Software Engineering Institute
Carnegie Mellon University

Contracting for Agile Software Development in the Department of Defense: An Introduction

Eileen Wrubel
Jon Gross

August 2015

TECHNICAL NOTE
CMU/SEI-2015-TN-006

Software Solutions Division

http://www.sei.cmu.edu

Table of Contents

List of Figures

List of Tables

Acknowledgments

The Software Engineering Institute (SEI) has many points of support for the development of a research agenda on Agile acquisition. The SEI author team would like to express our appreciation to all those who took time out of their busy schedules to support this effort via interviews, introductions, or providing us with resources. Their contributions were invaluable to this process.

We extend special thanks to the following individuals:

- Captain Daniel P. Taylor, United States Coast Guard (retired)
- Kelly Goshorn, former PEX Program Manager, Air Force Life Cycle Management Center /HBBD
- David I. Gill, Contracting Officer, Internal Revenue Service
- Brad Bernard, Deputy Chief Avionics Engineer, F-22 Modernization
- many additional contributors and supporters who prefer to remain anonymous

Thanks also go to our Agile Collaboration Group members, with whom we socialized and shaped this project, and whom provided excellent discussion and feedback on the business issues surrounding contracting for Agile development. We also appreciate continued opportunities to exchange information and ideas through the Association for Enterprise Information (AFEI) events focused on Agile in government.

Finally, we extend our gratitude to our supportive colleagues

- Carol Sledge
- Jeff Thieret
- Rachel Callison, our fearless research and reference librarian
- Gerald Miller, our unfailingly patient editor

Executive Summary

As iterative, incremental, or Agile software development methods continue to gain traction in the software industry, more and more Department of Defense (DoD) programs are taking notice of these methods, as a result of contractor proposals and program office staff research, outreach, and experience. The DoDI 5000.02, published in 2015, discusses iterative software development [DoD 2015]. Differences between Agile development lifecycles and more traditional waterfall-based approaches surface throughout the lifecycle, requiring modifications to traditional milestones, documentation, delivery, and progress monitoring activities. Contracting professionals, however, generally do not receive professional career field training to guide them in developing contracts that support these adaptations.

This technical note (TN) is part of the SEI's continuing exploration of Agile in the DoD. Our prior efforts have focused on providing tools to program office teams for successfully implementing Agile methods. Throughout our data gathering for this paper, dozens of interviews conducted for past efforts, our participation with various industry and academic groups, support to SEI customer programs, and our interactions with our own Agile Collaboration Group[1] members, we heard a frequent refrain: program office teams do not feel that they "speak the same language" as contracting officers with whom they must collaborate to create contracts that allow programs to fully realize the benefits of Agile methods, while simultaneously satisfying the contracting officer's objectives of developing fair contracts that protect government interests at an acceptable level of risk while in compliance with federal statutory requirements and agency policy guidelines.

This technical note, then, is intended primarily for contracting officers. The authors provide contracting officers with a basic background in Agile development principles, contrasting Agile briefly with waterfall-based software development paradigms with which they may be familiar, as a means to set the stage for understanding how contracting deliverables and structures may need to adapt. We explore the Federal Acquisition Regulation (FAR) provisions that grant contracting officers latitude to explore innovative business practices, and the collaborative support that should be received from the program office team during the contracting process. We address the elements of the 2015 DoDI 5000.02 that support incremental software development and the tailoring of program activities, to allay concerns about newer lifecycle models.

We address common concerns and misconceptions about risk associated with Agile development, supported with examples from actual programs and interpretive guidance from various federal agencies including the Government Accountability Office (GAO), the White House Office of Technology and Policy (OSTP), and the Office of Management and Budget (OMB). In each case, we provide contracting officers with concrete questions and actions that they can take to evaluate actions and deliverables proposed when developing a contract.

[1] The Agile Collaboration Group is a consortium of more than 150 representatives from more than 45 organizations across the SEI, DoD, federal agencies, defense contractors, academic institutions, and private industry.

Finally, we address overall structure of contracts: there is considerable discussion within the Agile community about whether fixed-price or cost-reimbursable structures are preferable for Agile. We do not attempt to divine a preferable approach: many variables affect the types of contracts that can legally be employed on any given program. Both approaches can produce viable contracts that effectively deliver mission capability and provide appropriate insight into cost, progress, and software quality. Thus, we discuss both kinds of approaches and considerations that enable either type to be used effectively.

This paper is not intended to provide detailed guidance that applies to every situation, contract language, or substitute for legal advice. Rather, the authors hope to provide contracting officers with a "running start" when they encounter a program that will employ Agile methods.

Abstract

This technical note (TN), part of an ongoing Software Engineering Institute (SEI) series on Agile in the Department of Defense (DoD), addresses effective contracting for Agile software development. Contracting officers do not receive career field education targeted at achieving successful outcomes with Agile software development methods. For the purposes of this TN, the SEI gathered data from program office team members, contractors, and contracting officers about the state of contracting activities involving Agile development. The authors conducted a series of interviews and mined past interviews and survey data on Agile software development to understand common questions and concerns and provide some real-world examples to address them. This TN offers a primer on Agile based on a contracting officer's goals, describes how program office teams need to support contracting efforts, and addresses common concerns about Agile and how those concerns can be mitigated in the contracting process.

1 Introduction

In recent years the federal government and the Department of Defense (DoD) have emphasized the necessity to shorten acquisition timelines to be more responsive to increasing operating tempo and warfighter need for more rapid capability development [OSD 2010, Lapham 2011]. In 2009, the Defense Science Board wrote that "The fundamental problem DoD faces is that the deliberate process through which weapon systems and information technology are acquired does not match the speed at which new IT capabilities are being introduced in today's information age" [Defense Science Board 2009].

Additionally, requirements for any given system are highly likely to evolve between the development of a system concept and the time at which the system is operationally deployed as new threats, vulnerabilities, technologies, and conditions emerge, and users adapt their understanding of their needs as system development progresses. Dr. Matthew Kennedy, formerly of the Defense Acquisition University (DAU), wrote that "Previous experience shows that changes within an SIS [software-intensive system] are inevitable, whether or not there are changes in requirements or technology" [Kennedy 2011]. With budgets constrained, ops tempos increasing, and requirements perpetually evolving, software development and acquisition practices must evolve in a way that facilitates faster capability deployment and flexibility in approaching system requirements.

