Project Planning and Management (PPM)

**Project Type Guide**



**Agile**

**Version 1.0**

**June, 2023**

**Table of Contents**

[1. Introduction to Agile Software Development 3](#_Toc462755997)

[1.1 What is Agile? 3](#_Toc462755998)

[1.2 Benefits of Agile 4](#_Toc462755999)

[2. Adopting Agile Software Development 5](#_Toc462756000)

[3. Project Team Organization in an Agile Project 7](#_Toc462756001)

[3.1 The Core Agile Team 7](#_Toc462756002)

[3.2 The Integrated Project Team 8](#_Toc462756003)

[4. Agile Development Lifecycle 8](#_Toc462756004)

[4.1 Project Validation Phase 10](#_Toc462756005)

[4.2 Release/Sprint Planning Phase 11](#_Toc462756006)

[4.3 Release Readiness Phase 12](#_Toc462756007)

[4.4 Deployment Phase 13](#_Toc462756008)

[5. Governance in Agile 13](#_Toc462756009)

[6. Agile Acquisition/Contracting and Budget Considerations 15](#_Toc462756010)

[7. Agile Artifacts 18](#_Toc462756011)

# Introduction to Agile Software Development

The U.S. Department of Housing and Urban Development (HUD) needs a more flexible IT development methodology for managing IT investments that integrates IT strategy development, enterprise architecture, capital management, project management and investment performance measurement. Today many agencies are concerned that their next information technology (IT) project will suffer the fate of recent and well-documented technology failures.

Agile development practices can help manage projects more effectively by producing faster results and responding more rapidly to changes in operations, technology, and budget. This Agile IT Project Guide provides details on how to adopt Agile practices within each Segment of the programs. This enables the business areas to achieve their IT goals while maintaining focus on the Department’s strategic objectives and vision. Agile software development, as with any development effort, requires thorough planning, which is critical to successful execution and implementation. In Agile software development, requirements and priorities are captured in a high level Product Vision, which establishes a definition of the scope of the project, specifies expected outcomes, and produces budgetary estimates.

Agile development can achieve these objectives through:

*“Single Family Housing has a one-stop shop for Mortgage Lenders, the public and other entities tied to the Housing Market. It turned out that HUD was on the “bleeding edge” of using Dynamic CRM in the way we were using it. “Bleeding edge” means we were the first customer using it the way we were using it so there was trial and error involved. Because of the Agile Development, all the error happened right in front of the customer. This was not something that the OCIO team was used to, so we had to adjust how customer service skills to better set the expectations of this kind of development process.” - HUD Program Area*

* Focusing on small, frequent capability releases *(that contains the features that the customers are looking for)*
* Valuing working software over comprehensive documentation *(contains the required quality that allows the project to deliver value not documentation)*
* Responding rapidly to changes in operations, technology, and budgets *(that adapts as the business circumstances change)*
* Actively involving users throughout development to ensure high operational value

## What is Agile?

On February 11-13, 2001, 17 representatives from Extreme Programming, Scrum, DSDM, AdaptiveSoftware Development, Crystal, Feature Driven Development, Pragmatic Programming and otherssympathetic to the need for an alternative to documentation driven, heavyweight softwaredevelopment processes convened at a summit. The result of the discussions produced the Agile Manifesto.

**AGILE CORE VALUES**

* *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 PRINCIPLES**

The Agile Manifesto is based on twelve principles:

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

Agile software development starts with a Product Vision and does not specify exact system features, but addresses the desired high-level functionality of the system (not the specific system features to achieve that functionality). Deliverables are the functional, working software (i.e., deployable code) that are produced through a repeatable iterative delivery process in a production environment. Cost and schedule are adhered to because there is a schedule of releases in which the contractor must produce deployable functionality.

## Benefits of Agile

The iterative nature of Agile development means features are delivered incrementally, enabling some benefits to be realized early as the product continues to develop. The following are some of the main benefits brought about by Agile:

* Stakeholder Engagement
* Transparency
* Reduced Risk through early discover of project issues and problems
* Predictable Costs and Schedule
* Focused on Business Value
* Focused on Users
* Improved Quality
* Better project Tracking and Monitoring
* Efficient use of time and funding resources through shorter time to deliver
* More rapid and flexible responses to changing requirements

Agile allows for increased transparency, less restrictive governance, and a dedication to delivering value over producing required documentation. In the following sections, this guide will explain the Agile process for managing a project under PPM V2.0, as well as important considerations for how the project team functions in an Agile environment, and how Agile Development impacts or affects governance boards, contracts and budgeting.

# Adopting Agile Software Development

The traditional waterfall methodology of software development is a progressive and linear process where requirements gathered in a pre-award environment strictly govern contractor performance in the post-award environment. This constitutes a long planning process resulting in highly detailed documentation, and rigorous work plans that dictate all facets of the development life cycle. At completion, the work product is delivered to the end users in one big production of deployment. Implementing Agile ensures that the software development is up to date and incorporates new requirements as they are identified. Unlike the traditional software development methods, there is not a need to spend a number of years in the development process only to realize that the world has actually moved past the original project requirements. With Agile, projects can better adapt their portfolio to meet the ever-changing business priorities.

Based on trends and emerging technologies, OMB has encouraged agencies to adopt Agile principles by breaking large requirements into smaller modules, to deliver functional, useful software throughout the period of performance. Agile represents a fundamental shift from industrial age processes to a modern management and development approach suited to the digital age. Agile practices help to make progress and development more transparent, enabling improved decision making by delivering more timely and accurate information. However, Agile (itself) is **not a solution**; Agile does not promise to solve all IT and program management problems, and may not be appropriate for use in all cases. Even successful adoption of Agile practices does not guarantee program success, as many variables that affect success lie outside the control of the government program manager and his team.

