Contracting Considerations for Agile Solutions
Key Agile Concepts and Sample Work Statement Language

Version 1.0

18 November 2019

OUUSD(A&S)
This page intentionally left blank.
# Table of Contents

1. Executive Summary 6
2. Purpose 7
3. Scope 8
4. Agile Primer 10
   4.1 Common Agile Frameworks 12
   4.2 Key Agile Terms 12
   4.3 Agile Connection to Lean 14
   4.4 Agile Connection to DevSecOps 15
5. Contracting for Agile Software Development FAQs 16
   5.1 Agile Software Development Basics 16
   5.2 How Contracts for Agile Software Development Are Different 18
   5.3 Developing the Contract Strategy 20
   5.4 Selecting Contract Type(s) 21
   5.5 Post Contract Award 22
6. Agile Development Strategy Considerations 24
7. Contracting Strategy Considerations 29
   7.1 Modular Contracting 29
   7.2 Using Service Contracts for Agile Development 31
   7.3 Contract Vehicles 32
   7.4 Contract Types for Agile Development Teams 33
   7.5 Challenge-Based Acquisition (ChBA) Approach 35
8. Sample Language and Guidance 37
   8.1 Requirements Considerations 37
   8.2 Minimum Viable Product (MVP) Considerations 43
   8.3 Agile Product Roadmap 44
   8.4 Agile Framework 47
   8.5 Lifecycle Considerations 49
   8.6 Product Backlog Management 50
   8.7 Agile Metrics 51
   8.8 Agile Teams, Roles, and Responsibilities 54
      8.8.1 Agile Roles and Responsibilities 56
         8.8.1.1 Product Owner (PO) / Business Representative 57
8.8.1.2  Technical Lead  58
8.8.1.3  Scrum Master  58
8.8.1.4  Contractor Project Lead  59
8.8.1.5  Agile Coach  60
8.8.1.6  DevSecOps Engineer  61
8.8.1.7  Site Reliability Engineer  62
8.9  DevSecOps  63
8.10  Training and Resources  68
8.11  Tools and Technology  69
8.12  Intellectual Property  69
Appendix A  Traditional vs. Agile Contract Strategies  71
Appendix B  Sample Contract Templates for Agile Delivery  73
Appendix C  Contracting for Agile References  74
Appendix D  Bibliography  78
Appendix A  Abbreviations and Acronyms  79
Table of Figures

Figure 1 – Agile Mindset, Values, Principles, and Practices.............................................................. 10
Figure 2 – Traditional vs. Agile Delivery Cycle ......................................................................................... 17
Figure 3 – Differentiating between Contract Requirements and Agile Requirements ................. 31
Figure 4 – Sample Traditional Product Roadmap .................................................................................... 45
Figure 5 – Sample Agile Product Roadmap ............................................................................................... 46

List of Tables

Table 1 – Mapping Agile to Traditional Reviews ..................................................................................... 50
Table 2 – Sample Table for Identifying Desired Sizing Approach for Backlog Items ................. 51
Table 3 – Agile Alignment to Traditional Metrics .................................................................................... 52
Table 4 – Sample Cost Metrics ................................................................................................................. 54
Table 5 – Comparison of Traditional and Agile Software Development Contract Elements 71
Table 6 – Sources of Contract Templates .................................................................................................. 73
Table 7 – Information Sources on Contracting for Agile Development .................................................. 74
1 Executive Summary

The Office of the Under Secretary of Defense for Acquisition and Sustainment (OUSD(A&S)) provides policy and governance for the Department of Defense (DoD), and is releasing a series of documents related to Agile adoption that address the challenges and complexities faced by programs transitioning from traditional (waterfall) to Agile practices. This guide provides an understanding of how Agile methodologies differ from traditional waterfall approaches and identifies key Agile focus areas that merit special attention when developing capabilities using Agile methodologies. Specifically, it offers actionable guidance and sample work statement language that programs could use as a starting point and tailor to meet their unique needs. The guide:

- Explains key differences between waterfall and Agile projects to increase understanding and context
- Presents frequently asked questions and responses
- Suggests considerations for selecting contract strategies
- Discusses the contractual impacts of an Agile software development approach on the following areas, and provides considerations, guidance, and sample language for each area:
  - Requirements
  - Minimum viable product (MVP)
  - Roadmap
  - Agile framework and methodology
  - Acquisition lifecycle considerations
  - Metrics
  - Roles and responsibilities
  - Training
  - DevSecOps
  - Tools and Technology
- Contains references to existing resources such as the US Digital Service TechFAR Handbook, the General Services Administration (GSA) and Defense Acquisition University (DAU).¹

¹ Source: [http://techfarhub.cio.gov](http://techfarhub.cio.gov)
2 Purpose

The DoD is working with Government and industry leaders to modify how DoD plans, acquires and delivers information technology (IT) products and services. Most legacy Government/DoD projects used a waterfall delivery approach or a “hybrid” approach that did not fully adopt Agile principles. The increasing complexity of systems, decentralization of tools and technologies, and rapid pace of change pose challenges in delivering meaningful, secure, and modern capabilities that meet user expectations on time and within budget. Addressing these challenges requires a significant change in approach to planning, acquiring, and delivering capability to the warfighter – specifically, a shift to Agile software development.

Transitioning from a waterfall to an Agile approach represents a true paradigm shift. It has an impact at all levels of the organization, including programs, projects, and enabling technologies. This guide provides Project/Program Managers and their teams with an understanding of how contracting for Agile development solutions differs from traditional waterfall approaches and identifies key focus areas that merit special attention when working with programs to develop software solutions using Agile methodologies. It also presents sample language that could be tailored for use in a work statement that, in turn, will assist Contracting Officers and Specialists with developing a contracting strategy.
3 Scope

This guide is intended to assist Program Managers to develop the contracting elements in their acquisition strategies for Agile software development that will also assist Contracting Officers and Specialists with developing a contracting strategy. All other members of the program team will also benefit from this material. While the guide offers actionable guidance and sample work statement language, it focuses primarily on the thought processes necessary to understand the elements and activities of the Agile software development approach.

There is no one prescribed approach to contracting for Agile software development. As in any procurement, the needs and objectives of the program must be considered when developing an acquisition strategy. The strategy should be thoughtfully developed based on considerations unique to Agile software development. This guide offers an introduction to some concepts (i.e., DevSecOps) that are unique to Agile software development and should be considered as part of the strategy and resultant contracting artifacts.

There is no one-size-fits-all solution for Agile software, and a strategy that worked well for one program may not work for another. Therefore, the sample language in this guide is intended to provide a starting point for developing a strategy rather than words to be simply copied and pasted without consideration of their intended purpose.

Key points regarding the scope of this guide include:

- **Agile is not a method of procurement, but a framework based on a set of principles:** Agile is based on values and principles that encourage frequent delivery of working solutions to users in order to gain fast feedback and enable continuous learning that is supported by a culture of collaboration.

- **Contracting Officers and Specialists should work closely with the program office and the Agile Team throughout the Agile project lifecycle:** The Contracting Officer or Specialist needs a basic understanding of how Agile software development differs from traditional waterfall development. The Program/Project Manager must engage Contracting Officers and Specialists early and often to enable development of contracting strategies that align with the Agile project vision and objectives.

---

2 This guide does not address purchases of warranties and licensing, as they apply to more than just contracts dealing with Agile software development.

3 Agile principles are based on Lean principles and other lessons learned from leaders in the program management world such as Edward Deming.
 Guidance focuses on software development solutions: Although Agile can be used for delivery of software, hardware, and other types of solutions, this guide centers mostly on software delivery.

Note: Section 4 of this document and The Agile 101 – Agile Primer paper published by OUSD(A&S) contain additional information on common Agile frameworks, Agile terms, and the differences between Agile and waterfall approaches. Review these materials if you are not familiar with Agile key principles.

How to Use this Document: For full context, read the entire document once. For specific, frequently asked questions pertaining to contracting for Agile software development, review Section 5, which contains common questions raised by Program Managers and Contracting Officers/Specialists. The questions offer a starting point to obtain basic answers and also point to additional sections within the document that offer more information related to each question.
4 Agile Primer

This section provides basic information on key Agile values, principles, practices, roles and responsibilities, and explains Agile ties to Lean and DevSecOps practices.

Agile, as previously noted, represents a true paradigm shift in the way work is planned, executed and structured. The structure impacts organizational governance, roles and responsibilities, and team structures, among other areas. To improve the likelihood of success in Agile software development, it is important that Program/Project Managers understand the mindset, values, and principles that Agile espouses. Basically, this starts with the mindset (a Lean mindset) that can be summed up for Agile purposes in 4 values and 12 principles (see Figure 1). At execution, programs apply practices that align to the Agile mindset, values, and principles.

The Agile mindset begins with four value statements to emphasize the factors that contribute directly to quickly delivering value to users while avoiding work that may not directly add user value. The four Agile values are:

1. **Individuals and interactions** over processes and tools
2. **Working software** over comprehensive documentation
3. **Customer collaboration** over contract negotiation
4. **Responding to change** over following a plan.

Because the pace of change is increasing, DoD needs to remain open to developing and delivering supporting tools and documentation incrementally. The traditional approach to product planning and delivery has focused on a predictive approach, whereby planning, requirements, and design activities are locked in as soon as possible and contracts are developed to support the initial design. The complex nature of projects, combined with the

---

4 Source: https://agilemanifestor.org
desire for a faster pace, has led to dynamics that sometimes become a point of friction between the Government and the contractor. Often, the friction is due to change requests that impact the scope of work and level of effort. Given this dynamic, if the program team is not confident in the requirements, or if the project duration is longer than a year, programs may want to consider incremental and iterative implementations using Agile approaches.

Agile accepts the nature of large, complex systems and acknowledges that upfront, predictive planning can be challenging and potentially more risky than incremental planning and delivery of work. The incremental deliveries characteristic of Agile allow users to receive value (or benefit as defined by the user) faster and, in return, provide feedback faster to the development team, thus enabling the Agile team to apply the user guidance in the work for the next incremental delivery. The smaller increments (batches) of work and reduced complexity can reduce initial investment costs and improve quality.

The Agile Manifesto also identifies 12 Agile principles: 5

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**.

---

5 Source: https://agilemanifestor.org
Many of the Agile principles are traceable to Lean principles, which were popularized by Toyota in manufacturing. From the concepts and principles defined in the Agile Manifesto arose a series of Agile frameworks and methodologies used today in software development.

4.1 Common Agile Frameworks

There are many Agile frameworks, each offering methodologies that apply variations of iterative development and continuous feedback. The most popular Agile frameworks include:

- **Scrum** – A lightweight, simple framework for teams to collaborate incrementally and iteratively on delivering value to the customer.
- **Kanban** – A framework that enables visualization of the flow of work and allows the team to monitor work in queue, work in progress, and the overall flow of work from inception through completion.
- **Scrum of Scrums** – A scaled version of Scrum, in which multiple Scrum teams work together on a large project or program.
- **Extreme Programming (XP)** – An Agile framework that focuses significantly on engineering and development practices to bring value to the customer by producing higher quality software. The five values of XP are communication, simplicity, feedback, courage and respect. Practices such as pair programming have been popularized by XP.

Additional Agile frameworks include:

- Large-Scale Scrum (LeSS)
- Dynamic Systems Development Method (DSDM)
- Disciplined Agile Delivery (DaD)
- Nexus Framework by Scrum.org
- Scaled Agile Framework (SAFe).

4.2 Key Agile Terms

Terminology may vary slightly across frameworks; however, important Agile terms noted in this guide include:

- **Users** – Those who will ultimately use the software solution. Users convey operational concepts and requirements/needs, participate in continuous testing activities, and provide feedback on developed capabilities. It is critical for the development team to have a clear understanding who the end-users are, to ensure the team focuses on meeting and exceeding their needs. A core Agile tenet is active user involvement throughout development.

---

6 See the Agile 101 – Agile Primer by OUSD(A&S) for more information on Agile.
7 The Kanban board concept originated from the 1978 book by Taiichi Ohno called “Toyota Production System – Beyond Large-Scale Production”, which described LEAN, Lean manufacturing and Kanban.
8 Source: [https://www.agilealliance.org/glossary/xp/](https://www.agilealliance.org/glossary/xp/)
9 Source: [http://www.extremeprogramming.org/](http://www.extremeprogramming.org/)
- **Epic** – A large body of work to be completed during development. Depending on the Agile framework, the epic can be too large to complete within a single sprint. Epics are further decomposed into smaller features and user stories. Epics may express business functionality or identify constraints placed on the product or system.

- **User Story** – The smallest unit of requirements written from the users’ perspective on how they will use the software. The Product Owner defines and prioritizes user stories via backlogs. User stories that cannot be completed within a single sprint should be divided into smaller elements. Each user story should have clear acceptance criteria.

- **Story Point** – A unit of measure to determine the size or amount of work a development team needs in order to complete a user story. More complex user stories will require more story points to complete development. Most Agile teams work together with the Product Owner to estimate the story points for each user story in the Product Backlog. Usually, story points are measured by numbers (for example, a story point of 1 is the smallest unit and all other work is assessed relative to that size), but other sizing techniques are also used. Story points are unique to each development team.

- **Definition of Done** – A shared and understood definition of the activities that must be completed for a given user story to be considered complete. The definition of done is used to provide clarity regarding the expected quality of work to meet the users’ needs.

- **Backlog** – A ledger of Agile requirements written mostly in the form of user stories and epics. Types of backlogs include the Product Backlog, Increment or Release Backlog, and Sprint Backlog. Each type of Backlog contains the requirements for the given program, product, increment, release, or sprint, respectively. The Product Owner, working with stakeholders, regularly grooms the backlog to ensure the work is clearly defined, in priority order, and is accompanied by reasonable estimates. The higher priority work is defined in the greatest detail and will be completed first, whereas work planned for future releases will be lower on the backlogs and be described with less fidelity.

- **Sprint** – A short cycle of work (notionally two to four weeks in duration) that focuses on completing a defined subset of project deliverables or usable functionality. Each sprint includes planning, designing, developing, integrating, testing, and demonstrating working software to the Product Owner, users, and other stakeholders.

- **Release** – The core element of the program structure, guiding how frequently the program delivers capabilities to the end users. The length of time between releases depends upon operational, acquisition, and technical factors that the developers should discuss with stakeholders across the user and acquisition organizations. As a general guideline, most releases should take less than six months (as championed by US Chief Information Officer (US CIO), the Government Accountability Office (GAO), and the Federal IT Acquisition Reform Act (FITARA)). Shorter release cycles have

---

10 Depending on the specific Agile framework, some backlogs can contain features and themes, which are additional levels of abstraction of user requirements.
several benefits, the most important being that the program deploys useful capability to the end-user faster.

- **Velocity** – The measure of the amount of work completed in a given sprint for a given Agile team. Velocity is measured by summing the total number of story points completed by the team. A team’s velocity over multiple sprints is a key metric to track a team’s performance and aids in planning and scheduling future work. Velocity is only applicable to a particular team and cannot be transferred as an estimation tool for another team, because each team measures level of effort, size and complexity of work differently in terms of story points.

- **DevSecOps** – All work related to ensuring the ability to continuously integrate and continuously deliver working code. DevSecOps encapsulates multiple areas (i.e., configuration management, automation, development, testing, security, integration, deployment, and operations). It encourages the concept of “shifting left” – where functional teams work together as early and often as possible – to reduce handoffs and include all functional areas in planning as early as possible.\(^\text{11}\)

- **Test-Driven Development (TDD)** – A practice that involves developing test cases and test scripts before developing the functioning code. The software is then developed and improved upon until it passes the test.

- **Vision Statement** – A short description of the desired end-state of the solution to be developed.

- **Minimum Viable Product (MVP)** – An Agile concept that refers to the first version of a product that provides value, data, and learning for the users.

- **Product Roadmap** – A high-level strategic plan to guide organizational vision and align Product Owner and stakeholder expectations for future development. The roadmap, like the backlog, should be revisited and updated regularly.

- **Kanban** – A framework used by Agile teams to make work visible. The most commonly recognized tool within the Kanban framework is the Kanban Board, which is used to track the flow of work and make work visible by showing work in queue and work in progress. The Kanban Board can also show blocked work and work dependencies.

- **Agile Roles** (further defined in Section 8.8.1) – Product Owner/Business Representative, Technical Lead, Scrum Master, Contractor Project Lead, Agile Coach, DevSecOps Engineer, and Site Reliability Engineer.

### 4.3 Agile Connection to Lean

Agile and DevSecOps practices are based on the Lean principles of:

- Delivering value quickly by managing the flow of work in order to increase the pace of feedback and opportunities for learning – To be able to deliver value quickly, the organization must have leadership support, training and staff capability, and a clear understanding of Agile and DevSecOps practices.

---

\(^{11}\) See the Agile 101 – Agile Primer paper published by OUSD(A&S) for more information on “shifting left”
• Building in quality at all levels of work – This means that there must be accountability at all levels of work. The Agile team must avoid single point quality checks in favor of quality checks at all levels of work, which involve everyone on the team from developers to testers, security, and operations team members. Standardizing and automating as much as possible to avoid issues resulting from inconsistent application of testing and quality measures will help reduce human error (e.g., environments will remain consistent if applied consistently).

• Eliminating waste – Developers should reduce “gold plating” or overbuilding based on assumptions of desired user value. Eliminating waste also encourages delivering capability in small batches to reduce complexity and possible rework resulting from issues that naturally rise from complexity, and adopting an MVP design to begin to gain valuable feedback for incremental improvements.

• Making decisions at the last reasonable moment – This really means that the team should avoid locking down a design early and instead let the data and feedback determine design updates.

