

# Chapter 2: Project Life Cycles, Phases, and Process Groups

## CHAPTER 2 PREVIEW

- Distinguish among predictive, iterative, incremental, and agile project life cycles as well as hybrid approaches.
- Discover how to split projects into phases to enable incremental value delivery using any type of project life cycle.
- Explore what is done during the overlapping Process Groups of Initiating, Planning, Executing, Monitoring and Controlling, and Closing.
- Understand the Agile Manifesto and the principles that underly agile project management methodologies.
- Discover the benefits of learning about unfamiliar project methodologies.

This chapter addresses the basics of the project life cycles, project phases, and Process Groups. First, project life cycles and phases are introduced. Then, life cycles are addressed in more detail: the chapter presents core concepts of predictive projects, including introducing the Process Groups, then goes on to present the core concepts of agile and hybrid projects.

Note that this chapter is intended to give you just enough information to put the discussions in this and the next module into a project management context. The more detailed discussions of project management processes are reserved for Modules 3 through 6.

## Project Life Cycles and Project Phases

Here we discuss the concept of project life cycles and how they are typically planned and executed, the characteristics of life cycles, options relating to cadence and life cycle types, and project phases.

### Progressive Elaboration

The *PMBOK<sup>®</sup> Guide—Seventh Edition* defines a **project life cycle** as “the series of phases that a project passes through from its start to its completion.” A **project phase** is “a collection of logically related project activities that culminates in the completion of one or more deliverables.” A life cycle provides a basic framework for managing a project, and the phases of the chosen life cycle do not change based on the specific project work required. According to the *PMBOK<sup>®</sup> Guide—Seventh Edition*, a **method** is “a means for achieving an outcome, output, result, or project deliverable.” The tools and choices made during these life cycles comprise a **methodology**, which the *PMBOK<sup>®</sup> Guide—Seventh Edition* defines as “a system of practices, techniques, procedures, and rules used by those who work in a discipline.”

Most projects, even those with just one phase, are typically planned and executed using what is called progressive elaboration.

The *PMBOK® Guide—Seventh Edition* defines **progressive elaboration** as “the iterative process of increasing the level of detail in a project management plan as greater amounts of information and more accurate estimates become available.”

You need to start planning somewhere even though you know information from other planning areas will influence the part of the plan you are working on. For example, when you are working on the initial schedule and cost estimates, you haven’t yet worked on the quality plan or the risk register. Quality control activities and risk responses take time and money, which will either directly impact the schedule and budget or take the form of reserves. Therefore, you will go back and revise these initial plans as you go through the various planning steps.

You will not start out with perfect estimates. As the relevant events get closer or earlier phases are completed, a more detailed level of information will be available and estimates will be more precise. Since the phases of agile projects are typically very short, the progressive elaboration in this case is meant to occur between each phase. (Changes to plans during a phase are actively discouraged.) As projects progress, project managers will conduct a **project review**, which the *PMBOK® Guide—Seventh Edition* defines as “an event at the end of a phase or project to assess the status, evaluate the value delivered, and determine if the project is ready to move to the next phase or transition to operations.”

Project managers view plan refinement as a natural part of planning that allows more precise baseline specification, measurement, and management as the project evolves.

## Characteristics of Life Cycles

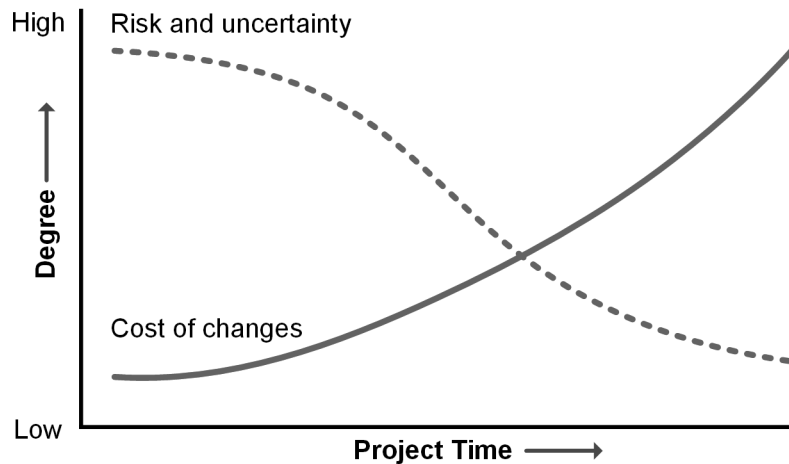
There are some common characteristics of project life cycles that can influence how an organization approaches complex or high-risk projects.

