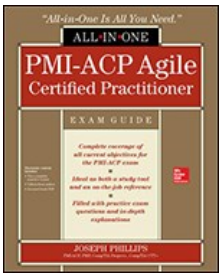

PMI-ACP Agile Certified Practitioner All-in-One Exam Guide

by Joseph Phillips

Chapter 1: Agile Principles and the PMI-ACP Mindset

Overview

In this chapter, you will

- Define the nine exam tasks of agile principles
- Explore the agile values for projects
- Define leadership qualities for organizational agility
- Learn the Agile Manifesto
- Compare agile project management approaches
- Review the qualities of agile project leadership

This is a book on how to pass, not just take, the PMI-ACP exam. In each chapter I'll dive into the exam specifics you'll need to know to clear that specific exam domain. This chapter is all about the agile principles and embracing the agile mindset. The PMI-ACP exam is more than just recalling facts about agile, and it'll test your ability to review a scenario-based question and then choose the best course of action—often based on the agile mindset.

You will need a strong foundation of agile principles—and if you've worked on agile projects, you likely have an understanding of how agile projects operate. I don't believe that the PMI-ACP exam is overwhelmingly difficult, but it can be tricky. Agile projects are tricky too—you must follow a framework, allow flexibility for the project team, and serve as a leader. On the PMI-ACP exam, you'll likely face questions for which more than one answer is acceptable, but to be scored correctly, you'll have to choose the best answer—not just an acceptable answer. I know, it's tricky.

Video For a more detailed explanation, watch the *Passing Your PMI-ACP Exam* video now.

The first exam domain, on agile principles and the agile mindset, is worth 16 percent of your exam; that's about 19 questions. While it's not the largest exam domain, it's an important one. Fortunately, the topics in this exam domain and chapter are pretty easy to grasp, and I suspect you already know much of this information. If you're nervous about your chances of passing the PMI-ACP exam, this chapter will help you build a strong foundation on agile and bolster your confidence.

Exam Coach Throughout the book I'll drop in these little coaching tips and tricks. These serve as a quick reality check, to encourage you, and to remind you that while there's lots of information to learn, know, and recall, you can do this. Thousands of others have passed the exam, and you can do it too. Be confident, do the work, and you'll get it done. Like an agile project, we're breaking down all this exam preparation into manageable chunks.

Introducing the Agile Principles and Mindset Tasks

This first exam domain is about understanding and applying the agile principles and mindset within your project team and within your organization. To have the agile mindset means that you're not a silo, but that you're leading others within your organization to also embrace the agile approach. If you're the only person in your organization that has the agile mindset, you're going to feel frustrated as not everyone is going along with the principles of agile. Your role is to be an agile evangelist and convert others to agile and all its glory.

To have the agile mindset, you first must understand agile, then embrace agile, and then help others learn and embrace agile, too. The agile mindset embraces the values of the Agile Manifesto (which I'll discuss later in the chapter), solid principles of agile project management, and effective practices and leadership in agile projects. The agile mindset also requires having a good attitude, learning to fail and learning from failure, and helping others understand what they need to know to deliver in the project.

Reviewing the Nine Tasks for Exam Domain I

You'll need to know the following nine tasks, all centered on the agile principles and mindset, for this first exam domain:

- Advocate for agile principles by modeling those principles and discussing agile values in order to develop a shared mindset across the team as well as between the customer and the team.

- Help ensure that everyone has a common understanding of the values and principles of agile and a common knowledge around the agile practices and terminology being used in order to work effectively.
- Support change at the system or organization level by educating the organization and influencing processes, behaviors, and people in order to make the organization more effective and efficient.
- Practice visualization by maintaining highly visible information radiators showing real progress and real team performance in order to enhance transparency and trust.
- Contribute to a safe and trustful team environment by allowing everyone to experiment and make mistakes so that each can learn and continuously improve the way he or she works.
- Enhance creativity by experimenting with new techniques and process ideas in order to discover more efficient and effective ways of working.
- Encourage team members to share knowledge by collaborating and working together in order to lower risks around knowledge silos and to reduce bottlenecks.
- Encourage emergent leadership within the team by establishing a safe and respectful environment in which new approaches can be tried in order to make improvements and foster self-organization and empowerment.
- Practice servant leadership by supporting and encouraging others in their endeavors so that they can perform at their highest level and continue to improve.

Applying the Nine Agile Principles

On your exam, you'll be faced with scenarios about these nine principles. For example, a typical exam question could be something like this:

> John is the project manager for his organization and he's leading an agile project. Some of the stakeholders are confused about the actual progress of the project and how agile is working. Of the following choices, which is the best method for John to convey actual project progress, create trust with the stakeholders, and keep everyone informed about the project progress?

A. Host a weekly status meeting with the project stakeholders

B. Create a project website and share status reports

C. Create an information radiator that includes a project burndown chart

D. Create a wall of status reports for each iteration within the project

While all the choices could work, you must choose the best answer, which is C, to create an information radiator. While we've not yet discussed the concept of an information radiator, that's a key term directly from the fourth task and it refers to a method that helps to communicate project status and is highly visible for everyone to see actual project performance. You must always choose the best answer, even if some of the other choices are good ideas, too. And don't worry, I'll be covering the terms in the preceding answer choices in this chapter.

Your experience as an agile project manager might differ from the principles I'm sharing with you here—and that's okay. I'm sharing the mainstream approaches that you'll be tested on. The PMI-ACP exam tests your knowledge of the most common agile methodologies and concepts, not the very specific applications and process tailoring you may have created and embraced in your organization.

Defining Agile Values for Projects

If there's one word to sum up agile projects, it'd be *value*. Value is what you and your organization care most about: it's the value of the work you do, why your organization exists, and the value you have for the people and their talents in the project and in the organization. Value is so important in agile projects, there's an entire exam domain just on creating value. Value touches all areas of an agile project—and all exam domains for your test. In this first exam domain, we need to discuss the value of doing and being agile.

The whole goal of project management is to get things done, and that's the first definition of value—reaching "done" in a project. The project manager, the project team (sometimes just called the development team), the customer, and all other

stakeholders need a common vision of what the definition of done is. You, the agile project manager, and the customers need to agree on what done looks like. Once everyone knows what constitutes done, you can work toward that vision. Of course, in agile, the requirements that equate to done can be evolving or changing throughout the project life cycle, but everyone needs a general idea of what done means.

Once you have that shared vision of the project's product, you can utilize the adopted agile approaches to continue to work to achieve that value. As changes enter the project, the changes are prioritized based on their value in relation to other components of the project. Agile, at its core, welcomes and expects changes throughout its life cycle, but the framework of agile has principles established to prioritize changes that affect the expected value of what the development team is working toward.

Project management is about getting things done. Agile projects have that same goal, but the management of the project is different than predictive projects. Agile projects, and the processes therein, have some special considerations you'll need to know for your career and for the PMI-ACP exam.

Managing Knowledge-Work Projects

Agile projects are most commonly associated with knowledge-work projects. Knowledge-work projects are driven by creativity and brain power more than projects like construction or moving a call center. This isn't to say that industrial-work projects are less valuable than knowledge-work projects; they're just different. Industrial-work projects, like construction, require up-front planning, exact designs, and specifications that are well known at the launch of a project. These types of projects follow a predictive life cycle where everything is predicted at the start of the project.

Progress in an industrial-work project is easy to see because you can view the results of the construction, for example, as it's happening. Progress in a knowledge-work project is tougher to see because lines of code don't mean much to a customer. This is the reason why one of our exam tasks in this domain is to maintain a highly visible information radiator—a wall on which to openly and easily share project information—which is vital to being open and transparent and showing honest progress.

Project management in industrial-work projects is also different than in knowledge-work projects. Industrial-work projects have a defined sequence, rely on exact answers and plans, measure performance based on results, and establish the project manager as the center of command and control over the project team. Agile project managers have a servant leadership role, meaning they help the project team by removing impediments, shielding the team from interruptions, and encouraging the team to be autonomous and self-directed.

Knowledge-work projects are more ambiguous and expect change, and the actual work is invisible because the work is taking place in your brain rather than at a construction site. Agile is best-suited for knowledge-work projects, like software development. Agile expects change and provides a framework to manage changes that will happen in its life cycle. Knowledge-work projects have less structure and more innovation, more creativity, than industrial-work projects.

Exam Coach	Knowledge-work projects have higher uncertainty than industrial-work projects because change is likely, they can be complex, and there is risk in their probability of success. Agile project management addresses change, complexity, and risk.

Comparing Empirical Processes and Defined Processes

If you and I were going to put together a store-bought packaged bookshelf, we'd unpack the bookshelf, dig through the instructions, and follow the defined plan. That's a simple analogy to defined processes. *Defined processes*, as the name implies, are processes for project work that have been defined for us. In construction, manufacturing, or any other industrial work, there are processes for the work and tasks that follow a defined and generally accepted approach to getting the work done. In a predictive project, where everything is planned upfront, the project manager and team are creating and using defined processes because they know exactly what the project is intended to create.

Empirical processes, by contrast, are based on observation, trial-and-error, and the experience of the person doing the work. Empirical processes are what you use when you put together a puzzle—or perform knowledge work. Agile projects rely on the knowledge worker to be creative and innovative, and to figure out the work to reach the desired results. You could bring ten developers together and have each of them create a simple "Hello, World" application based on the same set of requirements and you'd likely have ten different approaches to reach the same result in the application. Each developer would use their experience, observations, and creativity to build the app to get the "Hello, World" result. That's an example of empirical processes.

By embracing the values of agile, you're encouraging your dev team to utilize empirical processes to create value. Of course, in an agile project you'd want the ten developers to work together to create a common plan to create the value of the deliverable, rather than ten individuals working on their plans separately. We want teams to be collaborative and to have a common,

agreed-upon approach to solutions.

Embracing the Agile Mindset

A term you'll see throughout the *Agile Practice Guide*, on websites, and in other books is the *agile mindset*. The agile mindset is a way of thinking about and doing agile projects. Having the agile mindset is to exemplify the values and principles of agile in how you work, how you lead a project, and how you share your passion for agile with others. To fully embrace the agile mindset means that you first must follow a formal agile approach.