Before deciding to adopt Agile practices, program managers should first identify the best approach for the project as a whole and/or for any subprojects within it. This includes assessing the project’s volatility (to include requirements and technology), criticality, availability of resources, organizational culture, and availability and commitment of the customer and stakeholders. The table below provides general assessment areas to consider when deciding if a project should be run as an Agile effort.

|  |  |  |
| --- | --- | --- |
| **Consider Agile Practices** | **Assessment Areas** | **Consider Traditional Practices** |
| Requirements cannot be well defined upfront due to a dynamic operational environment. | **Requirements Stability** | Requirements have been relatively well defined by the operational sponsor. |
| Requirements can be decomposed into small tasks to support iterative development. | **Requirements Divisibility** | Requirements are tightly integrated and are difficult to decompose. |
| Users welcome iterative development and require frequent capability upgrades (<1 year). | **User Timelines** | Allow iterative development or lacks the ability to absorb frequent updates. |
| User representatives and end users are able to frequently engage throughout development. | **User Involvement** | Users cannot support frequent interaction with the development team or the target end user cannot be accessed. |
| Program scope is mostly limited to the application layer while using existing infrastructure. | **Program Scope** | Program spans core capabilities and underlying platform or infrastructure. |
| The government is responsible for primary systems integration. | **Systems Integration** | The government does not want to own systems integration responsibilities. |
| Capabilities are operational at a basic level, with some defects that can be addressed in future releases. | **System Criticality** | Program supports a critical mission in which defects may result in loss of life or high security risks. |
| Industry has relevant domain experience and Agile development expertise. | **Developer Expertise** | Agile development expertise is unavailable or lacks domain experience. |
| Program office has Agile training, experience, and/or coaches. | **Government Expertise** | Program office has no Agile experience or funding for Agile training or coaches. |
| Program contract strategy supports short Agile development timelines. | **Contracting Timelines** | Contract strategy cannot support short Agile development timelines. |
| Program Executive Office or subordinate has authority for most program decisions. | **Level of Oversight** | HUD Service Acquisition Executive (SAE) is the Milestone Decision Authority (MDA) and requires most decisions to be made at that level. |
| Development can be effectively managed by a small cross-functional government team. | **Team Size** | Many government stakeholders will be involved in the software development and decision-making process. |
| Government and developers can collaborate frequently and effectively. | **Collaboration** | Stakeholders physically located across multiple locations and have limited bandwidth to support frequent collaboration. |
| One or a few contractor(s) or teams can perform development. | **Complexity** | Many contractors are required to develop program elements. |
| Program can leverage test infrastructure and automated tests, and testers are active throughout development. | **Test Environment** | Extensive development and operational testing is conducted serially following development. Limited resources and tools available to conduct parallel development testing. |
| Leadership actively supports Agile development practices and provides “top cover” to use non-traditional processes and methods. | **Leadership Support** | Leadership prefers a traditional development approach or is unfamiliar with Agile practices. |

Table 1: Agile Assessment Criteria (*Source: Mitre's Defense Agile Acquisition Guide*)

The move to Agile requires time to learn new practices (training) and time to replace traditional development methodologies. Effective transition may require some tailoring because of the nature of government contractor relationships and current organizational processes and regulations. Prior to committing projects to the transition, project managers must understand and appreciate each stakeholder’s risk tolerance and legal responsibilities, and provide clear and compelling evidence that an Agile approach can reduce risk. However, Agile characteristically serves as a risk mitigation strategy, since early working software products reduce risk by validating requirements and performance characteristics rather than by conducting comprehensive paper analysis. The requirements process prioritizes steps in the development to deliver the highest priority capabilities to the user with each release. However, smaller scale development efforts naturally carry less risk, permitting a lightweight approach to documentation and review that is consistent with the lower risk.

# Project Team Organization in an Agile Project

There are organizational differences between traditional project management and Agile project management practices. One example is the organization of the project team as well as the roles and responsibilities that are held by its various members. It is important to understand how the project team operates within Agile. The team is comprised of a cross-functional, dedicated group of people who work collaboratively on the project. Project Management best practice dictates government and vendor personnel working together to ensure proper execution and implementation, as opposed to the more traditional model of vendor interaction that is rigid and hierarchical. Projects should operate in a trusting and cooperative manner across the government and contractor teams. This may require the government to refine or redefine existing roles and consider new roles specific to Agile.

Another key change to how the project operates in the way that it is governed. This will be addressed further in Section 5, Governance in Agile, but it is important to note that within Agile there is a shift from the traditional model of governing projects from the top down to one of trusting and empowering the project teams to make decisions and to collaborate on solutions. Projects also possess much more autonomy to manage decisions of sprint development and prioritization of the product backlog. Within Agile, the focus is on producing working functionality and not on comprehensive documentation. The governance bodies will still ensure documentation is complete and provides value, but the formal gate reviews will focus on overall preparation (during the planning phases) and technical reviews of demonstrations during releases and deployment.

The main Agile team includes the Product Owner, Scrum Master, the Development Team, and the Customer Relationship Coordinator. The broader extended team includes the members of the Integrated Project Team, which is comprised of OCIO stakeholders from key areas such as Enterprise Architecture, Security, Privacy, Operations, Acquisitions, and Information Technology Investment Management (ITIM).