• Taking a scientific approach – This includes making data-driven decisions based on testing specific hypotheses and iterating based on the findings. Examples include application of the MVP concept to enable testing of smaller pieces of functionality, experimentation, and increased collaboration with users through application of user-centered design concepts to test usability before implementing.

• Creating a learning culture based on respect – The program should promote continuous learning, improvement, and innovation, and encourage a blameless culture that allows teams and individuals to take calculated risks and learn openly.

4.4 Agile Connection to DevSecOps

DevSecOps applies to all the steps required to deliver work from initial request to operations. DevSecOps promotes integration of development, testing, security, deployment, and operations practices and tools to continuously deliver solutions. It also promotes the Lean principles of continuous integration and continuous delivery (CI/CD) of work by encouraging the team to devote attention to the end-to-end flow of value delivery, or the value stream.

DevSecOps encourages the development, testing, security and operations teams to collaborate early and often during the application life cycle. DevSecOps CI/CD practices also encourage automation to enable consistent and repeatable testing, security, integration, delivery, and deployment practices throughout the delivery pipeline. Further, it enables instantiation of consistent environments through the use of automated code, code-based configuration, and cloud technologies such as containerization and orchestration.

Key Takeaways: Understanding key Agile concepts and the principles upon which Agile is based will better position Program Management teams, contractor teams, and other Agile team members to shape successful Agile development strategies.
5 Contracting for Agile Software Development FAQs

The US Digital Service TechFAR Handbook (https://techfarhub.cio.gov/handbook) offers a wealth of information on procuring software solutions using Agile principles. The sample contracting language contained in Section 8 of this guide is predicated on an understanding of how contracting for Agile software development differs from contracting for waterfall development so that a Contracting Officer can develop a contracting strategy suited to an individual program and its needs. The following questions from the TechFAR Handbook are linked to sections of the handbook that provide additional information and sample contracting language, where appropriate. Additional questions provide greater insight into topics relevant to contracting for Agile software development and point Contracting Officers to other sources of information.

5.1 Agile Software Development Basics

TechFAR Handbook Questions

Q: Generally speaking, what is Agile software development, and how does it fit into the acquisition development lifecycle?

A: Agile software development is a method of software development that utilizes an iterative development process, designs services based on real user needs, and constantly improves software on the basis of continuous user feedback.

See https://techfarhub.cio.gov/handbook/general-considerations/ for more information.

Q: Are agencies authorized to shape their IT software acquisitions around Agile principles? The FAR does not expressly speak to Agile concepts such as refining technical solutions after contract award based on testing and customer feedback or buying a product with a process rather than an identified solution.

A: The principles of Agile software development are consistent with modular contracting, which is discussed in FAR Part 39, Acquisition of Information Technology. In addition, as a general matter, an agency may pursue acquisition practices that are not expressly endorsed in the FAR, including Agile software development, as long as they are not expressly prohibited by law (FAR 1.102(d)).

See https://techfarhub.cio.gov/handbook/general-considerations/#authorized-shape for more information.

Additional Questions:

Q: What’s the difference between a framework and a methodology?

There are many Agile frameworks that are based on Lean and Agile principles. Each framework contains one or more methodologies that interpret the application of the framework into a process or sequence of activities. Most of the methodologies apply variations of iterative development and continuous feedback, as Agile principles encourage fast feedback and continuous, incremental delivery of work in small batch sizes. Common Agile frameworks include Scrum, Kanban, and Extreme Programming (XP).

See the Agile 101 – Agile Primer paper for more details.

**Q: How does the Government choose which Agile Framework to use?**

A: The Government can choose to either define the specific Agile development framework (e.g., Kanban) that a program will use or allow contractors to propose and apply an Agile framework of their choosing. Programs experienced with a specific framework or programs seeking to hire a contractor to develop software as part of an existing environment may require a specific Agile framework. Alternatively, programs inexperienced with Agile development may find it more beneficial to allow the contractor to propose a framework with which the contractor has demonstrated success and to include Agile training and coaching as part of the required solution.

See the Agile 101 – Agile Primer paper for more details on Agile principles, frameworks, practices, and tools.

**Q: What is DevSecOps and how do I incorporate it into the contract?**

A: DevSecOps is a set of systems engineering principles that align with Lean and Agile principles. The main objective of DevSecOps practices is to establish a CI/CD pipeline to the users and continuous monitoring of working software. A successful DevSecOps strategy encompasses working with all stakeholders involved in planning, designing, developing, testing, securing, integrating, deploying, maintaining, and continually improving the product. The work statement for an Agile software development solution should either:

1. Describe the existing DevSecOps environment into which the contractor solution must integrate, OR
2. Require the contractor to propose a DevSecOps environment and capability to operate it based on specific accreditation requirements provided by the Government.

See [https://tech.gsa.gov/guides/what_is_devops](https://tech.gsa.gov/guides/what_is_devops) for more details.
See Appendix A for more details on Agile principles, frameworks, practices, and tools. See the DoD Enterprise DevSecOps Reference Design for more information on DevSecOps.\(^{13}\)

**Q:** What’s the “definition of done,” why does it matter, and how does it fit into the contract?

A: To ensure that users receive the results they expect, the Government must request the contractor to provide a “definition of done” so that is clear, comprehensive, and objective. The Request for Proposal (RFP) and resultant award should state that a definition of done will be established and agreed to by both the Government and the contractor at the beginning of each sprint and release cycle.

Section 4.2 contains a description of the definition of done and its relationship to user stories. Some of the sample language in Section 8 contains references to the definition of done that programs can tailor as needed.

**Q:** Is my role as the Contracting Officer different for an Agile project?

A: For an Agile project, the Contracting Officer is critical in facilitating the active and ongoing engagement between Government and contractor essential to realize the benefits of Agile. While contracting fundamentals remain the same, one of the greatest challenges that a Contracting Officer will face is ensuring that the contracting activities align with the Agile work cadence in terms of continuous reprioritization and delivery of work.

See the Agile 101 – Agile Primer paper for more details on Agile principles, frameworks, practices, and tools.

**Q:** I’ve never received Agile training. Do I need it?

A: Yes. All members of the Agile program team (e.g., Contracting Officer or Specialist; Program Management; Development; Testing; Security; Operations) will need a basic understanding of Agile principles and commonly used methodologies, and how Agile differs from traditional waterfall development to understand the impacts on a solicitation and resultant contract for an Agile solution.

See the Agile 101 – Agile Primer paper for more details on Agile principles, frameworks, practices and tools

### 5.2 How Contracts for Agile Software Development Are Different

**TechFAR Handbook Questions\(^{14}\)**

**Q:** FAR 15.203 requires agencies to identify requirements in their requests for proposals (RFPs). How does this requirement fit with Agile processes, which are based on the premise that it is not realistic to expect users to know exactly what they

\(^{13}\) Source: [https://dodcio.defense.gov/Portals/0/Documents/DoD%20Enterprise%20DevSecOps%20Reference%20Design%20v1.0_Public%20Release.pdf](https://dodcio.defense.gov/Portals/0/Documents/DoD%20Enterprise%20DevSecOps%20Reference%20Design%20v1.0_Public%20Release.pdf)

\(^{14}\) Source: TechFAR Handbook [https://techfarhub.cio.gov/handbook/requirements/#documentation](https://techfarhub.cio.gov/handbook/requirements/#documentation)
need before they see it and rely on refinement of system requirements based on testing and customer feedback after the contract is awarded?

A: Agencies using Agile software development can meet the requirements of FAR 15.203 by identifying a product vision and coupling it with an explanation of how the Agile process will be used to achieve the product vision. Rather than provide a set of “how to” specifications (or a Requirements Traceability Matrix), the product vision focuses on a desired outcome; this is similar to performance-based contracting, which the FAR has permitted for many years.

See https://techfarhub.cio.gov/handbook/requirements/#documentation for more information.

Q: Without having detailed system requirements documented up front, how will the Government ensure it has appropriate documentation and know whether a contractor is performing?

A: Agile software development requires documentation for user stories, acceptance criteria, tasks to be completed to fulfill the definition of done and ensure code quality and compliance with standards. The Agile software development approach involves a highly disciplined process with Government representation on the development team to set priorities and ensure that working software is compliant with contract or system requirements. Progress is monitored by the delivery of deployable code on time.

See https://techfarhub.cio.gov/handbook/requirements/#documentation for more information.

Q: FAR Subpart 7.5 states that contractors cannot perform inherently Governmental work. Because software requirements are refined after contract award, would the use of a contractor in an Agile software development contract be considered inherently Governmental?

A: The use of a contractor to provide assistance to the Government with software development does not fall into the category of inherently Governmental work as described in OFPP Policy Letter 11-0115 or FAR 7.503. Under Agile software development, the Government retains the responsibility for making decisions and managing the process; it plays a critical role in the Integrated Project Team (IPT) as the Product Owner by approving the specific plans for each iteration, establishing the priorities, approving any overall plan revisions that reflect the experience from completed iterations, and approving deliverables.

5.3 Developing the Contract Strategy

TechFAR Handbook Questions\textsuperscript{16}

Q: Are IDIQ contracts the only contract vehicle responsive enough to support the flexibility needed in Agile software development processes?

A: No. Contracts utilizing an Agile software development methodology may be used with any contract type, subject to limitations permitted by the contract vehicle (e.g., FAR Part 12).


Q: To avoid duplication, are agencies limited to only using existing enterprise-wide contracts for IT? What happens when existing contract vehicles do not provide for Agile software development services?

A: Contracting Officers need to make a determination whether an existing contract vehicle supports Agile software development. If existing contract vehicles do not support Agile software development or have limitations, the Contracting Officer is encouraged to use other contract vehicles.


Additional Questions:

Q: Is there a standard contracting strategy for writing a contract for Agile software development?

A: Although there is no single recommended contracting strategy for an Agile software development effort, establishing a flexible, modular contract approach is essential to success. Several factors drive the choice of a specific contracting strategy. The program must understand the operational and programmatic priorities, constraints, and considerations involved in Agile software development to design an appropriate contract strategy.\textsuperscript{17}

See Section 7 for more information.

Q: Modular contracting is the preferred approach to acquire software development in accordance with 41 U.S.C. §2308 – Modular Contracting for Information Technology, and implemented in FAR Part 39.103, but what does that really mean?


\textsuperscript{17} Source: AiDA (https://aida.mitre.org/agile/agile-contract-preparation/)
A: Modular contracting reduces program risk (because it involves managing smaller contracts rather than one monolithic contract), incentivizes contractor performance (by structuring contracts so that the Government is not required to procure additional increments), and enables the Government to continuously innovate and take advantage of the latest technology (by leveraging commercial development standards, when appropriate, in conjunction with agency architecture).

*See Section 7.1 for additional information.*

**Q:** What kind of discussions should I have during pre-solicitation to ensure that I’ve got everything I need for the Agile contract?

A: See Section 6 for more information.

### 5.4 Selecting Contract Type(s)

**TechFAR Handbook Questions**

**Q:** Must agencies only use fixed-price contracts to get the desired result?

A: The selection of a contract pricing structure for acquisitions using Agile software development is no different than selecting a structure for any other contract. Contracts utilizing Agile software development are not limited to fixed-price arrangements; the Contracting Officer is encouraged to select the pricing structure that will result in reasonable contractor risk and provide the contractor with the greatest incentive for efficient and economical performance.

*See https://techfarhub.cio.gov/handbook/pricing/ for more information.*

**Q:** When using a fixed-price contract, how could the line items be structured?

A: In a fixed-price contract, the line items may be structured by desired capabilities, increments, release cycles, and/or groups of resources/teams. The Government may also use optional line items to account for additional work or resources if needed.

*See https://techfarhub.cio.gov/handbook/pricing/#structured for more information.*

**Q:** Do incentives under FAR Part 16 work with contracts for Agile software development?

A: FAR Part 16 describes different incentives that may be used in contracts to ensure the best results. Recently, Government agencies have had success with fixed-price incentive contracts (FAR 16.204, 16.403), fixed-price contracts with award fees (FAR 16.404), and performance incentives (FAR 16.402-2). See sample language for Government contracts for Agile software development services in Section 8 of this guide.

See https://techfarhub.cio.gov/handbook/pricing/#incentives for more information.

Q: How does the Government ensure fair and reasonable prices when acquiring a process such as Agile software development?

A: The Government may determine whether prices are fair and reasonable in a contract utilizing an Agile software development methodology by requesting and evaluating pricing of the effort as a unit of measure that is equivalent to the proposed sprint/release cycle and demonstrating the correlation between the proposed technical solution in the Performance Work Statement (PWS) and the pricing.

See https://techfarhub.cio.gov/handbook/pricing/#incentives for more information.

Q: Doesn’t the fact that technical requirements are not defined substantially increase the risk of a protest?

A: The fact that technical requirements are developed through an Agile process should not increase the risk of a protest.


5.5 Post Contract Award

TechFAR Handbook Questions19

Q: FAR 42.302 lists the contract administration functions to be performed by the Government. When performing contract administration, agencies have noted challenges in committing staff to support Agile software development. Is Agile software development feasible given agencies’ limited resources?

A: Agencies must ensure adequate resources are applied to manage their contracts irrespective of the strategy used. Agile software development is no exception. While the process is highly interactive, the overall amount of work is not greater; work is simply performed differently to produce results faster.

See https://techfarhub.cio.gov/handbook/contract-admin/ for more information.

Q: Because Agile software development is a fluid process with requirements that are refined continuously and iteratively, how can the Government hold contractors accountable in an Agile environment?

A: Even though a key principle of Agile software development is that working software is the primary measure of progress, contractors are still responsible for meeting cost and

schedule goals. The Government holds the contractors accountable for producing working software consistent with the set sprint/release schedule and within budget.

See https://techfarhub.cio.gov/handbook/contract-admin/#accountable for more information.
6 Agile Development Strategy Considerations

This section contains key questions for a program to consider when developing components of an acquisition strategy based on program objectives.

Key Questions on Requirements

Requirements for traditional development usually contain hundreds or thousands of specific and detailed requirements that can take months or years to assemble. The resultant requirements are included in the RFP and, in turn, contractors propose detailed solutions that align with requirements. When using an Agile development approach, contracts should incorporate a leaner and more flexible and adaptable set of requirements that describes the user/business needs at a high level and enables refinement and reprioritization of requirements throughout the development lifecycle.

An initial high-level set of business needs (defined as epics/capabilities/features depending on the Agile framework) that describe the business goals/objectives allows the contractor and Government to work together after contract award to decompose the objectives into more detailed Agile requirements (user stories) prior to starting development activities for a given release or sprint. Agile requirements are prioritized and managed in backlogs (e.g., Product Backlog and Sprint Backlog).

- Does the program have a Product Vision statement and Product Roadmap for the project outcomes? What capabilities are needed in six months, nine months, or a year?
- Has the program developed an initial high-level Product Backlog and assigned a Government Product Owner (PO) to manage the Product Backlog? If not, will the contractor development team, led by the Government PO, help develop or refine the backlog further prior to starting the first sprint?
- How much Product Backlog definition work will be required with the contractor upon award and before the first sprint can begin? (Some planning for the MVP will have to take place upfront, and prior to the start of the first sprint. Prior to the start of the first sprint, the Agile team must initially define and commit to the Sprint Backlog, which is a subset of the Product Backlog.)
- Who is responsible for system integration (the Government, development contractor, or separate systems integrator)? Is an Enterprise Architect ensuring proper integration will occur across all products?
- Have infrastructure requirements been defined? (This is necessary to ensure that the “architectural runway” keeps pace with the Agile delivery cadence. This means that environments are ready for the team as needed and are not the source of potential bottlenecks. Requirements must span collaborative environments and any test environments (to include hardware-in-the-loop labs) that can be assimilated into the CI/CD pipeline.)
● What activities and resources are needed in addition to development (e.g., training materials)?

● What is the desired frequency of releases (e.g., weekly? monthly? three months? six months)? Understanding the release frequency will provide insight into how the team expects to work. Longer release windows organized by specific capability releases could be factors in determining how to structure the contract. However, for teams that plan to release capability frequently, the release cadence may not be a significant factor in the contract structure. As Agile programs mature and releases become more frequent, release cadence should become less of a factor in contracts.

● Is the contractor expected to use an existing development environment and technical infrastructure/design, or will the contractor be required to establish the development environment?

● What external system dependencies exist? Is this information documented to inform contractor proposals?

● How will the Government mitigate integration risk and configuration management if multiple contractor teams conduct parallel development efforts?

● Does the Government anticipate a planning sprint to clarify requirements and ensure the team is set up properly? If yes, how much time is planned for this activity?

● Will the Government expect the contractor to execute an innovation sprint periodically to allow room for refactoring, learning, and innovation? (This could impact the duration of the Agile release cycles.)

● How will the program ensure that security and information assurance requirements are considered? (For example, what must the program consider to ensure secure development and delivery of capability such as incorporating automated tests early and often (e.g., during code check-ins, delivery to an environment, or integration testing)?)

Section 8.1 provides additional information on Agile requirements, practices, and considerations.

**Key Questions on Minimum Viable Product (MVP)**

Agile methodologies use the term MVP to refer to the first version of a product that provides value, data, and learning. The MVP is a collection of features that provide just enough capability to accomplish basic functionality so that the user can provide feedback to influence future development sprints. The MVP should be a manageable product that enables users to employ the targeted solution and assess how it performs in a production environment to be incorporated into future iterations.

● Has the MVP been defined?

● If the MVP has been defined, how quickly does the Government require completion?

● If the MVP is not defined, will the contractor be asked to help define the MVP as part of the effort?
Section 8.2 provides additional information on MVP practices and considerations.

**Key Questions on the Product Roadmap**

A product roadmap provides a sequenced view of key features to be delivered over time. The roadmap is not a project schedule or Gantt chart that defines a set of time-based, dependent activities. It also does not show critical path. The product roadmap, like the product backlog, should be reviewed and updated periodically as work progresses. The Government Product Owner is responsible for the product roadmap.

