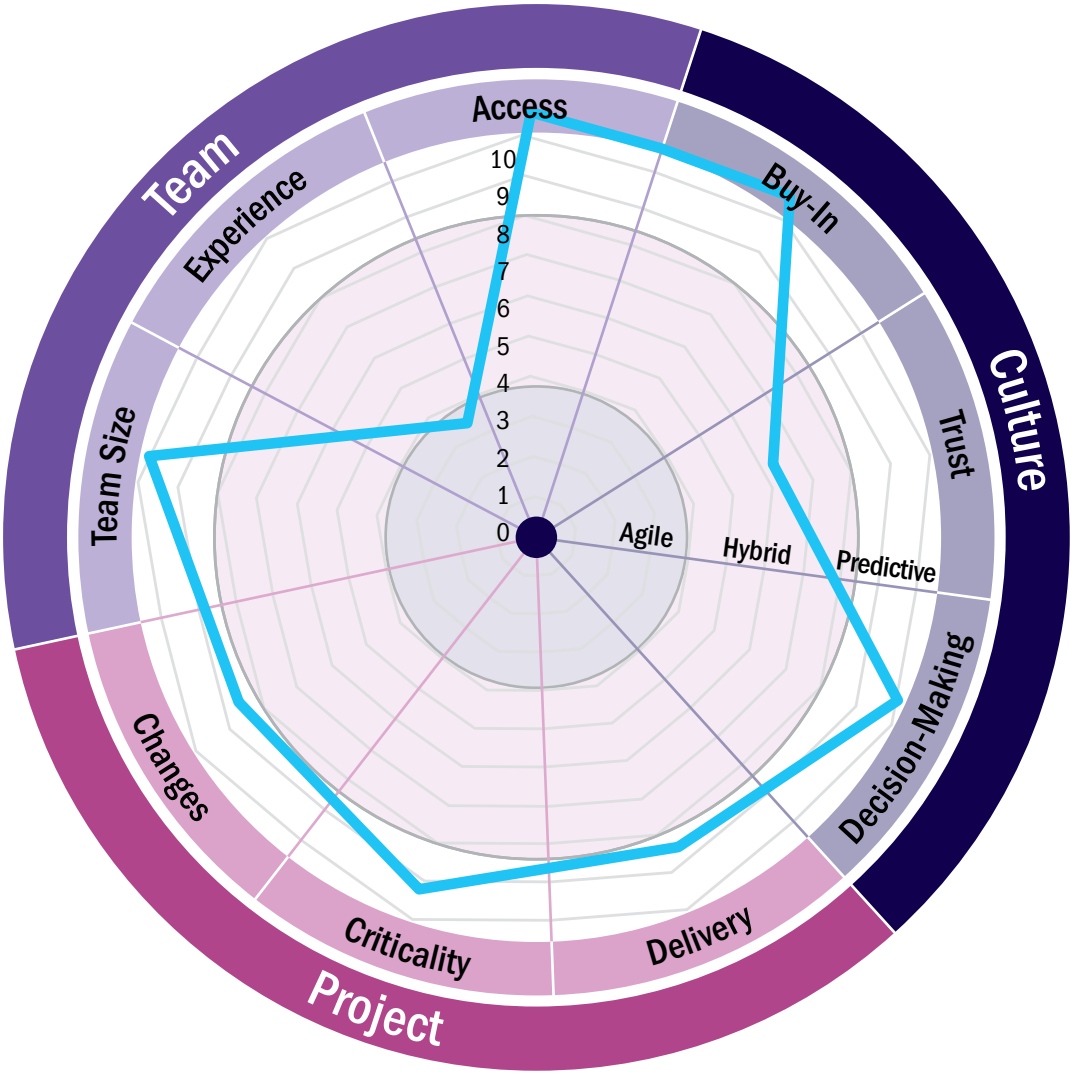


Figure X3-3. Military Messaging Example

As shown in Figure X3-2, high levels of buy-in and trust are evident for those who worked in an empowered way. The visual nature of the website made it easy to show new increments of functionality; but the system criticality was fairly high, with essential funds for the pharmacy at stake. As mentioned, there were very high rates of change, but the small, experienced team handled them well and had easy access to a knowledgeable business representative. The approach was very successful and extremely agile.

X3.2.4.2 Military Messaging System Example

In contrast to the first example, the second example is a large project to develop a military messaging system that had already been running for 5 years when the assessment was made (see Figure X3-3).

Buy-in for an agile approach was lacking because using an agile approach was not being considered. Trust in the vendors was mixed but generally respected. Decision-making was not local but instead

made by architecture and requirements committees. While elements of the design could be tested incrementally in a laboratory, they could not be gathered together for an end-to-end demonstration of functionality. Many lives were potentially at risk, so criticality was very high. Requirements were locked down because changes impacted so many subcontractor organizations.