You must know the rules, the processes, and the formalities of agile as a solid base to embracing the agile mindset. Once you completely understand the agile methodology that you and your organization will follow, then you can begin tailoring the process to suit your environment and projects. If a project manager begins changing the established agile approach without really understanding how the approach works, then risk can be introduced to the project, frustration and confusion can plague the project team, and stakeholders may lose engagement and support of the project. So, basically, know what you're doing before trying to change what you're doing.

The second step to fully embracing the agile mindset is to utilize agile to reach organization and customer goals, not just to "do agile." You might break a product into increments for several product releases, or help the customer find the most valuable requirements, or trim requirements from scope to meet time and costs constraints. To fully embrace the agile mindset, you'll use agile to create value, not just follow a nifty project management formula.

Reviewing the Declaration of Interdependence

The core people who established the guiding principles of agile project management released a document in 2005 called the "Declaration of Interdependence" to serve as a value system for agile project managers. The idea behind this document is that all participants in an agile project are interdependent on one another: the project manager, development team, customers, and other stakeholders all contribute to the same goal of creating value for the organization. Here's the content of this straightforward document for the agile community (see www.pmdoi.org):

We are a community of project leaders that are highly successful at delivering results. To achieve these results:

- We **increase return on investment** by making continuous flow of value our focus.
- We **deliver reliable results** by engaging customers in frequent interactions and shared ownership.
- We **expect uncertainty** and manage for it through iterations, anticipation, and adaptation.
- We **unleash creativity and innovation** by recognizing that individuals are the ultimate source of value, and creating an environment where they can make a difference.
- We **boost performance** through group accountability for results and shared responsibility for team effectiveness.
- We **improve effectiveness and reliability** through situationally specific strategies, processes and practices.[1]

While you likely won't have direct questions about this document on your PMI-ACP exam, you should know that this document clearly defines the concept of the agile mindset. These are some guiding principles you'll want to know for answering questions on the exam.

Being Agile and Doing Agile

As you might expect, there's a real difference between doing agile and being agile. You can accomplish *doing agile* by taking a seminar, following some preset rules and guidelines, and following a formula of sorts for an agile approach. Doing agile encompasses the mechanics of agile project management. *Being agile*, however, includes not only the mechanics of agile but also the agile mindset of engaging your project team, working collaboratively with stakeholders, providing open and honest communication, allowing leadership from anyone in the project, embracing change, and being excited about the project.

Being agile is more than just understanding the rules and principles of an agile approach. Being agile is understanding the depth of process application that's needed, how to tailor the approach, and how and why to bend the rules, and keeping the team and stakeholders engaged and focused on creating value. Being agile is the actualization of the agile mindset. While the PMI-ACP exam can't precisely test you on whether you're just doing agile or really being agile, having the agile mindset will greatly help you both pass the exam and find success as a PMI-ACP.

Inverting the Triple Constraints

If you're like most agile project managers, you already have a good grasp on the traditional, predictive approach to project management. In traditional project management, we often refer to the "iron triangle" (or "triple constraint") where we wrestle with the project to maintain the balance of schedule, costs, and scope. Scope in predictive projects that follow the iron triangle is the factor of project management that is fixed, and we generally try not to change scope—unless we'll accommodate the change in the other two factors of schedule and cost. In other words, for the project to be successful, scope is fixed, but the schedule and costs may vary.

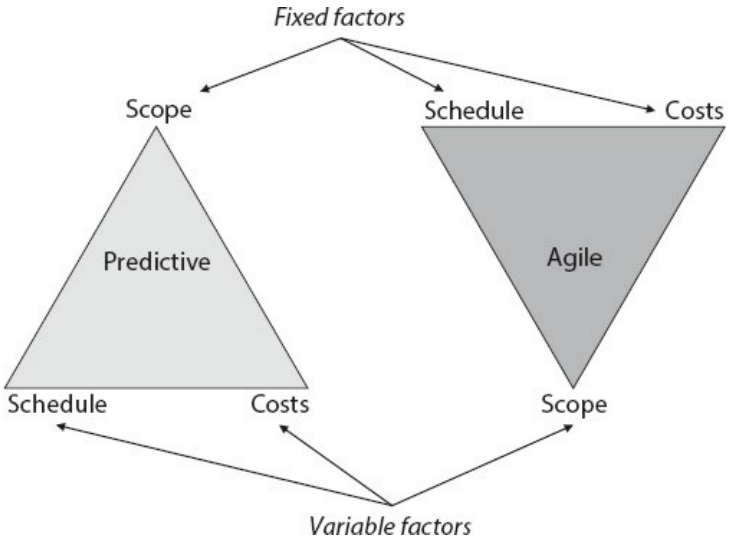

Figure 1-1: Agile inverts the common triple constraint of traditional (predictive) project management.

In an agile project we flip this equation and fix schedule and costs and allow scope to change as much as the customer likes, as shown in Figure 1-1. With this inverted triple constraint, we know how much the stakeholder can spend, based on the cost of the knowledge-work labor in the project, and we know the duration of the project, based on iterations of work or a realistic deadline that we must meet. Now the requirements of the project scope can grow or shrink to meet how much time and how much money we must spend. This introduces the prioritized product backlog, which I'll talk more about in adaptive planning, but for now, just know it's the prioritized product backlog that make up the project scope in an agile project.

Leading Organization Agility

If you're the only person in your organization who embraces agile, you'll feel anxious and frustrated as you work with others who don't embrace it. You may have the agile mindset, and that'll help you to some extent, but your team, stakeholders, and the rest of the organization won't be in the same mindset when it comes to getting things done. If your project team comes onboard and begins to embrace agile, that'll help the team get things done, but the rest of the organization will still be outside the agile mindset and how agile works. Finally, if the whole organization learns how agile works, then the whole organization can prosper, and the organization can move deeper into organizational change, continuous improvement, and some of the loftier goals of agile.

While your PMI-ACP exam won't focus too much on how you convert project teams and organizations to agile, it's part of the agile mindset: to teach others how agile works, why it creates value, and how it helps us all become more effective in our roles and responsibilities.

Thinking About Agile

As you prepare to pass the PMI-ACP exam, you'll be thinking a lot about agile. In your study efforts, look for opportunities to apply agile techniques both in your plan to pass the PMI-ACP exam and in your projects. Seeing how agile can create value, streamline project management, and empower others to innovate and be creative is crucial for exam (and agile) success. While you may be the sole individual in your organization to embrace the agile mindset, great things can come from converting individuals, then teams, and, hopefully, the entire organization to the agile mindset.

Doing Agile Project Management

While I just discussed the need to be agile, not just do agile, you have to start somewhere. Agile project management is really a

generic term to describe all the different flavors of agile: Scrum, XP, Lean, Kanban, and Crystal, to name a few. Whatever approach you use out there in the real world, you must first understand the rules, sometimes called *ceremonies*, of agile and how it all works. The PMI-ACP exam will first test you on doing agile—on understanding the rules and framework of agile—by testing your knowledge of these different agile flavors. You don't have to be an expert in every agile approach, but you need to know enough to understand the framework, recognize the terms, and follow the generally accepted practices. Then, when you've fully mastered the agile approach, you can begin tailoring the approach to work best in your environment—that's being agile.

Encouraging Others to Embrace Agile

To be an agile project manager, you need to understand the principles of agile and then evolve into being agile. That's what'll happen with your project team and with the organization. We all have to start somewhere—so the basic rules are a great place to start. By showing others, not just telling, how effective agile is, they'll want the approach in their life too. This is true for the project team and for others in the organization. You'll encourage others to try agile, and that means you'll be doing some teaching of how agile works, its goals, and how to best apply it. You'll likely see a question or two on this concept on the PMI-ACP exam, as it's part of being agile.

When we think about organizational agility, we must think from the organization's perspective. Many of the people in an organization are going to be resistant to changing to agile practices, let alone being agile. People don't like change, of course, so you'll need to think of things from their perspective, think of what concerns them, what they expect from a project. To encourage others to embrace agile means you'll need open and honest communication—no secrets in agile project management. You'll also need to constantly think about the values of the organization: what constitutes value for your customers, and how can you create that value quickly, effectively, and with quality?

[1]©2005 David Anderson, Sanjiv Augustine, Christopher Avery, Alistair Cockburn, Mike Cohn, Doug DeCarlo, Donna Fitzgerald, Jim Highsmith, Ole Jepsen, Lowell Lindstrom, Todd Little, Kent McDonald, Pollyanna Pixton, Preston Smith, and Robert Wysocki.

Reviewing the Agile Manifesto

If you've been around agile for more than a few minutes, you've likely heard of the Agile Manifesto. Technically, this document is called the Manifesto for Agile Software Development, but Agile Manifesto is the name you'll encounter most often. In 2001, a bunch of software developers and software project managers met up to discuss the nuances of software project management and to try to establish some general guidelines for managing software development projects. From that meeting the Agile Manifesto was created, and it has grown in popularity over the past decade and a half.

Exam Coach	You don't have to know about the origins of the Agile Manifesto or the names of the 17 people who signed the original document. I'm just providing some light history of the document here for reference.

The Agile Manifesto is a broad document that establishes the values of agile project management and provides some good advice for leading agile projects. The four values you want to know are

- Individuals and interactions over processes and tools
- Working software over comprehensive documentation
- Customer collaboration over contract negotiation
- Responding to change over following a plan

Let's take a closer look at each of these values.

Valuing Individuals and Interactions over Processes and Tools

The first value of agile project management is that we value individuals and interactions over processes and tools. Projects are performed by people, not by equipment, software, or tools and techniques. People get things done, people create the business value for the organization. In agile projects, we focus on collaboration, synergy, and creating a sense of community within our projects. We want people to feel safe, empowered, and trusted to do the right thing. We don't focus on doing processes just because we were instructed to do them. We take the time to think, to see the obvious, and to get things done.

Processes and tools are great for helping us to get things done, but sometimes organizations and project managers get bogged down in the bureaucracy of managing a project and fail to get out of the way and let the team do its work. That's the

goal in agile: if a process or tool doesn't add value to the project, we likely don't need it. Our focus is on collaboration, teamwork, and working together to get things done.