- Does the program have a roadmap? – Has the program developed a product roadmap to include in the solicitation? Or does the program intend to have the contractor help develop the roadmap as part of the effort?

Section 8.3 provides additional information on roadmap practices and considerations.

**Key Questions on Agile Frameworks**

- Does the program require a specific Agile framework (e.g., Scrum, Scrum of Scrums, XP, Kanban, Scrumban, Nexus, LeSS, DSDM, DaD, other?)? If not, is the contractor expected to propose an Agile software development methodology?
- Does the program have experience in the specified or selected Agile framework? If not, is training planned for entire team? Who will provide the training?
- Will the contractor will be expected to use Test-Driven Development (TDD)? Software developers responsible for developing or maintaining code should have TDD experience because supporting continuous and automated testing is an important part of DevSecOps good practices.
- Will the Government require specific Agile application lifecycle management tools, or will the contractor be able to propose a tool of choice? (This decision may be driven by existing tools and licenses available to the Government and/or used by other programs with interoperability needs)

Section 8.4 provides additional information on Agile methodology practices and considerations.

**Key Questions on Tailoring the Acquisition Lifecycle**

- For programs subject to [DoDI 5000.02](https://www.dod.mil/Portals/49/Documents/ProgramMgmt/DoDI_5000.02.pdf) or [DoDI 5000.75](https://www.dod.mil/Portals/49/Documents/ProgramMgmt/DoDI_5000.75.pdf), has the program performed a lifecycle impact assessment to identify the tailoring of standard program milestones and reviews needed to align with the Agile planning and development cadence (to account for incremental or progressive technical reviews, progressive elaboration of requirements, iterative and incremental updates to required lifecycle documents, and tailoring milestones)? If these items are not tailored for Agile, then the program will be subject to the traditional waterfall-based milestones, decisions, and artifacts.

Section 8.5 provides additional information on lifecycle practices and considerations.

**Key Questions on Agile Metrics**

- Has the program identified Agile metrics it will require?
Section 8.7 provides additional information on Agile metrics.

**Key Questions on Training**

- Will the contractor be required to provide Agile training for the Government team, or will the Government team receive Agile training (specific to the selected methodology) separately from the contract?
- Will coaching be required as part of the work to be performed by the contractor, or will coaching be provided by a third party? How many coaches will be needed? (More than one is recommended for scaled Agile implementations involving more than one Agile team.) It may be in the Government’s best interest to use an external party as an Agile coach for an objective perspective.

Section 8.10 provides additional information on training considerations.

**Key Questions on Roles and Responsibilities**

- Who is the Product Owner? The Product Owner role must be filled by a fully empowered and capable Government individual; this is critical to the success of an Agile program.
- Have all Agile-specific roles and responsibilities been identified, documented, and assigned based on the recommended roles within the selected framework (e.g., Scrum Master, Product Owner, Product Owner/Program Manager, DevSecOps Lead, etc.)?
  - Although not specific to DevSecOps, the following roles will be important to identify for DevSecOps:
    - Who will fulfill the Engineering Lead role?
    - Who will fulfill the Architecture Lead role?
    - Who will fulfill the Security Lead role?
    - Will there be a DevSecOps Lead role?
  - It is important to have a plan and commitment from the stakeholders to participate in the Agile process in order to provide feedback at Sprint Reviews and on incremental software releases. The product will only be as good as the participation of these stakeholders.

Section 8.8 provides additional information on key Agile roles and responsibilities.

**Key Questions on DevSecOps**

DevSecOps is a relatively new practice area for most programs, but is key for enabling continuous integration, continuous delivery, and continuous monitoring of solutions. DevSecOps also espouses automation of processes (e.g., automated testing of the solution functionality and security), and shifting left of team members so that they can engage early and often in the planning and execution of work.

DevSecOps is a broad topic. More information is available in other publications such as the DoD Enterprise DevSecOps Reference Design published by the DoD Chief Information
Office. For programs that are actively pursuing DevSecOps enablement, the following questions could help define the requirement:

- What are the cloud considerations and needs and how will the solutions be purchased?
- Are any specific tools required, or must the contractor have experience with a specific technology stack, to support the DevSecOps goals of CI/CD?
- If the contractor is responsible for developing and/or maintaining a DevSecOps pipeline, how will the contractor be required to demonstrate the capabilities and approach in this area? Will the contractor be required to explain and demonstrate its CI/CD pipeline to show that it can deliver continuously? If the Government already has a pipeline, will the contractor have to report its level of experience operating such a pipeline (e.g., experience with the tools and technologies; skillsets)?
- Who or what Government organization is responsible for the overarching Agile test strategy? Has it been coordinated with developmental test/operational test (DT/OT) organizations? (The Government must coordinate with DT/OT test organizations upfront and gain their buy-in of the contractor’s automated DevSecOps software stack with build-integrate-test built in.)
- Will the program seek a continuous Authority to Operate (c-ATO)? If so, how will this affect the DevSecOps requirements? This should be a collaborative effort between the program and engineering teams.

Section 8.9 provides additional information on DevSecOps practices and considerations.

---

7 Contracting Strategy Considerations

This section presents considerations for building a contracting strategy to support an Agile development effort, to include the benefits of modular contracting and factors for using existing contracts or establishing new contracts.

As noted in Section 5.3, there is no single recommended contracting path or strategy for an Agile development effort, but establishing flexible contracts is essential to success. Several factors drive the choice of a specific contracting strategy. The program must understand the operational and programmatic priorities, constraints, and considerations involved in Agile development to design an appropriate contract strategy. Many of these factors were identified in the form of questions in Section 6.

7.1 Modular Contracting

Modular contracting is a technique that uses one or more contracts to acquire IT systems in successive, interoperable increments. Modular contracting enables breaking down required support into simplified discrete elements that can be achieved through a variety of contracting options to shorten time to award. This enables the Government to develop a collection of contracts with different objectives to meet different requirements that support the overall program objectives.

Modular contracting is the preferred approach for acquiring major software IT systems in accordance with 41 U.S.C. §2308 – Modular Contracting for Information Technology and implemented by FAR Part 39.103, and may also be considered when acquiring non-major systems. It is intended to reduce program risk and to incentivize contractor performance while meeting the Government’s need for timely access to rapidly changing technology. When using modular contracting, an acquisition of a system of IT may be divided into several smaller acquisition increments that:

1. Are easier to manage individually than would be possible with one comprehensive acquisition
2. Address complex IT objectives incrementally in order to increase the likelihood of achieving workable systems or solutions for attainment of those objectives
3. Provide for delivery, implementation, and testing of workable systems or solutions in discrete increments, each of which comprises a system or solution that does not depend on any subsequent increment in order to perform its principal functions
4. Provide an opportunity for subsequent increments to take advantage of any evolution in technology or needs that occur during implementation and use of the earlier increments

21 Source: AiDA (https://aida.mitre.org/agile/agile-contract-preparation/)
22 FAR 39.002 - Definitions
24 Source: https://www.acquisition.gov/content/39103-modular-contracting
5. Reduce risk of potential adverse consequences for the overall project by isolating and avoiding custom-designed system components.

When planning a modular contracting strategy, it is important to understand the program needs and objectives in order to identify the contract vehicles needed to support the mission. The collection of modular contracts should be expected to change and evolve throughout the Agile development lifecycle, especially as scaling occurs and more development activities are added. For example, modular contracting solutions a program may need to acquire to support an Agile development effort could include, but are not limited to:

- **Sustainment of existing legacy systems** – until modernized or no longer needed
- **Cloud needs**
  - Infrastructure as a Service (IaaS) subscription needs – establish an accredited cloud-hosted foundational infrastructure layer for compatible platform environments (servers, network, storage, etc.). A full list of FedRAMP-approved IaaS providers is available on the FedRAMP Marketplace website\(^{25}\)
  - Platform as a Service (PaaS) subscription needs
- **Services**:
  - Cloud development and configuration – support to establish the cloud-based environment, and/or rehost or re-platform existing software applications in the environment
  - DevSecOps expertise – to support instantiation of a software delivery pipeline and demonstrate continuous software delivery
  - Agile development team/s – software developers and designers to develop new or update existing solutions using modern software development and DevSecOps practices (See Appendix B for two examples of contracting for Agile development teams as a Service)
  - Additional subject matter experts – to enable the Government to efficiently design, secure, and scale its solution architecture; they include cloud enterprise architects, platform engineers, cybersecurity engineers, software engineers, DevSecOps engineers, Site Reliability Engineers (SREs), etc.
  - Advisory and assistance (A&AS) support – program office support, Agile training, Agile coaching, etc.
- **Microservice Solutions** (independent software applications – compatible with PaaS/IaaS solution/s)
- **Cybersecurity** – to support security vulnerability protection during software coding development
- **Communication/collaboration tools** – to address purchase, installation, configuration, and licensing costs of any software required to support the management, configuration control, and visibility of the work being performed

---

\(^{25}\) Reference: [https://fedramp.gov](https://fedramp.gov)
7.2 Using Service Contracts for Agile Development

Agile development is characterized by continuous change and reprioritization of requirements. This calls for flexible contracts where cost and schedule are adjusted to allow the scope (priorities of capability needs in the backlog) to change. It represents a shift from traditional contracts where the scope (defined set of requirements) is fixed and does not change without an engineering change proposal and resultant contract modification. Agile methods such as daily collaboration between Government and contractor teams, a mutually agreed-upon definition of done for a sprint/release cycle, and metrics informing the Government about the quality of capability developed and delivered allow the Government to continually assess contractor performance toward achieving program outcomes within a fixed cost and schedule construct.

Service contracts can be used to acquire support for a variety of activities that will enable the Government to implement Agile development. In particular, service contracts enable the Government to contract for development teams by describing desired outcomes and enabling continuous reprioritization of work without requiring contract modifications. It is essential to differentiate between contract requirements and Agile requirements, as shown in Figure 3.

![Figure 3 – Differentiating between Contract Requirements and Agile Requirements](image)

Contracts for development teams can be executed using any contract type permitted by the FAR. Agile best practices recommend contracts with a short (one year or less) initial period of performance (PoP) and options for additional support. Shorter PoPs and options do not commit the Government to years of service by a particular contractor if the contractor’s performance does not meet expectations. If using a modular construct, multiple service contracts for development may be executed to develop discrete capabilities that fit into the overall solution managed by the Government. When leveraging service contracts for Agile
development teams, the Government should be the integrator and in full control of prioritizing the product backlog and managing the development process to ensure individual development contracts align with architecture, vision, and other development activities.

The Government can use a Statement of Objectives (SOO) or PWS to describe its objectives and desired outcomes (e.g., Vision or Capability Needs Statement) rather than a Statement of Work (SOW) defining a detailed set of requirements.

7.3 **Contract Vehicles**

The procedures in FAR Part 8.4 (Federal Supply Schedules), FAR Part 12 (Commercial Items), and FAR Part 13.5 (Simplified Acquisition Procedures for Commercial Items) can be applied to quickly execute contracts for services and microservices as a program evolves. FAR Part 8.4 and FAR Part 12 contracts are limited to Fixed-Price or Time and Materials (T&M) contract types. FAR Part 13.5 authorizes commercial item purchases up to $7M.

FAR Part 16.5 procedures may be used to award task orders against existing contracts to acquire platform and integration subscriptions, as well microservices and services. Examples include **Government-wide Acquisition Contracts (GWACs)** (e.g., NASA SEWP V, NIH NITAAC, GSA Alliant 2), Multi-Agency Contracts (MACs) (e.g., DISA Encore III, DoDIAC), and agency-specific Indefinite Delivery Indefinite Quantity (IDIQ) contracts (e.g., Air Force NETCENTS-2 and SBEAS, Navy NGEN and NGEN-R, and Army CHESS). When considering FAR Part 16 solutions, Program Managers need to carefully consider the Office of Management and Budget (OMB) Spend Under Management Memorandum (SUM) and Best in Class (BIC) solutions.

Another option is to consider establishing a new multiple-award IDIQ contract (leveraging FAR Part 15) or Blanket Purchase Agreement (BPA) (leveraging FAR Part 8.405-3) at the program, portfolio, enterprise, or Program Executive Office (PEO) level. A portfolio-, enterprise-, or PEO-level contract would enable many programs to leverage the contract and the benefits of its streamlined processes, allowing them to focus their strategy and energy on task orders and calls. New enterprise-level IDIQ contracts or BPAs require lead time to establish, but once in place they can offer streamlined procedures for an organization to execute orders or BPA calls as part of a modular contracting strategy.

For Agile development projects that include a prototyping element, “Other Transaction” agreements offer opportunities to structure flexible business arrangements to advance new technologies or to evaluate technical feasibility of new or existing technology in a

---

26 Source: NASA SEWP V GWAC: [https://www.sewp.nasa.gov/](https://www.sewp.nasa.gov/)
27 Source: NIH NITAAC GWAC: [https://www.sewp.nasa.gov/](https://www.sewp.nasa.gov/)
29 Source: DoDIAC MAC: [https://dodiac.dtic.mil/](https://dodiac.dtic.mil/)
30 Source: USAF NETCENTS 2 IDIQ: [https://www.netcents.af.mil/](https://www.netcents.af.mil/)
31 Source: US Army CHESS: [https://chess.army.mil/](https://chess.army.mil/)
Government environment. The OSD Other Transactions Guide and the Contracting Cone provide additional details guiding execution of Other Transaction agreements.

The Contracting Cone (https://aaf.dau.edu/contracting-cone/) outlines a variety of FAR and non-FAR contract strategies. The supporting materials provide details intended to enable collaborative discussions that will help programs to select the right strategy based on environment, constraints, and desired outcomes.

### 7.4 Contract Types for Agile Development Teams

Agile projects will likely call for a variety of contracts and contract types to meet the objectives of a program, as previously described in Section 7.1. When acquiring the services of Agile development teams (software developers and designers to develop new or update existing solutions using modern software development practices), selection of contract type should be informed by the contract strategy chosen for Agile development. If pursuing a service contract to acquire the services of Agile development teams, as discussed in Section 7.2, cost-type contracts are not allowed if using FAR 8.4 (Federal Supply Schedule) or FAR Part 12 (Commercial Items). Instead, Fixed-Price or T&M contracts should be used to acquire services using FAR Part 8.4 or Part 12. Cost-type contracts may be considered if using other contract strategies, such as IDIQ or MAC awards using FAR Part 16.5.

**Fixed-Price**

Typically used when requirements are well understood and well-defined, fixed-price contracts can and have been used to acquire the services of Agile development teams to accommodate the continuous reprioritization of requirements in the product backlog. This is made possible by fixing the price of the level of effort that the contractor should provide over a specified time period and enabling requirements in the product backlog to continuously evolve. The price is negotiated using specified hours in specified labor categories (e.g., software engineer, software developer, software tester, product manager).

Contract PoPs can be short (e.g., 6 months) to align with release cadence cycles and can include options to continue development support as needed. Alternatively, task orders or BPA calls can be executed on a recurring basis (e.g., per release) based on program objectives and desires. Because the scope of the contract is to provide Agile development services and not to define the Agile requirements (see Figure 3), contract modifications are not necessary to reprioritize, add, or delete Agile requirements from the product backlog. A clearly stated and agreed-to definition of done, established at the start of each sprint and release cycle, ensures the Government receives the expected incremental value from the contractor. Section 4.2 contains a description of definition of done and its relationship to user stories. Section 8 contains references to definition of done that can be tailored as needed.

One option is to use a capacity-team-based approach leveraging firm-fixed-price contracts to deliver Agile development services. Under this model, Agile teams deliver value within

---

33 Source: Contracting Cone https://aaf.dau.edu/contracting-cone/ot/
contractually defined technical constraints. An example of this model is provided in Appendix B.

**Time and Materials (T&M)**

The commercial sector often uses T&M contracts for Agile development teams, especially for smaller projects. T&M contracts can provide maximum flexibility for Agile development, especially at the start of a development effort when a team’s velocity (measure of amount of work completed in a given sprint for a given Agile team) has not yet been established. Since an Agile development strategy requires daily interaction between the Government and contractor staff as well as a mutually agreed upon definition of done for a sprint/release cycle, and metrics informing quality of capability developed and delivered, the controls needed to monitor contractor performance are built into the Agile process and provide oversight to ensure delivery of expected working capability and effective cost containment under a T&M arrangement. Structuring the program into smaller, frequent releases (e.g., six months) limits the risks often associated with a T&M contract because of the shorter PoP. Additionally, a T&M or Labor-Hour contract type may be beneficial if an agency is using a modular contracting approach under which one vendor has no direct or exclusive control over the outcome of the products. The Government must be the integrator in this scenario.

DoD programs have been successful in using T&M contract types to acquire the services of Agile teams by scoping increments into task requirement notices aligned to the product backlog, informed by user-prioritized needs. The Government must negotiate labor rates when using a T&M contract type, and establish a ceiling price. An example of this model is provided in Appendix B.

**Cost Reimbursement**

Cost-type contracts may be appropriate when the necessary labor hours and labor mix are not well understood. The Software Engineering Institute (SEI) at Carnegie Mellon University (CMU) Contracting for Agile Software Development in the Department of Defense: An Introduction technical paper offers the following perspectives on cost-type contracts for Agile development:

Cost-reimbursement contracts potentially allow for refinement of the requirements based on the evolution of the working system and the priority for functionality defined by the Product Owner. To be effective, this type of contract requires adequate Product Owner capability to manage and oversee the contracted work. Effective Product Owner capability and active interaction and collaboration, focused on delivery of working software, increases the success of developing the right working software. The flexibility to adjust to changing operational

---

34 Source: AiDA https://aida.mitre.org/models/agile-acquisition/tmrr-phase/contract-prep/
35 Source: TechFAR https://playbook.cio.gov/techfar/
system needs is built into the statement of work or objectives that accompanies the contractual funding constraint.