Iterative, incremental software development methodologies commonly referred to as "Agile" methods have been gaining ground in efforts throughout the DoD and federal agencies as a means to achieving these objectives for the acquisition of software-intensive systems and improving visibility into development execution to enable early detection of problems that can derail programs.

We have consistently written about cultural and behavioral shifts required on the part of program office teams and acquisition leadership to support the employment of Agile techniques [Lapham 2010, Lapham 2011, Lapham 2014, Wrubel 2014]. DoD contracting officers and program managers, while fulfilling critical roles in the acquisition process, approach the issues of software-intensive system development through different lenses. A program manager is responsible for the development and delivery of a system that meets mission needs; a contracting officer is responsible for ensuring that the contractual vehicles employed to meet those mission needs are in compliance with federal statutory requirements and agency policy guidelines. According to the Federal Acquisition Regulation (FAR) (Part 1.102-1 (b)),

> *All participants in the System are responsible for making acquisition decisions that deliver the best value product or service to the customer.* ***Best value must be viewed from a broad perspective and is achieved by balancing the many competing interests in the System.*** *The result is a system which works better and costs less* [FAR 2015, emphasis added].

Both program managers and contracting officers have different perspectives on "best value" as it pertains to their role in the process, and both must manage competing objectives such as risk (e.g., technical, financial, information), cost, schedule, desires for flexibility versus desires for predictability, socioeconomic factors in contracting, and a wide variety of other factors. The contracting officer must ensure that while a contract is in the best interest of the United States government, it also provides “impartial, fair and equitable treatment”[1] to contractors.

Throughout the SEI’s multi-year efforts to address adoption enablers for and barriers to Agile in DoD programs, program managers and engineering staff have frequently indicated that putting effective contracts in place to support Agile methods remains challenging, citing a communication barrier with their contracting officer counterparts: “We don’t speak their language, and they don’t speak ours.”[2] In other words, program office teams are having difficulty communicating their “best value” scenarios and achieving alignment with the contracting officer’s “best value” scenarios.

Currently, no formal Agile-related career-field education exists for DoD contracting officers. While Agile methods are explored in the curriculum at Defense Acquisition University, this is within the scope of IT career field coursework.[3] Most of our respondents indicated that contracting officers assigned to their contracts had little or no prior exposure to Agile software development efforts and had little opportunity to receive training from other sources—many of them learned “on the job” alongside their program office counterparts. Contracting officers with whom we spoke mirrored this assessment—they had little to no professional exposure to contracting for iterative, incremental software development methods prior to being tasked with supporting these acquisitions.

With the understanding that contracting officers typically come across Agile “cold,” the purpose of this technical note is to introduce Agile concepts and principles to contracting officers and link those concepts and principles to supporting federal and DoD publications and elements of the FAR. Understanding the underlying principles and framework for Agile is necessary for an understanding of the level of specificity at which requirements are documented and the level of involvement of the program office in development activities, determining what deliverables are necessary on a contract, and understanding how to monitor progress and quality of the software produced during the contract. We also address common misperceptions about the risk associated with leveraging Agile methods on DoD contracts, and successful examples from real programs.

The contracting community is a critical leader in the adoption of new innovation in the acquisition of new capability or services for the DoD. The program manager is a key role, both in leading the program to meeting user/warfighter needs and helping the contracting officer establish effective contracts “that deliver the best value product or service to the customer” [FAR 2015]. The authors intend to provide some initial knowledge of Agile methods and how they affect and are affected by contracts, so that the contracting officer can help improve government business practices to ensure more effective delivery of capability. We also intend to provide program managers with perspective on the support contracting officers will require to develop effective contracts for Agile development.

[1] FAR (Part 1.602-2)

[2] Interview respondent

[3] Reports from interview respondents, and also in AFEI 2012 *Agile in Defense Fall Workshop* [AFEI 2012]

Section 2 provides the contracting officer with a basic understanding of Agile methods and how the Agile lifecycle is different from traditional approaches.

Section 3 delves into the role of contracting officers, program office teams, and the Federal Acquisition Regulation.

Section 4 identifies elements of DoD acquisition guidance that support the implementation of Agile methods where appropriate.

Section 5 addresses common misconceptions about Agile methods and provides contracting officers with both examples from real programs and questions contracting officers should pursue to set up contracts that will successfully leverage Agile efforts.

Section 6 discusses how Agile methods can be supported using either fixed-price or cost-based contracts.

The appendices to this document provide additional reference material on Agile software development in the DoD and the research questions that guided our interview efforts.

2 What is Agile?

It is important to understand that Agile[4] is not one specific method; Agile software development is both a philosophy and an umbrella term for a collection of methods or approaches that share common characteristics.[5] To arrive at a brief working definition, we must first introduce the underlying tenets and principles. (Appendix B of this document contains a list of additional SEI publications on using Agile methods in the DoD; Appendix D of this document contains a glossary of common Agile terms used throughout this technical note.)

2.1 The Agile Manifesto and Defining Principles

The Agile Alliance provides some background on the genesis of Agile methods:

> *In the late 1990's several methodologies began to get increasing public attention. Each had a different combination of old ideas, new ideas, and transmuted old ideas. But they all emphasized close collaboration between the programmer team and business experts; face-to-face communication (as more efficient than written documentation); frequent delivery of new deployable business value; tight, self-organizing teams; and ways to craft the code and the team such that the inevitable requirements churn was not a crisis* [Agile Alliance 2001c].

A group of software industry practitioners and consultants, who became known as the Agile Alliance, developed and published key tenets known as the *Manifesto for Agile Software Development* [Agile Alliance 2001]:

Manifesto for Agile Software Development

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

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

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

[4] When using the term "Agile" throughout this paper, the authors refer to *software development* conducted according to principles and practices consistent with the Agile Manifesto and the Agile principles, using the definition provided in Section 2.2.

[5] Palmquist offers a more thorough discussion of the similarities and differences between waterfall-based development and Agile development [Palmquist 2014].

It is important to note that none of the elements on the right side of the list are absent; rather, they *support* and *add value* to the elements on the left side of the list:

- Tools and processes facilitate interactions between team members, as opposed to shoehorning these interactions into molds and patterns for the sake of process compliance.
- Documentation is developed to add value to development and sustainment of the code, rather than as evidence to prove compliance or completion.
- Contract negotiations must establish a collaborative work environment that enables effective decision-making and flexible response, rather than high-overhead change control processes. (This can also include early termination points to limit government risk for poor performance.)
- High-level plans must be flexible to allow for necessary evolution of system requirements; plans become more granular at the development level.