The project was large, with more than 300 people from one vendor alone, but each role had many experienced practitioners. Finally, access to the business/customer was challenging. Contract analysts serving as proxies typically replied or asked clarifying questions within 10 days. This long feedback loop severely reduced adaptability, pushing the project toward a predictive approach. Parts of the project could have been carved off and run as agile projects, but at the heart of the initiative was a single large project.

X3.3 Summary

Agile suitability filters are useful tools for identifying potential fits and gaps for agile approaches. These filters should not be used as definitive inclusion or exclusion gates but instead as topics for objective discussion with all interested parties.

Appendix X4
Contributors and Reviewers

The following individuals had input into shaping the content of *Agile Practice Guide*—Second Edition. Inclusion of an individual's name in this list does not represent their approval or endorsement of the final content in all its parts.

X4.1 Contributors and Reviewers

Adnan Maqbool
Junaid Ahmad, PMI-ACP, PMI-RMP, PMP
Phill Akinwale, CSM, SPS, PMI-ACP
Mehdi Alibakhshi, DASSM, PMI-PBA, PMP
Victor Andrades, MBA
Onana André, PhD, PMI-SP, PMP
Viviane Arazi, DASSM, PMP, PgMP
Yannick Arekion
Sivaram Athmakuri, PMI-ACP, PMI-PBA, PMP
Akbar Azwir, PSM II, PgMP, PfMP
Sharaf Azzain
Ken Bainey, MBA, BETI, PMP
Gabriela Batista Castro, MSc, SCPM, PMP
Martial Bellec, PMI-ACP, PMP, PgMP
Yan Bello, CDBA, PMP
Greta Blash, PMI-ACP, PMI-PMOCP, PgMP
Cecilia Boggi, DBA, PMI-ACP, PMP
Wassim Bouaziz, MBA, PMI-ACP, PMP
Margareth Fabíola S. Carneiro, PhD, PMI Fellow
Panos Chatzipanos, PhD, BC.WRE, Dr Eur Ing
Auster Chen
Ivan Darmawan
Christina Diamantopoulou Tolia, MBA, PSM II, PMP
Louisa Dixon, MBA, PMP
Alberto Dominguez, PMI-ACP, DASSM, PMP
Montes Eduardo, PSM I, PMP
Haikal El Abed, Eng, PMI-ACP, PMP
Peter Elek, MBA, PMP
Mohamed Elfouly, PhD, PMI-RMP, PMP
Lucie Ellis, SHRM-SCP, A-CSM, PfMP
Dmitrii Evteev
Jesse Fewell, CEC, PMI-ACP, PMP
Ali Forouzesh, PhD, OPM3cc, PfMP
Teresa Foster, CAE
Luis Eduardo Franca, PMI-ACP, PMP
Carlos Augusto Freitas, MSc, DAC, PMP
Nestor Gabarda Jr., ECE, PMI-ACP, PMP
Vinu Ganesun
Juan Gabriel Gantiva Vergara, MSc, PMI-RMP, PMP
Oswaldo Jose Gil Simbolo
Theofanis Giotis, PhD, PMP, PfMP
Scott M. Graffius, PMP
Akram Hassan, PMP
Myo Min Htoo, BEng, PMP, PgMP
Trikesh Kumar Hurchurn, PMI-ACP, PMI-PMOCP, PMP
Maaz Ibrahim
Shuichi Ikeda, CBAP, CSM, PMP
Yanis Ioualitene
M. Athar Januar, PMI-ACP, PMI-RMP, PMP
Favrot Jean-Luc, SPC, PMI-ACP, PMP
Palomino Jorge, MBA, PMI-ACP, PMP
Rami Kaibni, CBAP, PMP, PgMP
Ravi Kalluri, CSM, PMP
Betsy Kauffman
Krishnakumar Kesavan, SA, PMP
Azmathulla Khan, Hon. DB Counsel, PMP, PgMP
Evgeny Khotulev, BA, MA, PMP
Toshifumi Kimura, CSP-SM, PMI-ACP, PMP
Aboozar Kordi
Jiri Kratky, PMI-ACP, PMP