SEI noted that selection of cost-type contracts for Agile development was driven by the uncertainty of the requirements and the need to prioritize requirements based on changing operational needs. Fixed work cycles and software release cycles, with constraints on the amount of work scheduled for the fixed work cycles, enabled the Government to prioritize work.

The Government assumes more risk in a cost reimbursement arrangement than in a fixed-price contract, as the contractor is less incentivized to control cost.

Hybrid Contracts
In accordance with FAR 16.103(c), an agency might consider hybrid contracts that allow the agency to achieve a better match between the requirement and how the work is priced. Work for which there is a basis for firm pricing could be awarded for a fixed-price while work about which considerable uncertainty remains can be acquired on a cost-reimbursement, T&M, or Labor-Hour basis. IDIQ contracts can be structured to permit multiple contract types, allowing the agency to choose among the various pricing structures.37

7.5 Challenge-Based Acquisition (ChBA) Approach
ChBA is based on the concept that Government agencies can best perform acquisitions if they present the solution to be acquired as a need (the challenge) and potential providers are asked to propose innovative solutions that fill the need. Challenges are issued in terms of operational needs and are accompanied by descriptions of mechanisms for evaluating proposed solutions and contractual terms for provider participation. Typically, solutions take the form of simplified implementations, and evaluations assess how well a solution satisfies the need. Following the guidelines provided in this document, a well-crafted challenge, accompanied by clear, transparent, and effective assessment methodologies and appropriate contracting vehicles, leads to successful acquisitions.

The FAR encourages approaches such as ChBA: “...absence of direction should be interpreted as permitting the Team to innovate and use sound business judgment that is otherwise consistent with law and within the limits of their authority. Contracting officers should take the lead in encouraging business process innovations and ensuring that business decisions are sound...”

ChBA is especially appropriate in situations where the Government’s need is urgent and time critical, where no traditional solution seems viable, or where emerging technologies have the potential to provide non-traditional solutions. The ChBA can be applied to large, multi-year major system acquisitions, and should be well thought out and designed to drive to solutions that mitigate known risks.

37 Source: TechFAR https://playbook.cio.gov/techfar/
Several acquisition strategies are available for ChBA. The choice of strategy depends on circumstances: acquisition objectives, available time, complexity, technology ambiguity, challenger pool size, and acquisition scope. Some options include multiple-award IDIQ contracts for evaluation and procurement, Broad Agency Announcements (BAAs) for technology creation followed by competitive procurement, BAAs exclusively for intellectual property (IP) creation, and Other Transaction Authority (OTA) for prototypes and demonstrations, with a subsequent transition to a procurement. In all cases, applying the guidelines in this document to create a pool of qualified offerors, followed by successive evaluation cycles, should lead to a successful procurement that adheres to all Government regulations.38

**Key Takeaways:** There is no “right” or “most appropriate” contract type for acquiring the services of Agile software development teams. Agile projects may call for a variety of contract types to meet the objectives of a program. Close coordination among all members of the program team will help inform contract type selection.

8 Sample Language and Guidance

This section contains guidance and sample language for Agile work statements (e.g., PWS/SOO) for Agile development. The language is intended as sample language and meant to be tailored into a specific contract if appropriate and as needed. The work statement is developed by the Program Manager and provided to the Contracting Officer/Specialist as an element of the RFP package. Collaboration is necessary to ensure the work statement aligns with the selection of contract strategy and contract type.

The following guidance offers key considerations when contracting for Agile software development solutions.

8.1 Requirements Considerations

Programs executing a waterfall methodology produce a comprehensive set of detailed requirements against which contractors develop solutions. Upfront requirements planning, often referred as a "risk reduction" activity, can take anywhere from months to years to complete. The resulting requirements are included in the RFP, evaluated, and put on contract. To achieve the proposed and negotiated outcomes, requirements must be accurate and depict the desired product at a high level of detail and confidence.

Decades of upfront requirements planning have proven that it is difficult to identify all requirements at the start of a project, especially for large and complex systems or systems of systems. Over time, and through progressive elaboration of requirements and design, the program identifies changes to the original requirements and design that must be addressed and that may require change proposals and possible contract renegotiation, especially under fixed-price contracts.

By contrast, with an Agile approach the program does not presume it can define all requirements from the start. Rather, the program expects continuous elaboration of requirements and design due to an iterative and incremental approach to planning and releases that deliver smaller chunks of working software to users at frequent intervals. The advantage of performing the work in smaller chunks is that the Agile team receives feedback faster and more often, allowing the team to incorporate the feedback into the next incremental release.

Agile software development also embraces emergent design principles by leaving room for innovation and change to take place as the solution evolves rather than locking in requirements at contract award. To enable this approach, Program Managers should consider the following when developing the requirements document.

- **Maintain accountability over requirements and design:** Maintain Government ownership of and accountability for the requirements and design processes by taking an active role in the definition, prioritization, and delivery of value. This responsibility usually rests with the Business Representative. In an Agile Scrum methodology, the Business Representative role is recognized as the Product
In general, every Agile team should include one Business Representative with accountability for value delivery.

- **Define the vision and objectives rather than specific product requirements:** Organize the project outcomes around desired business value and project objectives rather than product specifications. The Government should use a SOO to define the vision and desired objectives rather than the overly prescriptive product and design requirements typically contained in a SOW.

- **Expect refinement of technical requirements post-award:** Expect and plan for technical requirements refinement to take place post-award through a disciplined and iterative process of testing and user feedback. The contract could also be configured to allow for an initial period of planning with the contractor to shape the work to a point where both the Government and contractor are comfortable with the scope of work for, at a minimum, the first increment or release.

- **Leave room for innovation:** If the program is open to the idea, communicate openness to considering proposed design alternatives from the contractor during the PoP to enable emergent design patterns. The Government should communicate that the features and requirements within scope are subject to change throughout the contract PoP, and encourage the contractor to suggest improvements based on practices and lessons learned from other engagements.

- **Identify an Agile requirements model (e.g., user stories, etc.):** Ensure that the program agrees on a defined Agile requirements model before Backlog development work begins. Similar to defining a traditional requirements model, Agile also has several variations of requirements models. Most center around a core of epics, Features, and user stories. The key is to select a model that can be used to define the Agile Product Backlog. If the program has not chosen a specific Agile model but is open to considering various options, then the program should request that the contractor provide an Agile requirements model.

- **Consider dependencies on external systems:** Often a system exists within a system of systems, in which case the system is needed by and/or depends on other systems. Program Managers should account for such dependencies by working with the program staff to identify all important dependencies, specifically if the dependency provides a required capability or feature that must be documented in the SOO/PWS/SOW.

- **Consider emergent design patterns such as User Centered Design (UCD):** Determine if the program is open to prototyping or modeling some of the capabilities or features by developing user designs (or user concepts) without developing actual code. This can be performed by applying UCD practices (e.g., developing personas, use cases, wireframing, tools that support observation and analytics of user behavior, and workflows). Applying UCD practices can prove cost...

---


40 Source: TechFAR [https://playbook.cio.gov/techfar/](https://playbook.cio.gov/techfar/)
effective as a result of working with users to refine the design before any code is written.

- **Design deliverables that align with Agile methodology**: Traditional projects often include a detailed prescriptive list of what the vendor should deliver and how the artifacts are accepted. The list of artifacts that usually asks for lengthy documents are contained in the contract data requirements list (CDRL), which varies greatly from contract to contract. An extensive list of deliverable documents is not required for Agile; when applied appropriately, Agile is a structured and disciplined process that encourages an incremental approach to design and delivery. As a result of the incremental approach, the Government will receive “empirical” evidence of performance in the form of working software. The Government should request “just enough documentation” to validate that the capabilities are being delivered in compliance with agency governance and IT system lifecycle requirements. To avoid excessive upfront documentation, the program should

  - Distinguish between high-value documents and those that simply “check the box” or do not contribute to the value or quality of the product being delivered.
  - Work with stakeholders to nominate traditional deliverable documents that do not add value to Agile development for exclusion.
  - Streamline the format of the required documents as appropriate and work with the decision authorities to align on an incremental and iterative approach to document completion (to follow the cadence of Agile delivery and align with emergent design principles).
  - Consider possible efficiencies that could be gained by changing the format and delivery of the artifact. For example, consider whether the artifact can be automatically generated from the project tools being used by the program.

*Sample work statement language for requirements (tailor as needed):*

> “The contractor will work in a team-based Agile environment. The Product Owner will specify high-level requirements/capabilities to the Agile team. As in typical Scrum-based Agile processes, the Agency Product Owner will work together with the team to develop and estimate user stories and establish acceptance criteria. These acceptance criteria will specify expected functionality for a user story, as well as any non-functional requirements that must be met in the development of the story. The Agency Product Owner, supported by subject matter experts and business analysts, will determine whether or not acceptance criteria have been satisfied.”

> “The [project] desires development of the [product] solution to include [software development]/[COTS integration and customization], test and evaluation, information...”

41 Source: TechFAR ([https://playbook.cio.gov/techfar/](https://playbook.cio.gov/techfar/))
assurance, operational test support, and system training. This [project], with optional objectives, will result in a delivered and operational system based upon the attached objectives. This [optional: limited deployment] system will have the required operable system interfaces, receive an approved [[Authority to Operate or Interim Authority to Operate] or [continuous Authority to Operate]], and meet the system/subsystem requirements. [optional: This effort will support the Systems Engineering Technical Review (SETR) process for system approval and acceptance.]

**Objective 1**
The system developer will participate in Engineering, Programmatic, Design, Information Assurance, and Logistics reviews. [Insert additional project-specific details.] The system developer will be responsible for environment stand-up for selected users. This objective will have a period of performance of 3 months.

Specifically, the Contractor shall annotate, refine, and prioritize the Backlog consisting of epics and user stories that are maintained in [tool]. The Contractor shall conduct release and sprint planning on the subset of technical requirements and high-level design elements to be satisfied during each sprint and subsequent release. The Contractor shall capture and document the technical requirements and acceptance criteria for each sprint.

Optional Objective(s)
The following describes “optional” new capabilities or strategic training that may be exercised at the Government’s discretion to design, enhance, configure, customize, test, implement, train and document new capabilities that will be required to deliver new enhancement and/or training on behalf of the [Agency/Service/Organization]. These optional CLINs will be considered for award as the [product] environment matures over the period of performance. The timing of the optional CLINs and placement within subsequent option years represents the Government’s best estimates for when those services will be required.”

Additional language for consideration:

“The Contractor shall evolve [solution] capabilities to support Government prioritized features.

While forecasted features, and requirements are stated below, consistent with Agile- and Sprint-based development, the features and requirements are subject to change based on Sprint planning during the period of the contract.

The forecasted features [listed below] may be accelerated, changed, or interchanged within the scope as determined in program increment or sprint planning.”
Sample work statement language for user centered design and technical design work (tailor as needed):

“5.2.1 USER CENTERED DESIGN (UCD)

The Contractor shall provide a single UCD team, made up of at a minimum [x] FTEs with the appropriate technical skillsets, which shall provide on-going support to [organization] staff offices and Lines of Business (LOBs).

[Optional, if desiring a research plan before the UCD team can start:] The Contractor shall develop a UCD Research Plan that it will use for its UCD process, identifying the types of questions that it will ask and collection mechanisms.

During the UCD process, the Contractor shall leverage existing personas representing the users, or create new ones if none exist. A persona is a user archetype used to help guide decisions about product features, navigation, interactions, and even visual design. In most cases, personas are synthesized from a series of ethnographic interviews with real people, and then captured in 1–2 page descriptions that include behavior patterns, goals, skills, attitudes, and environment, with a few fictional personal details to bring the persona to life.

The Contractor shall update the backlog generated during this process based on data-driven feedback from users and various stakeholders to help drive prioritization of future capabilities.

The Contractor shall work with the [organization] LOB/office to develop and assess the epics, user stories, Requirements Backlog, and Business Processes. At the application project level, the Product Owner will prioritize User Stories, creating a backlog of work. After the prioritized backlog has been set, the Contractor shall create work items to deliver the work. All user stories will be uploaded to [tool] and must be accepted by the Product Owner.

Based on the results of user meetings, the Contractor shall provide a proof of concept to the Contracting Officer’s Representative (COR), including a wireframe (in a computer-generated format). The UCD team shall deliver screen flows and/or clickable prototypes to demonstrate understanding of the requirement and solution approach. The Contractor shall create visual designs to demonstrate the visual appeal and layout of the project and shall annotate them to show how the design satisfies user-specific requirements.

The Contractor shall generate the appropriate technical epics/user stories. The work items shall then be mapped into sprints based on scope and complexity. [optional: if the Requirements Traceability Matrix (RTM) is maintained to track specific functional and technical requirements to Backlog stories:] The RTM shall identify/finalize and track the release sprint to which each functional or technical..."
requirement is assigned, listing all system/component requirements and detailing how and where they are addressed in the system design.]

[Suggested Deliverables:
A. UCD Research Plan
B. Proof of Concept (e.g., wireframe, screen flow, prototype)
C. Epics/User Stories/Requirements Backlog]

TECHNICAL DESIGN

Prior to the start of each sprint, the Contractor shall analyze the selected user stories, the impact on security, including Authority to Operate (ATO), and develop a Technical Requirements Document for each sprint that captures the technical design. The Contractor shall validate the technical design against engineering/data architecture requirements.

The Contractor shall capture and provide the details of any system services that will be affected by the upcoming Sprint in the Technical Requirements Document for each release. The Contractor shall capture the information for all new features and enhancements, as well as for the entire release package. The Contractor shall perform and document the following:

1. Based on the requirements elaboration, the Contractor and Product Owner shall agree on a Definition of Done, as captured in the Technical Requirements Document for the upcoming Sprint. The Definition of Done shall be based on the user stories to be completed and the functionality to be delivered.
2. Document … [identify specific expectations in terms of design artifacts and deliverables]

Note: Delivery of a TRD with each sprint/release should not contradict the Agile Principle of prioritizing working software over comprehensive documentation. The TRD can comprise the already required Agile products such as user stories, definitions of done, and test cases rather than creating new material/documentation. It can also be updated iteratively so that the information is expanded over time and in alignment with work to be performed.

Deliverables:
A. Technical Requirements Document

... OPTIONAL TASK FOR USER CENTERED DESIGN...

ADDITIONAL UCD (OPTIONAL TASK 1)
[Program] may exercise this optional task for additional UCD teams to support system enhancement projects following the UCD process described in Section [5.2.1] of this PWS. The specific scope of functionality to be delivered for each exercise in the optional task will be determined with Product/Project business owners prior to exercising the optional task.

The Contractor shall provide a UCD team (approximately [x] FTE resources per team made up of the appropriate technical skillsets) in one-month increments to achieve a specified scope of work as defined by the Product Owner.

8.2 Minimum Viable Product (MVP) Considerations

An MVP is often defined during project initiation and refined during the planning period. The MVP consists of a selection of features that provide just enough capability to the end-user to permit evaluation while also prompting users to give feedback to influence future sprints. The MVP should be a manageable piece of work that enables the program to see how the solution performs in a production environment. There may be situations in which the MVP cannot be operational; in such cases the program must align on the definition of “delivery.” For example, the MVP could be “launched” in a staging area that mimics production to allow the end-user to test the capabilities and provide the program with feedback.

The program uses the MVP concept to identify capabilities that will be deployed, generally during the program’s first release. The contractor and Government can work together to define or refine the MVP after contract award to ensure alignment with the vision and goals. In some cases, the Government may define a duration or no-later-than time for delivery of the MVP. The MVP delivery could be used when defining a payment schedule.

The MVP does not minimize the need for the remaining features that are not part of the MVP. Rather, the MVP allows the project team to focus on a subsection of the features with the intention of delivering meaningful capability to the end-user faster in order to receive feedback earlier. The team will use the feedback to make decisions on the next set of features and enhancements, and to determine when enough capability has been delivered.

Sample MVP language – if the Government has defined an MVP and will require the contractor to refine it after contract award (tailor as needed):

"The Contractor shall collaborate with the business representative and other subject matter experts to [review] the minimum viable product (MVP). The MVP description in Section (x) identifies the features that provide just enough capability to the end-users to enable them to provide feedback to influence future development sprints. The MVP will be delivered in the first release."

43
Sample MVP language – if the Government has **not** defined an MVP and will require the contractor to propose an MVP solution that may be refined after contract award (tailor as needed):

"The Contractor shall propose an MVP solution containing features that provide initial capability to the end-user while also encouraging end-users to provide feedback to influence future development sprints. The MVP will be delivered in the first release. The Contractor shall collaborate with the business representative/Product Owner and other subject matter experts to refine the MVP after contract award."

### 8.3 Agile Product Roadmap

Traditional projects using a waterfall methodology usually require a detailed Integrated Master Schedule (IMS) to develop, test, integrate, and deliver a product (see Figure 4). The schedule assumes that the team has confidence in the activities through final product delivery and generally expects minimal variance in schedule or cost. If the schedule changes significantly (usually exceeding 10%) as a result of changes in requirements, design, or scope, the team must engage in a time-consuming replanning effort to update the IMS. Changing the IMS has downstream impacts on resources and cost as the schedule usually shifts to the right. Resulting contract changes and the overall impact on the current work are high.
The Agile roadmap (Figure 5) provides a prioritized, value-driven view of work to be performed that also aligns to the program vision. The roadmap depicts the capabilities or features to be developed over time while avoiding overly prescriptive plans. Because Agile roadmaps are defined at a higher level of abstraction, they allow the teams to collaborate on lower level requirements and design details.