The Agile Alliance also documented 12 principles that underlie the tenets in the manifesto [Agile Alliance 2001b]:

We follow these principles:

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

With this basic understanding of the philosophy of Agile development, we can move on to understanding what happens in practice. This understanding will provide a foundation for understanding how the Agile-based development model behaves differently from traditional waterfall-based development models, and how those differences manifest themselves in program execution.

2.2 A Definition of Agile Software Development

Distilling the guidance from the tenets of the manifesto and the 12 supporting principles, one solid definition of Agile is

> *An iterative and incremental (evolutionary) approach to software development which is performed in a highly collaborative manner by self-organizing teams within an effective governance framework with "just enough" ceremony that produces high quality software in a cost effective and timely manner which meets the changing needs of its stakeholders* [Ambler 2004].

We can extend this definition by describing the behavior of an Agile team, as follows:

> *In Agile terms, an Agile team is a self-organizing cross-functional team that delivers working software, based on requirements expressed commonly as* ***user stories****, within a short timeframe (usually 2-4 weeks). The user stories often belong to a larger defined set of stories that may scope a release, often called an* ***epic****. The short timeframe is usually called an* ***iteration*** *or, in Scrum*[6]*-based teams, a* ***sprint****; multiple iterations make up a* ***release****. The team's progress toward completion of the iteration is measured via the team's* ***velocity****. While the code produced within an iteration is useable, it may not have enough functionality to be released to the end user until the multiple iterations that make up a release are completed* [Lapham 2011].

Agile methods involve successive iterations of software development, each iteration producing working software, and enough documentation to develop and support the associated code base. Understanding how Agile teams produce code allows us to understand how acquisition and contracting guidance must be adapted.

Throughout the rest of this paper, when using the term "Agile," the authors refer to *software development* conducted according to principles and practices consistent with the Agile Manifesto and the Agile principles, using the definition provided in this section.

2.3 Mapping of Manifesto to 12 Principles

We can map the tenets of the Agile Manifesto to the 12 supporting principles (some principles are mapped more than once). Items highlighted in ***bold italic*** in Table 1 are of special note for contracting officers and are themes consistently addressed throughout this technical note.

6 http://www.mountaingoatsoftware.com/agile/scrum

Table 1: Mapping of Agile Manifesto Tenets and Supporting Principles

Tenets	Principles
Individuals and interactions over processes and tools	***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. 8. Agile processes promote sustainable development. The sponsors, developers, and users should be able to maintain a constant pace indefinitely. 11. The best architectures, requirements, and designs emerge from self-organizing teams.
Working software over comprehensive documentation	1. Our highest priority is to satisfy the customer through early and continuous delivery of valuable software. 3. ***Deliver working software frequently***, from a couple of weeks to a couple of months, with a preference to the shorter timescale. ***7. Working software is the primary measure of progress.*** 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.***
Customer collaboration over contract negotiation	**1. Our highest priority is to satisfy the customer through early and continuous delivery of valuable software.** ***4. Business people and developers must work together daily throughout the project.*** 6. The most efficient and effective method of conveying information to and within a development team is face-to-face conversation. 8. Agile processes promote sustainable development. The sponsors, developers, and users should be able to maintain a constant pace indefinitely.
Responding to change over following a plan	***2. Welcome changing requirements, even late in development. Agile*** **processes** ***harness change for the customer's competitive advantage.*** 9. Continuous attention to technical excellence and good design enhances agility. 12. At regular intervals, the team reflects on how to become more effective, then tunes and adjusts its behavior accordingly.

2.4 How Does Agile Differ from Traditional Software Development Methods?

Contracting officers on DoD programs are likely to be familiar with phased approaches to software development (such as waterfall): system requirements are thoroughly documented and provided to a contractor, and the software development progresses through distinct phases of design, development, and test, resulting in a final delivery of completed software. A rigorous Engineering Change Proposal (ECP) process is implemented to tightly control any changes to requirements or design, generally requiring a contractual action. In the acquisition process, the software development phases map to milestone reviews such as the Preliminary Design Review (PDR), Critical

Design Review (CDR), etc. Each milestone must be met before progressing further with design, coding, and test.

Under Agile,[7] an exhaustive set of requirements is not locked down at the start of the program. Rather, Agile development assumes that system and software requirements will evolve over time, rather than be definitized prior to system development. With Agile, high-level vision for the system is defined up front,[8] but specific requirements are fixed at the iteration (or "sprint") level according to an established cadence. The development team and user representative (generally referred to as a "product owner") agree to a set of requirements to accomplish during the defined time interval associated with the iteration. This serves to time-box the delivery of software: increments are completed on a regular, predictable basis. At the end of a sprint or increment, prioritized software requirements are agreed to for the next development iteration. The understanding of the user requirements evolves, guided by the high-level vision (or roadmap), as the software product continues to be developed (see Figure 2 for a graphic representation).

The development team's capacity for execution (typically referred to as velocity[9]) is used as a boundary for the number of stories to which the team commits during any given iteration. Requirements refinement, design, development, and testing are all completed within the scope of the increment. Testers and other specialists[10] are either members of the development team, or working in very close coordination. When the working software is completed at the end of each iteration, the development team and the product owner can readily assess that the increment has achieved functional and quality commitments as described in the requirements agreed to for that iteration. The code delivered at the end of each iteration is *production-quality* code.

It is important to note that "delivery" and "deployment" of software products developed under Agile methods are not synonymous. Code fielding is not required at the completion of each increment or even after several increments (see Figure 2—in this example production quality code is delivered at the end of each iteration, but deployed only as key capabilities are available and suitable for fielding at the release level). However, as each iteration is developed, production-quality code is built upon.

Figure 1, excerpted from the GAO's 2012 report, *Effective Practices and Federal Challenges in Applying Agile Methods*, provides a good high-level visualization of the differences between the Agile approach and a more traditional waterfall-oriented approach [GAO 2012].

7 Appendix D contains a glossary of Agile terms.

8 The system under development may be exclusively software being developed by the Agile team, or the software may be a *part* of a broader system.

9 Velocity measures are unique to a software team, derived from historical data about that team's performance. As such, velocity is useful in allocating work to a software increment, *for a particular team under local conditions*, but is not an effective tool for comparisons or work allocation across different teams [Sliger 2008, Hayes 2014].