Elton Kuzniewski, PMI-ACP, PMI-PMOCP, PMP
Jorge Lamadrid Guerrero, PMI-ACP, PMI-PBA, PMP
Chris Lawson, EMBA, PMI-PMOCP, PgMP
Laura Lazzerini Neuwirth, AgilePgM, PMI-PMOCP, PMP
Enrique Ledesma, PMP
Chia Kuang Lee, PhD, CQRM, PMP
Islam Mahmoud, PMI-ACP, PMI-CP, PMP
Douglas Martin, CSP, PMI-ACP, PMP
Dora Luz Mejia, PMI-ACP, PMP, PgMP
Duane Mengo, PMI-ACP, PMI-PBA, PMP
Kunihiko Mishima
Abhishek Mishra, MBA, PMP
Adrian Mladin
Hiromi Nakatani, PMI-ACP, PMI-PMOCP, PMP
David Newman, GPM-b, PMI-ACP, PMP
Nguyen Si Trieu Chau, PMP, PgMP, PfMP
Nguyen Tuan Le Giang, PMP, PgMP, PfMP
Mai Nishio, PMP
Babasola Adeboye Osibo
Thomas Owens
Emre Ozmen, PhD, PMP
Anand Britto Packiam, PMI-ACP
Maillard Patrick, MBA, PMP
Emma Perennes, MA, PMI-ACP, PMP
Maria Paula Perides, PhD, PMP
Luigi Punzo, MBA, CISA, PMP
Zulfiqar Ali QaimKhani, PMI-ACP, PMP, PfMP
Hossein Rahmatjou
Syed Ali Raza, PMI-RMP, PMP
Tashfeen Riaz, DAVSC, PMP, PgMP
Diane Roberts, PMP
Matheus Rocha, PMI-PMOCP, PMP
Carlos Ericson Rodriguez Castro, PMI-ACP, PMI-PMOCP, PMP
Vugar Rustamli, PMI-ACP, PMP
Edgardo Sergio Safranchik, MSc, PMI-ACP, PMP
Yishai Sandak, MSc, PMI-ACP, PMP
Danny Seow Wei Jie, PMI-PBA, PMP, PgMP
Courtney Shar
Denys Shcherba
Najab Siddiqui, PMP
Giorgos Sioutzos, MSc, CBAP, PMI-PBA
Joseph Sisto, MA, PMI-ACP, PMP
Anca Slusanschi, MSc, CMInstD, PMP
Craig Smith
Cynthia Snyder Dionisio, MBA, PMI-RMP, PMP
Roël Soeleman
Mauro Sotille, PMI-RMP, PMI-PMOCP, PMP
Fernando Souza, MSc, PMI-ACP, PMP
Sreeshaj Sreedhar, PSM I, PMI-ACP, PMP
Chandramouli Subramanian, PhD, PMI-ACP, PfMP
Awadalsaid Tara MSc, PMI-ACP, PMP
Christian Tavera, MBA, PMP, PgMP
Laurent Thomas, PhD, SPC, PMP
Yassine Tounsi, PMI-ACP, PMI-CP, PMP
Meiko Tourista, CCMP, Prosci CCP, PMI-PMOCP
Cristina-Elena Ungureanu, ITIL v4 MP, PMP
Tony Van Krieken, DASM, PMI-ACP, PMP
Alvaro Vargas, PMP
Luis Fernando Vitteri Quiros
Catherine Watson
Jack Wu, PMI-ACP, PMP
Priscila Z Vendramini Mezzena, CSM, PMI-ACP, PMP
Alan Zucker, DAC, PMI-ACP, PMP

X4.2 PMI Team Members

Mike Griffiths, PMI-ACP, PMP
Kelly Heuer, PhD, CAPM
Christie McDevitt, APR
Sarah Philbrick
Lenka Pincot, PMI-ACP, PMP, PfMP
Americo Pinto
Scott Shemo
Kim Shinners

References

[1] Project Management Institute (PMI). (2021). *A Guide to the Project Management Body of Knowledge (PMBOK® Guide)*—Seventh edition. PMI.

[2] Project Management Institute (PMI). (2025). *A Guide to the Project Management Body of Knowledge (PMBOK® Guide)*—Eighth edition. PMI.

[3] Barrero, J. M., Bloom, N., & Davis, S. J. (2021). *Why Working From Home Will Stick* (NBER Working Paper No. 28731). National Bureau of Economic Research. https://www.wfhresearch.com

[4] Project Management Institute (PMI). (2024). *AI Essentials for Project Professionals*. PMI. https://www.pmi.org/standards/ai-essentials-for-project-professionals

[5] Project Management Institute (PMI). (2026). *Manifesto for Enterprise Agility*. PMI.

[6] Project Management Institute (PMI). (2013). *Managing Change in Organizations: A Practice Guide*. PMI.

[7] Beck, K., Beedle, M., van Bennekum, A., Cockburn, A., Cunningham, W., Fowler, M., Grenning, J., Highsmith, J., Hunt, A., Jeffries, R., Kern, J., Marick, B., Martin, R. C., Mellor, S., Schwaber, K., Sutherland, J., & Thomas, D. (2001). *Manifesto for Agile Software Development*. https://agilemanifesto.org.

[8] Sidky, A. (2025). *Ahmed Sidky*. https://www.ahmedsidky.com/

[9] Drucker, P. F. (1959). *The Landmarks of Tomorrow*. Harper & Brothers.

[10] Drucker, P. F. (1966). *The Effective Executive: The Definitive Guide to Getting the Right Things Done*. Harper & Row.

[11] Drucker, P. F. (1999). *Management Challenges for the 21st Century*. Harper Business.