- **Cost and staffing curve.** In general, costs and staffing for a predictive project life cycle are low during the earlier Process Groups—Initiating and the first pass at Planning—because few team members are on board and the project is using fewer resources. Costs increase quickly during the later Process Groups of Executing and Monitoring and Controlling, when most of the project work is being done. Costs begin to decrease at the end of Executing and Monitoring and Controlling as deliverables are completed and resources are released. Costs decrease more quickly during the final Process Group, Closing. On an agile or hybrid project, early phases may likewise have lower costs if teams start small, and costs could ramp up if the work shows promise and additional resources are devoted to the project. Teams may also be kept the same size, in which case the cost and staffing curve will be more flat and, as a side benefit, predictable. Stable teams could be small or large (e.g., to represent many cross-functional interests).

- **Risk/uncertainty curve.** In general, uncertainty about a project’s ability to meet its objectives decreases as the project progresses through the project life cycle. This makes sense. At the beginning of any project, all risk is in the future. As the project proceeds, decisions are made, knowledge and experience increase, and steps can be taken to manage risk more effectively. For example, a predictive project aiming at producing a new type of manufacturing equipment has more risk during Initiating and Planning, because the sponsor and the project team don’t know how or if the objectives can actually be achieved in proposed designs and if the designs can be translated into production within scope, budget, and time frames. Once a project is accepted by the customer, no further risk exists. On an agile/hybrid project, as phases are completed and their deliverables are accepted, the level of risk and uncertainty gradually is reduced.
- **Cost of changes curve.** For a predictive project, the least expensive time to change scope or product characteristics is early on, before work has been done and resources spent. As the project work progresses, the ability to make changes without significantly affecting cost and schedule (and possibly other measures such as quality and project team motivation and engagement) decreases. Changing course may mean that investments made as a result of earlier decisions have been wasted. Work based on a mistake or incorporating a flawed element has to be redone. Changes may have a “ripple effect” on work that has already been completed. A team working on new manufacturing equipment could handle this issue by proving their design concepts through a prototype test before proceeding to building the equipment. (Prototyping is an example of an agile mindset.) This would decrease the risk of later changes to project baselines. Agile, by design, works to keep the cost of changes lower throughout a project so that changes can be accommodated even late in the project.

The way in which the cost of change increases as a project progresses through various phases is shown in Exhibit 1-3. Note that the phase names used in this graphic may be different from the phase names used elsewhere in these materials or those you will encounter in the field. Organizations may use many different names to describe product phases.

### Exhibit 1-3: Risk and Uncertainty Decrease, but Cost of Changes Increases as Project Progresses



Source: Project Management Institute, *A Guide to the Project Management Body of Knowledge (PMBOK® Guide)—Sixth Edition*, Project Management Institute, Inc., 2017, Figure 1-3, Page 549. Material from this publication has been reproduced with the permission of PMI.

Not all projects are alike, however. In some projects, costs are front-loaded or rise quickly—for example, when a project requires very expensive architectural planning up front. In some projects, uncertainty is very difficult to manage. For example, a new building may include features that have never been built before or materials that have never been used in construction of this type. To apply heightened levels of managerial oversight, organizations can divide complex predictive projects into more phases, or they can use iterative, incremental, or agile life cycle types.

### Cadence Choices

The *PMBOK® Guide—Sixth Edition* discusses how cadence choices refer to the timing and frequency expected for project deliverables. These choices are broken down into four delivery options:

- **Single delivery.** These projects have a single delivery of any and all intended deliverables at the end of the project.
- **Multiple delivery.** These projects have multiple deliveries throughout the course of the project, though they may not be on a fixed schedule. Deliveries may be developed sequentially or as separate parts of the product.
- **Periodic delivery.** These projects have multiple deliveries, but the schedule is fixed, such as biweekly or monthly.
- **Continuous delivery.** These projects deliver features to customers immediately upon creation, such as for digital products. This works best with project teams that are stable and do not change over time.

## Life Cycle Types

The *PMBOK® Guide—Sixth Edition* mentions four main types of project life cycles that work for particular types of projects. These are summarized in Exhibit 1-4. There can also be a hybrid life cycle, which combines aspects of all four types.