10 Such as certification and accreditation representatives

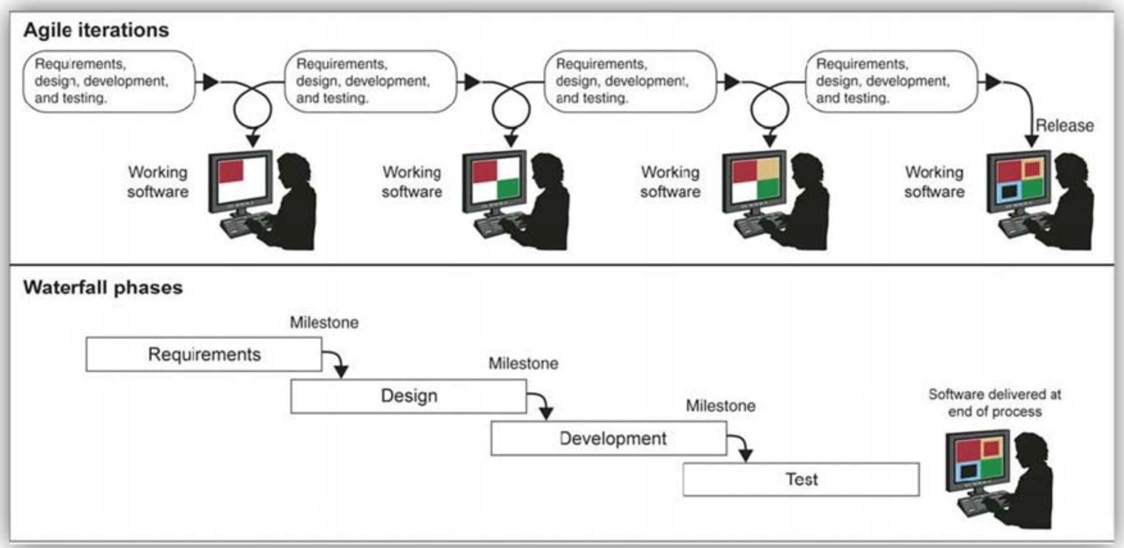

Agile iterations
Requirements, design, development, and testing.
Requirements, design, development, and testing.
Requirements, design, development, and testing.
Requirements, design, development, and testing.
Working software
Working software
Working software
Working software
Release
Waterfall phases
Requirements
Milestone
Design
Milestone
Development
Milestone
Test
Software delivered at end of process

Figure 1: Agile Versus Waterfall Software Delivery

Figure 2 provides a more detailed overview of the Agile lifecycle showing how individual iterations comprise releases. The roadmap represents the vision and overall direction of the program. The product backlog represents high-level requirements, which are then refined during each development sprint. The grey arrows in the diagram represent the continuous, collaborative involvement of critical stakeholders. Progress along the roadmap is achieved incrementally, with production quality code delivered at defined intervals. Depending on the mission needs, the government may choose to deploy interim releases that contain militarily useful capabilities.

(Section 2.6 provides additional resources for gaining more in-depth insight into Agile development, and Appendix D provides a glossary of some common Agile terms used in this technical note.)