[12] Edmondson, A. C. (1999). Psychological Safety and Learning Behavior in Work Teams. *Administrative Science Quarterly*, *44*(2), 350–383.

[13] Clark, T. R. (2020). *The 4 Stages of Psychological Safety: Defining the Path to Inclusion and Innovation.* Berrett-Koehler Publishers.

[14] Helfand, H. (2020). *Dynamic Reteaming: The Art and Wisdom of Changing Teams*. O'Reilly Media.

Bibliography

Agile Alliance (n.d.). *Women in Agile Initiative.* https://agilealliance.org/resources/initiatives/women-in-agile-initiative/

Agile Business Consortium. (2022). *Remote Working in an Agile Team: A White Paper by the Agile Research Network*. https://www.agilebusiness.org/resource-report/white-paper-remote-working-in-an-agile-team.html

Anderson, D. J. (2010). *Kanban: Successful Evolutionary Change for Your Technology Business*. Blue Hole Press.

Asana. (2025, January 25). *Team Structures: Types and Best Practices*. https://asana.com/resources/team-structure

Basili, V. R., Caldiera, G., & Rombach, H. D. (1994). *The Goal Question Metric Approach.* Institute for Advanced Computer Studies, University of Maryland. cs.umd.edu

Blank, S. (2005). *The Four Steps to the Epiphany: Successful Strategies for Products That Win*. K&S Ranch.

Blank, S. (2013, May). Why the Lean Start-Up Changes Everything. *Harvard Business Review*. https://hbr.org/2013/05/why-the-lean-start-up-changes-everything

Brown, T. (2009). *Change by Design: How Design Thinking Transforms Organizations and Inspires Innovation*. Harper Business.

Cagan, M. (2018). *Inspired: How to Create Tech Products Customers Love* (2nd ed.). Ascent Audio.

Cohn, M. (2024). *Don't Estimate the Sprint Backlog Using Task Points*. Mountain Goat Software Blog. https://www.mountaingoatsoftware.com/blog/dont-estimate-the-sprint-backlog-using-task-points

Croll, A., & Yoskovitz, B. (2013). *Lean Analytics: Use Data to Build a Better Startup Faster*. O'Reilly Media.

Doerr, J. (2018). *Measure What Matters: How Google, Bono, and the Gates Foundation Rock the World with OKRs*. Portfolio/Penguin.

Duhigg, C. (2016, February 28). What Google Learned From Its Quest to Build the Perfect Team. *The New York Times Magazine*.

Edmondson, A. C. (2018*). The Fearless Organization: Creating Psychological Safety in the Workplace for Learning, Innovation, and Growth*. Wiley.

Edmondson, A. C., & Lei, Z. (2014). Psychological Safety: The History, Renaissance, and Future of an Interpersonal Construct. *Annual Review of Organizational Psychology and Organizational Behavior, 1*(1), 23–43.

Forsgren, N., Humble, J., & Kim, G. (2018) *Accelerate: The Science of Lean Software and DevOps*. IT Revolution Press.

Google re:Work. (2015). *Guide: Understand Team Effectiveness (Project Aristotle)*. Google LLC. https://rework.withgoogle.com/intl/en/guides/understanding-team-effectiveness

Google re:Work. (n.d.). *Set Goals with OKRs*. Google re:Work Guide. rework.withgoogle.com

Gothelf, J. (2020, December 17). Use OKRs to Set Goals for Teams, Not Individuals. *Harvard Business Review*, https://hbr.org/2020/12/use-okrs-to-set-goals-for-teams-not-individuals

Gren, L., & Pettersson, N. (2024). *Investigating Team Maturity in an Agile Automotive Reorganization* [Preprint]. arXiv. https://arxiv.org/abs/2409.11781

Hyper Drive. (2020, September 22). *Agile CoE vs. Agile CoP: Building Your Internal Coaching Platform*. https://hyperdriveagile.com/articles/agile-co-e-vs-agile-co-p-building-your-internal-coaching-platform-6

IDEO.org. (2015, June). *The Field Guide to Human-Centered Design*. IDEO.org. https://www.ideo.com/journal/design-kit-the-human-centered-design-toolkit

IT Revolution. (2023, February 28). *Four Fundamental Team Topologies*. https://itrevolution.com/articles/four-team-types/

Jeffries, R. (2019, May 23). *Story Points Revisited*. RonJeffries.com. https://ronjeffries.com/articles/019-01ff/story-points/Index.html

Kamani, V. (2025, January 13). *7 Agile Metrics That Matter for Successful Project Management*. Arkenea blog. https://arkenea.com/

Kumospace. (2024, March 22). *Virtual Teams: Examples & Benefits.* https://www.kumospace.com/blog/virtual-teams-examples

Learning Loop. (n.d.). *Team Topology.* https://learningloop.io/glossary/team-topology

LinkedIn Talent Solutions. (n.d.). *How to Engage and Retain Remote Teams*. https://business.linkedin.com/talent-solutions/resources/talent-engagement/top-employee-retention-strategies

LogRocket. (2024, March 21). *An Introduction to Dynamic Reteaming*. https://blog.logrocket.com/product-management/introduction-to-dynamic-reteaming/

Magennis, T. (2011). *Forecasting and Simulating Software Development Projects: Effective Modeling of Kanban & Scrum Projects Using Monte Carlo Simulation*. CreateSpace.