**Exhibit 1-4: Project Life Cycle Types**

Approach	Requirements	Activities	Delivery	Goal
Predictive	Fixed	Performed once for the entire project	Single delivery	Management of cost
Iterative	Dynamic	Repeated until correct	Single delivery	Correctness of solution
Incremental	Dynamic	Performed once for a given increment	Frequent smaller deliveries	Speed
Agile	Dynamic	Repeated until correct	Frequent small deliveries	Customer value via frequent deliveries and feedback

Source: Project Management Institute, *Agile Practice Guide*, Project Management Institute, Inc., 2017, Table 3-1, Page 18. Material from this publication has been reproduced with the permission of PMI.

Each of these project life cycles may manage uncertainty by using **rolling wave planning**. The *PMBOK® Guide—Seventh Edition* describes this as “an iterative planning method in which the work to be accomplished in the near term is planned in detail, while the work in the future is planned at a higher level.” The first wave of project work appears on the horizon, and the team busily prepares to manage it. While that wave dissipates, the team turns its attention to planning its response to the next wave. It can see this wave more clearly now, and it may understand how to manage it better because of the experience and deliverables generated during the previous wave. Agile projects in particular rely on rolling wave planning between phases.

Note that the four project life cycle types work on a continuum, so the iterative and incremental types tend to overlap with other life cycle types. This means that, when depicted as seen in Exhibit 1-5, the values given for an iterative project may overlap more with the predictive (left) column and the values given for an incremental project may overlap more with the agile (right) column.

**Exhibit 1-5: Project Life Cycle Continuum**

Predictive	Iterative — Incremental	Agile
Requirements are defined up-front before development begins	Requirements can be elaborated at periodic intervals during delivery	Requirements are elaborated frequently during delivery
Deliver plans for the eventual deliverable. Then deliver only a single final product at the end of the project timeline	Delivery can be divided into subsets of the overall product	Delivery occurs frequently with customer-valued subsets of the overall product
Change is constrained as much as possible	Change is incorporated at periodic intervals	Change is incorporated in real-time during delivery
Key stakeholders are involved at specific milestones	Key stakeholders are regularly involved	Key stakeholders are continuously involved
Risk and cost are controlled by detailed planning of mostly knowable considerations	Risk and cost are controlled by progressively elaborating the plans with new information	Risk and cost are controlled as requirements and constraints emerge

Source: Project Management Institute, A Guide to the Project Management Body of Knowledge (PMBOK® Guide)—Sixth Edition, Project Management Institute, Inc., 2017, Figure X3-1, Page 666. Material from this publication has been reproduced with the permission of PMI.

**Predictive**

The *PMBOK® Guide—Seventh Edition* describes the **predictive approach**, also called the fully plan-driven approach, as “a development approach in which the project scope, time, and cost are determined in the early phases of the life cycle.” A predictive cycle can also be called a waterfall cycle, because each step flows down to the next without repeating prior steps. (However, progressive elaboration will still occur during planning.) The ability to develop detailed plans in advance of execution allows the project manager to predict an unfolding sequence of activities that will take place in each phase, therefore avoiding risk and reducing costs. One can associate predictive life cycles with the image of a falling line of dominoes. If you’ve seen a falling dominoes display, you know that individual dominoes are first carefully laid out in an exact pattern. The “domino master” (or project manager, for our purposes) tips the first domino. Unless there has been a fault in planning or some unexpected and uncontrollable external force intervenes (imagine the effect of an earthquake), the dominoes will fall in the anticipated order and produce a satisfying result. (Clearly, a project manager needs to stay more involved than this, but it is just an analogy.) Similarly, in a predictive project life cycle, the phases unfold and develop in the anticipated manner. Complex domino displays and projects with predictive life cycles can also unfold in sequential and overlapping phases.

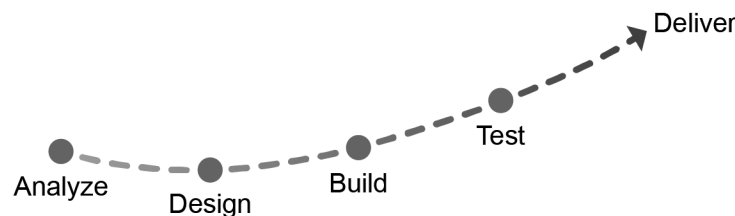
Each phase focuses on work that is essentially different from that in the preceding phase and may involve different team members. The deliverable is not in a form acceptable to the customer until the end of the project.