## The Core Agile Team

While some of the roles may vary based on specific project, four roles will recur in almost all Agile projects.

1. The **Product Owner** is the authoritative representative who manages the product backlog(s) and requirements prioritization, communicates operational concepts to the development team, and provides continual feedback to development team on their developments, demonstrations, storyboards, mockups, etc.
2. The **Scrum Master** facilitates the Scrum processes, enforces the development team's rules, and keeps the team focused on tasks.
3. The **Development Team** typically is the contractor team of software developers, including software and security engineers, data specialists, testers, quality assurance, and configuration managers.
4. The **Customer Relationship Coordinator (CRC)** is the liaison between the project team and the Program Area and ensures coordination and integration.
5. The **Agile Coach** is an expert in Agile methodologies and can conduct on-the-job-training for the program office staff. They can make invaluable contributions by guiding the program office to identify and improve roles, processes, and techniques, while addressing any adoption issues.

It is ideal in an Agile project that there is at least one member of the team with Agile experience. That person can provide guidance and knowledge to the other staff and is committed to leading teams through successful adoption of Agile practices.

## The Integrated Project Team

Depending on the needs and timing of the project, the Integrated Project Team (IPT) will be comprised of various OCIO stakeholders, as well as representation from the Program Area. They will assist with gathering requirements and provide input into product backlog prioritization. The IPT will also collaborate in each sprint retrospective, where priorities can be reassigned and requirements can change. They will also work to provide feedback to the Product Owner whenever possible. These members will also be critical in determining and executing the implementation strategy for the releases, as well as reviewing testing results and preparing for governance reviews.

The Product Owner will chair the IPT while the Business Owner and the CRC will also have vital roles in ensuring the IPT functions properly. Other key personnel might be needed from areas such as Enterprise Architecture, Operations, Privacy, Security, ITIM, and contracting, as well as other stakeholders as identified by the Product Owner, CRC, and Business Owner throughout the project. The composition of the IPT is determined autonomously by the Product Owner, Business Owner, and the CRC, since they are best suited to know what the project needs in terms of experience and subject matter.

The following section will outline the Agile life cycle under PPM V2.0. Within this section, the specific processes will be outlined and connected to the roles and responsibilities of the Agile project team.

# Agile Development Lifecycle

The objective of this section is to provide guidelines and instructions on applying the Agile best practices and executing the project using Agile development within PPM V2.0. To benefit fully, users should have a good working knowledge of PPM and Agile best practices.

The Agile approach to system development differs from the traditional “waterfall” method. Under the traditional approach, each System Development Life Cycle (SDLC) phase is completed and approved before the next phase can begin. For example, all requirements are defined and approved before completing the functional and system design. These designs, in turn, are completed, and approved before construction and testing. Under Agile, the system is defined, constructed, and implemented in a series of smaller efforts known as *sprints*. In effect, each sprint becomes an iteration of most of the Agile Framework phases, from planning through deployment of the developed functionality. The figure below shows a high-level overview of the Agile process.