Maurya, A. (2012). *Running Lean: Iterate from Plan A to a Plan That Works* (2nd ed.). O'Reilly Media.

Plattner, H., Meinel, C., & Leifer, L. (Eds.). (2011). *Design Thinking: Understand – Improve – Apply*. Springer.

PMI® Disciplined Agile®. (n.d.). *Organizing Agile Teams at Scale*. https://www.pmi.org/disciplined-agile/agility-at-scale/tactical-agility-at-scale/large-agile-teams/organizing-agile-teams

PMI® Disciplined Agile®. (n.d.). *Practice: Facilitating Remote Teams*. https://www.pmi.org/disciplined-agile/team-lead/practice-facilitating-remote-teams

PMI® Disciplined Agile®. (n.d.). *Value Streams*. https://www.pmi.org/disciplined-agile/process/value-streams

Project Management Institute (PMI). (2017). *Navigating Complexity*. PMI.

PsychSafety.com. (2024, March 28). *Google's Project Aristotle and Psychological Safety* [Blog post]. https://psychsafety.com/googles-project-aristotle/

Ries, E. (2011). *The Lean Startup: How Today's Entrepreneurs Use Continuous Innovation to Create Radically Successful Businesses*. Crown Business.

Staats, J. (n.d.). *Dynamic Reteaming: How We Thrive by Rebuilding Teams* [Experience report]. Agile Alliance. https://agilealliance.org/resources/experience-reports/dynamic-reteaming-how-we-thrive-by-rebuilding-teams/

Wellable. (2023). *Hybrid, In-Person, or Fully Remote?* https://www.wellable.co/blog/hybrid-in-person-or-fully-remote/

Wodtke, C. (2016). *Radical Focus: Achieving Your Most Important Goals With Objectives and Key Results*. Boxes and Arrows Press.

Glossary

acceptance-test-driven development (ATDD). A method of collaboratively creating acceptance test criteria that are used to create acceptance tests before delivery begins.

adaptive approach. A development approach in which the requirements are subject to a high level of uncertainty and volatility and are likely to change throughout the project.

agile. A term used to describe a mindset of values and principles as set forth in the *Manifesto for Agile Software Development*.

agile approach. A development approach that applies the mindset, values, and principles of the *Manifesto for Agile Software Development (Agile Manifesto)* through practices that emphasize collaboration, adaptability, and frequent delivery of value.

agile coach. An individual with knowledge and experience in agile who can train, mentor, and guide organizations and teams through their transformation.

agile life cycle. An approach that is both iterative and incremental to refine work items and deliver frequently.

***Agile Manifesto*.** The original and official definition of agile values and principles.

agile mindset. A way of thinking and behaving underpinned by the four values and 12 principles of the *Agile Manifesto*.

agile practitioner. A person embracing the agile mindset who collaborates with like-minded colleagues in cross-functional teams.

agile principles. The 12 principles of agile project delivery as embodied in the *Agile Manifesto.*

Agile Suitability Filter. A structured decision aid used to assess whether a project or initiative is well suited for agile approaches based on factors such as uncertainty, complexity, and stakeholder involvement.

antipattern. A known, flawed mode of work that is not advisable.

artificial intelligence (AI). The programming of machines with patterns and processes similar to those observed in—and by—humans and human interactions.

backlog. An ordered list of work to be done, often written as user stories, and prioritized by the business to manage and organize an adaptive or agile project's work. See also *product backlog*.

backlog planning. An activity in which the team, directed by the product owner, reviews and prioritizes items in the backlog to prepare for upcoming iterations or releases.

backlog refinement. Progressive elaboration of content in the backlog and the prioritization and reprioritization of it to identify the work that can be accomplished in an upcoming iteration.

behavior-driven development (BDD). A system design and validation practice that uses test-first principles and plain-language scripts.

blocker. See *impediment*.

burnup chart. A graphical representation of the work completed toward a milestone.

cadence. A rhythm of activities conducted throughout the project. See also *timebox*.

center of excellence (CoE). An organizational team or function that provides leadership, recommended practices, support, and training for a specific discipline across the enterprise.

collective ownership. A project acceleration and collaboration technique whereby any team member is authorized to modify any project work product or deliverable, thus emphasizing team-wide ownership and accountability.

colocation. An organizational placement strategy where the project team members are physically located close to one another to improve communication, working relationships, and productivity.

community of practice (CoP). A group of people who share a common interest in a discipline or topic and who collaborate to exchange knowledge, improve skills, and develop practices.

continuous delivery. The practice of delivering feature increments immediately to customers, often through the use of small batches of work and automation technology.

continuous integration. A practice in which team members' work products are frequently comingled and validated with one another.

continuum of approaches. Often referred to as hybrid life cycles, a continuum of life cycle approaches mixes components of predictive approaches and elements of iterative/incremental agile approaches. The specific mix of approaches varies per implementation, so there is a variety of attributes a continuum approach may exhibit.

cross-functional team. A team that includes practitioners with all the skills necessary to deliver valuable product increments.