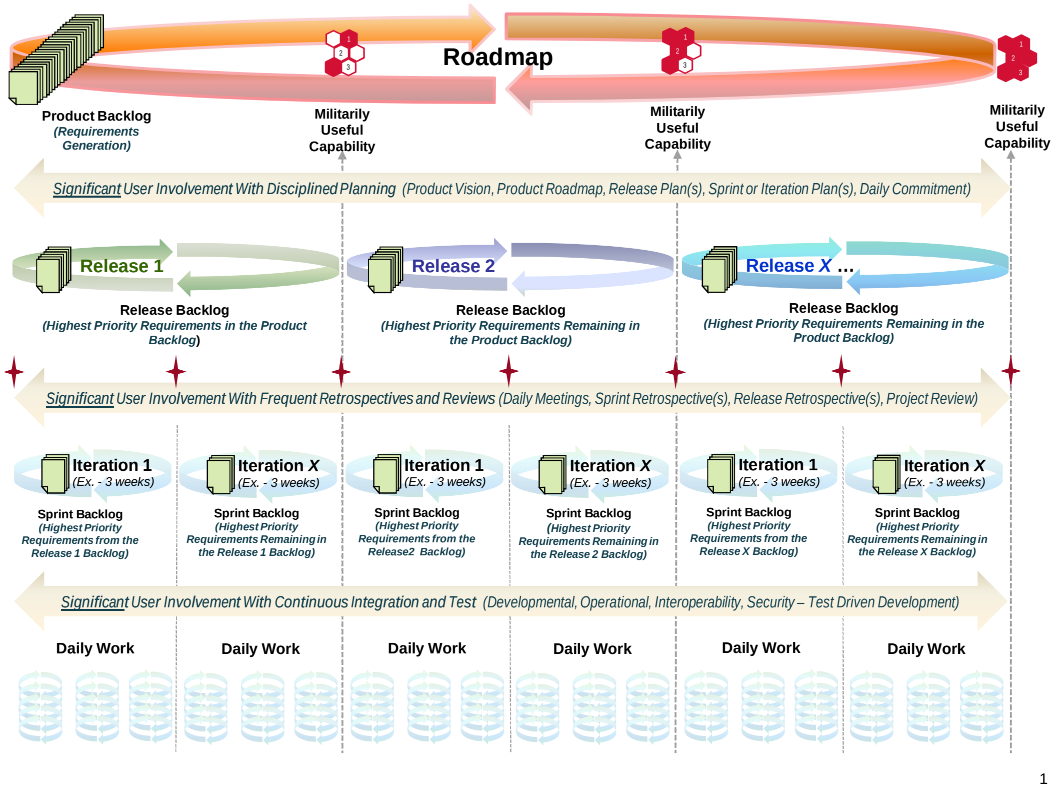

Roadmap
Product Backlog
(Requirements Generation)
Militarily Useful Capability
Militarily Useful Capability
Militarily Useful Capability
Significant User Involvement With Disciplined Planning (Product Vision, Product Roadmap, Release Plan(s), Sprint or Iteration Plan(s), Daily Commitment)
Release 1
Release Backlog
(Highest Priority Requirements in the Product Backlog)
Release 2
Release Backlog
(Highest Priority Requirements Remaining in the Product Backlog)
Release X …
Release Backlog
(Highest Priority Requirements Remaining in the Product Backlog)
Significant User Involvement With Frequent Retrospectives and Reviews (Daily Meetings, Sprint Retrospective(s), Release Retrospective(s), Project Review)
Iteration 1 (Ex. - 3 weeks)
Sprint Backlog (Highest Priority Requirements from the Release 1 Backlog)
Iteration X (Ex. - 3 weeks)
Sprint Backlog (Highest Priority Requirements Remaining in the Release 1 Backlog)
Iteration 1 (Ex. - 3 weeks)
Sprint Backlog (Highest Priority Requirements from the Release2 Backlog)
Iteration X (Ex. - 3 weeks)
Sprint Backlog (Highest Priority Requirements Remaining in the Release 2 Backlog)
Iteration 1 (Ex. - 3 weeks)
Sprint Backlog (Highest Priority Requirements from the Release X Backlog)
Iteration X (Ex. - 3 weeks)
Sprint Backlog (Highest Priority Requirements Remaining in the Release X Backlog)
Significant User Involvement With Continuous Integration and Test (Developmental, Operational, Interoperability, Security – Test Driven Development)
Daily Work
Daily Work
Daily Work
Daily Work
Daily Work
Daily Work
1

Figure 2: Agile Lifecycle [Palmquist 2014]

In *Parallel Worlds: Agile and Waterfall Differences and Similarities*, Palmquist addresses how these lifecycle differences between waterfall-based and Agile approaches may appear in program execution [Palmquist 2014]. Requirements are fixed at a more granular level; reviews of the work product happen more frequently and assess each individual increment rather than a "big bang" development. Table 2 contrasts the two approaches at a high level. Additional detail can be found in Palmquist.

Table 2: Differences Between Agile and Waterfall [Palmquist 2014]

Traditional Principles	**Agile Instantiation**
Plan the work—especially the budget, schedule, and deliverables—to the maximum extent possible before beginning any design or code.	• Near-term plans contain more detail, while plans further out on the time horizon contain fewer details. • The overall vision is broken down into a roadmap, which is further broken down into release plans, which are further broken down into sprint or iteration plans, which are further broken down into daily plans. • Requirements are prioritized. • Cost and schedule estimates are prepared for each capability at a high level. Relative estimation versus absolute estimation is employed. • Frequent planning sessions (at the beginning of each iteration) result in detailed, high-fidelity plans. • Risks are assessed and risk mitigation influences planning.

Traditional Principles	Agile Instantiation
Lock down requirements to prevent gold-plating and scope creep.	• No requirements can be added to an iteration once it has started. • New requirements are evaluated by the stakeholders and prioritized thus preventing gold-plating and scope creep.
Institute multiple reviews to provide senior leadership oversight as well as to serve as gates for continued work.	• The customer is involved in all aspects of planning and testing. The customer (in the form of the product owner) is involved daily. • There are reviews at the end of each iteration that serve as gates to further work.
Move forward in a step-by-step, sequential manner and only when all parts of the previous steps were complete.	• The code base is integrated and tested daily. • The code base must pass all tests before and after integration. Regression testing is typically done each night.
Capture all details with extensive documentation.	• There is an overall plan. • There are requirements descriptions. • There are cost and schedule estimates. • There are risk assessments. • There is training material (as appropriate). • There is documentation (as appropriate). • There are lessons learned (based on retrospectives).

Metrics are also treated differently between traditional and Agile approaches. Section 4.2 discusses some of the metric differences.

2.5 What Are the Benefits of Agile Methods?[11]

Agile methods show greater promise in enabling organizations to adjust to changing requirements and rapidly field software as compared to other development approaches such as the waterfall approach. In contrast to waterfall-based projects, Agile seeks to deliver small but functioning software in increments that eventually build up to the full desired capability. In this manner, users (or their representatives) can begin to interact with the software system earlier. Users receive some minimal capability early rather than waiting until the end of the entire waterfall lifecycle to receive any working software. This can reduce lifecycle costs by eliminating the development of unnecessary and unwanted features. Additional benefits seen from using Agile methods include

- early insight by the users into the actual design and implementation of the solution
- the ability to modify requirements and priorities throughout the lifecycle allows for flexibility to adapt to a changing environment
- opportunities to potentially deploy the solution in stages, putting capability in warfighter hands sooner, while delaying less critical functionality to later releases
- opportunities to "fail fast" and make timely adjustments if the early solution ideas turn out to be flawed; little time or money is spent before that learning occurs, and redirection can be implemented (this includes early opportunities to address performance problems with contractors)

[11] The material in Section 2.5 was reproduced in its entirety from *Innovative Contracting Case Studies*, a report of the White House Office of Science and Technology Policy [OSTP 2014].

- an explicit understanding on the part of the development and acquiring organizations that the requirements are expected to evolve and are a natural part of software development and ensuring value is delivered to the customer [Highsmith 2000, Nidiffer 2014, Kennedy 2011].

2.6 Further Reading

While a basic background on Agile approaches is necessary for further discussion, the objective of this paper is not to provide a comprehensive definition of Agile terms and lifecycle components. The resources documented in this section provide foundational knowledge on Agile methods, which will aid in discussing and applying the concepts described in this paper.

There is considerable literature available to further enhance one's understanding of the specific elements of Agile. In developing definitions and figures, the SEI drew from a number of sources including the following:

1. http://www.agilealliance.org/the-alliance/what-is-agile/
2. http://www.aspe-sdlc.com (*Agile Glossary: Words and Terms Common to Agile Methods*)
3. http://www.telerik.com/agile-project-management-tools/agile-resources/vocabulary.aspx
4. http://www.accurev.com/wiki/agile-glosssary
5. http://www.develop.com/agiledemystified

The SEI's report, *Parallel Worlds: Agile and Waterfall Differences and Similarities*, provides more detailed assessment of similarities and differences between Agile and traditional methods [Palmquist 2014]. A list of SEI publications on leveraging Agile software development in the acquisition environment is provided in Appendix B.

The U.S. chief information officer (CIO) is continuing to help encourage a change in the way the federal government acquires information technology (IT).[12] In 2010, the *25 Point Implementation Plan* started a cultural shift in U.S. government approaches to IT.[13] In August 2014, the U.S. CIO announced the release for public comment of two publications that will continue the encouragement of a culture change in the federal government to improve and simplify the modernization of government acquisition: the *Digital Services Playbook* and the *TechFAR Handbook* [U.S. CIO 2014]. The support for culture change in federal IT acquisition and development continues.