Although the scope is well-defined at the beginning, planning for later phases may not be as detailed. As the project life cycle proceeds, rolling wave planning may be used to plan the next phase in detail. Scope changes are controlled carefully, however. Changes in requirements that occur in late phases can create significant changes to a project's schedule and budget. As a result, any changes to the project scope are carefully managed.

Predictive life cycles are best suited to projects that have easily defined, well understood deliverables. Think about the manufacturing of a new school bus model. Although the project may include some new engine or dashboard technology and new kinds of seats or seating configurations, the customers know the requirements for a school bus and the suppliers know what it takes to manufacture one.

Exhibit 1-6 shows an example of a predictive life cycle.

**Exhibit 1-6: Predictive Life Cycle**



Source: Project Management Institute, *Agile Practice Guide*, Project Management Institute, Inc., 2017, Figure 3-2, Page 21. Material from this publication has been reproduced with the permission of PMI.

## Iterative and Incremental

Iterative and incremental life cycles are different. For example, their goals differ. The iterative life cycle's goal is correctness of the solution; the incremental life cycle's goal is speed. However, there is significant overlap between the two types. Also, an iterative life cycle uses increments to some degree and an incremental life cycle uses iterations to some degree. How iterations and increments are applied differs between each method, as is seen by reviewing their definitions from the *PMBOK® Guide—Sixth Edition*.

In an **iterative life cycle**,

the project scope is generally determined early in the project life cycle, but time and cost estimates are routinely modified as the project team's understanding of the product increases. Iterations develop the product through a series of repeated cycles, while increments successively add to the functionality of the product.

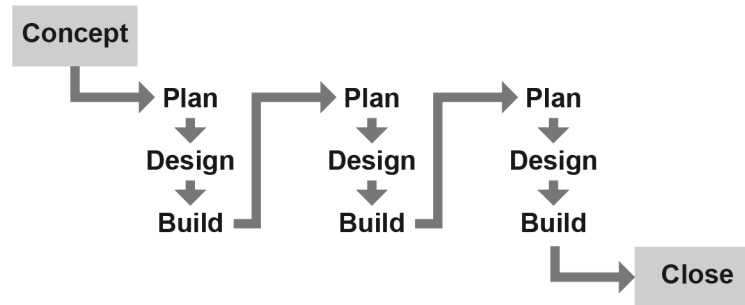
The reason this life cycle is called iterative is that repeating the cycles is the key element used to drive value by ensuring that this is the correct solution. While there are increments, the increments are not distinct deliverables but rather features or refinements of a single deliverable per phase. (The *PMBOK® Guide—Seventh Edition* defines a **feature** as “a set of related requirements or functionalities that provides value to an organization.”) Repetition (or iteration) of one or more project activities deepens the project team’s (and possibly the customer’s) understanding of the product, thereby allowing them to progressively develop plans to avoid risk. Rolling wave planning may be used to add detail to plans for the next iteration. A project with one or more prototypes but only one final deliverable is an example. The prototypes could be used to add functionality or take the project in an unexpected direction.

**An incremental life cycle is**

an adaptive project life cycle in which the deliverable is produced through a series of iterations that successively add functionality within a predetermined time frame. The deliverable contains the necessary and sufficient capability to be complete only after the final iteration.

In an incremental project, the deliverable (product, service, or other result) is run through a series of predictive or waterfall iterations (one iteration per phase), each of which produces a feature, component, or unit of the overall deliverable. While this method uses iterations to produce each partial deliverable, it is the increments that are the most important, because each increment can be used to gather early customer feedback or to fast-track other phases/projects. The increment might also be released to operations. Speed of delivery is enhanced by planning each increment one at a time. Unlike a predictive project, this allows requirements to be dynamic for the parts of the project that have yet to be planned and executed. Note that the overall deliverable will be considered finished—i.e., it contains all agreed-upon capabilities, qualities, or features—only after the last iteration. For example, an incremental townhome complex project completes a block of similar units at a time. The first few units are used to demonstrate various options to potential buyers, who can select customizations for the units that are yet to be built. Some units are sold and people move into them before the final units are even started. Exhibit 1-7 shows an example of a life cycle featuring an incremental development approach.



**Exhibit 1-7: Sample Life Cycle with Incremental Development Approach**

Source: Project Management Institute, *PMBOK Guide*®—*Seventh Edition*, Project Management Institute, Inc., 2021, Figure 2-10, Page 44. Material from this publication has been reproduced with the permission of PMI.