Valuing Working Software over Comprehensive Documentation

One of my pet peeves is documentation: reports, meeting minutes, manuals, and other items documenting why or how the team accomplished something. In my experience, these documents are standard tasks for management rather than tasks with meaning and value. Sure, some documentation is good, but if no one is reading what's been written, who cares?

In agile, we are pragmatic and focus on creating working software, not comprehensive documentation. If the team is having to spend time documenting decisions and work processes rather than staying "in the flow" of creative problem solving and getting things done, it's a big waste. No one is going to spend time reading lengthy reports, manuals, and documentation. Value is in working software. Value is in results. Agile focuses on the priorities of the customer, not documentation that is glanced over and then tossed in the recycle bin. We do need some documentation, but the documentation should be barely sufficient, completed at the last responsible moment, and sometimes, just because. "Just because" means that sometimes, like in the case of regulations or strict policies, we have to document things.

Valuing Customer Collaboration over Contract Negotiation

Consider the customer–vendor relationship in traditional project management, where proposed changes to the product the vendor is creating for the customer often involve some negotiation and haggling. Now consider how agile expects change, how agile priorities can shift based on the customer needs and realizations of value, and how the project scope in an agile project varies at the start of the project. Rather than negotiating with the customer regarding each change to the project, in agile, the customer and the project manager and team all work together to create the best solution for the customer.

Valuing customer collaboration over contract negotiation doesn't mean there isn't a contract between the customer and the vendor, but that the terms of the contract are defined, with flexibility in mind, and then we focus on collaborating through the agile framework. We don't have to haggle over a shift in priorities, but instead we'll focus on what's creating value, accepting some tradeoffs, and working through the project collaboratively, rather than bemoaning and dreading every meeting with the customer. Collaboration is a key theme in agile project management.

Responding to Change over Following a Plan

The Agile Manifesto is about software development projects, not industrial-work projects like construction projects. Construction projects rely on a predictive life cycle, where design specs, blueprints, engineer drawings, architects, and the customer are all in agreement about what the project is intended to create before the project team begins executing. In a predictive life cycle, the project manager and project team are often averse to change because of the time and monies that have been invested in planning.

Of course, in agile projects, we expect and even welcome change. Change is not a big deal like it is in a predictive life cycle. Changes to the project scope can, and do, occur often, but agile also considers changes in the technology the development team is working with. Technology can change daily. The customer expectations can change. The marketplace can shift. If the project manager and team are married to a plan, then they ignore the opportunity to deliver value, or at least are resistant to the change. Agile welcomes change, and our planning is at a high level initially, and then specific to our current iteration. I'll talk more about iterations and iteration planning when we get into adaptive planning in Chapter 5—something to look forward to.

Embracing the 12 Principles of Agile

In addition to the four values of the Agile Manifesto, there is a supporting document that defines the 12 principles of agile project management. Know these for your exam. As you prepare to pass the PMI-ACP exam, embrace these 12 principles of agile project management. While the PMI-ACP exam likely won't ask specific questions about these 12 principles, knowing these will help you recognize the best answer on your test.

Create Value Through Continuous Delivery Value is in working software, not documentation, meetings, or great ideas. The highest priority of an agile project team is to satisfy the customer by delivering value. Above all other choices on the PMI-ACP exam, look for opportunities to deliver value to the customer through continuous delivery of working software as soon as possible in the project.

Working with Changing Requirements Agile welcomes change at any point in the project, even at the end of the project. The goal is to create value for the customer, so if a change will add value, bring it on. This principle is "Welcome changing requirements, even late in development. Agile processes harness change for the customer's competitive advantage."

Delivering Working Software Frequently Agile isn't a surprise party: create value quickly for the customer through timeboxed iterations. This principle is "Deliver working software frequently, from a couple of weeks to a couple of months, with a preference to the shorter timescale."

Working with Business People Often in predictive projects there can be an "us against them" mentality, but not in agile. Agile projects collaborate with business people. This principle is "Business people and developers must work together daily throughout the project."

Creating Results This principle is straightforward: "Build projects around motivated individuals. Give them the environment and support they need, and trust them to get the job done." As in the Agile Manifesto, results are created by people, not by equipment, tools, or processes. Agile empowers the project team rather than the project manager to be in a command-and-control position.

Communicating Face-to-Face Face-to-face communication is one of the best ways to communicate and to share information. That's the thrust of this principle: "The most efficient and effective method of conveying information to and within a development team is face-to-face conversation."

Building Working Software Variance analysis, earned value management, and performance measurement are all great, but the real measure of progress is working software. This principle is "Working software is the primary measure of progress."

Providing Manageable Environments It's no secret that overworking the project team creates tension, defects, and frustration. Agile projects promote sustainable development by establishing a workable pace in a manageable environment. This principle is "Agile processes promote sustainable development. The sponsors, developers, and users should be able to maintain a constant pace indefinitely."

Paying Attention to Details The devil is in the details, and agile is no exception. By paying attention to details, being committed to excellence, and practicing good design, the development team is creating value for the customer. This principle is "Continuous attention to technical excellence and good design enhances agility."

Keeping Things Simple Simple doesn't mean easy, but it's focused on the prioritized requirements, approachable, and creates reliable solutions through stable architecture, good code, and valuable software. Agile projects don't need to create more than what's needed. This principle is "Simplicity—the art of maximizing the amount of work not done—is essential."

Relying on Self-Led Teams Agile teams are self-led and self-organizing, and they work collaboratively on the software solutions. The project manager gets out of the way and lets the team do the work as it sees fit. This principle is "The best architectures, requirements, and designs emerge from self-organizing teams."

Adjusting the Project Practices One of the beautiful things about agile is that there is opportunity to pause and reflect on what's working, or not working, and then to adjust. Rather than continue to suffer, as in some predictive projects, agile teams have an opportunity to communicate honestly and make corrective actions in the work to improve the project. This principle is "At regular intervals, the team reflects on how to become more effective, then tunes and adjusts its behavior accordingly."

Comparing Agile Project Approaches

There are lots of different approaches to doing agile and no single way is the best way. For your PMI-ACP exam, you'll need to know a bit about each of these common agile approaches, but you won't have to know everything about every single approach—just the good stuff. In this section I address the seven most common agile approaches that you'll likely see on your exam. I cover them to a depth that's reflective of what's popular in the industry. I'm not going into every facet of every approach, but will provide detail where detail is warranted.

The two most common agile approaches in the software development industry, and on the PMI-ACP exam, are Scrum and XP. You'll likely see questions about these two approaches the most on your exam, but you'll also likely see questions about the other approaches at a high level. I seriously doubt you'll get many questions about the nuances of any of these approaches. The PMI-ACP exam is about being agile, not specifically about Scrum, XP, Kanban, or even Crystal.

Reviewing the Basics of Scrum

As of today, Scrum is arguably the most popular agile project management approach. Scrum uses timeboxed iterations, called *sprints*, to create prioritized requirements for the customer. Sprints last from two to four weeks. Scrum is based on three core principles of agile software project management: transparency, inspection, and adaptation. Know these three principles and how to apply them in a Scrum project.

- **Transparency** Transparency begins with a clear understanding among all stakeholders of what the definition of "done" is. Transparency communicates with a common language, evidence of accomplishments in each iteration, and an understanding of what everyone on the project is doing.
- **Inspection** Scrum utilizes intermittent inspections of the work to ensure quality. The inspection of the work catches defects before they're released into production (called *escaped defects*) and gives the team opportunity to fix the glitches before the defects cause the work to become unacceptable.
- **Adaptation** Scrum is agile, meaning it's flexible, adaptive, and allows tailoring of the processes being used in the project. If a process within a project isn't adding value, you don't just live with the broken process, you adjust the broken process. It's silly to continue to run a project on and on when there's something that's not working.

Exploring the Five Scrum Values

There are five Scrum values that all project team members, including the project manager, strive to adhere to:

- **Commitment** Commit to achieving the goal of each sprint.
- **Focus** Focus on completing the goals of the sprint and the goals of the team.
- **Openness** Be open and transparent about the challenges of the project work.
- **Respect** Respect each other as capable, independent people.
- **Courage** Do the right thing and work through tough problems.

Leading a Scrum Project

There are just seven steps in a Scrum project—that's it! I'll talk more about each of these steps later in the book, but for now, here are the seven steps to the Scrum framework:

1. The product owner creates a prioritized list of everything the project might deliver. This list is the product backlog.
2. The team and product owner meet in sprint planning and the team decides how much work it can take on in the next sprint. The team pulls requirements from the prioritized product backlog that it can achieve in the sprint. This chunk of work becomes the sprint backlog.
3. The team decides who'll do what and creates the items in the sprint backlog for the current sprint. The team will meet each day for a 15-minute meeting, called the Daily Scrum, to share progress updates.
4. The project manager, called the ScrumMaster, helps keep the team working toward the sprint goal.
5. A sprint review happens at the end of each sprint to demonstrate for the product owner what the team has accomplished.
6. After the sprint review the team participates in a sprint retrospective to discuss what did or did not work in the last sprint. This gives the ScrumMaster and the team an opportunity to adjust the processes and work for the next sprint.
7. The whole process repeats itself by the project team selecting the next chunk of prioritized requirements from the backlog and getting to work in the next sprint.

Sounds easy, right? Well, the Scrum approach is easy to understand, but implementing it in an organization can be tricky—especially the first time. Like most things, with practice it does get easier and easier, and as people begin to see value in the results, more and more people in the organization will want to adapt the Scrum approach.

Identifying the Scrum Roles and Responsibilities

There are three distinct roles in the Scrum environment and they all work together throughout the project. First up is the product owner. The product owner has one job: manage the product backlog. As previously mentioned, the product backlog is the long list of prioritized project requirements. The product owner arranges these items from most valuable to least valuable. The ordering and content of the product backlog can change for each sprint, but what's in the backlog and the ordering of the items is transparent to the whole team. The management of the product backlog is called *grooming* or *refining*.