This specific work on contracting for Agile software development is heavily influenced by the work done in the United Kingdom and the European community on contracting for Agile approaches to software development. Two key publications are very valuable resources for more detailed treatment of the legal and contracting approaches. The publications are

- *Practices for Scaling Lean & Agile Development: Large, Multisite, & Offshore Product Development with Large-Scale Scrum.* Tom Arbogast, Craig Larman, and Bas Vodde. http://www.agilecontracts.org [Arbogast 2012]

[12] https://cio.gov/

[13] https://www.dhs.gov/sites/default/files/publications/digital-strategy/25-point-implementation-plan-to-reform-federal-it.pdf

- *Agile Contracts. Creating and Managing Successful Projects with Scrum*. Andreas Opelt, Boris Gloger, Wolfgang Pfarl, Ralf Mittermayr. http://onlinelibrary.wiley.com/book/10.1002/9781118640067 [Opelt 2013]

Both of these books are written within the European legal context, but provide valuable insights for any contracting officer or their contracting officer representative.

In the next section, we discuss the role of the contracting officer and the program office, in addition to elements within the FAR that support innovation and collaboration consistent with Agile approaches.

3 The Contracting Officer, the FAR, and Agile

The contracting officer, by his or her responsibility and authority, falls squarely in the middle of defining the appropriate balance between the two sides of the Agile software development manifesto tenet "Customer collaboration over contract negotiation." This balance must be established by both the contract and the trust that builds up by effective performance of the program office and the contractor. The contract governing Agile development must provide sufficient structure to protect all parties and achieve the desired mission outcomes, while offering flexibility for adaptation of software requirements within the agreed-on scope of the system. In this section we briefly discuss the responsibilities of the contracting officer and the acquisition team and how the expectations set forth in the FAR align with principles and tenets of Agile development. We also discuss the FAR's emphasis on flexibility to support innovation in business practices.

3.1 Contracting Officers and the Acquisition Team: Definitions and Authority

The contracting officer is a critical leader in the FAR System,[14] as a key member of the acquisition team with a unique responsibility: "Contracts may be entered into and signed on behalf of the Government only by contracting officers."[15] It is important to understand the specific authority and responsibility of contracting officers from the FAR. The specifics will help focus on the needs of the contracting officer in contracting for software development according to Agile principles.

> *FAR 1.602-1 Authority.*
>
> (a) *Contracting officers have authority to enter into, administer, or terminate contracts and make related determinations and findings. Contracting officers may bind the Government only to the extent of the authority delegated to them. Contracting officers shall receive from the appointing authority (see 1.603-1) clear instructions in writing regarding the limits of their authority. Information on the limits of the contracting officers' authority shall be readily available to the public and agency personnel.*
>
> (b) ***No contract shall be entered into unless the contracting officer ensures that all requirements of law, executive orders, regulations, and all other applicable procedures, including clearances and approvals, have been met*** [FAR 2015, emphasis added].

Taking a look at the FAR definition of the roles of Acquisition Team[16] members helps to lay the foundation for how a contracting officer can think about building effective contracts for Agile software development.

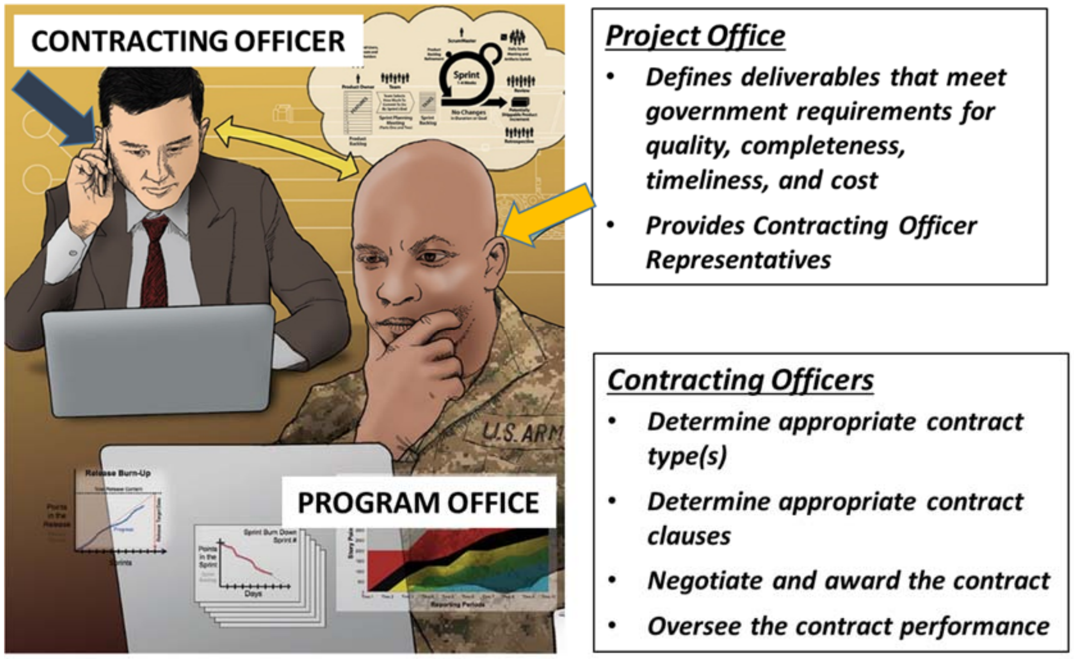
[14] For the Department of Defense, Defense Federal Acquisition Regulation Supplement (DFARS) are the guiding policy. The DFARS provides DoD implementation and supplementation of the FAR. The DFARS contains requirements of law, DoD-wide policies, delegations of FAR authorities, deviations from FAR requirements, and policies and procedures that have a significant effect on the public.

[15] FAR Subpart 1.6—Career Development, Contracting Authority, and Responsibilities; 1.601 General.

[16] "The Acquisition Team consists of all participants in Government acquisition including not only representatives of the technical, supply, and procurement communities but also the customers they serve, and the contractors who provide the products and services" [FAR 2015].