Crystal family of methods. A collection of lightweight agile software development methods focused on adaptability to a particular circumstance.

cycle time. The total elapsed time from the start of a particular activity or work item to its completion.

Cynefin Framework. A sensemaking framework that helps leaders and teams categorize situations into domains (clear, complicated, complex, chaotic, and confused) to decide on appropriate actions.

daily coordination meeting. A brief, daily collaboration meeting in which the team reviews progress from the previous day, declares intentions for the current day, and highlights any obstacles encountered or anticipated. (Previously, this was commonly referred to as a standup meeting.)

daily scrum. A brief, daily collaboration meeting in which the team reviews progress from the previous day, declares intentions for the current day, and highlights any obstacles encountered or anticipated.

daily sync. See *daily scrum*.

definition of done (DoD). A checklist of all the criteria required to be met before a deliverable can be considered ready for customer use.

definition of ready (DoR). A checklist for a user-centric requirement that has all the information the team needs to begin working on it.

demonstrations (demos). Events in which a team presents completed work to stakeholders to obtain feedback and validate progress.

design thinking. A nonlinear, iterative process that teams use to understand users, challenge assumptions, redefine problems, and create innovative solutions to prototype and test.

development approach. A method used to create and evolve the product, service, or result during the project life cycle such as an adaptive, predictive, or hybrid approach.

DevOps. A collection of practices for creating a smooth flow of delivery by improving collaboration between development and operations staff.

DevOps Research and Assessment (DORA) metrics. A set of widely used measures—deployment frequency, lead time for changes, change failure rate, and time to restore service—that assess software delivery performance.

DevSecOps. An extension of DevOps practices that integrates security activities into every stage of the development and operations pipeline.

Dynamic Systems Development Method (DSDM). An agile project delivery approach.

enterprise agility. The ability of an organization to quickly sense and respond to change across strategy, structures, processes, and people, extending agile principles beyond software and individual teams.

Extreme Programming (XP). An agile software development method that leads to higher quality software, greater responsiveness to changing customer requirements, and more frequent releases in shorter cycles.

feature-driven development (FDD). A lightweight agile software development method generated by the perspective of elements that clients value.

flow. The measure of how efficiently work moves through a given process or framework.

flow metrics. Quantitative measures such as throughput, work in process, and cycle time that indicate how efficiently work items move through a process.

framework. A basic system or structure of ideas or facts that support an approach.

generalizing specialist. A person who combines deep expertise in one area with broad skills and the willingness to learn across multiple domains to support team needs.

goal question metric (GQM). A measurement approach in which goals are defined, questions are formulated to determine achievement of those goals, and metrics are identified to provide the required data.

hybrid approach. A combination of elements from both adaptive and predictive approaches that is useful when there is uncertainty or risk around the requirements.

I-shaped. A person with a single deep area of specialization and no interest or skill in the rest of the skills required by the team. See also *T-shaped.*

impact mapping. A strategic planning technique that acts as a roadmap to the organization while building new products.

impediment. An obstacle that prevents the team from achieving its objectives. Also known as a *blocker*.

increment. A functional, tested, and accepted deliverable that is a subset of the overall project outcome.

incremental approach. An adaptive development approach in which the deliverable is produced successively, adding functionality until the deliverable contains the necessary and sufficient capability to be considered complete.

incremental life cycle. An approach that provides finished deliverables that the customer may be able to use immediately.

information radiator. A visible, physical display that provides information to the rest of the organization, enabling timely knowledge sharing.

iteration. A short cycle of development during which a product or deliverable is released or further matured. See also *sprint* and *timebox*.

iteration planning. A meeting to clarify the details of the backlog items, acceptance criteria, and work effort required to meet an upcoming iteration commitment. See also *sprint planning.*

iterative approach. A development approach that focuses on an initial, simplified implementation then progressively elaborates, adding to the feature set until the final deliverable is complete.

. **kaizen events.** Events aimed at improvement of the system

.

kanban board. A visualization tool that shows work in process to help identify bottlenecks and overcommitments, thereby allowing the team to optimize the workflow.

Kanban Method. An agile approach inspired by the original Kanban inventory control system and used specifically for knowledge work.