Next up is the development team, sometimes just called the dev team. The development team is responsible for sizing the requirements of the product backlog and getting work done in each sprint. The development team is self-organizing and self-led, and its members are called *generalizing specialists* because they can often do more than one function on the team. An ideal Scrum team has no less than five people and no more than eleven people. Some people argue that seven people on a Scrum team is perfect.

Finally, there's the ScrumMaster. This is analogous to the project manager role, but a ScrumMaster is less focused on command and control and more focused on being a servant leader. The ScrumMaster ensures that everyone understands the rules of Scrum, removes impediments for the team, facilities Scrum meetings, helps the product owner groom the backlog, and communicates the vision of the project to everyone that's involved.

Reviewing the Scrum Ceremonies

Scrum ceremonies sounds so formal, but it's the terminology for the meetings and events within the Scrum practice. Scrum ceremonies are straightforward, but there are some guidelines for each of the events:

- **Backlog refinement meeting** This event is where the product owner, the ScrumMaster, and the development team work together to discuss the backlog items and prioritize the items.
- **Sprint planning meeting** The team determines how much work they can take on from the prioritized backlog for the next sprint. This determination is based on estimates of the items in the product backlog and past sprints. The selected items from the product backlog become the sprint backlog and the goal of the sprint.
- **Daily Scrum** This is a 15-minute meeting, sometimes called a standup meeting because everyone stands for the duration. The Daily Scrum happens every day at the same time, in the same place. This meeting is for the development team and the ScrumMaster only. In the meeting each person addresses the entire team by answering three specific questions:
    - What have I accomplished since the last Daily Scrum?
    - What will I accomplish before the next Daily Scrum?
    - Are there any impediments blocking my work?
- **Scrum of Scrums** In a big Scrum project there might be multiple teams working together. Rather than having a huge Daily Scrum, the teams meet separately and then a representative from each team meets in a Scrum of Scrums to report on each team's progress. The team representatives answer the same three questions, but for the team rather than as individuals. In addition, a fourth question is often addressed: Will our team be putting something in another team's way?
- **Scrum of Scrums Scrum** Yes, this is getting silly, but you should know it. If there are many teams, you can create another layer of Scrum meetings with representatives from the Scrum of Scrums. This isn't very common, but it's just the type of question scenario that you could see on your exam.
- **Sprint review** At the end of each sprint, the development team demonstrates for the product owner, the ScrumMaster, and other key stakeholders the work they've accomplished in the past sprint. This review is an opportunity for the development team, not the ScrumMaster, to show the results of what they've created. It's also an opportunity for the product owner to offer feedback on whether the work has reached the done stage, and, if it hasn't, describe what's missing and elaborate on corrections or modifications for the increment of work created.
- **Sprint retrospective** After the sprint review, and before the next sprint planning meeting, the development team meets to discuss three items: people, product, and process. The team discusses what's worked well with each item in the project, what needs improvement, and the feedback from the product owner from the sprint review meeting. Note that *people* means the project team and the ScrumMaster; this isn't a gossip session, but an opportunity to give an assessment and offer suggestions for improvement, done with respect for one another.

Exam Coach	This is a quick primer on Scrum. Throughout the book I'll reference Scrum as needed. The important thing to remember here is that the PMI-ACP exam doesn't test you specifically on Scrum, or other agile methodologies, but on how to perform agile. You'll only need to know the fundamentals of Scrum and these agile approaches. Don't get bogged down in trying to learn all that you can about each of these agile approaches.

Reviewing XP

XP is short for eXtreme Programming, an agile approach that is focused, of course, on software development. XP also uses iterations of development, but these iterations last for two weeks in a typical project. The requirements are called *user stories* and the development team sizes up the user stories to see how much work they can accomplish in the next iteration. At the beginning of the project, and as needed through the project, the XP team will host a release planning meeting. The purpose of the release planning meeting is to create the high-level estimate for the project work and define when the product will be released, either through increments or as one final deliverable.

Spike is an XP term that you should know. There's an architectural spike, which is an iteration to set up the environment, ensure the design is not overly complicated, and examine the plan's feasibility to accomplish the project goals. A risk spike aims to mitigate or eliminate risks that have been identified in the project and threaten project success.

Reviewing Five XP Values

In an XP project, the development team utilizes pair programming. As the name suggests, *pair programming* means that programmers work together in pairs; one person codes and the other person checks the code. When thinking through these values, keep in mind the concept of pair programming. XP has the following five core values, which aren't much different than the values you've already seen in the Agile Manifesto:

- **Simplicity** Find the simplest thing that could possibly work. Keeping things simple means removing complexity and waste in the development.
- **Communication** Transparent, face-to-face communication is best for a project team.
- **Feedback** Fail fast and fail early to get feedback on what's not working before getting too invested in the project approach.
- **Courage** All the code is visible to everyone all the time on an XP project. It takes courage to put your work out there for others to review, inspect, and edit.
- **Respect** The team respects each other, and respects others' culture, values, and how they work to get results. The success, or failure, of the project is everyone's responsibility.

Introducing the XP Roles and Responsibilities

There are four roles that you'll work with in an XP project. Know these four roles for your PMI-ACP exam:

- **Coach** The project manager as a coach is more of a mentor, a facilitator, and she coaches the people on the project team, helps get things done, and serves as the hub of communications for the project stakeholders.
- **Customer** This is the person that represents the business, and the role is similar to that of the product owner in a Scrum project. The customer helps prioritize requirements and is onsite with the development team.
- **Programmers** These are the development team members for the project. These members are also known as developers or coders.
- **Testers** These are the people that test the code before the code is released into production. The tester role can be fulfilled by a programmer.

Learning the 13 XP Core Practices

There are 13 XP core practices that wrap around different layers of a circle, as depicted in Figure 1-2. While I do think it's important to know these core practices, you don't need to memorize them for your PMI-ACP exam. Be more familiar with the mindset and intent of these practices and how to apply them—that'll help you more. Let's take a quick look at these 13 core practices now:

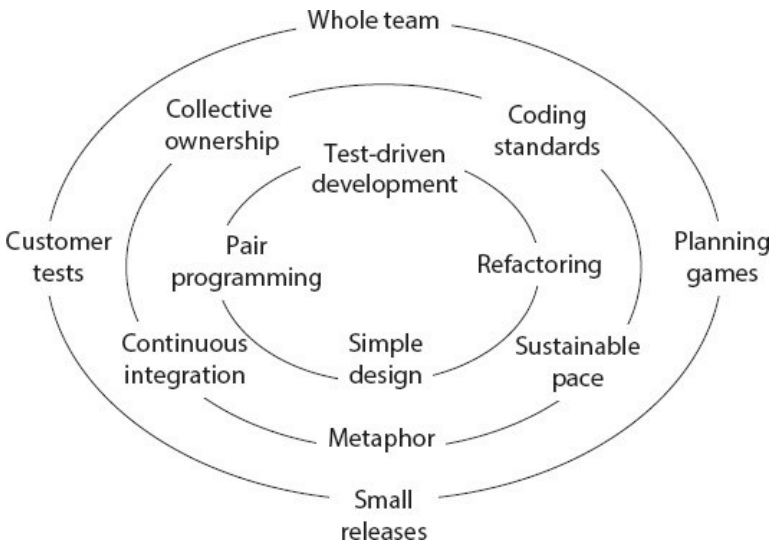

Figure 1-2: The 13 XP core practices are situated on different layers of a circle.

- **Whole team** The whole team acts as a collaborating unit where team members can serve more than one role and are ideally colocated. The individuals in the team are sometimes called generalizing specialists.
- **Planning games** The team plans for iterations and incremental releases. Iterations are the two-week segments of work, and releases happen within six months.
- **Small releases** With two-week iterations, the development team can create, test, and release the code in quick, small releases for fast feedback and corrections as needed.
- **Customer tests** The code is tested from the customer perspective to confirm that the code meets the expectations of the customer before releasing the code to production.
- **Collective code ownership** Any pair of programmers on the development team can edit anyone's code.
- **Code standards** All programmers on the development team follow defined and communicated standards for developing the code.
- **Sustainable pace** As in Scrum, the team works at a sustainable pace to get the iteration done.
- **Metaphor** A metaphor for the project is used to explain the goal and function in plain, understandable language for all stakeholders.
- **Continuous integration** By integrating code frequently throughout the iteration, issues are discovered quickly and are easier to fix than when integrating code right before testing.
- **Test-driven development** An acceptance test is written before the code is written so the developers know what it takes to pass the acceptance test and can program accordingly.
- **Refactoring** Refactoring involves cleaning up the code to remove waste, redundancy, dependent connections, and shortcuts. This "mess" that developers are cleaning up is called *technical debt*. Technical debt can pile up and become trickier and heavier to clean up the longer the mess is ignored.
- **Simple design** Do the simplest thing that can work. Keep it simple.
- **Pair programming** Developers work in pairs; one person codes while the other checks the code. The pair switches roles periodically.

Like Scrum, I'll mention XP as needed throughout the book, but you won't have real in-depth questions about the nuances of XP on your exam. Be familiar enough with these principles and intent to answer agile questions about the approach.

Working with Kanban

Kanban (pronounced "con bon") is a Japanese word that means signboard. The Kanban approach started at Toyota as a way to help visualize the flow of work through a system. Figure 1-3 is an example of a simple Kanban that shows the work that's in

queue (the Backlog column), the work that's in progress (the middle three columns), and the items that have been completed (Done column). The work that's in progress is called the WIP and that's what the team is working on right now. The signboard is often a big white board and items are pulled from the leftmost column and moved progressively to the right columns either by erasing and rewriting the items in each column or by using sticky notes. This use of low-tech, high-touch tools to help manage and visualize the project workflow is an agile concept.