---

42 From GSA Tech Guides https://tech.gsa.gov/guides/develop_an_agile_product_roadmap/
The initial version of the roadmap should, at a minimum, depict the capabilities and/or features to be delivered in the MVP (organized by sprint) and align to the overall product vision. Depending on the contracting strategy, it may be desirable to request the contractor to develop the roadmap as an initial activity, which in turn depends on the state of planning at the time of solicitation and the overall strategy for support. If the Government has developed an initial roadmap, it should be included in the RFP. As with any plan, the roadmap can change if it continues to remain in alignment with the vision and scope of the contract work.

To avoid re-creating an IMS, the program should define the roadmap at a high enough level to show the desired direction of the program and what it wants to achieve in terms of value (described as capabilities, or perhaps features). The roadmap should not be task or activity oriented, but outcome/goal oriented, so that it leaves enough room for the team to innovate and learn along the way.

*Sample work statement language if the Contractor is required to develop or work with an already existing product roadmap (tailor as needed):*

"As an initial activity, the contractor shall [receive a product roadmap from the Government] or [lead the effort to develop a product roadmap]. The items identified in the roadmap must align to the overall product vision and be organized by [capabilities/features]. The team shall use the roadmap to organize the Backlog, prioritize work, and ensure that the appropriate [capabilities/features] are being delivered at the right time. The Team can adjust the product roadmap, in consultation

---

43 From GSA Tech Guides [https://tech.gsa.gov/guides/develop_an_agile_product_roadmap/](https://tech.gsa.gov/guides/develop_an_agile_product_roadmap/)
with the Product Owner, prior to each [release/sprint] planning event. The Team shall, in consultation with the Product Owner and facilitated by the ScrumMaster, allocate backlog items to Sprints that align to the roadmap. The Contractor shall deliver the MVP within the allotted timeframe of [x] months. End-user feedback shall be documented and reviewed with the Product Owner.”

If the Government has developed a roadmap, the RFP can request the contractor to propose a planning sprint to develop the backlog and commit to the MVP as a team after contract award. Sometimes this activity is contained within what is called Sprint 0. Usually the Sprint 0 activity has about the same duration as an actual Sprint, or perhaps is a little longer to allow for initial planning.

Sample work statement language for Sprint 0 – assuming the roadmap has been developed by the Government (tailor as needed):

"As an initial activity, the contractor shall work with the Government [business representative/Product Owner/Product Team] in a Sprint 0 planning activity to [develop/review] the product roadmap as a team, and agree on necessary adjustments within scope of the [vision] [and] [statement of objectives] defined in the [SOO/PWS/SOW]. The duration of Sprint 0 will be [x] weeks."

8.4 Agile Framework

The Government Program Manager/Requirement Owner can choose to define the specific Agile framework (e.g., Scrum, Kanban) that a program will use. Alternatively, the Government can allow contractors to select and propose an Agile framework of their choosing. For either option, there are important factors to consider. Government ownership of the process is a critical, strategic consideration. Owning the process organically has proven successful for mature Agile programs. However, this requires upfront planning, training of staff, and a commitment to continual learning and improvement.

Programs with experience using an Agile framework stand a better chance of success in owning the process and hiring a contractor with specific skillsets to meet specific objectives (as opposed to hiring a contractor to deliver a product). Alternatively, programs that are new to Agile may find it more beneficial to allow the contractor to propose a framework the contractor has demonstrated success using and include Agile training and coaching as part of the requirement.

Sample work statement language for an Agile Scrum-based methodology – Program-specific Agile framework; framework methodologies and processes to be agreed upon after contract award by Agile team (tailor as needed):
“The Contractor shall provide the [organization] with Agile team(s), using Agile Scrum to deliver the solution. The Contractor teams are expected to be high-performing, cross-functional teams. The Contractor shall propose cross-functional teams that will support all application development and operations activities defined in this task. Capacity of a team will be monitored and measured using stories to ensure optimal productivity. In collaboration with the Government, the Contractor shall provide stories with the requisite information. A story represents any unit of measurable work that provides value to the [AGENCY NAME] and will serve as the basis for measurable work/deliverables.

Stories shall include the following information:
1) Title
2) Description
3) Level of Effort (e.g., story points, hours, etc.)
4) Business Value
5) Definition of Done
6) Acceptance Criteria
7) Dependencies
8) Other attributes that characterize the story
9) Type of Work (Appendix [x] – Work Types)

Stories can fall into one of four states:
a) Proposed: The story is complete (contains the required attributes) but has not been approved by the COR.
b) Approved: The story has been approved and work can begin.
c) In-Progress: The story is approved, assigned to a team, and is currently under development.
d) Done: The story has been completed (meeting the definition of done and all acceptance criteria) and is accepted by the COR.

Prior to executing any work, the Contractor shall develop (in conjunction with stakeholders) a user story and obtain approval. It is assumed the creation of stories will require participation from multiple stakeholders (Business Units, etc.). The Contractor is responsible for the coordination of all activities required to complete each story. The Contractor is responsible for tracking and decomposing each story down to the working level (generally requiring less than 10 business days to complete the story). The Contractor shall clearly show traceability and dependencies between stories. At the request of the COR, the Contractor shall develop milestones and a detailed schedule for more complex stories (such as a large server upgrade or a large database migration.)
The [AGENCY NAME] will evaluate the Contractor’s performance on this task on the basis of:
- Meeting the definition of done for each story, sprint, and release to accept the work:
  • Delivering stories, sprints, or releases that consistently meet the definition of done.
- Work estimation:
  • Making consistently accurate estimates of capacity (i.e., hours, story points, etc.) and actual capacity used.

8.5 Lifecycle Considerations

Agile software development calls for tailoring that will impact acquisition lifecycle stages, milestones, and required documentation. For programs that must comply with standard DoDI 5000.02 or DoDI 5000.75 milestone decisions and reviews, Agile software development activities may be aligned with key lifecycle milestones. However, many of the documentation reviews should account for the iterative nature of document development. In tailoring of milestone decision and reviews, some of the requirements and reviews may be fulfilled by Agile functional equivalent events.

Agile software development mitigates the need to deliver completed architecture, requirements, and design documents upfront. Agile projects do produce these documents; however, requirements and design artifacts will not be fully complete prior to the start of development work. Instead, the artifacts will be developed iteratively in a collaborative environment and elaborated upon prior to each release or sprint. Note that there are functional equivalents in Agile as indicated in Table 1 from a U.S. Marine Corps pilot program.

The sample below applies to a DoD program that must comply with DoDI 5000.02 milestone decisions and reviews and has provided a crosswalk of how the reviews can be accomplished while still executing in Agile.

Sample work statement language for providing a crosswalk of acceptable Agile artifacts (tailor as needed):

“The Contractor will continue refining the proposed solution in support of Systems Requirements Review(s) (SRR) event(s) as part of the Systems Engineering Technical Review (SETR) process that will support the final solution. This [Task order] will support the eventual Initial Release Planning Review (IRPR), Systems Functional Review (SFR), Preliminary Design Review (PDR), Critical Design Review (CDR), Test Readiness Review (TRR), Systems Verification Review (SVR) and other Systems Engineering Technical Review (SETR) events as required. Documents will be developed iteratively and in alignment with Agile planning and execution of work. Technical reviews will account for the iterative nature of the development work. See Table 1 for alignment of
technical reviews and supporting artifacts to Agile functional equivalents. The Contractor is responsible for providing...”

Table 1 – Mapping Agile to Traditional Reviews

<table>
<thead>
<tr>
<th>Technical Reviews in the Agile Process</th>
<th>Traditional Analogous Systems Engineering Technical Review</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Initial Release Planning Review (IRPR)</strong></td>
<td>System Requirements Review 2 (SRR2)</td>
</tr>
<tr>
<td>● Focused on Initial Release and corresponding sprints</td>
<td>System Functional Review (SFR)</td>
</tr>
<tr>
<td><strong>Infrastructure Review (IR)</strong></td>
<td>(Incremental PDRs will be conducted at the sprint levels)</td>
</tr>
<tr>
<td>● Proposed Hardware Infrastructure</td>
<td></td>
</tr>
<tr>
<td>● Estimated Virtualized Resource Pool</td>
<td></td>
</tr>
<tr>
<td><strong>Release Planning Reviews (RPR)</strong></td>
<td>Systems Functional Review (SFR)</td>
</tr>
<tr>
<td>● Oversight will be delegated to the Agile Review Board</td>
<td>● Subsequent release SFR</td>
</tr>
<tr>
<td>● Focused on follow-on release and corresponding sprints</td>
<td></td>
</tr>
<tr>
<td><strong>Sprint Planning/Reviews</strong></td>
<td>Sprint Preliminary Design Review (SPDR)**</td>
</tr>
<tr>
<td><strong>Daily Build/Test/Integration</strong></td>
<td>Critical Design Review (CDR)</td>
</tr>
<tr>
<td><strong>Sprint Demonstration Review</strong></td>
<td>N/A</td>
</tr>
<tr>
<td>*** Completed products are demonstrated to the Product Owner</td>
<td></td>
</tr>
<tr>
<td><strong>Release Demonstration</strong></td>
<td>Integration Readiness Review (IRR)</td>
</tr>
<tr>
<td></td>
<td>Test Readiness Review (TRR)</td>
</tr>
<tr>
<td><strong>Sprint and Release Retrospectives</strong></td>
<td>Continuous Process Improvement (CPI)</td>
</tr>
<tr>
<td>● Assessment opportunity to determine what went well and what did not for sprint/releases</td>
<td></td>
</tr>
<tr>
<td><strong>Systems Verification Review (SVR)</strong></td>
<td>Systems Verification Review (SVR)</td>
</tr>
<tr>
<td><strong>Operational Test Readiness Review (OTRR)</strong></td>
<td>Operational Test Readiness Review (OTRR)</td>
</tr>
</tbody>
</table>

8.6 Product Backlog Management

The Product Owner is accountable for developing and prioritizing the Backlog and for ensuring items in the Backlog include acceptance criteria necessary for the Government to accept the work as complete. The contractor may be required to support the maintenance of the Backlog, but ultimate accountability should remain with the Government in terms of ensuring value delivery that meets Government customer expectations. The Government should state if it requires a specific tool to maintain the Backlog.
Sample work statement language for working with the Product Backlog (tailor as needed):

"The Contractor shall work closely with the Government [Business Representative/Product Owner/Product Team] to ensure that the Product Backlog, stored in [software/tool], is continually and regularly maintained in alignment with the Agile cadence so that the necessary Release and Sprint Backlog items (epics and user stories) are defined with acceptance criteria and prioritized by the [Business Representative/Product Owner/Product Team] prior to the start of each release and sprint. The Contractor shall prioritize development in accordance with the Program Backlog."

The program must identify how sizing will be established for each item type (e.g., epic, user story) in the Backlog (see Table 2). The program office should be comfortable with the selected sizing method, as this will inform complexity sizing and level of effort for pricing sprints throughout the development lifecycle. Backlog tool selection may factor into the decision, especially if the Government has invested in licenses and training for a specific tool. If so, the Government may wish to specifically require use of the tool to manage Backlogs. (The same logic can apply to other areas such as DevSecOps and software development in terms of the technologies already in use.) The contractor should understand how the Program team sizes Backlog items and the intervals at which the items are reviewed and sized. Alternatively, the Government should request that the contractor propose sizing approaches and cadence (review periods) for Backlog items.

Table 2 – Sample Table for Identifying Desired Sizing Approach for Backlog Items

<table>
<thead>
<tr>
<th>Backlog Item Type</th>
<th>Sizing for Complexity or LOE</th>
<th>Review</th>
</tr>
</thead>
<tbody>
<tr>
<td>Epic</td>
<td>Story Points</td>
<td>At Product Backlog Review</td>
</tr>
<tr>
<td>Feature</td>
<td>Story Points</td>
<td>At Release Backlog and/or Sprint Backlog</td>
</tr>
<tr>
<td>Story</td>
<td>Story Points</td>
<td>At Release Backlog and Sprint Backlog</td>
</tr>
<tr>
<td>Task</td>
<td>Story Points or Duration in Time</td>
<td>Sprint Backlog</td>
</tr>
</tbody>
</table>

8.7 Agile Metrics

Rigidity or prescriptiveness regarding Agile metrics is counter to the Agile methodology. In Agile, tracking progress does not align to traditional DoD metrics to validate cost, schedule, performance, and quality of programs and projects. While cost, schedule and performance continue to have meaning in an Agile program, they must be viewed and managed differently for Agile, as shown in Table 3.
The list below provides general guidance on Agile metrics but is not intended to be all-encompassing or definitive. A baseline of metrics should focus on value delivery, quality, and performance (Agile process adoption). Sample metrics include:

- **Delivery Metrics**: related to pace of delivery and progress
  - Progress against the roadmap (i.e., number of features completed; number of sprints completed; release burndown). The Release Burndown (also known as Release Burnup) is a good visual indicator of whether the team will be able to complete all the items in the Product Backlog assuming the Release Velocity.
  - Progress against the Product Backlog (i.e., number of story points completed; number of stories delivered; percentage of total Backlog completed; release burndown).
  - Progress against the sprint (i.e., number of story points completed; number of stories delivered; percentage of total Sprint/iteration work completed; Sprint/iteration burndown). Specifically, the Sprint/iteration burndown metric allows the team to track work progress and make predictions regarding whether the team will complete the work on time.
  - Progress against the release (i.e., number of story points completed; number of stories delivered; percentage of total release work completed; percentage of Product Backlog completed).

- **Quality Metrics**: related to product quality
  - Testing results (e.g., number or percentage of tests that passed/failed)
  - Number of issues found per sprint (defect count)
  - Issue type histogram (bar chart that organizes issues by type and count to give a sense of the issues that are the biggest challenge to progress)

---

**Table 3 – Agile Alignment to Traditional Metrics**

<table>
<thead>
<tr>
<th>Traditional DoD Measures</th>
<th>Characteristics</th>
<th>Agile Measures</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Cost</strong></td>
<td>Dollars Resources</td>
<td>Work Elements Velocity/Capacity</td>
</tr>
<tr>
<td><strong>Schedule</strong></td>
<td>Time Milestones Planning</td>
<td>Sprint/Release Planning Release Burn-up/down Charts Sprint Burn-down Charts</td>
</tr>
<tr>
<td><strong>Performance</strong></td>
<td>Customer satisfaction Customer capabilities Quality (of product) Performance (of contractor)</td>
<td>User Story Completion and Acceptance Test Coverage</td>
</tr>
<tr>
<td><strong>Quality</strong></td>
<td>Design and Architecture Development (of code) Test and Integration</td>
<td>DevSecOps metrics (e.g. Test automation) Value Delivered Defects Corrected and Escaped</td>
</tr>
</tbody>
</table>
Scatter plots of stories and time to completion (a more advanced analysis tool that provides a method to quickly isolate outlier stories to understand why some stories took longer to complete).

- **Process/Performance Metrics**: related to adherence to the Agile framework and process
  - Ensure that Agile ceremonies are defined. The ceremonies are usually events organized around planning, demonstrations, retrospectives, or daily standups. Documentation of the Agile process will ensure that all team members and leadership understand the framework, roles, responsibilities, and expectations.
  - Roles defined and filled: If the Government does not define a specific Agile framework or methodology, it should avoid being overly prescriptive in roles, but should ensure that the roles are identified, documented, and assigned as part of the original RFP response, or early after award. Once the work begins, the Government should request information on the team composition and whether the positions are filled. Specifically, the Government roles related to the business representation (e.g., Product Owner) are critical to the success of the project and to ensuring that the right value is delivered according to the right priorities set by the Business Representative / Product Owner.
  - Integration of Contracting, Testing, Security, and Operations personnel as part of the Agile team
  - Number of stories removed from sprint
  - Story completion ratio and story point completion ratio (= percent complete/percent planned)
  - Number of blockers (understanding blocker sources and reasons could help reduce the potential of future blockers)
  - Training participation. This is another important metric. Training all team members and senior leadership in the Agile framework before the start of the project will help improve overall success. All contractor staff must also be trained.
  - DevSecOps metrics such as lead time for changes (time from requirement definition to delivery), build frequency, deployment frequency, mean time to restore service, and change fail rate are key software delivery performance indicators.\(^4\)

- **Cost Metrics**: related to cost measurements
  
  Cost can be determined based on the number of hours planned and delivered against stories within a sprint. The costs per story metric are subjective and vary according to the complexity of the stories in a sprint and the number of team members contributing to that sprint (see Table 4). Work elements are typically

\(^4\) Source: 2018 Accelerate State of DevOps Report by DORA
not equivalent to time, and programs must take this into consideration when evaluating these metrics. As described previously, teams may also need several sprints to “normalize” their estimates/velocity, and the cost per work element may fluctuate in the beginning. This metric depends heavily on the accuracy of the estimates and stability of the team’s velocity. Recommended metrics to manage cost include those listed in Table 4.\textsuperscript{45}

<table>
<thead>
<tr>
<th>Metric</th>
<th>Used to Answer</th>
</tr>
</thead>
<tbody>
<tr>
<td>Total work elements committed to sprint</td>
<td>Projected cost of work scheduled</td>
</tr>
<tr>
<td>Total hours planned per story in sprint</td>
<td>Projected cost per story</td>
</tr>
<tr>
<td>Actual hours logged against each story in sprint</td>
<td>Actual cost of work performed</td>
</tr>
<tr>
<td>Actual work elements completed in sprint</td>
<td>Cost of work completed for potential delivery</td>
</tr>
</tbody>
</table>

Table 4 – Sample Cost Metrics

8.8 Agile Teams, Roles, and Responsibilities

Prior to the start of development work, the program should decide on the number of development teams needed based on the size and scope of the development effort. At a minimum, the program should ensure that team sizes do not exceed the recommended size for an Agile software development team (Scrum guidance recommends 7 plus or minus 2). The program should require that the team members have, at a minimum, two specific skillsets each (e.g., developer and tester). Efficient Agile teams have members with multiple skillsets, which reduces the chance of bottlenecks resulting from skillset constraints. The program should require that team members be dedicated full time and not spread across multiple teams.