Project managers prefer using iterative and/or incremental cycles when:

- The project is too complex to plan from the start and feedback or lessons learned can be used to improve later phases.
- The scope and objectives are subject to change as stakeholders learn more.
- Partial or incremental deliverables are valuable to stakeholders. This is a distinct difference between the predictive and iterative approaches. The deliverable is available only at the end of the predictive or purely iterative project; incremental projects can provide deliverables at each phase that might be put to use. The overall project may not be considered complete yet, but the increments provided so far could be used to train end users, to prove that useful work is getting done, and so on. Incremental deliverables are avoided when a partial product would be detrimental to the final deliverable, such as a product that provides value to the customer only once it is fully complete, that would provide too much information to competitors without fully satisfying customers, or that would confuse customers.

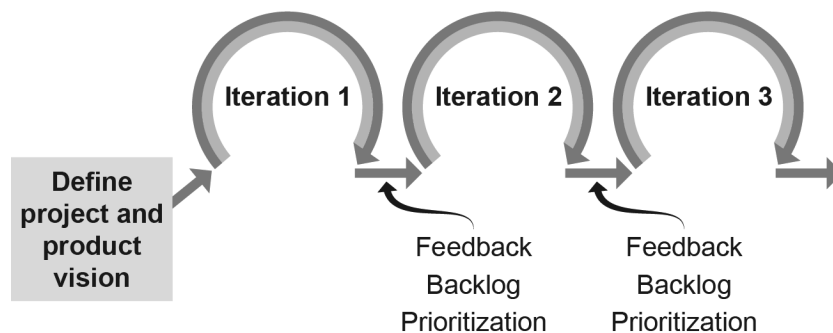
Another advantage of iterative or incremental life cycles is that the project manager can plan the next phase while the project team executes the current phase. In this way, only a high-level vision of the scope of work needs to be determined initially. Elements of the plan may be only tentatively assigned to a particular iteration. However, once work begins on an iteration, scope changes to that iteration are subject to integrated change control.

A project can be iterative or incremental and still be closer to the predictive end of the spectrum. This might be the case if the project phases are long and each phase involves detailed planning based on feedback gained from the prior phase, including the client's reaction to the partial deliverables received so far. When a project is iterative and incremental and has other features like very short phases, then it is likely an agile project.

## Agile (Adaptive)

The *PMBOK® Guide—Seventh Edition* defines **agile** as “a term used to describe a mindset of values and principles as set forth in the Agile Manifesto.” An agile life cycle is also known as the change-driven or adaptive approach. According to the *PMBOK® Guide—Seventh Edition*, the **adaptive approach** is “a development approach in which the requirements are subject to a high level of uncertainty and volatility and are likely to change throughout the project.” While an iterative life cycle is a little incremental and an incremental life cycle is a little iterative, an agile life cycle is equally iterative *and* incremental. Both are vital to success. The increment to produce is approved before the start of a given iteration. The iteration is used to minimize requirements uncertainty and the potential for plans to get rapidly out of date by planning for only the upcoming iteration in detail. Agile life cycles differ from iterative and incremental life cycles that are closer to the predictive end of the spectrum in that iterations are very rapid (usually two to four weeks in length), which allows this approach to address risk and costs as issues emerge. The agile approach is intended to facilitate projects that expect a high amount of change. This requires a high degree of ongoing stakeholder involvement. Exhibit 1-8 shows a sample life cycle with an adaptive development approach.