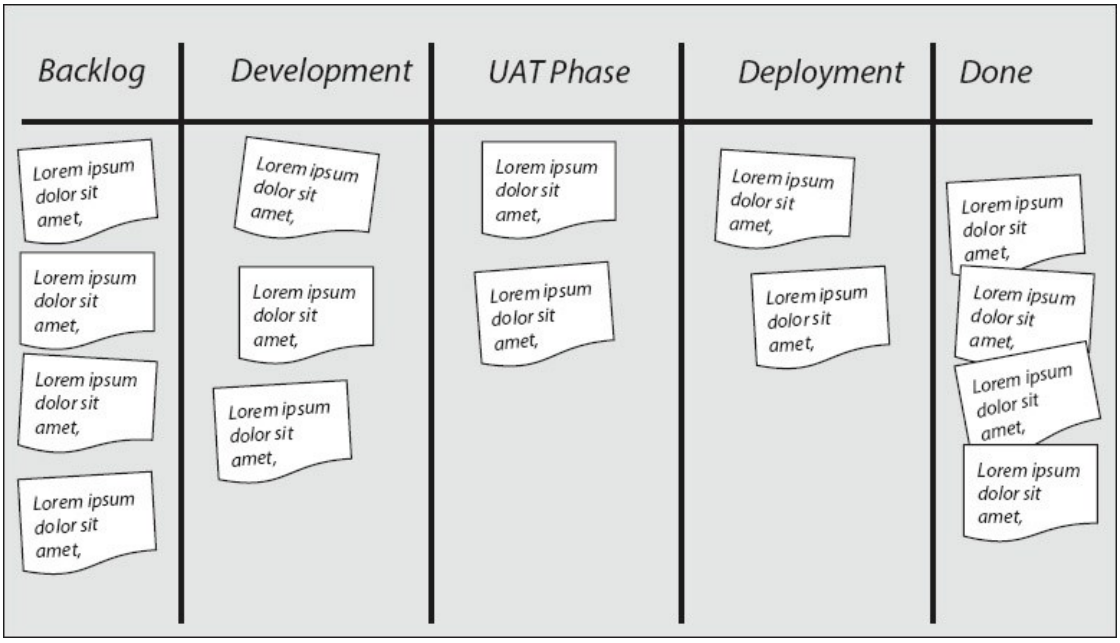

Figure 1-3: Kanban boards visualize the flow of the work in a project.

Kanban is a pull system, pulling work from the queue into the WIP and workflow. When a team member completes an item, it triggers the next item to be brought into the system. When there are additional steps, such as testing, the item pulled into the next task column can also trigger other team members to address or pull that item in their queue of work. Kanban doesn't use iterations like Scrum and XP, but just continues to pull to-do items into the WIP and through the established workflow. The items in queue can be prioritized, but Kanban is about getting things done, rather than creating increments.

Reviewing the Five Kanban Principles

Kanban has five principles you should know:

- **Visualize** The big principle of Kanban is the most obvious: you visualize the workflow that you and the team have created. Your workflow can have as many steps (columns) as needed from the queue to completion.
- **Limit the WIP** Don't pull too many items into WIP in an attempt to increase productivity. Switching between work items creates waste, bottlenecks, and defects. By limiting the WIP, your team members focus on completing one item at a time before bringing new work into the workflow.
- **Manage the workflow** By visualizing the work and controlling the WIP, you and the development team can better control defects, easily track work items, and better manage changes.
- **Clearly define the process and policies** When everyone understands the policies and how agile processes work, any changes to processes are made based on value, rather than subjective reasoning.
- **Collaborate** The team works together to get the work done, but also works as a team to consider any changes to the processes or workflow.

The most important thing about Kanban is limiting the WIP. By limiting the WIP, the development team is more effective and can complete the work faster. Little's Law, identified by John Little, states "the average number of items in a queuing system equals the average rate at which items arrive multiplied by the average time that an item spends in the system." In other words, the more items that are in the WIP queue, the longer it will take the team to complete the items in queue. It's like standing in line at the grocery store: the longer the line you're in, the longer it takes the cashier to process all the orders ahead of you. If your team can minimize waste and increase speed, then the duration of the queue will diminish. Too many items in WIP equates to too much motion—and motion is waste.

Introducing Lean Product Development

Lean Product Development, commonly called Lean, began in Toyota manufacturing as a refinement of Henry Ford's manufacturing approach, but it's been adapted to agile and other environments. Like other agile approaches, Lean visualizes what needs to be done, creates requirements based on the customer's definition of value, and includes opportunity for learning and process improvement throughout the project.

There are seven principles within Lean that you should recognize for your exam:

- **Eliminate waste** Waste is anything that is a detriment to the customer-identified value. Waste includes partially done work, the addition of unnecessary features, rework in test and fix cycles, and waiting too long to test after creating the code.
- **Quality is built in** As a project manager, you likely know that quality is planned into a project, not inspected in. That's the same principle here: you build quality into the project by keeping things simple in the code, refactoring often, and utilizing continuous integration.
- **Create knowledge** Architecture validation happens as the code is created, offer early and frequent releases for customer feedback, have daily builds, perform continuous integration, and use modular architecture to add features and changing requirements in future builds. Through learning we create more knowledge.
- **Defer commitments** By not making irreversible decisions until the last responsible moment, you're creating opportunities for better design, features, and flexibility in the software. You don't want to lock in design solutions until it's necessary to do so.
- **Deliver fast** Fast delivery creates value, keeps customers from changing their minds about the established and agreed-upon product requirements, allows the development team to be self-led and self-organizing to get the work done, and reduces defects because of higher quality demands.
- **Respect people** Development teams are given goals and the empowerment to achieve those goals, leaders are more entrepreneurial than traditional project managers, and the development team is made up of technical experts that can deliver on requirements. We treat people as professional colleagues, not as underlings to do our bidding.
- **Optimize the whole** We consider the whole project: the people, processes, alignment with organizational goals, and how everything must contribute to the value of the product.

In addition to these seven principles of Lean, there are also the following seven areas of waste we aim to remove from the project. These are often called Poppendieck's Seven Wastes of Software Development, as they were identified by Lean experts Tom and Mary Poppendieck.

- **Partially done work** Value is in working software, not partially done work. The development team should work on and complete one item at a time.
- **Extra features** Adding bells and whistles that the customer hasn't requested is not a value-add. This is scope creep and robs the project of time and energy that should be spent on the identified requirements.
- **Relearning** When information goes missing, due to poor knowledge management, and teams members have to relearn the information, that's wasted time and energy in the project.
- **Handoffs** When information is passed from one person to another and another, rather than through a centralized communication system, the message and information can change and that contributes to waste.
- **Delays** Waiting time is wasted time. When team members are waiting for information or outputs from stakeholders, this is waste.
- **Task switching** Hopping from task to task is wasteful, sucks time, and creates partially done work. Instead, the development team should focus on completing one task at a time.
- **Defects** Defects cause rework, create frustration, and can cause stakeholders and the project team to lose support of the project. Defects that make it through the testing process and are released into production are called escaped defects.

Working with DSDM

The Dynamic Systems Development Method (DSDM) is one of the predecessors of today's agile project management and relied on a business case to show value and a feasibility study to determine if the development team could create the architecture and requirements the customer identified. While you likely won't see many questions about DSDM on the PMI-ACP exam, you should recognize these eight principles:

- Business need is the primary goal.
- Deliver working software on time.
- Collaborate with the development team and the business people.
- Quality cannot be comprised.
- Build incrementally based on firm foundations.
- Develop iteratively.
- Provide clear and consistent communications.
- Demonstrate control.

The project manager and the development team work together to ensure that all eight of these principles are implemented in the project. You cannot shortcut any of the eight principles without affecting the quality of the project management process and the quality of the deliverable presented to the project customer.

Utilizing Feature-Driven Development

Feature-Driven Development (FDD) is an iterative approach to software development that bases its progress on the client's values of features the software will provide. Like other agile approaches, FDD utilizes a backlog of features that are prioritized based on customer values then created through iterations of project work. Figure 1-4 shows the FDD process.

There are five stages to the FDD process:

- **Develop an overall model** This stage establishes a high-level description of the project scope. The scope is broken down into domain models and passed through a peer review to reach consensus on the models that will be part of the project's overall model to be created.
- **Build the features list** The domain models are decomposed into subject areas. The subject areas are broken into business activities and features. Features are written as <action><result><object>, such as "Calculate the total price for a purchase" or "Validate a user login." Features are small enough that they won't take longer than two weeks to create.
- **Plan by feature** A features list is compiled, and a development plan is created. The development plan assigns ownership of features to developers.
- **Design by feature** The chief programmer chunks out the set of features that can be created in the two-week iteration by the class of programmers. The chief programmer also creates a sequence diagram for the selected features and updates the overall model. Design by feature repeats for each iteration of the project work.

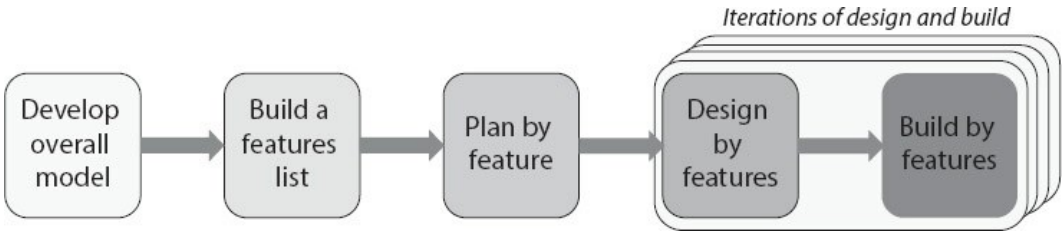

Figure 1-4: Feature-Driven Development has five stages in its development process.

- **Build by feature** The development team develops the code, then the features pass through unit testing, and finally code inspection ensures that each feature is of quality and ready to be added to the build. Build by feature also repeats for each iteration of the project work.

Throughout the process, the project manager utilizes configuration management to document and track changes to the code. The team participates in regular builds to ensure the features they're creating work with the existing build. The compiled code

is demonstrated for the client on a regular basis (often every two weeks). The project manager also tracks the completion of milestones and overall progress.

Working with Crystal

The last agile approach you need to be familiar with is Crystal. Crystal isn't a singular approach to agile project management, but a family of methodologies created by Alistair Cockburn. Crystal uses eight different schemes of agile project management based on several factors, such as the complexity of the project, the number of project team members, and the criticality of the project. Sometimes, these approaches are described as project criticality and the weight of the processes used in the project. The higher the priority of the project, the more weight and ceremony is needed to control the project.