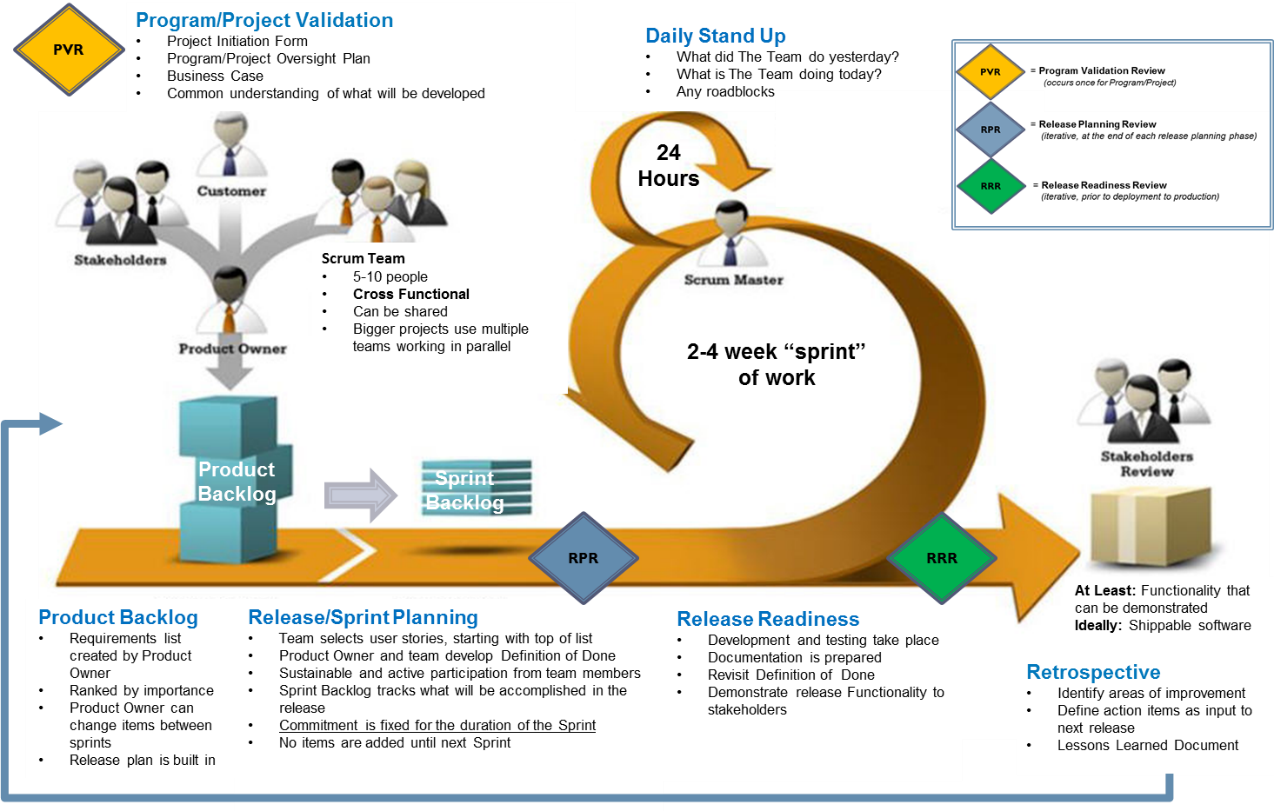


Figure 1: HUD's PPM Agile Scrum Methodology Framework

Agile projects must produce the required documentation. Not all documents are fully completed at the conclusion of each phase. In some cases, the artifact is reviewed and updated as part of each sprint. For example, once the content of a specific sprint has been determined, the project team will update the Product Backlog and Technical Architecture & System Design Document, etc., as needed. At the final Sprint Closeout, each document should be complete and include the details of all the functionality provided by the solution. Once created, supporting documents such as the Technical Architecture & System Design Document and the O&M Manual may not need to be updated during a given sprint, but should be reviewed during each sprint. The graphic below shows the necessary artifacts within each phase and how the project should expect to iterate and refine documents throughout the project lifecycle.

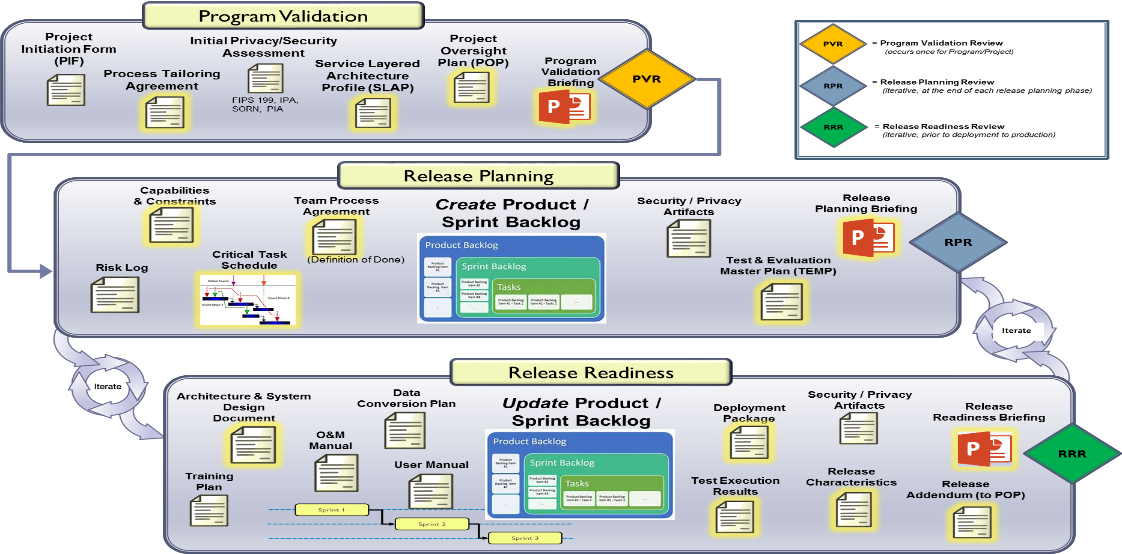


Figure 2: PPM Agile Phases and Artifacts

In the following subsections, the Agile lifecycle is described in detail. Activities, governance touchpoints, and roles throughout the lifecycle are documented and explained.

## Project Validation Phase

The Project Validation Phase is the initial planning phase, and represents the first formal phase of the project lifecycle. During Project Validation, the Development Team and Integrated Project Team, led by the Product Owner and CRC, analyzes and defines the approach for realizing the concept, and identifies potential alternatives for their feasibility and appropriateness.

The Validation begins when there is justification for implementing the proposed solution. During Validation, the technical approach for meeting the business need is defined and a project team begins to take shape.

* **Definition**. The Project Validation Phase represents the initial formal phase of the project lifecycle. During this phase, the project team analyzes and defines the approach for accomplishing the concept and reviews the alternatives for feasibility and appropriateness. The project team further develops and describes how the business will operate once the approved system is implemented. The request for initial funding for the system is submitted. This phase seeks to define the project from a business unit’s perspective and start comprehensive planning for the project.
* **Entry Criteria.** Entry to this phase requires a conceptual solution approved by the Technical Review Subcommittee (TRC).
* **Gate Review.** The Project Validation Review (PVR) establishes that the project has done some initial planning and has developed a structured approach for realizing the concept.
* **Exit Criteria**. Exit from this phase requires the successful completion of the PVR.

## Release/Sprint Planning Phase

When the project first starts, the Product Backlog must be created. This phase initiates the Product Backlog and project plans, the high-level architecture, and the security and privacy assessments used to initiate required activities based on the high-level architecture and the project objectives. In the Release Planning Phase, project planning is completed; planning documents are reviewed and approved; and approval is granted as part of the Release Planning Review (RPR) to proceed to the Sprint Planning Phase.