Because Agile encourages a commitment to continuous learning and experimentation, it is important that the contractor commit to the same principle and enable the team members to continually learn and stay abreast of the latest in Agile delivery methods.

The program should also establish a transition strategy for moving from one contractor to another and/or to Government resources and require the contractor (either as a basic contract requirement or a contract option) to deliver artifacts (e.g., technical data, source code) and services necessary to execute the transition strategy.

Sample work statement language for requesting specific roles and defining responsibilities (tailor as needed):

\phantom{

\begin{tabular}{|l|c|}
\hline
Metric                                      & Used to Answer                                      \\
\hline
Total work elements committed to sprint     & Projected cost of work scheduled                    \\
Total hours planned per story in sprint     & Projected cost per story                            \\
Actual hours logged against each story in sprint | Actual cost of work performed                      \\
Actual work elements completed in sprint    & Cost of work completed for potential delivery       \\
\hline
\end{tabular}

\textsuperscript{45} Source: AiDA https://aida.mitre.org
"The Contractor shall provide the number of teams and the team composition with the requisite skillset(s) to perform the work described in this task. The maximum number of total team members for this task is x. The Contractor teams shall supply x hours of effort (x-members x x-hours) each x-month period of performance. Each team member shall provide a minimum of two (2) core skillsets; for example, one team member could be both a developer and a tester. There is no maximum number of skillsets that a team member may possess."

The following items must be addressed:

1) The Contractor shall clearly identify the transition strategy (if applicable) for this task. (The incumbent’s proposal will not require a transition strategy.)

2) A key attribute of Agile teams is that each team member is fully dedicated to a single team. As a result, the [AGENCY NAME] expects that Agile teams will be staffed by individuals assigned to a single team.

3) The Contractor shall identify the total number of teams proposed. No fewer than ([x]) and no more than ([x]) teams may be proposed. Regardless of the number of teams proposed, the required core skillsets must be properly addressed. (See: Required Core Skillsets per Team/Task.)

4) The Contractor shall ensure that the teams’ skillsets continue to evolve to keep pace with technology.

5) The Contractor shall describe the approach to ensuring the quality of the teams’ work products and shall provide metrics/measures to track each team’s quality and productivity.

6) The Contractor shall explain the strategy for filling vacancies on a team and ensuring that capacity remains constant while vacancies are being filled. Please be mindful of the impact of vacancies when proposing smaller team sizes.

8) The Contractor shall complete the table below describing the staff required to support this task.

Required Core Skillsets per Team

Each team must include: (x) Software Testers / Quality Assurance Specialists and (x) Software Developers. Note: If a team member is also Key Personnel, the team member’s proposed skillsets for Task 2 must be clearly identified in his/her resume.

Required Core Skillsets per Task:

- User Experience (UX)
• Help Desk / Customer Support
• Technical Writer – including user manual/guide development
• Senior Software Developer
• Business Analyst
• Senior Software Tester

Note: A team member cannot count the same core skillset twice. For example, the same team member cannot be both a “Software Developer” and a “Senior Software Developer.”

The Contractor shall provide a separate table for each team being proposed. Remember that no fewer than \(x\) and no more than \(x\) teams, with a total of \(x\) individuals per team, may be proposed.

<table>
<thead>
<tr>
<th>Team #</th>
<th>Team Member</th>
<th>Core Skillset 1</th>
<th>Core Skillset 2</th>
<th>Additional Skillset(s)</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td></td>
<td>Ex.: Software Developer</td>
<td>Ex.: Software Tester / Quality Assurance Specialist</td>
<td></td>
</tr>
<tr>
<td>2</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>(n)</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

### 8.8.1 Agile Roles and Responsibilities

All parties must understand who will fulfill key roles (contractor or Government).

- It is recommended that the Government own the definition and prioritization of requirements and actively participate in the continuous planning effort. As such, the expectation is that a business representative (e.g., Product Owner in Agile Scrum) is a full-time Government employee and an integral team member in Agile planning, review, and acceptance events.
- The Government should ensure that the Design Lead is a full-time, organic role. Ultimate design strategy and decisions must be owned by the Government. The Government should allow the team to provide design recommendations in order to foster innovation and new ideas, but accountability must rest with a Government role to ensure that design supports the desired business intent and adheres to agency and Government guidance.
8.8.1.1 Product Owner (PO) / Business Representative

The Product Owner / Business Representative, an organic Government role, maintains responsibility for delivering business value. This means the Product Owner must maintain accountability over the Product Backlog and must work with the team and support organizations to define the scope/goal, priority of work, and acceptance criteria within each sprint. During the sprint, the Product Owner must be available to address issues related to the Backlog. The Product Owner must also accept or reject the work at the end of each sprint by deciding whether the delivered product meets the acceptance criteria and provides the desired business value. The Product Owner will ensure that the appropriate user representation is available for demonstrations at the end of each sprint. Depending on the specific Agile framework, the Product Owner might also assign a business value to each user story before and after the sprint.

Sample work statement language describing the Product Owner role (tailor as needed):46

"The Product Owner is the project’s key stakeholder and is a staff member of the Government Program Office. The Product Owner shall be responsible for establishing a vision of what the product team wishes to build and conveying the vision to the Contractor’s scrum or development team. The Product Owner’s responsibilities include, but are not limited to:

- **Creating and Managing the Product Backlog**: The Product Owner is responsible for establishing and maintaining the Product backlog with a complete, prioritized list of features containing descriptions of the desired functionality of the product.

- **Providing Clarification on Requirements during Agile Ceremonies**: Working with the Scrum Master, end-users, and other subject matter experts, the Product Owner shall hold recurring backlog refinement sessions to continuously refine and prioritize requirements and decision making during sprint planning and implementation. The Product Owner shall ensure the sprint backlog reflects the highest priority work to be delivered.

- **Maintaining Accountability for Minimum Viable Product**: The Product Owner shall work with the team to define the minimum viable product (MVP) that will serve as the first product release.

- **Developing, Maintaining, and Assessing Progress Against the Roadmap**: The Product Owner shall maintain accountability for the Product Roadmap that will identify the capabilities to be delivered by sprint or release. The Product Owner shall review progress after each sprint to assess whether the project is on pace for delivery according to the product roadmap and work with the team to adjust work and schedule as needed.

- **Attending Sprint Ceremonies**: The Product Owner shall attend all sprint ceremonies, including the daily Scrums/Stand-ups and the end-of-sprint demonstrations. The Product Owner shall ensure that the appropriate users are

46 Some of this pulled from [https://tech.gsa.gov/guides/Agile_Contracts_PWS_Template/](https://tech.gsa.gov/guides/Agile_Contracts_PWS_Template/)
available for the demonstrations to capture appropriate and timely user feedback.

- **Holding a User Acceptance Review**: The Product Owner shall work with the COR / ACOR to inspect and accept / approve Contractor work delivered by each sprint.

The Product Owner role will be filled by the Government, as the Product Owner is a representative of the needs of the Government user.

### 8.8.1.2 Technical Lead

The Technical Lead serves as an enabler in terms of ensuring that the technical architecture is planned and executed to keep pace with the development activities.

*Sample work statement language describing the Technical Lead (tailor as needed).*

“The Technical Lead is responsible for coordination among the Contractor’s development team, the Product Owner, and the key technical stakeholders to ensure that the technical architecture needs are planned and accounted for in support of the new business features/capabilities. Specific activities include:

- Participate in planning, definition, and high-level design of the solution and explore solution alternatives
- Enable the Continuous Delivery Pipeline through appropriate design guidelines
- Actively participate in the Continuous Exploration process as part of the Continuous Delivery Pipeline, especially in identification, planning, and execution of technical work
- Define subsystems and their interfaces, allocate responsibilities to subsystems, understand solution deployment, and communicate requirements for interactions with solution context
- Work with customers, stakeholders, and suppliers to establish high-level solution intent and the solution intent information models and documentation requirements
- Establish critical non-functional requirements
- Support technology/engineering aspects of Program planning and execution
- Provide oversight and foster Built-In Quality and Team and Technical Agility.”

### 8.8.1.3 Scrum Master

The contractor can fill the role of the Scrum Master (for Agile frameworks that incorporate Scrum). The Scrum Master serves as the facilitator for Agile Scrum ceremonies. The Scrum Master also facilitates the daily Scrum meetings and works with the team to remove or

---

47 Pulled from [https://tech.gsa.gov/guides/Agile_Contracts_PWS_Template/](https://tech.gsa.gov/guides/Agile_Contracts_PWS_Template/)
address any impediments or blockers. In larger, scaled Agile implementations, the Scrum Master also serves as a technical connection point between the other teams. Generally, in a scaled Agile implementation involving multiple Agile teams, a Scrum-of-Scrums meeting is held periodically to allow the teams to discuss interdependencies. These meetings are attended by the Scrum Masters of each team at minimum.

Sample work statement language describing the Scrum Master (tailor as needed):

“The Contractor shall provide a Scrum Master responsible for facilitating Agile ceremonies and discussions among development and customer team members. [If expecting multiple teams, include “There will be one Scrum Master for every Agile Team.”] The Scrum Master is responsible for coaching the team on the Scrum processes and removes impediments for the team. Scrum Master role activities include, but are not limited to:

- **Facilitating Scrum Meetings**: The Scrum Master coordinates and facilitates Scrum meetings, such as Sprint planning, Backlog Refinement, Sprint Reviews, and Sprint Retrospectives, and provides sufficient records.
- **Defining Team Performance Metrics (if metrics not defined)**: The Scrum Master works with product leadership and the team to define and deliver relevant metrics that are formulated and utilized for meeting project objectives.
- **Delivering Team Performance Metrics (if metrics defined)**: The Scrum Master works with product leadership and the team to deliver relevant metrics that are formulated and utilized for meeting project objectives (See section [x] for metrics to be reported).
- **Planning and Coordinating Team Training**: When appropriate, the Scrum Master shall plan and coordinate training for the team on the Scrum processes. The Scrum Master also helps to remove team roadblocks.”

8.8.1.4 Contractor Project Lead

The Contractor Project Lead is responsible for coordinating and delivering the work requested per the SOO/PWS/SOW.

Sample work statement language describing the Project Lead – assuming use of Agile Scrum methodology (tailor as needed).\(^49\)

---

\(^49\) Source: [https://tech.gsa.gov/guides/Agile_Contracts_PWS_Template/](https://tech.gsa.gov/guides/Agile_Contracts_PWS_Template/)
“The Contractor shall provide a Project Lead as the primary point of contact for coordination among the Contractor, the Product Owner, and the Contracting Officer and COR / ACOR. The Project Lead’s responsibilities include:

- Coordinating with the contracting organization’s program office to enable timely problem resolution
- Properly aligning staffing requirements
- Facilitating product reporting in line with Agile delivery methods. Per Agile Scrum, the Contractor’s Project Lead will be expected to work with the Product Owner and Scrum Master to develop Product / Sprint plans and reporting in line with Agile Scrum delivery approaches.”

8.8.1.5 Agile Coach

The Agile Coach helps program management staff and team to implement and manage Agile practices. The Agile Coach should have extensive experience and expertise with Agile, and with the selected Agile framework and its specific methodologies. Additionally, the Agile Coach should have experience with organizational change management in terms of assisting in transitioning organizations to Agile. This includes empowering leadership with the information they need to support and encourage Agile practices.

Sample work statement language describing the Agile Coach (tailor as needed):

The Contractor shall provide an Agile Coach. The following skills and experience are required for a person to be considered for this position:

- A minimum of five years’ experience as an [Agile Framework] Coach with demonstrated success in this role at multiple organizations

- A leader in the Agile community who has helped numerous organizations make a successful transition to Agile

- Demonstrated ability to work collaboratively with high-level managers and successfully guide and implement the change to Agile in organizations that have little previous experience with Agile software development

- In-depth understanding of organizational dynamics and demonstrated ability to achieve the culture change needed when successfully transitioning to Agile

- Breadth and depth of experience in the tools and techniques of the selected Agile framework (e.g., Scrum and XP) including a proven track record of continuous improvement in knowledge and implementation of Agile
· Experience guiding program managers using Agile in a large program that includes multiple Agile teams, releases, and projects that operate in parallel (e.g., Scrum of Scrums)

· Knowledge and understanding of how to implement architecture in an Agile environment, how to combine the use of component and feature teams, and how to coordinate multiple teams when managing Agile software development for large-scale enterprise development

· Understanding of the constraints, and demonstrated ability to apply means and methods to overcome these constraints when implementing Agile in a federal contracting environment.

The Agile Coach will be responsible for coaching the Agile Teams on the framework; attending Agile ceremonies; supporting the Scrum Master(s) as needed; reporting on Agile process performance; and working with the program leadership to improve Agile adoption and performance.

Consider a coach or process expert to enable successful Agile execution practices. Program Managers should consider whether the Government teams require dedicated, full-time Agile coaches, especially when they are first restructuring to transition to Agile.

Sample work statement language for Agile coaching needs (tailor as needed):

"The Contractor shall provide Agile coaches with the necessary qualifications and experience related to the recommended/selected Agile framework. The Coach will support the adoption of Agile principles and practices, and work with the Program/Project Manager, Product Owner(s), Scrum Master(s), and others as needed to address issues related to Agile process implementation or performance."

8.8.1.6 DevSecOps Engineer

The DevSecOps Engineer is accountable for the continuous integration and continuous delivery (CI/CD) of products. This can entail multiple specialties in the areas of software development, configuration control, testing, security, automation, containerization, orchestration, cloud services, and even open source technologies. There is also a large, and growing, collection of tools that support CI/CD activities. The Program Manger/Requirement Owner and technical subject matter experts must work with the Contracting Officers and Specialists to determine what specialties and tooset experience the DevSecOps Engineer must possess if this position is to be filled by a contractor. Ideally, this position should be filled by the Government to maintain in-house knowledge of the DevSecOps architecture.
Sample work statement language describing the DevSecOps Engineer (tailor as needed):

“The DevSecOps Engineer is accountable for overseeing the continuous integration and delivery (CI/CD) of the working product, which entails the following responsibilities:

- Working with the Development, Engineering, Architecture, and Testing teams to define the set of tools and processes for the continuous delivery pipeline
- Working with the Security Lead to ensure that the tools are compliant with security requirements
- Working with the Security Lead to streamline the ATO process from a tools and technical process perspective
- Working with the Testing Lead to automate testing where possible
- Working with the Operations Lead to ensure that the CI/CD processes align with operational requirements.

The DevSecOps Engineer must also have hands-on experience working with tools and technologies in, at a minimum, two of the following areas: software development, configuration control, testing, security, automation, containerization, orchestration, cloud services, open source technologies.”

8.8.1.7 Site Reliability Engineer

The Site Reliability Engineer (SRE) is accountable for ensuring the continual uptime and performance of the system in conformance with an agreed-upon service level agreement (SLA). The SRE is also a developer. The SRE and the operations teams are usually incentivized to maintain system performance at or above the minimum SLA levels.

Sample work statement language describing the Site Reliability Engineer (tailor as needed):

“The Site Reliability Engineer (SRE) is accountable for maintaining the uptime and performance of the solution in accordance with the agreed-upon service level agreement (SLA) by:

- Providing deep technical troubleshooting for escalated issues that involve the most technically complex or large-scale components and the affected users.
- Developing and/or using troubleshooting, monitoring, and reporting tools to analyze the root cause of serious and impactful technical issues and building stable and sustainable solutions and improvements.
- Working closely with customer care, escalation teams, product management, and engineering to seek solutions for customers and drive tasks toward completion.
- Driving and improving the whole lifecycle of operational readiness – from inception and design through deployment, operation, and refinement.
- Developing tools, operational improvements, and automated solutions that enable self-service configuration changes, speed deployments, and improve
monitoring in support of mission-critical customer-facing applications and environments

- Assisting the software engineering team to ensure accurate monitoring and metrics are built into the applications before deployment to production
- Participating in an on-call rotation
- The SRE must also have hands-on experience working with tools and technologies in, at minimum, two of the following areas: software development, configuration control, testing, security, automation, containerization, orchestration, cloud services, open source technologies.”

8.9 DevSecOps

“DevSecOps is an organizational software engineering culture and practice that aims at unifying software development (Dev), security (Sec) and operations (Ops). The main characteristic of DevSecOps is to automate, monitor, and apply security at all phases of the software lifecycle: plan, develop, build, test, release, deliver, deploy, operate, and monitor.” DevSecOps principles highlight the need to include security requirements upfront to successfully achieve continuous integration and continuous delivery (CI/CD pipeline), continuous monitoring and instrumentation/measurement of working software to the customer.

CI describes the concept of integrating smaller pieces of code and functionality more frequently to allow teams to integrate and validate changes. As a result, the team benefits from a consistent, repeatable, and automated way to build, package, and test code. CD is the next step in terms of delivering the application in a consistent, repeatable, and automated manner to specific environments and includes continuous monitoring.

CIO.gov defines continuous monitoring as “... a risk management approach to cybersecurity that maintains an accurate picture of an agency’s security risk posture, provides visibility into assets, and leverages use of automated data feeds to quantify risk, ensure effectiveness of security controls, and implement prioritized remedies.” A well-designed continuous risk management process will transform risk assessment and monitoring from a static activity to a continual monitoring activity that is enabled by automated processes and tools.

The DoD Enterprise DevSecOps Reference Design further elaborates that, ‘Continuous monitoring is an extension to continuous operation. It continuously inventories all system components, monitors the performance and security of all components, and logs application and system events.”