Crystal utilizes methodology, techniques, and policies to manage the project. In addition, Crystal focuses on people, interaction, community, development team skills, talents, and communication as its core principles. From lightest criticality to most serious project criticality, here are the eight levels of Crystal:

- Crystal Clear—small, easy projects
- Crystal Yellow
- Crystal Orange
- Crystal Orange Web
- Crystal Red
- Crystal Maroon
- Crystal Diamond
- Crystal Sapphire—top priority projects

The bigger the project, the more complex it is, and the more people involved, the more formal the project and the heavier the project management processes will become.

Exam Coach	Crystal isn't something that you'll need to know a great deal about for your exam. Just know that the colors of the Crystal approach represent different priority levels. I suspect you may see one or two questions on this approach.

Leading an Agile Project

Leading an agile project is different than leading a predictive project. Predictive projects plan everything, for the most part, up front. Agile projects follow a methodology of not being too invested in planning and instead focusing on handling chunks of work throughout the life cycle. An agile project manager can go by lots of different names: ScrumMaster, coach, team leader, or even facilitator. The responsibility of the agile project manager will vary based on the methodology selected and the organization, but as a rule, the agile project manager follows more of a servant leadership role and ensures that the team has the resources, knowledge, and tools to complete their project work. Sometimes the agile project manager is known to "carry the food and water for the project team." This doesn't mean you're literally carrying jugs of water and snacks for the team, but that you're getting the team the stuff they need so the team can continue to be productive.

Within agile there are four types of life cycles you'll be leading as a project manager, and you should recognize these characteristics for your exam:

- **Agile** Working with dynamic requirements for the development team to deliver working software frequently through small releases
- **Incremental** Working with dynamic requirements for the development team to quickly deliver working software through increments of product releases
- **Iterative** Working with dynamic requirements for the development team to create a single delivery of the software solution at the end of the project
- **Predictive** Working with fixed requirements for the project team to create a single delivery of the product, manage costs, and manage the project schedule

While the agile, incremental, and iterative life cycles have overlapping characteristics, there are some difference in the delivery

of the product the project is creating. Of course, predictive is the traditional approach to project management that you'll often see in industrial-work projects.

Comparing Management and Leadership in Agile Projects

There's an obvious difference between management and leadership. Management is about getting things done and leadership is about getting the project team to want to do what needs to be done. Leadership is about aligning, motivating, and inspiring the project team. Management is about command and control of the project tasks and directing people to get stuff done. Sure, there are some cute idioms about management and leadership, but the truth is that you need to know both approaches to complete a project successfully. Agile projects, and your PMI-ACP exam, will lean toward the attributes of leadership over management.

Make no mistake: all agile projects need management to ensure that the ceremonies, framework, and processes are being followed. Call that management facilitation, or mentoring, or coaching—it's all management to get things done. You, the agile project manager, must have some control over the project without getting in the way of the project team, creating unnecessary bureaucracy, or creating layers of approval that can delay delivering value quickly for the customers. Servant leaders focus on the purpose of the project, the people working on the project, and then the processes needed to complete the project.

Serving as a Leader

The project manager doesn't always have to be the leader in an agile project. Agile supports the idea of emergent leadership, where anyone can become the leader based on what's happening in the project. You will be seen, at least initially, as the project leader. You'll need to embrace these seven attributes of servant leadership:

- **Promoting self-awareness** Servant leaders use emotional intelligence to understand and control their emotions and to understand the emotions of others. This also includes how the servant leader behaves, responds, and takes time to think before speaking or acting.
- **Listening** Servant leaders listen to the project team and the stakeholders. This means listening to the whole message, body language, and nonverbal clues to really understand the message's meaning. By understanding the message, you can better serve the needs of the project.
- **Serving the project team** Servant leaders protect the project team from interruptions, remove any impediments the team reports or the project manager identifies, often communicate the project vision to everyone, and provide what the team needs to do their work.
- **Helping people grow** Servant leaders provide opportunities for people to learn from the project work and from one another. Often, when people are challenged and interested in what they're doing, they're happier, create better results, and are willing to take on tasks. Servant leaders promote collaboration and peer learning.
- **Coaching versus controlling** Servant leaders coach and mentor rather than command and control. The project manager may have to enforce certain rules and policies of the organization but should do so in a way that explains the policies so the team understands why they need to follow the policies.
- **Promoting safety, respect, and trust** Servant leaders ensure a safe environment for the team to experiment, to be creative and innovative without a fear of repercussions should they fail in their innovations. The team members need to show each other respect and need to trust one another. Trust also means the team must learn to trust the process, the approach, and be willing to collaborate with one another to create value.
- **Promoting the energy and intelligence of others** Servant leaders promote and encourage the team to be creative, to try new things, and to collaborate on ideas. The project team should be self-organizing to determine what's the best use of their time without the project manager being in control of every decision. The team members should be encouraged to practice emergent leadership where anyone can become a leader for the project team.

Visualizing Transparency

Transparency is a concept you'll see promoted in every agile approach. Transparency means that we're honest about both good news and bad news, and that we don't hide information about progress or setbacks in the project. To visualize transparency means that the project manager and the development team are open about their work's progress, the actual code of the work, and how well they're meeting the iteration's goals and objectives. For example, in Scrum, the team commits to taking on a chunk of the product backlog and then throughout the sprint communicates daily about what's been accomplished and any issues they've experienced in their progress.

Visualization is another key agile concept. Through signboards, charts, and dashboards, we openly communicate our progress. As previously mentioned, an information radiator is an agile concept of a wall of signs, charts, or even electronic displays that "radiates" information about the project work. An information radiator may be referred to informally as "a big, visible chart" or the stuffier "informative workspace." The information radiator visualizes that the team has nothing to hide from stakeholders, the project manager, and each other. Everything, good or bad, is out there for everyone to see.

Creating a Safe and Trustful Environment

I've seen, and perhaps you've seen it too, a manager encourage the team to experiment and try new methods of getting work done only to later criticize the team when their innovations didn't work out. That's just awful. In agile, project managers embrace innovation, empower the team to try new things, to be creative—all without the fear of repercussions. Because agile, for the most part, uses short bursts of work, if something isn't working out, the failure is considered fast, thereby avoiding a long, drawn-out failure by sticking to a plan that is doomed. It's better to experiment, fail early, learn from the mistake, and move on.

Fast failure is good for several reasons:

- The team now knows something that didn't work.
- New success is based on failed innovations.
- Failure is a learning opportunity for the whole project team.
- Failure without repercussion promotes trust and continued innovation.

The project team should be empowered to try new approaches, make decisions, learn from what didn't work, adjust their approaches, and try again. Too much studying, planning, and analysis can slow things down versus simply experimenting to see what'll happen. This isn't to say that the development team doesn't think before acting, but rather they don't overthink a problem that they could test and then move forward based on the results. An agile project manager wants the team to experiment and discover new ways of doing things to find innovations.

Encourage Emergent Leadership

Emergent leadership means that leaders emerge from the project team at different times of the project. Different team members will lead various initiatives of the project work based on what's happening within the project. When a new leader emerges, ideally a power struggle will not ensue over who's leading and all team members will acknowledge that others have talents to offer and should be followed when those talents are needed. This isn't a formal voting process to identify the new leader; rather, leaders are self-selected in an effort to help the team, not themselves.

Emergent leadership doesn't mean that the agile project manager ceases to offer leadership as new leaders emerge from the project team. The project manager has four distinct leadership roles:

- **Directing** When the team first comes together, called the *forming stage*, the project manager needs to direct the team, as they'll likely have low competence about the project's values and goals, but a high commitment to getting things done.
- **Coaching** When the teams shifts into storming, where there's a power struggle and conflict among the team, the project manager offers directions and supportive behavior to help the team move forward in the process of team development.
- **Supporting** As the team settles into their project roles, called the *norming phase* of team development, the project manager scales back directing but continues supportive behavior.
- **Delegating** When the team is working collaboratively and the project is moving forward, the team is in the *performing stage* of team development. During the performing stage, the project manager offers minimal directing and low supportive behavior. This means the project manager gets out of the way and allows emergent leaders to lead.

Chapter Summary

You're done with Chapter 1. Congrats!

In this chapter, we talked about the agile principles and the agile mindset you'll need to master for the PMI-ACP exam. Remember that the agile domain of Agile Principles and Mindset has nine tasks that center on the values of agile projects. Agile projects are creative, knowledge-work projects that deal with invisible work and complex requirements to create value for the project customers. This complex work requires the use of trial and error, observation, and innovation through empirical

processes rather than the use of defined processes as in industrial work.

You also reviewed the Agile Declaration of Interdependence, which outlines the values and principles of an agile project team. This document has just six items that create a framework of goals and behaviors for agile project managers and the project team. Based on this document, you also reviewed the difference of being agile and doing agile. Of course, your goal is to be agile, to embrace the agile mindset, and create a safe environment for your team to create value for the project customer. You'll also work to educate others in your organization about agile and why it's something the whole organization should embrace rather than just one individual or just one project team.

In this chapter you also took a quick look at the Agile Manifesto and its four values for the project. This is the document that defines how agile projects value people and interactions over processes and tools, working software over comprehensive documentation, customer collaboration over contract negotiation, and responding to change over following a plan. You also explored the 12 principles of agile that support the goals of the Agile Manifesto.

Next, you walked through the various agile project management approaches. This discussion included an in-depth look at Scrum and XP, two approaches you'll likely see on the PMI-ACP exam. Other, less popular approaches you might encounter on the PMI-ACP exam include Kanban, Lean Product Development (Lean), Dynamic Systems Development Method (DSDM), Feature-Driven Development (FDD), and Crystal. Finally, we wrapped things up with a frank talk about agile project leaders and the concept of emergent leadership.

Keep working toward your goal of earning the PMI-ACP credential. You've already done more than many who say they want to earn the PMI-ACP: you've read the first chapter of the guide that aims to get you there. Keep working smart, stay focused, stay confident. You can do this!

Key Terms

Agile life cycle
: The agile life cycle works with dynamic requirements for the development team to deliver working software frequently through small releases.