> *1.102-4 Role of the Acquisition Team.*
>
> *(a) Government members of the Team must* ***be empowered to make acquisition decisions within their areas of responsibility****, including selection, negotiation, and administration of contracts consistent with the Guiding Principles. In particular, the contracting officer must have the authority to the maximum extent practicable and consistent with law, to determine the application of rules, regulations, and policies, on a specific contract.*
>
> *(b) The authority to make decisions and the accountability for the decisions made will be* ***delegated to the lowest level within the System****, consistent with law.*
>
> *(c) The Team must be prepared to perform the functions and duties assigned. The Government is committed to provide training, professional development, and other resources necessary for maintaining and improving the knowledge, skills, and abilities for all Government participants on the Team, both with regard to their particular area of responsibility within the System, and their respective role as a team member. The contractor community is encouraged to do likewise.*
>
> *(d) The System will foster* ***cooperative relationships*** *between the Government and its contractors consistent with its overriding responsibility to the taxpayers.*
>
> *(e) The FAR outlines procurement policies and procedures that are used by members of the Acquisition Team. If a policy or procedure, or a particular strategy or practice, is in the best interest of the Government and is not specifically addressed in the FAR, nor prohibited by law (statute or case law), Executive order or other regulation, Government members of the Team should not assume it is prohibited. Rather, 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*** [FAR 2015, emphasis added].

Contracting officers, then, are encouraged by the FAR to adapt business practices to support innovative methods and techniques, so far as those adaptations are consistent with the FAR, federal law, and agency policy and regulation. The other members of the Acquisition Team need to provide the contracting officer with the resources, information, and support required to do so. The system itself encourages cooperative, collaborative relationships between the parties and the delegation of decision-making. The Agile Manifesto and the 12 supporting principles emphasize collaboration between the parties, consistent with the FAR. The FAR emphasis on making decisions at the lowest level (within the constraints of the law) is also consistent with the collaborative emphasis of Agile software development and its iterative approach to the discovery and evolution of system requirements.



Project Office

- ***Defines deliverables that meet government requirements for quality, completeness, timeliness, and cost***
- ***Provides Contracting Officer Representatives***

Contracting Officers

- ***Determine appropriate contract type(s)***
- ***Determine appropriate contract clauses***
- ***Negotiate and award the contract***
- ***Oversee the contract performance***

Figure 3: Differing Perspectives (Adapted from Hayes [Hayes 2014])

3.2 Stakeholder Objectives and Collaboration

The focus of the contracting officer responsibility is not only about the mission or the business needs. The contracting officer is dependent on the program office, both prior to and after contract award. The flow of requirements and needs comes through the program office. Regardless of the software development approach preferred by the program office, the legal and regulatory environment is unchanged for the contracting officer. When contracting for Agile software development efforts, the contractual agreements must fit into the same legal and regulatory framework that traditional acquisition programs fit. Of course, the chosen Agile software development approach does not override these requirements on a contracting officer; we will demonstrate through this paper that Agile principles also do not inherently conflict with a contracting officer's responsibilities. The Agile software development focus on delivery of "working software" and "customer collaboration" enable the contracting officer to monitor, administer, and even terminate the contract as provided under the law.

The program office likewise is dependent on the contracting officer to ensure an effective contract is put in place to accomplish the mission and business needs. As we discussed previously, the contracting officer must act in the best interest of the government while balancing a variety of competing interests.

The specific requirement on a contracting office is to ensure all legal requirements, procedures, and approvals have been met *before* awarding the contract. The contracting officer needs the support of subject matter experts across the acquisition team to satisfy this sweeping requirement. FAR 1.602-2(c) requires the contracting officer to obtain support from all specialists necessary for meeting responsibilities for "effective contracting, ensuring compliance with the terms of the contract, and safeguarding the interests of the United States in its contractual relationships."[17]

This support for the contracting officer is analogous to the multi-disciplinary approach to Agile teaming discussed earlier. Throughout the Agile software development process, the multidisciplinary team works together to develop the software incrementally: software engineers engage with information security agencies, test groups, operational specialists (users), and other stakeholders on a continuous basis. The program office team should also be engaged on an ongoing basis with subject matter experts that overlap with those required to support contract development. Integrating those personnel/functions into the discovery process that supports the contracting officer's work should be straightforward and ensure that the legal/contracting objectives are aligned with the mission objectives of the program.

3.3 Flexibility to Innovate

As we continue to review the FAR, we see again that it encourages behaviors consistent with Agile principles:

> *FAR 1.602-2 Responsibilities.*
>
> *Contracting officers are responsible for ensuring performance of all necessary actions for effective contracting, ensuring compliance with the terms of the contract, and safeguarding the interests of the United States in its contractual relationships. In order to perform these responsibilities**, contracting officers should be allowed wide latitude to exercise business judgment*** [FAR 2015, *emphasis added*].

Contracting officers should be able to use their good professional judgment about contract vehicles, terms, and incentives that make the most sense to get the best value outcome for the government. Many individuals we encountered in the course of our interviews for this paper, and our other research efforts, have indicated that contracting officers can find their options regarding contract types curtailed by local agency policy. When this is the case, contracting officers and their supporting program managers should work within the confines of that guidance to determine what additional features of the contract might afford flexibility to most appropriately meet the mission needs of the program, as provided by the FAR.

The FAR (Part 1.102 (d)) provides important flexibility:

> *The role of each member of the Acquisition Team is to exercise personal initiative and sound business judgment in providing the best value product or service to meet the customer's needs. In exercising initiative,* ***Government members of the Acquisition Team may assume if a specific strategy, practice, policy or procedure is in the best interests of the Government and is not addressed in the FAR, nor prohibited by law (statute or case law), Executive order or other regulation, that the strategy, practice, policy or procedure is a permissible exercise of authority*** [FAR 2015, emphasis added].

To help the program be successful in an Agile software development contract, program managers must realize that contracting professionals need support that demonstrates the actions and out-

[17] FAR 1.602-2 Responsibilities

comes desired both meet the mission needs and are consistent with all appropriate statutory, regulatory, and policy requirements. Contracting officers must exercise the latitude within the FAR to achieve these objectives when software is developed in an Agile manner. The role of the program office needs to shift to include helping to ensure that a successful contract can be awarded. Only through a successful contract, that supports Agile software development, can the mission and business needs be satisfied.

In the following section, we address the elements of the newest revision of DoDI 5000.02 that support iterative software development (such as Agile software development) and the tailoring of acquisition processes to support such implementations.

4 Incremental Development in the DoDI 5000.02

Contracting officers can find support for leveraging Agile software development methods in the current DoDI 5000.02, released in January 2015. While the new guidance does not name any specific development methodology (such as Agile), its guidance supports both iterative software development and the adaptation of business processes in the acquisition program that are commensurate with iterative software development approaches.