51 Source: CIO.gov https://www.cio.gov/agenda/cybersecurity/continuous-monitoring/
Instrumentation (or measurement) of DevSecOps elements provides visibility and insight into bottlenecks in the pipeline or issues related to quality, security, or operation of the product.

A successful DevSecOps strategy encompasses working with all stakeholders involved in planning, designing, developing, testing, securing, integrating, deploying, continually monitoring, and improving the product. The goal of DevSecOps is to ensure that the team works together to remove bottlenecks related to product planning, development, testing, security, and operations by reducing handoffs in favor of close coordination as early and as often as possible. The DevSecOps lead is accountable for defining a technology stack to enable the necessary capabilities to support DevSecOps practices. Training staff members and developing a culture of continuous learning is also necessary for improvement, as the technology continuously changes.

The concept of “shifting left” is important in DevSecOps. It involves engaging testing, security, and operations teams to actively participate as early as possible. Preferably, one representative from each function should be part of the Agile team to remove siloed “handoffs” and create group ownership and accountability of the delivery pipeline.

If the program is considering including DevSecOps practices within the scope of work, then the program should consider the contractor’s role in integrating development, testing, security, and operations practices within the overall DevSecOps strategy. The contractor should have experience standing up a software delivery pipeline, which is a collection of tools and process that enable the CI/CD of capability to the user. Depending on the program’s familiarity with respect to DevSecOps, the contractor could be asked to help implement a pipeline, or work within an existing pipeline.

Modern software development practices include use of open source tools to help improve the speed and quality of software development. Many of the tools allow development teams to take advantage of already developed, commonly used code and technical features that abstract out some of the low-level and repetitive work. This allows the development team to focus on developing high-value capabilities. Many open source tools align with DevSecOps practices; however, due to agency and/or Government policies, the program should request an itemized list of open source tools the contractor intends to use that can be reviewed by the Government.

Program Managers should consider the questions below when shaping their DevSecOps requirements. The Contracting Officer and Specialist can expect to receive the following information from the program to support the Agile solicitation and resulting contract:

- What is the DevSecOps strategy?
- What intellectual property (IP) constraints must be considered?
- What cloud-based costs should be factored in?
- What is the commitment of the testing, security and operations teams to “shift left”? Must certain testing- and security-related activities take place prior to final approval that cannot “shift left” or be integrated into the CD pipeline?
● Are there hardware considerations that could impact the program’s end-to-end delivery pipeline (i.e., weapons systems).

For additional detailed technical questions related to DevSecOps, the DoD CIO has developed a checklist of questions (accessible on http://software.af.mil/), which are designed to ensure that the program has a clear understanding of the Program’s DevSecOps approach.

Sample work statement language describing the DevSecOps strategy (tailor as needed):

“The Contractor shall propose a clear DevSecOps strategy and explanation of the continuous delivery pipeline. After contract award, the Contractor shall work with [Program] to increase continuous integration and continuous delivery capability by progressively layering more sophisticated automation components into the continuous integration/continuous delivery pipeline while also reducing the time to move approved code from development to production.

In support of this effort, the Contractor will bring to bear industry and Government practices for elements of DevSecOps such as code repository management, configuration management, branching/merging code, building automation, test code coverage, and automated scanning tools. The Contractor shall also look for practices to streamline and integrate security practices within the continuous integration/continuous delivery pipeline [optional: that is aligned with the DoD Enterprise DevSecOps initiative].

The DevSecOps support required includes:
1) Installing and configuring application lifecycle management, continuous integration/continuous delivery, and continuous monitoring tools.
2) Designing, implementing, and maintaining the DevSecOps continuous integration/continuous delivery pipeline for cloud and on-premises deployments.
3) Recommending and implementing industry-based practices for continuous monitoring.
3) Training and coaching development teams in utilizing the application lifecycle management and continuous integration/continuous delivery tools.
4) Establishing and maintaining standards for use of the application lifecycle management and continuous integration/continuous delivery tools.
5) Recommending industry-based practices for incorporating testing, security, and operations representation into the Agile Team early and continuously in the application lifecycle.
6) Recommending industry-based approaches and tools for instrumenting DevSecOps elements to provide greater visibility, transparency, and awareness related to pipeline bottlenecks, security, and operational issues.”

For programs interested in following the DevSecOps guidance provided in the DoD Enterprise DevSecOps Reference Design, the following sample language is a starting point for capturing the desired DevSecOps capabilities as recommended within version 1.0 of the
Reference Design. The DoD Enterprise DevSecOps Reference Design provides implementation and operational guidance to solution capability providers, consumers, applications teams and authorizing officials. Specifically, the Reference Design describes the DevSecOps lifecycle, supporting pillars, and DevSecOps ecosystem; lists the tools and activities for DevSecOps software factory and ecosystem; introduces the DoD enterprise DevSecOps container service that provides hardened DevSecOps tools and deployment templates to the program application DevSecOps teams to select; and showcases a sampling of software factory reference designs and application security operations.

Sample work statement language describing the desired capabilities in accordance with the DoD Enterprise DevSecOps Reference Design – for a new (greenfield) implementation (Tailor as needed):

The Contractor shall propose a clear DevSecOps strategy and explanation of the continuous delivery pipeline that aligns with the guidance provided in the DoD Enterprise DevSecOps Reference Design published by the DoD Chief Information Officer. After contract award, the Contractor will work with [Program] to increase continuous integration and continuous delivery capability by progressively layering more sophisticated automation components into the continuous integration/continuous delivery pipeline while also reducing the time to move approved code from development to production.

In support of this effort, the Contractor will bring to bear industry and Government practices for elements of DevSecOps such as code repository management, configuration management, branching/merging code, building automation, test code coverage, and automated scanning tools. The Contractor will look for practices to streamline and integrate security practices within the continuous integration/continuous delivery pipeline [optional: that is aligned with the DoD Enterprise DevSecOps initiative].

The DevSecOps support required includes:
1) Installation and configuration of application lifecycle management, continuous integration/continuous delivery, and continuous monitoring tools.
2) Design, implementation, and maintenance of the DevSecOps continuous integration/continuous delivery pipeline for cloud and on-premises deployments.
3) Recommending and implementing continuous monitoring industry-based practices.
3) Training and coaching of development teams in utilizing the application lifecycle management and continuous integration/continuous delivery tools.
4) Establishment and maintenance of standards for use of the application lifecycle management and continuous integration/continuous delivery tools.
5) Recommending industry-based practices for incorporating testing, security, and operations representation into the Agile Team early and continuously in the application lifecycle.
6) Recommending industry-based approaches and tools for instrumenting DevSecOps elements to provide greater visibility, transparency and awareness related to pipeline bottlenecks, security and operational issues.
Per the DoD Enterprise DevSecOps Reference Design, the following capabilities must be addressed when standing up a software factory:

- **Use of Open Container Initiative (OCI) compliant and Cloud Native Computing Foundation (CNCF) certified Kubernetes platform to enable interoperability from one Kubernetes installation to the next and allow flexibility to choose between vendors if needed**
- **Use of hardened containers to maintain security**
- **Use of the DoD Centralized Artifact Repository (DCAR) to hold the hardened VM images and hardened OCI compliant container images of DevSecOps tools, container security tools and common programming platform components (e.g., COTS or open source products) that the DoD program software teams can utilize as a baseline to facilitate the authorization process.**
- **Use of a Sidecar Container Security Stack (SCSS) - A new service that is enabled by DevSecOps and the container-based Kubernetes runtime environment is the Sidecar Container Security Stack (SCSS). This security stack enables: correlated and centralized logs, container security, east/west traffic management, a zero-trust model, a whitelist, Role-Based Access Control (RBAC), continuous monitoring, signature-based continuous scanning using Common Vulnerabilities and Exposures (CVEs), runtime behavior analysis, and container policy enforcement. In addition to the components in the sidecar, there are a few services that support the security sidecar. These include:**
  1. Program-specific Log Storage and Retrieval Service
  2. Service Mesh
  3. Program-specific artifact repository
  4. Runtime Behavior Analysis Artificial Intelligence (AI) service
  5. DCAR for the hardened containers
  6. Common Vulnerabilities and Exposures (CVE) Service / host-based security to provide CVEs for the security sidecar container
- **Vendors will be required to work within a government furnished DevSecOps environment in order to get full source code access, and in order to be able to perform full functional and security tests**
- **Use of one CI/CD orchestrator per software factory**
- **Support for automation**
  - Development teams developing application code and Infrastructure-as-Code (IaC) to improve quality of integrations, releases and deployments.
  - Security teams strive to automate security by implementing security compliance checking and auditing as code (SaC)
  - Both IaC and SaC are treated as software and go through the rigorous software development and processes including design, development, version control, peer review, static analysis and test
8.10 Training and Resources

Training everyone involved (both Government and contractor staff) is critical to the success and adoption of Agile. Training helps reduce common problems, where teams practice what they believe to be Agile but is not Agile. After initial Agile training is received, Agile teams also need a plan for sustained learning and coaching to ensure proper execution and to be able to practically apply the newly learned skills. Team members should have more than one skillset and training should support this.

Ensure that the contractor has experience in the Agile methodology. The Government should have the ability to review team member skillsets and establish minimum standards through the solicitation and resulting contract. The PWS should communicate that notification is required in advance if resource commitment to the effort changes (full time/part time/transitioning off). The program may want to include that the contractor must identify the number of years/types of Agile projects each resource worked on and provide resumes.

Sample language from https://tech.gsa.gov/guides/Agile_Contracts_PWS_Template/ (tailor as needed):

"The Contractor shall provide qualified personnel with relevant experience and domain knowledge in line with this task's performance work statement, in terms of necessary skills at the requisite level of knowledge and experience. Broadly, a team assigned to build the {product name} is expected to have experience with: {List relevant domain here} such as building and testing web-based or mobile applications, user-centered design practices, usability testing, user experience design, visual design, specific code languages, cloud deployment, open-source login / authentication services, Agile and scrum methodologies, etc."

Ensure that development team members have more than one skillset (e.g., development and testing). Having the ability to shift to an activity that is lagging gives the team flexibility to move to a task that requires more support. This reduces delays overall, especially if there are dependencies on the lagging activity.
**Sample work statement language for multiple skillset needs (tailor as needed):**

"The Contractor shall ensure that development team members have more than one of the following skillsets as indicated by their resumes and experience: [list skillsets here: e.g., software development, web development, testing, automated testing, etc.].”

### 8.11 Tools and Technology

The work statement should provide a picture of the technical landscape and the expectations pertaining to the enabling tools, as listed below, to ensure contractor staff is qualified to support the tools as needed. If requesting the contractor to provide guidance on tools, the statement should account for necessary tools and technology to support Agile project management and requirements management.

- Is there an expectation that the development team uses Government-provided tool(s) or is the contractor expected to bring its own tools? License requirements for use must be evaluated.
- The contractor should account for necessary tools to support DevSecOps practices (e.g., CI/CD technology stack).
- The contractor should identify all open source tools, or desire for open source tools.
- If there are any custom tools, the contractor should consider possible IP issues.

**Sample work statement language for communicating tool usage expectations (tailor as needed):**

[Program’s] technical landscape includes proprietary Commercial Off-the-Shelf (COTS)-based frameworks; [optional: if willing to review open source solutions: however, a shift to include open source frameworks in the future is anticipated]. [Program] expects that the current development and test architecture will evolve. For Application Lifecycle Management (ALM), [Program] currently uses [provide list of tools here]].

### 8.12 Intellectual Property

Per DoD Instruction 5010.44 – Intellectual Property (IP) Acquisition and Licensing, acquiring and licensing IP is vital for maintaining functionality, sustainability, ongoing improvement, and affordability of weapon and information systems. The Government should acquire IP while also considering the rights and fair treatment of IP owners. [1]

The DoD should also consider the following when shaping the IP strategy:

- DoD should only pay for the IP that is needed.
- DoD should only pay once for the same IP.
- Consider differed ordering/contract options to acquire IP at a later date.
- Consider having the bidders propose an IP strategy as part of source selection evaluation.

Sample work statement language (and consideration items for developing sample language) to address IP (tailor as needed):

[Due to the unique nature of every solution, the IP strategy will vary. However, consider these items when developing the language:

- What will be required in terms of access to code, data, licensing, and knowledge to affordably operate, maintain, modernize, and sustain the system over the life of the solution?
- Will there be a need for specially negotiated licenses to acquire customized IP deliverables?
- Is it clear how the solution will be delivered and whether the source code and data will accessible by the Government?
- Does the IP strategy consider what would happen to the data rights/access in the event of early termination of the contract by either party? A lack of clear guidance could put the Government at risk of loss of access to vital data required to maintain the solution moving forward.
- What specific tools will be required to provide authorized access, retention, integration, sharing, transferring, and conversion of IP deliverables?]

[Additional sample language for implementing system interfaces:]

"The [solution integrator] will develop all major systems interfaces to be based on non-proprietary, widely supported, and consensus-based standards (if available and suitable).

The [solution integrator] will provide authorized access, retention, integration, sharing, transfer, and conversion of IP deliverables. The [solution integrator] will [ensure that processes and tools are in place / use the existing tools in place] to support product configuration management, data loss prevention, and data sharing and exchange."

Key Takeaways: This section provides sample work statement language for Agile development contracts. The language can and should be tailored to fit the needs of a program based on the Agile approach, resource needs, level of Agile knowledge, and other factors.
Appendix A  **Traditional vs. Agile Contract Strategies**

Contract awards for traditional development programs are based on evaluation of proposed technical solutions to meet detailed Government requirements, as they are understood at the time the requirements are developed, with the least amount of risk to the Government. Contracts are designed to lock in details and schedules to align with programmatic milestones and events. Contractors are incentivized to deliver to schedule and effectively manage agreed-to costs with as little deviation as possible.

Agile software development calls for awarding contracts that are based on a contractor’s proven ability to develop and deliver software and other capabilities using Agile methodologies. The contract should enable the Government to adjust and reprioritize work throughout the project lifecycle. Creating an effective contract is a challenging, but essential, element in attaining the benefits of Agile software development practices. Traditional long contracting timelines that impose costly change requests present major hurdles to the desired small, frequent releases encouraged by Agile. Contracting strategies for Agile software development must be designed to support the short development and delivery cycle timelines. Table 5 presents a comparison of traditional and Agile contract elements.

<table>
<thead>
<tr>
<th>Contracts for Traditional Software Development</th>
<th>Element</th>
<th>Contracts for Agile Software Development</th>
</tr>
</thead>
<tbody>
<tr>
<td>Rigid processes with long timelines not designed for constant change</td>
<td>Contract Timelines</td>
<td>Streamlined contracting processes to adjust to continuous reprioritization</td>
</tr>
<tr>
<td>Completion contracts requiring upfront definition of requirements</td>
<td>Contract Type</td>
<td>Not limited to, but often, Level-of-effort/term-type contracts</td>
</tr>
<tr>
<td>Defined in detail up front</td>
<td>Requirements</td>
<td>High-level objectives defined upfront to. Ability to reprioritize requirements throughout project lifecycle. Detailed requirements identified ahead of each development cycle.</td>
</tr>
<tr>
<td>Contract management when modifications are necessary or contractual issues arise</td>
<td>Contract Management</td>
<td>Continuous management of Government-contractor relationships and continuous Government participation in Agile team throughout lifecycle</td>
</tr>
<tr>
<td>Contractor incentivized to deliver against fixed requirements</td>
<td>Incentives</td>
<td>Contractor incentivized to be efficient and collaborative</td>
</tr>
<tr>
<td>Award made based on the strength of the technical solution</td>
<td>Technical Evaluation</td>
<td>Award based on the strength of the team and experience with Agile</td>
</tr>
<tr>
<td>The requirements and design</td>
<td>Confidence Rests on...</td>
<td>The team’s qualifications and ability to work together to deliver capability</td>
</tr>
</tbody>
</table>

---

53 Incentives are an important part of contracting, especially when programs are transforming to use Agile approaches for the first time and should be considered. Work on guidance for incentives is ongoing.
<table>
<thead>
<tr>
<th>Contracts for Traditional Software Development</th>
<th>Element</th>
<th>Contracts for Agile Software Development</th>
</tr>
</thead>
<tbody>
<tr>
<td>The requirements according to specifications</td>
<td>Focus is on delivering...</td>
<td>Value based on continuous feedback and continuous test</td>
</tr>
<tr>
<td>Maybe during testing... after development activities are complete</td>
<td>When is value received?</td>
<td>Iteratively, during sprint and release cycles</td>
</tr>
<tr>
<td>Milestone events and end of period of performance</td>
<td>Delivery</td>
<td>Continuous in accordance with agreed to backlog and sprint/release cycles</td>
</tr>
</tbody>
</table>

Agile software development requires documentation for user stories, acceptance criteria, tasks to be completed to meet the definition of done, code quality, and standards compliance. The Agile software development approach involves a highly disciplined process with Government representation on the team to set priorities and ensure that working software is compliant with contract or system requirements. As an active participant on the Agile team, the Government can closely monitor development progress and quickly discover whether a contractor is off track. The Government monitors progress by the timely delivery of deployable code in accordance with acceptance criteria, definition of done, code quality, and standards compliance.

Incentives are an important consideration when contracting for Agile delivery. Under an Agile software development effort, the team should focus on speedy and efficient contracting processes to support short delivery cycles. Thus, the use of complicated contract incentive structures (e.g., incentive fees, award fees) is not recommended when using a services-type contract for Agile software development. Not only are these types of incentives time-consuming and resource-intensive to administer, but under a service contract the Government drives the process, schedule, and requirements. As a result, creating an incentive for “early delivery” or “cost savings” may not be achievable because the contractor is not able to control these outcomes. Despite the noted concerns, the program can still consider using a performance-based contract when contracting for services as long as the program considers the independence of the metrics and whether Government requirements could potentially be a potential factor in whether the metric is achieved or not.