The Sprint Planning Phase is the beginning of the sprint iterations, which includes detailed planning for the current sprint iteration, and the set of user stories transferred from the Product Backlog owned by the Product Owner to the Sprint Backlog owned by the Agile Development Team. The functionality to be included in the sprint may include features from the Product Backlog that could not be completed during the previous sprint, changes to previously implemented features, or completely new features or functionality. The agreed-upon functionality must be attainable within the standard sprint period that was established for the project.

* **Definition.** Sprint planning occurs before each sprint begins to confirm the understanding and completeness of the user stories by the Development Team and enable the team to size each user story in story points. The Product Owner, IPT, and the Development Team review the outstanding stories from the Product Backlog, size them, and work to determine which ones will be developed during sprint iteration.

The Scrum Master facilitates a conversation between the Product Owner and the Development Team. The Product Owner decides which stories are considered the highest priority to the release and which will generate the highest business value. The IPT and Development Teams have the power to voice concerns or impediments. This is a good practice since the team may be aware of a legitimate impediment keeping the team from moving forward. When the team commits to accomplish the work, the Product Owner adds the corresponding stories into the Sprint Backlog. At this point, the Product Owner allows the Scrum Master and Development Team to decompose the Sprint Backlog items into tasks. The Agile Development Team and the Product Owner agree on the scope and the tasks to be performed in the sprint iteration.

Once the user stories are selected for inclusion in the sprint, the Development Team decomposes the user stories into tasks. The Development Team assigns a time estimate to tasks, typically in hours, with a goal of keeping tasks under one day. These refined task estimates, in combination with the team’s actual velocity, generate a higher degree of certainty of the team’s capacity for delivery. Velocity is defined as the amount of functionality measured by story points delivered by the Agile Development Team during one iteration. It is also used to measure the amount of work the team can complete in any given time period. The sprint planning session produces a Sprint Backlog that defines the set of user stories the Development Team has committed to delivering during the sprint iteration.

* **Entry Criteria**. Initial project planning is completed and a successful Program Validation Review (PVR) is conducted.
* **Gate Review.** Initial Release Planning Review (RPR) to validate and approve the Product Backlog and other initial project artifacts.No formal reviews are required to complete subsequent Sprint Planning and continue to the Sprint Activity.
* **Exit Criteria**. Creation of a Sprint Backlog estimates of tasks to complete each sprint user story, and criteria to demonstrate completion of each sprint user story in scope.

## Release Readiness Phase

The Release Readiness Phase includes completing the tasks to develop the features required in each user story for the sprint. This includes meeting any applicable development standards and performing unit and system test to validate the functional and non-functional requirements specified in the user stories. User Acceptance is achieved by demonstrating user story functionality to the Product Owner and selected stakeholders in a User Acceptance Testing environment.

* **Definition.** In each sprint, the Product Owner and Scrum Team define specific functionality to be coded, tested, and implemented within a set timeframe. For each sprint, the project team performs the functional equivalent of the following:

1. The Product Owner and the Development Team develop the Sprint Backlog, a subset of the Product Backlog, revise the Technical Architecture and Design, Test Plan, etc., for the functionality included in the Sprint. The required artifacts are selected in the Release Planning Phase and initial drafts are created in the first Sprint. For each subsequent sprint, the artifacts will be updated accordingly. Develop functionality and complete testing for backlog items included in the sprint.
2. Perform Stakeholder Review where the Development Team demonstrates the developed functionality and, if applicable, gains consensus approval from the stakeholders to deploy the functionality into Production.
3. Perform necessary artifact and Sprint Retrospective and gain agreement of the product owner on the completed functionality.
4. Deploy the baselined product with the functionality approved in the sprint. Depending on how the team approaches implementation, this may consist of pushing the new functionality to the Product Owner demonstration environment. When requested, the Product Owner demonstration environment can be updated with the most recent version of the product baseline.
5. Any planned functionality not implemented during the Sprint is added back to the Product Backlog for consideration in subsequent sprints.

Each day during the Release Readiness Phase, the Development Team conducts a Daily Stand Up. The purpose of the Daily Stand Up is to review progress toward the sprint iteration goals, assess any risk to achieving their objectives or goals, report any impediments, and adjust their plans to achieve them. This meeting has specific guidelines:

1. The standup starts precisely on time
2. All are welcome, but normally only the core roles speak
3. The standup length is set to 15 minutes
4. The standup should happen at the same location and same time every day
5. During the standup, each team member answers three questions:

* What have you completed since yesterday?
* What are you planning to complete today?
* Any impediments/obstacles?

It is the role of the Scrum Master to facilitate resolution of these impediments, although the resolution should occur outside the Daily Stand Up itself to keep it under 15 minutes. The objective of the Release Readiness Phase is to develop the features and satisfy the completion criteria for each user story selected for the sprint iteration.

In the Sprint Retrospective, all Development Team members reflect on the past sprint and identify tasks that would foster continuous improvement to the sprint iteration process. Two main questions are asked in the Sprint Retrospective. What went well during the sprint? What could be improved in the next sprint? Many times, the team will prioritize items to be improved, and only select one or two, so that it can hold itself accountable for real progress at the next Sprint Retrospective. The Scrum Master role facilitates the Retrospective meeting and documents the findings. There is a three-hour time limit.

* **Entry Criteria.** Sprint Planning is complete for the sprint iteration.
* **Gate Review.** ARelease Readiness Review is conducted any time functionality is intended to be deployed to Production. However, this review is done without impeding the ability of the Development Team to continue into their next sprint.
* **Exit Criteria**. The completion criteria associated with the Sprint user stories has been satisfied. The required artifacts are completed and reviewed by the appropriate reviewers. The Sprint demo for the Stakeholder Review of the completed features has been scheduled. Sprint Retrospective is conducted and improvements and refinements to the Product Backlog are documented.