### Exhibit 1-8: Sample Life Cycle with Adaptive Development Approach



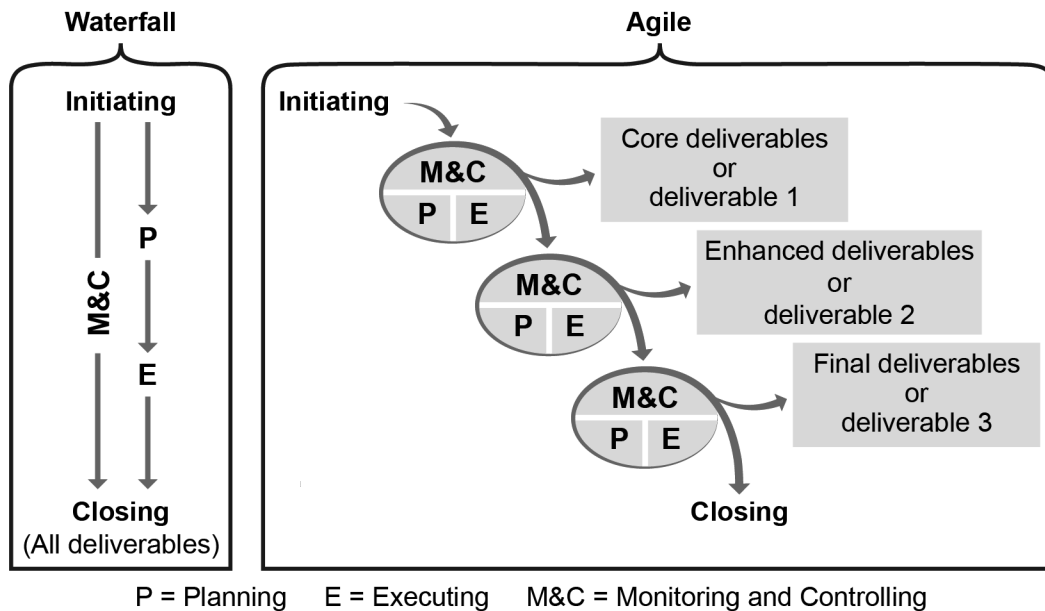
Source: Project Management Institute, *PMBOK Guide®—Seventh Edition*, Project Management Institute, Inc., 2021, Figure 2-11, Page 45. Material from this publication has been reproduced with the permission of PMI.

Planning for a single iteration means that planning is a regular event rather than being addressed up front. When the team meets in its iteration planning session, they determine how many of the highest-priority requirements can be completed in the current iteration. Projects with agile life cycles collect ideas for improvements and new features and feed them continually into the next iteration. Predictive projects with some degree of iteration will collect these ideas throughout the project but hold the most disruptive of them for a new, subsequent project—version 2.0.

Exhibit 1-9 compares a waterfall (predictive) project, on the left, with an agile project that includes features of both iterative and incremental life cycles, on the right. The path of the agile project is like a series of loops, so it is iterative. Iterative projects can be thought of as a spiral that starts with a central idea at its core that gets broader and more complex each time

it loops around through the Planning, Executing, and Monitoring and Controlling Process Groups. For example, testing conducted during Executing and interpreted during Monitoring and Controlling can influence the next round of Planning. The project also produces valuable deliverables or results in increments after each phase. Because iterations are brief and feedback is gathered continuously, the agile life cycle can handle change even late in the project in a cost-effective manner.

**Exhibit 1-9: Waterfall (Predictive) Life Cycle versus Agile Life Cycle**



The exhibit shows how project phases intentionally iterate or repeat certain activities as the project team develops a deeper and deeper understanding of the product while incrementally releasing better and better deliverables or new deliverables per phase. An example could be a billing system for a large utility company. The first phase could provide the company with a system with core deliverables such as the framework plus basic functionality. Subsequent phases could make the system more user-friendly or decrease transaction times or increase the number of interactions that end users (the company’s customers) can perform. Note that the example shows just one Initiating and one Closing Process Group, but these could also be repeated in each phase.

The agile cycle is best for situations when scope and requirements cannot be determined or must remain flexible to adapt to changing enterprise environmental factors such as competitor actions. Agile projects use a product backlog to control scope and prioritize activities to be performed in a given iteration.

There are two very different approaches that agile frameworks can take:

- **Iteration-based agile.** In iteration-based agile methods (such as Scrum), projects have brief phases called iterations or sprints. (The terms are interchangeable; the *PMBOK® Guide—Seventh Edition* defines a **sprint** as “a short time interval within a project during which a usable and potentially releasable increment of the product is created.”) Iterations are timeboxed, meaning that they are of uniform duration. Therefore each phase will likely have the same costs and will be straightforward to budget (assuming that the same resources are used in each phase). The iteration is done at the end of the set time period, meaning that while the plan is to start and complete all work (by selecting just enough work to likely fill the iteration), some planned tasks may not get done and will need to be completed in the next iteration.
- **Flow-based agile.** In flow-based agile (such as the Kanban Method), the process is still iterative, but work is not restricted to a particular iteration cycle. It is called flow-based because there is a continuous flow of work as features are started/completed on demand rather than new work being added only at the start of an iteration cycle. The primary control is instead a strict limit on how much work-in-process (WIP) can exist at any time. The limit is typically fairly low to help identify issues quickly. As a task or feature is completed, new capacity is created for the team to add another task/feature. The team convenes whenever there is available capacity to discuss what to add next. The plan is to always refill the WIP limit to its maximum as long as there is work to be done. For example, if a team is working on three features at a certain point and one feature (including testing) is completed, then the team meets to select and add another feature to fill up the queue even though the other two features are still being worked on.