Agile Manifesto
: The Agile Manifesto is a broad document that establishes the values of agile project management. The four values are individuals and interactions over processes and tools; working software over comprehensive documentation; customer collaboration over contract negotiation; responding to change over following a plan.

Agile mindset
: The agile mindset is a way of thinking about and doing agile projects. Having the agile mindset means you exemplify the values and principles of agile in how you work, how you lead a project, and how you share a passion for agile with others.

Backlog refinement meeting
: A backlog refinement meeting is a Scrum event where the product owner, the ScrumMaster, and the development team work together to discuss the backlog items and prioritize the items.

Coach
: This is an XP role that is like a project manager but is more of a mentor and facilitator. The coach mentors people on the project team, helps get things done, and serves as the hub of communications for the project stakeholders.

Code standards
: An XP concept is that all programmers on the development team follow defined and communicated standards for developing the code.

Collective code ownership
: An XP concept is that any pair of programmers on the development team can edit anyone's code.

Crystal
: Crystal is an agile approach that uses eight different schemes of agile project management based on several factors, such as the complexity of the project, the number of project team members, and the criticality of the project.

Daily Scrum
: This is a daily 15-minute meeting during which the team members share progress updates; it is sometimes called a standup meeting because everyone stands for the duration. The Daily Scrum happens every day at the same time, in the same place.

Development team
: The development team is responsible for sizing the requirements of the product backlog and getting work done in each sprint. The development team is self-organizing and self-led, and its members are called generalizing specialists because they can often do more than one function on the team. An ideal Scrum team has no less than five people and no more than eleven people.

Dynamic Systems Development Method (DSDM)
: DSDM is one of the predecessors of today's agile project management and relied on a business case to show value and a

feasibility study to determine if the development team could create the architecture and requirements the customer identified.

Empirical processes
Empirical processes are based on observation, trial and error, and the experience of the person doing the work. Agile projects rely on the knowledge worker to be creative, innovative, and to figure out the work to reach the desired results.

Feature-Driven Development (FDD)
FDD is an iterative approach to software development that bases its progress on the client's values of features the software will provide.

Incremental life cycle
The incremental life cycle works with dynamic requirements for the development team to quickly deliver working software through increments of product releases.

Industrial-work projects
Industrial-work projects utilize defined approaches to complete processes and tasks; the project team members know exactly what to do and what to expect in the project and in the project management approach. Construction is an example of an industrial-work project.

Inverted triple constraint
Agile projects invert the traditional triple constraints (or iron triangle) so that time and cost are fixed, but the scope is now flexible.

Iterative life cycle
The iterative life cycle works with dynamic requirements for the development team to create a single delivery of the software solution at the end of the project.

Kanban
Kanban is a Japanese word that means signboard. This approach started at Toyota and it helps to visualize the flow of the work through a system.

Knowledge-work projects
Agile projects are knowledge-work projects and are driven by creativity and brain power, rather than brawn and effort applied to defined processes and tasks. Software development is an example of a knowledge-work projects.

Lean Product Development
Commonly called Lean, this agile approach uses a visualization of what needs to be done, creates requirements based on the customer's definition of value, and includes opportunity for learning and process improvement throughout the project.

Little's Law
A theorem that states "the average number of items in a queuing system equals the average rate at which items arrive multiplied by the average time that an item spends in the system." In other words, the more items that are in the WIP queue, the longer it will take the team to complete the items in the queue.

Metaphor
An XP practice is to create a metaphor for the project. A metaphor explains the goal and function in plain, understandable language for all stakeholders.

Pair programming
This is an XP approach where developers work in pairs; one person codes while the other checks the code. The pair switches roles periodically.

Predictive life cycle
The predictive life cycle works with fixed requirements for the project team to create a single delivery of the product, manage costs, and manage the project schedule.

Product backlog
The product backlog is the long list of prioritized project requirements. The product owner is responsible for maintaining the product backlog.

Product owner
This is a role in the Scrum agile methodology that describes the individual that manages the product backlog for the project.

Refactoring
Refactoring involves cleaning up the code to remove waste, redundancy, dependent connections, and shortcuts.

Scrum of Scrums Scrum
This is a meeting for larger Scrum projects where multiple teams are working together. Rather than having a huge Daily Scrum, the teams meet separately and then a representative from each team meets in a Scrum to report on each team's progress. The team representatives answer the same questions as in the Daily Scrum, but for the team rather than individuals. In addition, a fourth question is often posed: Will our team be putting something in another team's way?

ScrumMaster
The ScrumMaster ensures that everyone understands the rules of Scrum, removes impediments for the team, facilities Scrum meetings, helps the product owner groom the backlog, and communicates the vision of the project to everyone that's involved.

Sprint planning meeting

In this Scrum event, the development team determines how much work they can take on from the prioritized backlog for the next sprint. This determination is based on estimates of the items in the product backlog and past sprints. The selected items from the product backlog become the sprint backlog and the goal of the sprint.

Sprint retrospective

This Scrum event is held after the sprint review and before the next sprint planning meeting. The team discusses what's worked well with people, product, and processes in the project, what needs improvement, and the feedback from the product owner from the sprint review meeting.

Sprint review

This Scrum meeting is held at the end of each sprint. The development team demonstrates for the product owner, the ScrumMaster, and other key stakeholders the work they've accomplished in the past sprint. This review is for the product owner to offer feedback on whether the work has reached the done stage, and, if it hasn't, describe what's missing and elaborate on corrections or modifications for the increment of work created.

Sprints

Scrum uses timeboxed iterations, called sprints, to create prioritized requirements for the customer. Sprints last from two to four weeks to complete the selected requirements for the current iteration.

Technical debt

Technical debt is sloppy code, shortcuts, and redundancies that need to be cleaned up as the project moves forward. Technical debt can accumulate and cause the project code to become more complex.

Test-driven development

In test-driven development, an acceptance test is written before the code is written so the developers know what it takes to pass the acceptance test and can program accordingly.

The Declaration of Interdependence

This agile methodology document serves as a value system for agile project managers. The idea behind this document is that all participants in an agile project are interdependent on one another: the project manager, development team, customers, and other stakeholders.

WIP (Work in progress)

The work in progress is what the team is working on right now.

XP

XP is short for eXtreme Programming and is an agile approach that is focused on software development. XP also uses iterations of development, but these iterations last for two weeks in a typical project.

Questions and Answers

1. You are the agile project manager for your organization and your current project is to manage a team developing software that will affect the business practices of the organization. When working on an agile project, processes, tools, documentation, and plans are necessary. However, which of the following should be the initial focus? ?

A. Scope of the project

B. People involved in the project

C. The individuals and interactions involved

D. Definition of success as defined by the project manager

2. A diverse agile team has been assembled by upper management, and there is confusion about the requirements and definition of success of the project. What might have been a better method of creating the team? ?

A. Select a person from each team with interest in the project

B. Allow a self-organizing team from the people that helped create the requirements and definition of success

C. Have the business partner select the members she wants on the project

D. Have the business partner hire a team of contractors that have to be "sold" on the project

3. What is one item that should be accomplished during a sprint planning meeting? ?

A. The product owner shares the updated backlog, and the team discusses it to ensure a shared understanding.

B. Three questions—what have I accomplished?, what do I plan to accomplish?, and are there any roadblocks?—are answered by each participant.

C. The product owner decides if the product is done.

D. The scope and costs of the project are renegotiated.

4. A good practice in keeping a project on track is to minimize waste. Which of the following could be eliminated from a project to maximize value?

A. Daily meetings

B. Unnecessary features

C. Testing

D. Sprint reviews

5. Tom is the project manager for his organization and he's educating the project team and key stakeholders about Scrum. Some of the team members are confused about the activities that happen after the sprint review. What activity should Tom indicate is completed after the sprint review?

A. Product backlog refinement

B. Sprint planning meeting

C. Daily Scrum

D. Sprint retrospective

6. Samantha is the ScrumMaster for her organization and she's working with the team and the product owner to refine the product backlog. What is the role of the team in this meeting?

A. Prioritize the work items

B. Add features to the project backlog

C. Estimate and refine the work items

D. Identify fixes

7. Which of the following describes how Kanban differs from agile?

A. Kanban teams plan their work in sprints or iterations.

B. Kanban teams work on a project as a whole.

C. Kanban teams employ a pull system.

D. There are no WIP limits in Kanban.

8. Pat has been selected to lead the RGW Project with Scrum methodologies. The project team is excited about this decision. Select which trait has made the team happy about this decision.

A. Pat likes to work within the box.

B. Pat has been with the company for 12 years.

C. Pat is honest and explains why decisions are made.

D. Pat will lead according to the project plan.

9. You are the project manager of the JLK Project utilizing Scrum. Several stakeholders have lost interest in the project and rarely attend meetings, offer input, or give feedback. What is a good practice to avoid this pitfall going forward?

A. Making project meetings mandatory

B. Meeting with each business partner on a one-on-one basis

C. Proving that everyone's ideas have value and taking advantage of small-scale experiments

D. Sticking strictly to the project plan

10. You happen to bump into a business partner of the project you are working on, and he asks you how many defects were found in the prior week. Since you don't know the exact answer, what should you do?

A. Take a guess

B. Tell the business partner you will send an e-mail to him with the information when you get back to your desk

C. Take him into the team area, where much information regarding the project is displayed

D. Ask the business partner to call the team leader to get a correct answer

11. The agile team you are working with encourages each other to suggest innovative ideas. Most ideas are tested to determine whether they will be successful or may not work very well. How is this accomplished without delaying the project? ?

A. The new ideas are only tested if there is sufficient time in the project plan.

B. The ideas are tested during short iterations and end before the next iteration.

C. Only one idea is accepted by the leader during an iteration.

D. Only the leader determines what ideas to test and what to put aside.

12. What does the term *emergent leadership* mean in an agile environment? ?

A. A new leader is assigned for each iteration.

B. A team member takes over the leadership role.

C. A team member tries a new approach after getting approval from the team.

D. A leader isn't in place until the team selects one of the team members.

13. How is an agile team better prepared to suggest solutions to a business request versus a team that is not using an agile methodology? ?