## Deployment Phase

The Deployment Phase has the Operations team implementing the system according to the Deployment Plan. Processes and procedures associated with operations and maintenance support also are implemented during this phase. In the Deployment Phase, the developed system is fully deployed to end users.

* **Definition.** During the Deployment Phase, the tested solution is integrated into HUD’s enterprise-wide environment. The objective of the Deployment Phase seeks to implement the system according to the defined deployment plan.
* **Entry Criteria.** Pilot execution (if applicable) and control artifacts have been updated to reflect status, including the schedule, actual costs, and risk register. If no pilot is required, deployment readiness documents are approved, and the work necessary to go to final deployment is complete.
* **Checklist**. Completion of the Deployment Review is required during this phase and for transition into the subsequent O&M activity.
* **Gate Review.** The Deployment Review ensures that outcomes match the stated goals of the system.
* **Exit Criteria.** Successful completion of the Deployment Review are required for exit from this phase.

# Governance in Agile

The purpose of governance within an Agile project differs from the traditional governance model. Instead of top-down oversight, there is an emphasis on collaboration between the governance body team and the project team. Instead of only having touchpoints at the control gate reviews, governance bodies are given transparency throughout the project lifecycle. This means that governance boards can be informal participants in sprint reviews and retrospectives, daily scrums, and can view documentation and status reports in real-time. The big difference between Agile governance and traditional governance is the *alignment with the value statements* in the Manifesto for Agile Software Development where the Governance should be more closely aligned with: Individuals and Interactions rather than Processes and Tools; with working solutions [more than just software] rather than comprehensive documentation; with Customer collaboration rather than contract negotiation; and with responding to change rather than following a new plan.

While more detailed information about the Agile lifecycle integration with the governance phase gates is in Section 4, there are a couple of key notes that should be highlighted here. When the project is conducting sprint planning, there is no review and approval required to allow them to move forward with executing the sprint. This allows the project team to focus on delivering functionality not documentation. This is also in accordance with one of the tenets of Agile that the project team is allowed to work autonomously to determine their sprint strategy and backlog prioritization.

The most important difference in the governance model is that Release Readiness Reviews are conducted outside of the project’s iterations in order to not “break the sprint.” The project does not come to a halt to wait for the phase gate to be completed, but rather the Development Team continues with their next sprint while the Product Owner and IPT participate in the Review and incorporate feedback in the Product Backlog and supporting documentation.

There are multiple benefits of this model of governance. For the governance body, there is less documentation to review and documentation can be viewed in real-time instead of waiting for formal submittal to a phase gate. The phase gates are streamlined and do not focus on compliance of comprehensive documentation, but rather emphasize reviewing data to ensure the project is operating efficiently and successfully building the solution. The project team feels empowered to make many decisions that would require approval and oversight in a more traditional model, thus allowing them to work more efficiently and undisturbed.

The Agile Project Guide establishes a set of required phase gate reviews that differs from the traditional PPM control gates of Initiation, Planning, Execution & Control, and Closeout. The table below lists the gates designed for the Agile approach and their high-level descriptions.

| **Agile Phase Gate Review** | **Phase Gate Description** |
| --- | --- |
| Project Validation Review (PVR) | Verifies that there is a structured idea of how project will be managed with the Project Oversight Plan. Ensures initial privacy, security, and architecture assessment has been done. Review is held with the Technical Review Sub-Committee (TRC). |
| Release Planning Review (RPR) | Initial Release Planning Review (RPR) before the first sprint to validate and approve the Product Backlog and other initial project artifacts. No formal reviews are required to complete subsequent Sprint Planning and continue to the Sprint Activity.  Verifies that the project management plan, subsidiary plans, and technical/engineering plans are adequate and consistent with all applicable standards, regulations, and guidelines. Review is held with the Technical Review Sub-Committee (TRC). |
| Release Readiness Review (RRR) | Confirms that production enabling systems, processes, and materials are in place, including operations and maintenance support. Review is conducted prior to a release being put into production. Review is held with the Technical Review Sub-Committee (TRC). |
| Deployment Review | The Deployment Review ensures that outcomes from deployment match the stated goals of the system. Lessons Learned are captured and successful handoff to O&M takes place. This final review is held with the Technical Review Sub-Committee and attended by the project leaders and O&M staff. |

Table 2: PPM V2.0 Agile Phase Gate Review Descriptions

# Agile Acquisition/Contracting and Budget Considerations

This section is meant as an introductory primer on Agile acquisition/contracting considerations rather than a new process that should be implemented and followed. Implementing Agile practices can be a challenge for acquisition/contracting, but it is a critical element in obtaining the benefits.

To develop an appropriate acquisition strategy, consider using the practices described in FAR part 39, Acquisition of Information Technology,” to the maximum extent practicable.

A primary component of part 39 is the use of modular contracting strategies that address the risk of technical obsolescence inherent to the IT environment. In acquiring IT for the government, obsolescence is a primary concern and discourages the use of waterfall development. Modular contracting is beneficial to acquiring IT and the strategy should seek to enable these benefits by employing the principles described in FAR part 39.

FAR Guidance at 39.103(e)

To avoid obsolescence, a modular contract for information technology should, to the maximum extent practicable be awarded within 180 days after the date on which the solicitation is issued…..To the maximum extent practicable, deliveries under the contract should be scheduled to occur within 18 months after issuance of the solicitation.