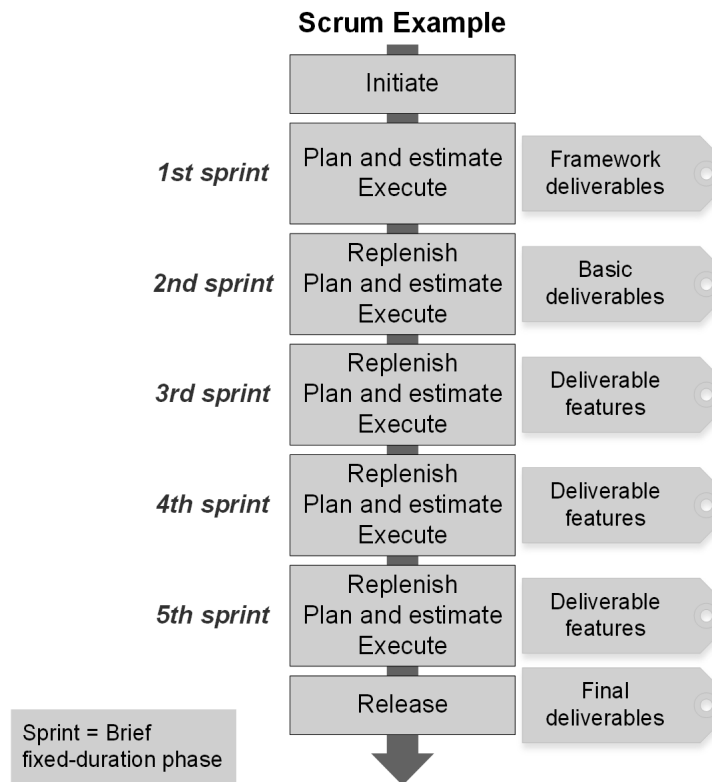
The following are key features of agile life cycles:

- Highly structured meetings are used to plan for iterations or add tasks to the queue, indicate daily status, demonstrate working deliverables, or retrospect (also called replenish). Retrospective meetings review what has been completed and create action plans to continuously improve. Iteration-based meetings are at the same time in every iteration, but flow-based teams need to select appropriate schedules for meetings since there are no distinct cycles.
- At the end of each iteration (or at selected points in flow-based agile), deliverables are ready for customer review, meaning that the features are stable, complete, and finished but customer acceptance is not required.
- Sponsor and customer representatives review the product for the purpose of providing feedback. Incremental deliverables are valuable and reassuring to the customer.

Exhibit 1-10 illustrates an iteration-based agile life cycle. The example shows Scrum, an agile framework. After the project is initiated, the team meets regularly in a series of sprints, which

are often as short as two or three weeks. The team plans and estimates for just that period and then implements that plan. The first half of the regular Scrum meeting for a sprint is dedicated to replenishment (a retrospective of the prior sprint). The second half of the meeting is devoted to planning and estimating for the upcoming sprint. The plan is executed between meetings, and the process is repeated until the final deliverable is ready for release. Each sprint results in a more developed product with more features or better functionality.

**Exhibit 1-10: Iteration-Based Agile Life Cycle (Scrum Example)**



**Hybrid**

A hybrid life cycle is a combination of adaptive and predictive approaches. Instead of being assigned to the whole project, predictive and adaptive processes are assigned to different elements within the project, based on the element’s qualities and requirements. It is up to the project management team to determine which is the appropriate life cycle for a project and to ensure that it has the requisite flexibility to encompass all project elements. The hybrid approach may be best applied when there is uncertainty around project requirements or when deliveries can be modularized.

**Project Phases**

Projects can be divided into phases to control the work and to integrate the project’s final product, service, or result into the ongoing operations of the organization for which the project

is being performed. With the exception of iterations for agile projects, a phase is usually a major subdivision of a project.

One or more project phases comprise the project's life cycle. Simple predictive projects often have only one phase. In a predictive project with multiple phases, the phases could each occupy a fairly long duration, such as several months, but each phase will produce something that can be delivered to the customer. In an incremental or agile project, there are almost always multiple phases, but these phases will often be very brief, such as a week or two.