Contract incentives are often used when the Government does not have the capacity or control to monitor contract performance. However, on Agile contracts, the interactions between the Government and the contractor are significantly greater. Therefore, the Government can manage contractor performance as a result of the inherent built-in collaboration and communication cadence built into the Agile software development process. The frequent collaboration reduces the need for incentive contracts, which can be burdensome and potentially confrontational.
Appendix B  Sample Contract Templates for Agile Delivery

The references shown in Table 6 offer two approaches for contracting for Agile development teams as a Service. Neither template includes language for acquiring platform, infrastructure, or DevSecOps solutions. As described in Section 7.1, a modular contracting approach can be used to separately procure such elements using other contracting strategies. These examples illustrate approaches that are currently being used to acquire Agile development services and are not intended to represent the only methods for acquiring Agile development services. Each program is unique, and information offered in the templates should be tailored to address specific program needs and objectives.

Table 6 – Sources of Contract Templates

<table>
<thead>
<tr>
<th>Source</th>
<th>Description and Link</th>
</tr>
</thead>
</table>
| 18F    | An 18F template for acquiring development services using a time and materials (T&M) contract type with a not-to-exceed ceiling. 18F offers a sample work statement structure, deliverables tied to a Quality Assurance Plan (QASP), and suggested language for instructions to offerors (Section L) and evaluation factors (Section M). 18F also provides links to additional resources, such as a sample RFP for Agile development services developed by the US Tax Courts leveraging the 18F T&M for Agile development services approach.  
| DAU    | A PWS template for Firm-Fixed-Price (FFP) work using a “Capacity Team Based” approach. The PWS clearly provides specific recommendations to consider for successful use of the template, which include:  
1) A Federal Representative(s)/Employee(s) (typically referred to as the “Product Owner”) with enough time to dedicate to the team(s)  
2) A continuous flow or backlog of requirements (stories)  
3) Future requirements that require the same skillsets (developer, tester, etc.) to complete.  
4) Managed processes (to include a substantial cadre of process engineers, Agile coaches, Scrum Masters, etc.)  
## Appendix C  Contracting for Agile References

Sources of information about Agile contracting.

<table>
<thead>
<tr>
<th>Organization</th>
<th>Reference Name</th>
<th>Reference Location and Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>18F</strong></td>
<td>Modular Contracting</td>
<td><a href="https://modularcontracting.18f.gov">https://modularcontracting.18f.gov</a> A modular contracting toolkit put out by 18F that covers</td>
</tr>
<tr>
<td><strong>18F</strong></td>
<td>18F Agile BPA Ordering Guide</td>
<td><a href="https://agile-bpa.18f.gov/ordering-guide/">https://agile-bpa.18f.gov/ordering-guide/</a></td>
</tr>
<tr>
<td><strong>DAU</strong></td>
<td>Agile Acquisition 101 Video</td>
<td><a href="https://media.dau.edu/media/Agile+Acquisition+101/1_qtvcltya">https://media.dau.edu/media/Agile+Acquisition+101/1_qtvcltya</a> A good, brief video that provides an overview of Agile software development, fielding, DevSecOps and Agile contracting.</td>
</tr>
<tr>
<td><strong>DAU</strong></td>
<td>Contracting for Change</td>
<td><a href="https://media.dau.edu/media/Contracting+for+Change+%28Agile%29%2F1_03dksk42">https://media.dau.edu/media/Contracting+for+Change+%28Agile%29%2F1_03dksk42</a> A brief video on Agile contracting</td>
</tr>
<tr>
<td><strong>DAU</strong></td>
<td>Contract Type Chart</td>
<td><a href="https://www.dau.edu/tools/t/Comparison-of-Major-Contract-Types-Chart">https://www.dau.edu/tools/t/Comparison-of-Major-Contract-Types-Chart</a> A comparison chart of contract types</td>
</tr>
<tr>
<td><strong>Developed by MITRE for OUSD(A&amp;S)</strong></td>
<td>Contracting Cone</td>
<td><a href="https://aaf.dau.edu/contracting-cone/">https://aaf.dau.edu/contracting-cone/</a> Specific FAR-based and Non-FAR based contracting approaches that can be used today for Agile contracting.</td>
</tr>
<tr>
<td><strong>Developed by MITRE for OUSD(A&amp;S)</strong></td>
<td>Adaptive Acquisition Framework</td>
<td><a href="https://aaf.dau.edu/aaf/">https://aaf.dau.edu/aaf/</a></td>
</tr>
<tr>
<td>Organization</td>
<td>Reference Name</td>
<td>Reference Location and Description</td>
</tr>
<tr>
<td>--------------</td>
<td>-------------------------</td>
<td>-------------------------------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>DoD</td>
<td>OTA Guide</td>
<td>&quot;This tool helps you to select the right pathway and provides you detailed guidance for each path&quot;</td>
</tr>
<tr>
<td>DoD CIO</td>
<td>DevSecOps Assessment Questions</td>
<td>URL TBD Questions designed to give an in-depth understanding of the Program’s approach to DevSecOps, as it relates to each topic area.</td>
</tr>
<tr>
<td>GSA</td>
<td>PWS Template PP</td>
<td><a href="https://tech.gsa.gov/guides/Agile_Contracts_PWS_Template/">https://tech.gsa.gov/guides/Agile_Contracts_PWS_Template/</a> Sample Performance Work Statement template for an Agile BPA. Perf</td>
</tr>
<tr>
<td>GSA</td>
<td>Agile Contracts - Task Order Template</td>
<td><a href="https://tech.gsa.gov/guides/Agile_Contracts_TaskOrder_Template/">https://tech.gsa.gov/guides/Agile_Contracts_TaskOrder_Template/</a> &quot;...provides contract language guidance and is intended to serve as a template for the Government representative during the development of a Task Order under an Agile BPA.&quot;</td>
</tr>
<tr>
<td>GSA</td>
<td>Agile Contracts Interview Questions for Agile Vendors</td>
<td><a href="https://tech.gsa.gov/guides/Agile_Contracts_Interview_Questions_for_Agile_Vendors/">https://tech.gsa.gov/guides/Agile_Contracts_Interview_Questions_for_Agile_Vendors/</a> &quot;...provides scenarios and questions that are intended to guide the Government representative during the interview and evaluation phase where a vendor representative has been shortlisted and is being considered for an Agile delivery RFI or RFP&quot;</td>
</tr>
<tr>
<td>GSA</td>
<td>GSA Fast Lane Agile BPAs</td>
<td><a href="https://www.gsa.gov/buying-selling/purchasing-programs/gsa-schedules/schedule-features/blanket-purchase-agreements">https://www.gsa.gov/buying-selling/purchasing-programs/gsa-schedules/schedule-features/blanket-purchase-agreements</a></td>
</tr>
<tr>
<td>GSA</td>
<td>FAR</td>
<td><a href="https://www.acquisition.gov/">https://www.acquisition.gov/</a> Federal Acquisition Regulation</td>
</tr>
<tr>
<td>MITRE</td>
<td>Accelerate Acc</td>
<td><a href="https://accelerate.mitre.org/contracting/">https://accelerate.mitre.org/contracting/</a></td>
</tr>
<tr>
<td>Organization</td>
<td>Reference Name</td>
<td>Reference Location and Description</td>
</tr>
<tr>
<td>---------------------------</td>
<td>---------------------------------------------</td>
<td>------------------------------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>MITRE</td>
<td>Acquisition in the Digital Age (AiDA) Ac</td>
<td><a href="https://aida.mitre.org/agile/agile-contract-preparation/">https://aida.mitre.org/agile/agile-contract-preparation/</a> Externally facing MITRE web site containing actionable guidance on Agile contracting, and references for further reading</td>
</tr>
<tr>
<td>MITRE</td>
<td>Adapting</td>
<td><a href="http://conferences.computer.org/stc/2014/papers/5034a007.pdf">http://conferences.computer.org/stc/2014/papers/5034a007.pdf</a> Adapting Agile Processes for Military Acquisition Programs</td>
</tr>
<tr>
<td>OMB</td>
<td>TechFAR Handbook</td>
<td>&quot;The TechFAR Handbook focuses on provisions of the FAR that are most relevant to digital services acquisitions and explains how agencies can align their applications of FAR authorities with contemporary development [5] approaches...&quot;</td>
</tr>
<tr>
<td>OSTP and OMB</td>
<td>Innovative Contracting Case Studies</td>
<td><a href="https://obamawhitehouse.archives.gov/blog/2014/08/21/buying-what-works-case-studies-innovative-contracting-0">https://obamawhitehouse.archives.gov/blog/2014/08/21/buying-what-works-case-studies-innovative-contracting-0</a> &quot;an iterative, evolving document that describes a number of ways Federal agencies are getting more innovation per taxpayer dollar – all under existing laws and regulations&quot;</td>
</tr>
<tr>
<td>SEI</td>
<td>Contracting for Agile SW Dev. in the DoD</td>
<td><a href="https://resources.sei.cmu.edu/library/asset-view.cfm?assetid=442499">https://resources.sei.cmu.edu/library/asset-view.cfm?assetid=442499</a> Contracting for Agile Software Development in the Department of Defense: An Introduction</td>
</tr>
<tr>
<td>SEI</td>
<td>Agile Software Teams: How They Engage with Systems Engineering on DoD Acquisition Programs</td>
<td><a href="https://resources.sei.cmu.edu/library/asset-view.cfm?assetid=2959433">https://resources.sei.cmu.edu/library/asset-view.cfm?assetid=2959433</a> Agile Software Teams: How They Engage with Systems Engineering on DoD Acquisition Programs</td>
</tr>
<tr>
<td>SEI</td>
<td>RFP Patterns and Techniques for Successful Agile Contracting</td>
<td><a href="https://resources.sei.cmu.edu/library/asset-view.cfm?assetid=484056">https://resources.sei.cmu.edu/library/asset-view.cfm?assetid=484056</a> SEI (Lapham) and National Defense Industrial Association (NDIA), Request for Proposal (RFP) Patterns and Techniques for Successful Agile Contracting</td>
</tr>
<tr>
<td>SEI</td>
<td>Contracting for Agile</td>
<td><a href="https://resources.sei.cmu.edu/library/asset-view.cfm?assetid=442499">https://resources.sei.cmu.edu/library/asset-view.cfm?assetid=442499</a></td>
</tr>
</tbody>
</table>
| The US Digital Service    | The Digital Services Playbook                | https://playbook.cio.gov/ "...a playbook of 13 key “plays” drawn from successful practices from the private sector and Government..." }
<table>
<thead>
<tr>
<th>Organization</th>
<th>Reference Name</th>
<th>Reference Location and Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>The US Digital Service</td>
<td>TechFAR Hub</td>
<td>The TechFAR Hub was created by the US Digital Service to provide a connection point between those who are doing digital service acquisition and those who want to. It is part of a greater collaborative effort with the Office of Federal Procurement Policy, Acquisition Innovation Labs, and GSA’s Acquisition Gateway</td>
</tr>
</tbody>
</table>
Appendix D  Bibliography


## Appendix A  Abbreviations and Acronyms

<table>
<thead>
<tr>
<th>Acronym</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>AiDA</td>
<td>Acquisition in the Digital Age (MITRE-hosted web site)</td>
</tr>
<tr>
<td>ALM</td>
<td>Application Life Cycle Management</td>
</tr>
<tr>
<td>ATO</td>
<td>Authority to Operate</td>
</tr>
<tr>
<td>BIC</td>
<td>Best in Class</td>
</tr>
<tr>
<td>CD</td>
<td>Continuous Delivery</td>
</tr>
<tr>
<td>CDR</td>
<td>Critical Design Review</td>
</tr>
<tr>
<td>CDRL</td>
<td>Contract Data Requirements List</td>
</tr>
<tr>
<td>CI</td>
<td>Continuous Integration</td>
</tr>
<tr>
<td>CLIN</td>
<td>Contract Line Item Number</td>
</tr>
<tr>
<td>CO</td>
<td>Contracting Officer</td>
</tr>
<tr>
<td>COR</td>
<td>Contracting Officer’s Representative</td>
</tr>
<tr>
<td>COTS</td>
<td>Commercial Off-the-Shelf</td>
</tr>
<tr>
<td>CPI</td>
<td>Continuous Process Improvement</td>
</tr>
<tr>
<td>CPMP</td>
<td>Contractor Project Management Plan</td>
</tr>
<tr>
<td>DAU</td>
<td>Defense Acquisition University</td>
</tr>
<tr>
<td>DFARS</td>
<td>Defense Federal Acquisition Regulation Supplement</td>
</tr>
<tr>
<td>DoD</td>
<td>Department of Defense</td>
</tr>
<tr>
<td>FAR</td>
<td>Federal Acquisition Regulation</td>
</tr>
<tr>
<td>FFP</td>
<td>Firm-Fixed-Price</td>
</tr>
<tr>
<td>FISMA</td>
<td>Federal Information Security Management Act</td>
</tr>
<tr>
<td>FITARA</td>
<td>Federal IT Acquisition Reform Act</td>
</tr>
<tr>
<td>GAO</td>
<td>Government Accountability Office</td>
</tr>
<tr>
<td>GSA</td>
<td>General Services Administration</td>
</tr>
<tr>
<td>GWAC</td>
<td>Governmentwide Acquisition Contract</td>
</tr>
<tr>
<td>IDIQ</td>
<td>Indefinite Delivery and Indefinite Quantity (contract strategy)</td>
</tr>
<tr>
<td>IMS</td>
<td>Integrated Master Schedule</td>
</tr>
<tr>
<td>IP</td>
<td>Intellectual Property</td>
</tr>
<tr>
<td>IR</td>
<td>Infrastructure Review</td>
</tr>
<tr>
<td>IRPR</td>
<td>Initial Release Planning Review</td>
</tr>
<tr>
<td>IRR</td>
<td>Integration Readiness Review</td>
</tr>
<tr>
<td>IT</td>
<td>Information Technology</td>
</tr>
<tr>
<td>Abbreviation</td>
<td>Full Form</td>
</tr>
<tr>
<td>--------------</td>
<td>-----------</td>
</tr>
<tr>
<td>LOB</td>
<td>Line of Business</td>
</tr>
<tr>
<td>LOE</td>
<td>Level of Effort</td>
</tr>
<tr>
<td>MAC</td>
<td>Multi-Agency Contract</td>
</tr>
<tr>
<td>MVP</td>
<td>Minimum Viable Product</td>
</tr>
<tr>
<td>NDIA</td>
<td>National Defense Industrial Association</td>
</tr>
<tr>
<td>OCIO</td>
<td>Office of the Chief Information Officer</td>
</tr>
<tr>
<td>OMB</td>
<td>Office of Management and Budget</td>
</tr>
<tr>
<td>OSD</td>
<td>Office of the Secretary of Defense</td>
</tr>
<tr>
<td>OT</td>
<td>Other Transaction</td>
</tr>
<tr>
<td>OTA</td>
<td>Other Transactions Authority</td>
</tr>
<tr>
<td>OTRR</td>
<td>Operational Test Readiness Review</td>
</tr>
<tr>
<td>OUSD</td>
<td>Office of the Undersecretary of Defense</td>
</tr>
<tr>
<td>PDR</td>
<td>Preliminary Design Review</td>
</tr>
<tr>
<td>PEO</td>
<td>Program Executive Office</td>
</tr>
<tr>
<td>PGI</td>
<td>Procedures, Guidance and Information</td>
</tr>
<tr>
<td>PM</td>
<td>Program/Project Manager</td>
</tr>
<tr>
<td>PO</td>
<td>Product Owner</td>
</tr>
<tr>
<td>PoP</td>
<td>Period of Performance</td>
</tr>
<tr>
<td>PWS</td>
<td>Performance Work Statement</td>
</tr>
<tr>
<td>RFP</td>
<td>Request for Proposal</td>
</tr>
<tr>
<td>RPR</td>
<td>Release Planning Review</td>
</tr>
<tr>
<td>SCA</td>
<td>Fortify Static Code Analysis (tool)</td>
</tr>
<tr>
<td>SEI</td>
<td>Systems Engineering Institute</td>
</tr>
<tr>
<td>SETR</td>
<td>Systems Engineering Technical Review</td>
</tr>
<tr>
<td>SFR</td>
<td>Systems Functional Review</td>
</tr>
<tr>
<td>SOO</td>
<td>Statement of Objectives</td>
</tr>
<tr>
<td>SOW</td>
<td>Statement of Work</td>
</tr>
<tr>
<td>SPDR</td>
<td>Sprint Preliminary Design Review</td>
</tr>
<tr>
<td>SRR</td>
<td>Systems Requirements Review</td>
</tr>
<tr>
<td>SUM</td>
<td>Spend Under Management Memorandum</td>
</tr>
<tr>
<td>SVN</td>
<td>Subversion (source code configuration management tool)</td>
</tr>
<tr>
<td>SVR</td>
<td>Systems Verification Review</td>
</tr>
<tr>
<td>Acronym</td>
<td>Description</td>
</tr>
<tr>
<td>---------</td>
<td>--------------------------------------------------</td>
</tr>
<tr>
<td>T&amp;M</td>
<td>Time and Materials (contract)</td>
</tr>
<tr>
<td>TDD</td>
<td>Test-Driven Development</td>
</tr>
<tr>
<td>TRR</td>
<td>Test Readiness Review</td>
</tr>
<tr>
<td>UCD</td>
<td>User Centered Design</td>
</tr>
<tr>
<td>US CIO</td>
<td>United States Chief Information Officer</td>
</tr>
<tr>
<td>UX</td>
<td>User Experience</td>
</tr>
<tr>
<td>WBS</td>
<td>Work Breakdown Structure</td>
</tr>
</tbody>
</table>