Acquisition activities in the Agile process for delivering IT capability will differ significantly from the traditional system development acquisition process. The IT acquisition process will be agile to respond to a dynamic technology environment and to address unique challenges, such as cyber threats.

**Short-Duration Projects**

Information capabilities will be delivered as a series of short-duration projects that deliver incremental capabilities in shorter timeframes as defined in approved roadmaps. A project manager will be assigned to each project, and performance will be assessed using performance-based metrics to promote transparency and accountability. Projects will be executed in a time-boxed manner to closely match the commercial IT development cycle and deliver capability more rapidly. Development efforts will focus on what can be achieved in the short term based on low-risk technology and balanced with user-determined priorities. Traditional program phases, milestones, and accompanying program reviews must be restructured or replaced and will include refashioned milestone reviews conducted more frequently as in-process reviews at key decision points within the new integrated governance structure.

While HUD is in the process of transitioning into an Agile organization, it is important to examine impact of Agile development on the various acquisition processes and contracting/budgeting guidelines. While Agile is a project type under PPM V2.0 it is important to follow the current processes and procedures, it’s equally important to consider the information provided below when making new decisions around Agile acquisition/contracting and budget considerations as the Federal government continues to promote modular contracting as a viable means to deliver value.

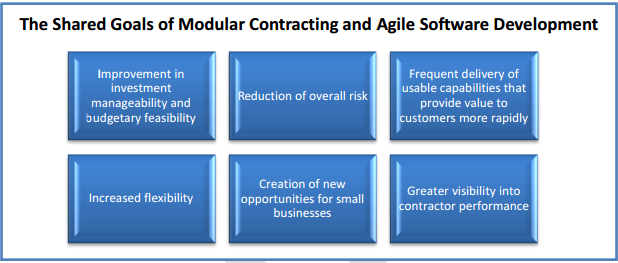


Figure 3: Shared Goals of Modular Contracting and Agile Software Development

**Contracts Considerations:**

In its June 2012 Report, Contracting Guidance to Support Modular Development/Office of Management and Budget (OMB) recommends three strategies for enabling Agile development: indefinite delivery vehicles (IDVs), stand-alone contracts, and successive contracts. Each strategy has its own utility, and the selection should be informed by the characteristics of the software development needs. For an enterprise-wide software development effort, an IDV offers the advantage of consistency among the project teams and the ability to divide the product vision into smaller chunks (modules) that can be delivered through task orders. In addition, and IDV can enable the use of multiple contractors for different elements of the product vision or provide for a system integrator that can provide project management assistance to the overall program.

A stand-alone contract is useful for smaller products that can be developed in the near term and do not require extensive integration with legacy systems. A stand-alone contract will require more support from the acquisition team to support the agile development effort and make end users available to inform the release planning process. For products that are more complex in nature, or will require extensive integration with legacy systems, a successive contract strategy that adheres to the guidelines of FAR 39.103 “Modular Contracting,” may be required.

Once the Acquisition Strategy is set, the type of contract must be determined. There is no formula for selecting the right contract type as it depends largely of the nature of the requirement, and the priority of that effort within the broader agency requirement.

**Fixed Price Contracts** can be challenging because they are reminiscent more of the “big-bang’ procurement approach, where vendors are asked to deliver a predetermined contractual outcome that often is influenced by rigorous price competition. Fixed-price contracts may have an adverse impact on the agile development life cycle by inhibiting the incorporation of lessons learned into subsequent release planning, because unforeseen challenges must be accommodated through contractual administration (e.g., modifications and change orders). This can lengthen the time between releases and reduce the return on software investment given the risk of technical obsolescence.

**Cost-Type Contracts** offer the advantages of schedule and cost flexibility, which are important in an agile development effort. Cost-type contracts also can be incentivized to motivate performance, and can enable functionality to be delivered at a lower cost than a fixed-price contract because the government has more control over funding decisions and the progress of development. Cost contracts are well suited for agile development because they allow the incremental delivery of working software, though they require a level of experience and expertise from the acquisition team and strong support from the acquisition leadership. Cost-type contracts also have their own unique requirements for cost analysis [as described in FAR 15.404-1(c)], though these requirements offer significant advantages over an evaluation based on total product price. In addition, cost-type contracts require additional administration and oversight because contractors have less incentive to control total costs. Applying an award or incentive fee structure to contractual performance will mitigate this risk but such structures require significant post award surveillance.

**Time-and-Materials and Labor Hour Contracts (TM/LH)** are worth consideration, given the focus of agile development on the continuous delivery of functionality. TM/LH contracts allow the government to purchase a not-to-exceed (NTE) amount of service in support of the overall product vision. This is consistent with the guidance that the scope of an agile development effort should be bound only by schedule and budget constraints. As long as the NTE amount is not breached, a TM/LH type contract for agile development can enable the delivery of functionality in excess of what was originally estimated, so long as it can be linked to the product vision. TM/LH contract types require the acquisition professional to document in a determination and findings document why no other contract type is suitable. Therefore, a thorough review and analysis of FAR subpart 16.6 should be conducted prior to selecting this contract type. In supporting the determination and findings document, consider using and incentive structure whereby future business can be earned through effective management and reporting of hours burned. This offers a way to control costs and promote efficiency, thereby addressing a criticism associated with this contract type.

**Other Considerations:** Incentive fees and other structures such as award terms have a utility for agile development. These incentives can be predicated on successful completion of individual releases, which can be included in modular contracting structures described in FAR 39.103.