A. The agile team is made up of business people.

B. The agile team worked with the business partner to prepare the project plan.

C. The agile team and business partners work together throughout the project life cycle.

D. The business partners review statuses and ask for recommendations.

14. As a PMI-ACP candidate you need to recognize the terms and techniques utilized in agile projects. Which one of the following is the best example of an empirical process? ?

A. Following a plan

B. Observing results and adapting the code

C. Making collaborative decisions

D. Working alone on a project requirement

15. You are the agile project manager for your organization and you're helping your development team better understand agile. You have reviewed the Agile Manifesto with your team. Which one of the following is not an attribute of the Agile Manifesto? ?

A. Promoting collaboration

B. Expecting changes to the product backlog

C. Documenting how the code is to be developed

D. Accepting tradeoffs with the vendor

16. Elizabeth is the project manager of an XP project. Her manager stops by to check on the progress and notices that the developers are working in groups of two rather than individually. What is this approach called? ?

A. Pair programming

B. Pared programming

C. People-People Programming

D. P2P (Programmer-to-Programmer)

17. Jane is a senior project manager in your company. Wally is a new project manager who is not familiar with the Scrum methodology. Jane explains the five Scrum values to Wally. Which one of the following is not one of the five values of Scrum? ?

A. Commitment

B. Focus

C. Respect

D. Innovation

18. Erin is a new project manager who is working on a Scrum project. She's confused on the difference between a sprint review and a sprint retrospective. What's the difference?

A. Sprint reviews are demonstrations of the work completed in the sprint. Sprint retrospectives are demonstrations of all the compiled work completed in the project.

B. Sprint review are for lessons learned. Sprint retrospectives are for product demonstrations.

C. Sprint reviews are for product demonstrations. Sprint retrospectives are for lessons learned.

D. Sprint reviews discuss what's worked in the sprint. Sprint retrospectives are done at the end of the project for a lessons learned opportunity.

19. You are the project manager of the BNK Project for your organization. This project is utilizing the XP framework. What is your title in this XP project?

A. Project manager

B. Product manager

C. Coach

D. Team leader

20. Your organization uses the Lean agile project management approach. One of the principles of Lean is to remove waste. Which one of the following is an example of waste in a Lean project that should be removed?

A. WIP

B. Bottleneck identification

C. Motion

D. Colocation

Answers

1. **C**. While processes and tools will likely be necessary on the project, the focus of the agile team's attention should be on the individuals and interactions involved. Projects are accepted by people, people debate scope, and people negotiate the definition of done. Focusing early on developing the individuals involved in the project and emphasizing productive and effective interactions help set up a project for success.

   A is incorrect because the scope of the project will likely change throughout the project. B is incorrect because this choice isn't the best answer. People involved in the project can include all the stakeholders, rather than only interactions of the individuals involved. D is incorrect because although the definition of success is important, it's not the initial focus of the project manager.
2. **B**. A self-organizing team composed of the people who created the requirements and defined success has a higher level of ownership and pride in the requirements and design, so the members don't have to be "sold" on the project. Ideas that come from outside resources need to be sold to the team for the implementation to be successful, which sometimes creates a challenging task.

   A is incorrect because although people on teams that were not part of the requirements gathering may have more merit on paper, they may want to implement the project differently than what was originally envisioned. C is incorrect because this solution creates the same problem of choosing people who are already invested in the value and understanding of the project goals. D is incorrect because it's best to utilize internal resources that already understand the project goals rather than to hire contractors who have to be sold on the value of the project.
3. **A**. The product owner should share the updated backlog items and ensure the entire team has a good understanding of how to move forward.

   B is incorrect because the questions regarding accomplishments and roadblocks are answered during Daily Scrums to uncover any obstacles. C is incorrect because the sprint planning meeting is not where the project's success is discussed. D is incorrect because the scope of the project is based on the prioritized requirements.
4. **B**. In addition to eliminating the introduction of unnecessary features, value can be maximized by eliminating partially done work, delays, extra processes and features, task switching, waiting, moving information or a deliverable, defects, and handoffs. A, C, and D are incorrect because daily meetings, testing, and sprint reviews are just a few of the activities required for a successful project.
5. **D**. After a sprint review but before the sprint planning meeting, a sprint retrospective should be held to gather lessons learned and look for opportunities for improvement. This allows the team to consider the owner's feedback and implement improvements before the next sprint.

   A is incorrect because product backlog refinement is done in product refinement meetings. B is incorrect because sprint planning meetings happen after the product refinement meeting. C is incorrect because the Daily Scrum is hosted every day at the same time and same location.
6. **C**. The team is responsible for estimating and refining work items. Each time the team refines their estimates with a higher level of detail, the product owner might need to adjust the priority of those items.

A is incorrect because the product owner and the team collaborate on prioritizing the requirements; ultimately, the prioritization is the responsibility of the product owner. B is incorrect because the team doesn't add features to the backlog. D is incorrect because the team doesn't identify fixes to the project in the refinement of the product backlog.

7. **C**. The main difference is that Kanban teams employ a pull system, which means that when an item of work is completed, it triggers a pull to bring in the next item in the queue to work on. Kanban teams also work off a Kanban board that displays columns of processes, each with a work in progress (WIP) limit.

   A is incorrect because agile teams, not Kanban teams, work in sprints before moving on to the next sprint backlog. B is incorrect because Kanban teams don't work on the whole project at once, but in chunks of requirements. D is incorrect because there is a WIP limit in Kanban.

8. **C**. Because Pat is honest and will communicate to the team the reasons for his decisions, he is a suitable ScrumMaster and will be less focused on command and control and more focused on being a servant leader.

   A is incorrect because agile calls for flexibility and adaptability in a project. B is incorrect because Pat may be with the company for many years and may be very competent, but may not know enough about the project to move it to success. D is incorrect because the leader must be able to accept changes and be willing to adjust plans.

9. **C**. Allowing stakeholders to suggest new ideas, and then giving them a chance to experiment with new concepts, proves that everyone's ideas have value. Nothing is more discouraging than having a good idea disregarded. Keeping stakeholders engaged allows the opportunity that agile projects present for small-scale, localized experiments in a supportive, low-risk environment.

   A is incorrect because mandatory meetings won't bolster stakeholder engagement. B is incorrect because meeting with each business partner one-on-one may be a good way to communicate, but it's not the best choice available. D is incorrect because agile plans change based on the product backlog.

10. **C**. A core task of the agile leader is to be transparent, not only about progress, but also about issues and roadblocks. It is normal to walk into an agile team's area and see graphs showing velocity, defect rates, and results of retrospectives including what is working well and what needs improvement. When the leader is open and honest, it fosters openness where people can be less guarded and focus on improvements.

    A is incorrect because guessing isn't an accurate or transparent way to communicate the project status on defects. B is incorrect because e-mailing the answer may be viable, but it's not the best way to be open and transparent about the project. D is incorrect because asking the business partner to call the project team lead can be a disruption and doesn't provide the readily available information.

11. **B**. New processes or techniques are tested during short iterations, enabling the team to either implement the successful idea or not, and before the next iteration. This is part of the concept of being innovative, and if necessary, failing fast, learning from the experiment, and moving on.

    A is incorrect because additional time doesn't need to be accounted for when iterations are typically between two and four weeks, and a decision will be made quickly. C is incorrect because there is no limit to good ideas and the team should decide whether to move forward with a concept. D is incorrect because the team is self-led and can determine what ideas should be tried and implemented.

12. **C**. An emergent leader is a team member who takes the initiative to try a new process or idea once they have the team's approval.

    A is incorrect because emergent leadership allows a leader to emerge at any point in the project, not by assignment. B is incorrect because emergent leadership is not about a formal role and assignment as leader. D is incorrect because leaders emerge based on conditions in the project, not by assignment or voting.

13. **C**. Working with business partners almost daily throughout the project life cycle is far more valuable and collaborative than meeting occasionally. The team hears what the business would like the results to be, and meeting face-to-face is far better than reading requirements, documents, and e-mails, or even a conference call. Another benefit of daily interactions is that the business partners will learn what task or activity might be far more expensive than another solution or take far longer than they expect.

    A is incorrect because the team is made up of developers, not just business people. B is incorrect because while the team collaborates with the business partner, the team creates a plan based on the sprint backlog. D is incorrect because the business partners may review statuses and ask for recommendations, but this isn't the best option available. For your PMI-ACP exam, you must always choose the best answer, even if more than an one answer is acceptable.

14. **B**. Empirical processes require observation, creativity, and trial and error, so observing results and adapting the code is the best answer.

    A is incorrect because following a plan describes a defined process. C is incorrect because making collaborative decisions isn't an empirical process. D is incorrect because working alone can involve utilizing either an empirical process or a defined process.

15. **C**. One of the values of the Agile Manifesto is that we value working software over comprehensive documentation. Documentation should be light and minimally sufficient.

    A is incorrect because team collaboration is an aspect of the Agile Manifesto. B is incorrect because responding to changes over following a plan is also a value of the Agile Manifesto. D is incorrect because customer collaboration over contract negotiation is a value of the Agile Manifesto.

16. **A**. XP utilizes pair programming where one developer codes and the second developer checks the code.

    B, C, and D are incorrect choices as these are not valid descriptions of pair programming in XP.

17. **D**. Innovation, while encouraged, is not one of the five values of Scrum. The five values are commitment, focus, openness, respect, and courage.

    A, B, and C are incorrect because these choices are part of the five values of Scrum.

18. **C**. Sprint reviews are for product demonstrations. Sprint retrospectives are for lessons learned.

    A, B, and D are incorrect descriptions of the sprint reviews and sprint retrospectives.

19. **C**. A project manager in an XP environment is called a coach.

    A, B, and D are incorrect as XP calls the project manager a coach, not a project manager, product manager, or team leader.

20. **C**. Motion is a waste in Lean projects. Moving items from person to person is motion and that takes time and creates waste. If there's not a clear workflow of how items move through the project processes, then there's waste.

    A is incorrect because work in progress (WIP) is needed in Lean and is not waste. B is incorrect because bottlenecks are wasteful, but the identification of a bottleneck is not waste. D is incorrect because colocation is a desirable aspect for the project team.