Large Scale Scrum (LeSS). A product development framework that extends Scrum with scaling guidelines while preserving the original purposes of Scrum.

lead time. The total elapsed time between a customer's request and the delivery of a usable product or feature that fulfills that request.

Lean Software Development. An adaptation of Lean manufacturing principles and practices to the software development domain and is based on a set of principles and practices for achieving quality, speed, and customer alignment.

Lean Startup. An entrepreneurial approach that emphasizes building minimal viable products (MVPs), testing assumptions with customers quickly, and using validated learning to guide product development.

life cycle. The process through which a product is imagined, created, and put into use.

low-code/no-code platforms (LCNC). Tools that enable applications to be built primarily through configuration and visual modeling rather than extensive hand-coding, thereby allowing faster delivery with appropriate governance for quality, security, and maintainability.

minimum business increment (MBI). The smallest deliverable piece of functionality that has value to the business. The MBI helps the organization focus on realizing that value quickly.

minimum marketable product (MMP). Focused on release, this is the smallest version of a product with enough features to be offered to the market and deliver value customers will pay for or adopt. The goal is to capture market value quickly.

minimum viable product (MVP). A concept used to define the scope of the first release of a solution to customers by identifying the fewest number of features or requirements that would deliver value. Focused on learning, the MVP is the smallest version of a product that allows a team to validate assumptions and gain feedback from real users. The goal is to reduce uncertainty about whether the product is valuable or usable.

mobbing. A technique in which multiple team members focus simultaneously and coordinate their contributions on a particular work item.

modeling. Creating simplified representations of systems, solutions, or deliverables such as prototypes, diagrams, or storyboards.

Monte Carlo analysis/simulation. A method of identifying the potential impacts of risk and uncertainty using multiple iterations of a computer model to develop a probability distribution of a range of outcomes that could result from a decision or course of action.

Net Promoter Score℠ (NPS®).[10] An index that measures the willingness of customers to recommend an organization's products or services to others.

objectives and key results (OKRs). A goal-setting framework used to define and track objectives and their outcomes. OKRs are typically used in business settings to align individual, team, and organizational goals with measurable results.

organizational change management (OCM). A comprehensive, cyclic, and structured approach for transitioning individuals, groups, and organizations from the current state to a future state with intended business benefits.

pairing. A technique of partnering two team members to work simultaneously on the same work item.

pair programming. Pairing work that is focused on programming.

pivot. A planned course correction designed to test a new hypothesis about the product or strategy.

PMI® Disciplined Agile® (DA®). A process-decision framework that enables simplified process decisions around incremental and iterative solution delivery.

portfolio Kanban. A visualization tool for managing the flow of initiatives or projects at the portfolio level, enabling prioritization, tracking, and the balancing of demand against capacity.

predictive approach. A development approach in which the project scope, time, and cost are determined in the early phases of the life cycle.

product. An artifact that is produced, is quantifiable, and can be either an end item or a component item.

product backlog. An ordered list of user-centric requirements that a team maintains for a product. See also *backlog*.

product life cycle. A series of phases that represent the evolution of a product throughout its creation and delivery, from conception through growth, maturity, and decline/retirement.

product management. The integration of people, data, processes, and business systems to create, maintain, and evolve a product or service throughout its life cycle.

product owner. A person responsible for maximizing the value of the product and accountable for the end product.

product team. A cross-functional group of individuals responsible for defining, building, and evolving a product to deliver value to customers and stakeholders. Typically formed for the duration of the product life cycle.

[10] Net Promoter®, NPS®, NPS Prism®, and the NPS-related emoticons are registered trademarks of Bain & Company, Inc., NICE Systems, Inc., and Fred Reichheld. Net Promoter Score℠ and Net Promoter System℠ are service marks of Bain & Company, Inc., NICE Systems, Inc., and Fred Reichheld.

progressive elaboration. The iterative process of increasing the level of detail in a project management plan as greater amounts of information and more accurate estimates become available. See also *rolling wave planning*.

project charter. A document issued by the project initiator or sponsor that formally authorizes the existence of a project and provides the project manager with the authority to apply organizational resources to project activities.

project management office (PMO). An organizational entity typically established as a department or team and primarily tasked with centralizing activities related to the management of portfolios, programs, and/or projects. The nature of these activities can vary according to the unique needs of each organization.

project team. A set of individuals performing the work of the project to achieve its objectives.

psychological safety. A shared belief within a team that members will be accepted, can speak up, take risks, and express ideas or concerns without fear of negative consequences.

RACI (responsible, accountable, consulted, informed) matrix. A type of responsibility assignment matrix that uses responsible, accountable, consulted, and informed statuses to define the involvement of stakeholders in project activities.

refactoring. A product-quality technique whereby the design of a product is improved by enhancing its maintainability and other desired attributes without altering its expected behavior.