For larger efforts, an IDV approach provides flexibility to award task orders for each release plan. If the total value is under $108M, a single-award indefinite-delivery, indefinite-quantity (IDIQ) contracted can be defended.

Hybrid contract types hold promise for agile software development. IDVs enable hybrid contracts, where fixed price orders can be let for project and release planning exercises. Then TM/LH or cons-reimbursable contract types can be used for the development of working software.

Different contracting approaches that can accommodate considerable uncertainty for IT systems must be developed. For Agile projects, since end-state is not well defined up front, a different approached must be considered.

IDIQ contracts and blanket purchase agreements save time by simplifying acquisition planning and technical evaluation for these services while providing access to a range of sources that can be quickly considered to perform these tasks to support a range of IT projects being undertaken. A review of available exisiting contract vehicles is needed to determine if they support Agile development. A process should be developed to transition existing contracts to a modular contracting format or for phasing out certain contract vehicles that do not effectively support Agile software devleopment.

**Overview of performance-based acquisition techniques appropriate for Agile development:**

In every Agile project, it is important that the project scope of work allow for flexibility and change, this allow you to add great value throughout the acquisition process by using a performance-based acquisition (PBA) technique. The PBA enables the acquisition professional to focus on functionality and strong adherence to project plans. According to FAR Subpart 37.6, Performance Based Acquisition (PBA) is a method of preparing service contracts that emphasizes the service outcomes the Government would like the contractor to provide. The use of PBA has been encouraged by the Office of Management and Budget to drive down the costs of contracts while improving contractor performance.

Through the planning phases, the acquisition professional should require an integrated project team (IPT), who will work together to create a high level product vision that will ensure consistency with HUD IT visions and goals. PBA will be used to acquire IT services under FAR part 39, during the final development and implementation of agile.

# Agile Artifacts

This section provides a summary description of a wide variety of Agile artifacts that are used with Agile Software Development. For additional information about the phases and where the artifacts are used, please refer back to Figure 3 PPM Agile Phases and Artifacts. It is important to remember that these artifacts are living documents and will be updated and refined after a sprint is completed. This ensures information and data are accurately recorded, and that the documents provide continuous value to the project team and other stakeholders. The matrix below identifies each artifact and provides a high-level description of the document’s purpose.

See the PPM V2.0 updated Artifacts for Agile Development: <http://portal.hud.gov/hudportal/HUD?src=/program_offices/cio/ppm/PPMV20Artifacts>

|  |  |
| --- | --- |
| Artifact | Artifact Description |
| Program Validation Phase | |
| Project Initiation Form (PIF) | Formally creates the project and provides high-level information |
| Project Tailoring Agreement (PTA) (tailored for agile) | Allows for tailoring of the documents required for the project |
| Security / Privacy Initial Assessment  (Same as PPM V2.0) | Security Assessments is designed to provide analysis about the security considerations involved with the project  Initial Privacy Assessmentis designed to assess whether a Privacy Impact Assessment (PIA) or System of Records Notice (SORN) is required |
| Service Layered Architecture Profile (SLAP) | The Project Service Layered Architecture Profile (SLAP) is developed by the EA Team and documents specific design constraints and design recommendations for each project. The SLAP is intended to be included in the acquisition package (if applicable). |
| Program / Project Oversight Plan (POP) | Similar to PPM’s Project Management Plan, provides information about how the project will be managed and executed. |
| PVR Briefing Deck | Provides necessary data and information during the PVR |
| Release Planning Phase | |
| Capabilities and Constraints | Documents capabilities of the current and/or proposed system or solution as well as initial known constraints |
| Team Process Agreement | Provides a Definition-of-Done as well as guidance on how the team is structured and how releases will be handled by the team |
| Risk Log (Same as PPM V2.0) | Provides analysis and mitigation strategies about risks and issues the project faces |
| Critical Task Schedule | Schedule that identifies and associates the inter-dependencies of tasks critical to the project’s schedule |
| Product / Release Backlog | Repository of requirements and features that will be prioritized and selected for each sprint |
| Security / Privacy Artifacts (Same as PPM V2.0) | Various security and privacy artifacts that may be required. For more information: <http://hudatwork.hud.gov/HUD/cio/po/i/it/security/templates> |
| Test and Evaluation Master Plan | Defines the testing and quality assurance processes the project will follow |
| Release Planning Review Deck | Provides necessary data and information during the RPR |
| Release Readiness Phase | |
| Architecture and System Design Document | Identifies architecture and system design impacts and considerations |
| Training Plan (Same as PPM V2.0) | Identifies necessary training for new system or solution |
| O&M Manual (Same as PPM V2.0) | Provides a manual on necessary O&M activities and considerations |
| Data Conversion Plan |  |
| User Manual (Same as PPM V2.0) | Provides a resource for system users to understand how to use the system |
| Security / Privacy Artifacts (Same as PPM V2.0) | <http://hudatwork.hud.gov/HUD/cio/po/i/it/security/templates>  IPA, PIA, SORN if applicable |
| Deployment Package | Under Development (Guidance Document) |
| Test Execution & Results | Provides results summary about execution of testing that is outlined in the Test and Evaluation Master Plan |
| Release Characteristics | Under Development |
| Release Addendum (to POP) | Provides a summary of the features and controls for the software build. It identifies and describes the version of the software being delivered, including the major changes since the last Release Addendum was issued |
| Release Readiness Review Briefing Deck | Provides necessary data and information during the RPR |

Table 3: Agile Artifacts within PPM V2.0