The creation of "one or more deliverables" is the distinguishing element of a project phase. Sometimes the deliverables produced in a phase may not actually be released into operations, depending on the release strategy, but because they are functioning deliverables, they could potentially be released. In any case, they will usually be shown to the customer or stakeholders to demonstrate that the team has been producing valuable results using the customer's resources.

Phases are always appropriate for an agile project. When are phases appropriate for a non-agile project? Apply the following criteria to decide:

- There is a distinct goal or objective to meet in the phase that can impact later phases.
- The work to be done in a phase is mostly or fully completed in that phase, resulting in a distinct deliverable.
- Deliverables are produced and turned over to customers before completion of the entire project.
- One of the deliverables has a different delivery date than the others.
- The people, organizations, or resources for the phase differ from that for other phases. (When the work required in each phase differs significantly, a good portion of the project team will also differ from phase to phase. For example, one phase may involve architects while another may involve electricians and structural engineers.)
- Unique processes or controls are needed in each phase.

Each phase has its own full or partial set of Process Groups. For example, during a feasibility phase, there would be an Initiating phase to authorize the feasibility study, Planning to determine feasibility scope and so on, and then Executing, Monitoring and Controlling, and Closing, resulting in a completed feasibility study. If the results look promising, then the next phase would begin. Other phases could include concept development, milestone reviews, or examination of lessons learned.

Phases can be identified using a variety of attributes. For example, a simple name or number (Phase A, Phase 1, Sprint 1), a duration (Week 1, Month 1), or a resource requirement (which building it will take place in, specific equipment that is needed) can be used. Entrance and exit criteria can also be used to identify project phases, such as the completion of certain

deliverables, including any required documentation and successful quality or integration test results.

At the end of a project phase is a **phase gate**, defined in the *PMBOK® Guide—Seventh Edition* as “a review at the end of a phase in which a decision is made to continue to the next phase, to continue with modification, or to end a project or program.” The project manager and the team will assess their work and progress. Sponsors or product owners will analyze accomplishments, the ratio of benefits to costs, and changes in risks or the organization’s own strategic priorities. The decision will be made to continue with or cancel the project. If necessary, remedial or corrective action can be taken before beginning the next phase in the life cycle. Different organizations call this point by different names, for example, milestone, kill point, stage gate, or phase review.

What are the phases of the project life cycle? The answer is, it depends. There is no consensus in the project management profession or even in a given industry. Organizations select or develop their own life cycles and phases and may choose different ones for different types of projects. (From this perspective, an organization’s guidelines about how to structure a project are an example of an organizational process asset, or OPA.)

Organizations’ approaches to structuring projects may be distinguished by:

- The names and order of phases they use, for example:
  - Concept.
  - Construction.
  - Testing.
  - Turnover.
- The degree to which the phases are predictive, iterative and incremental, or agile.
- Which phases are sequential or overlapping.

### **Sequential versus Overlapping Project Phases**

Project phases can be sequential or overlapping, and a project can have both types, though they may not occur concurrently.

With sequential phases, the next phase begins only after the prior phase is complete—deliverables are accepted and the phase has been closed. Sequential phases can be used when a deliverable is needed as an input to a later phase, when additional staff or resources are unavailable, or when risks need to be minimized. When using sequential phases, there are fewer opportunities to compress the schedule.

With overlapping phases, work in one phase may overlap another phase or be done simultaneously. This is possible if the teams, resources, and deliverables are independent of one another or only partly dependent on one another. Overlapping allows project managers to use schedule compression tools such as fast tracking (simultaneous activities). Overlapping may require increasing the size of the project team and resources. It increases some risks, primarily the risk that progressive elaboration in an earlier phase will lead to rework in a later phase. Another risk is added complexity due to the need for greater integration of the work.

An agile project with a fixed project team will typically use sequential phases because all the team is occupied on the same iteration. This makes sense, because when a project is subject to high degrees of uncertainty and possible change, the end of a sequential iteration presents a point at which adjustments can be made, before the project moves to the next phase. Agile projects can use overlapping phases, but this will generally be possible only on larger agile projects with multiple small project teams. Each subteam will perform tasks on its own, and the overall work will need to be integrated. This maximizes the amount of time spent on execution and reduces management overhead. An overall integration team can then focus on overall improvements as opposed to just those defined by the scope of an individual iteration.