Responsible, accountable, facilitator (RAF) model. A responsibility assignment model that clarifies who is responsible for doing the work, who is accountable for the outcome, and who facilitates the process.

retrospectives. Regularly occurring workshops in which participants explore their work and results to improve both the process and product.

review. A meeting held at the end of an iteration to demonstrate the work that was accomplished during the iteration. See also *sprint review*.

rework. Action taken to bring a defective or nonconforming component into compliance with requirements or specifications.

rolling wave planning. An iterative planning technique in which the work to be accomplished in the near term is planned in detail, while the work in the future is planned at a higher level. See also *progressive elaboration*.

Scaled Agile Framework (SAFe®). A knowledge base of integrated patterns for enterprise-scale lean–agile development.

Scrum. An agile framework for developing and sustaining complex products, with specific roles, events, and artifacts.

Scrum@Scale. A framework that extends Scrum principles to multiple teams by using a scaled structure of roles, artifacts, and events to coordinate work across an organization.

scrum master. The coach of the development team and process owner in the Scrum framework who removes obstacles, facilitates productive events, and defends the team from disruptions.

Scrum of Scrums. A technique to operate Scrum at scale for multiple teams working on the same product, coordinating discussions of progress on their interdependencies, and focusing on how to integrate the delivery of software, especially in areas of overlap.

Scrum team. The combination of the development team, scrum master, and product owner used in Scrum.

self-organizing team. A cross-functional team in which people fluidly assume leadership as needed to achieve the team's objectives.

servant leadership. The practice of leading the team by focusing on understanding and addressing the needs and development of team members to enable the highest possible team performance.

smoke testing. The practice of using a lightweight set of tests to help ensure that the most important functions of the system under development work as intended.

special interest group (SIG). A community of individuals who share a focus on a specific topic or area of practice and collaborate to deepen knowledge and share experiences.

spike. A timeboxed user story or task that is created to research a question or resolve a problem.

sprint. A timeboxed interval within a project during which a usable and potentially releasable increment of a product is created. See also *iteration* and *timebox*.

sprint backlog. A list of work items identified by the Scrum team to be completed during the Scrum sprint.

sprint planning. A collaborative event in Scrum in which the Scrum team plans the work for the current sprint. See also *iteration planning*.

sprint retrospective. See *retrospective*.

sprint review. A collaborative review session where the team demonstrates the work that was completed during the sprint to stakeholders and solicits their feedback. See also *review*.

Stacey matrix. A framework that categorizes work by the degree of certainty of requirements and agreement among stakeholders, often used to identify where agile approaches are most appropriate.

story mapping. A visual practice for organizing work into a useful model to help understand the sets of high-value features to be created over time, identify omissions in the backlog, and effectively plan releases that deliver value to users.

story point. A unit used to estimate the relative level of effort needed to implement a user story.

sustainability. The practice of delivering outcomes at a pace and in a manner that can be maintained over time, balancing productivity with respect for people, materials, and the environment. This effort includes supporting team well-being and minimizing negative environmental impacts so that work remains viable for future generations.

swarming. A technique in which multiple team members focus collectively on resolving a specific impediment.

T-shaped. Refers to a person with one deep area of specialization and broad ability in the rest of the skills required by the team. See also *I-shaped*.

team charter. A document that establishes the values, agreements, and operating guidelines for the team.

technical debt. The deferred cost of work not done at an earlier point in the product life cycle.

test-driven development (TDD). A technique where tests are defined before work is begun so that work in process is validated continuously, enabling work with a zero-defect mindset.

throughput. The number of items passing through a process.

timebox. A short, fixed period of time in which work is to be completed. See also *cadence*, *iteration,* and *sprint*.

user experience (UX). The process of enhancing the user experience by focusing on improving the usability and accessibility to be found in the interaction between the user and the product.

user story. A brief description of an outcome for a specific user, which is a promise for a conversation to clarify details.

value stream. An organizational construct that focuses on the flow of value to customers through the delivery of specific products or services.

value stream map. A display of the critical steps in a process and the time taken in each step used to identify waste.

velocity. A measure of a team's productivity rate at which the deliverables are produced, validated, and accepted within a predefined interval.

weighted shortest job first (WSJF). A prioritization method that ranks work by dividing its cost of delay by its job size, favoring items that deliver the highest value in the shortest time.

WIP limits (work-in-process/work-in-progress limits). A constraint on the maximum number of work items allowed to be active (started but not finished) in a workflow stage or system at one time.

Work in process/progress (WIP). The project tasks or activities that are in process/progress but have not yet been completed.

working agreements. Explicit, shared guidelines created by a team that describe how members will collaborate, communicate, and make decisions together. Working agreements support accountability, alignment, and effective interactions within the team.

Index

A

B

C

D

E

F

G

H

I

J

K

L

M

N

O

P

Q

R

S

T

U

V

W

X