DoDI 5000.02 now includes illustrative models of programs, based on predominant characteristics, and advises acquisition personnel to "use the models as a starting point for structuring unique programs" [DoD 2015]. It also makes clear that incremental approaches are not limited simply to IT systems, but can be utilized on weapon system and other hardware/software hybrid programs. Model Programs 2 and 3 and Hybrid Model Programs A and B embrace these approaches, emphasizing

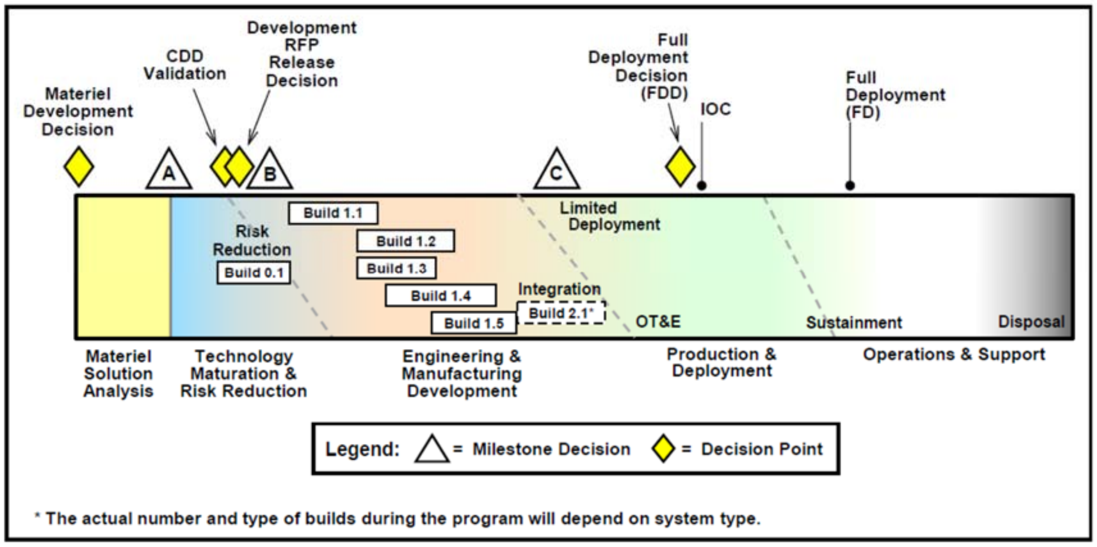
- small, testable builds with agreed-to definitions of "done" (also see Appendix D)
- each build features testable functionality that demonstrates progress

In this section we will highlight Agile-friendly provisions in the model programs and tailoring guidelines from DoDI 5000.02.

4.1 Software Intensive Programs, Model Program 2

Model Program 2, Defense Unique Software Intensive Program is characterized as

> *dominated by the need to develop a complex, usually defense unique, software program that will not be deployed until several software builds have been completed.* ***The central feature of this model is the planned software builds – a series of testable, integrated subsets of the overall capability – which together with clearly defined decision criteria, ensure adequate progress is being made before fully committing to subsequent builds.***
>
> 1. *Examples of this type of product include military unique command and control systems and significant upgrades to the combat systems found on major weapons systems such as surface combatants and tactical aircraft.*
> 2. ***Several software builds are typically necessary to achieve a deployable capability.*** *Each build has allocated requirements, resources, and scheduled testing to align dependencies with subsequent builds and to* ***produce testable functionality to ensure that progress is being achieved****. The build sequencing should be logically structured to flow the workforce from effort to effort smoothly and efficiently, while reducing overall cost and schedule risk for the program* [DoD 2015, emphasis added].



Materiel Development Decision
CDD Validation
Development RFP Release Decision
A
B
C
Full Deployment Decision (FDD)
IOC
Full Deployment (FD)
Risk Reduction
Build 0.1
Build 1.1
Build 1.2
Build 1.3
Build 1.4
Build 1.5
Integration
Build 2.1*
Limited Deployment
OT&E
Sustainment
Disposal
Materiel Solution Analysis
Technology Maturation & Risk Reduction
Engineering & Manufacturing Development
Production & Deployment
Operations & Support
Legend: = Milestone Decision = Decision Point
* The actual number and type of builds during the program will depend on system type.

Figure 4: Model 2: Defense Unique Software Intensive Program [DoD 2015]

Figure 4, reproduced from DoDI 5000.02, demonstrates Model Program 2. Model 2 emphasizes multiple software builds to achieve deployable capabilities, but the software is still fielded in a single deployment. In the case of tactical aircraft, for example, avionics software cannot be fielded to deliver military capability without the rest of the system (the aircraft itself). Even though the software is not deployed throughout the lifecycle, the multiple small increments allow the program to monitor progress and quality, incrementally integrate and test software, and facilitate early identification of any software-related risks. This in turn reduces integration risk with the other components of the platform.

4.2 Incrementally Deployed Programs, Model Program 3

Model Program 3, Incrementally Deployed Software Intensive Program (shown in Figure 5 below), is described as a program that will deploy multiple increments over time, as typically seen in IT systems.

> *It also applies to upgrades to some command and control systems or weapons systems software where deployment of the full capability will occur in multiple increments as new capability is developed and delivered, nominally in 1- to 2-year cycles. The period of each increment should not be arbitrarily constrained. The* ***length of each increment and the number of deployable increments should be tailored*** *and based on the logical progression of development and deployment for use in the field for the specific product being acquired.*
>
> 1. *This model is distinguished from the previous model by the rapid delivery of capability through* ***multiple acquisition increments, each of which provides part of the overall required program capability****. Each increment may have several limited deployments; each deployment will result from a specific build and provide the user with a mature and tested sub-element of the overall incremental capability. Several builds and deployments will typically be necessary to satisfy approved requirements for an increment of capability. The identification and development of technical solutions necessary for follow-on capability increments have some degree of concurrency, allowing subsequent increments to be initiated and executed more rapidly.*

2. *This model will apply in cases where commercial off-the-shelf software, such as commercial business systems with multiple modular capabilities, are acquired and adapted for DoD applications. An important caution in using this model is that it can be structured so that the program is overwhelmed with frequent milestone or fielding decision points and associated approval reviews.* ***To avoid this, multiple activities or build phases may be approved at any given milestone or decision point, subject to adequate planning, well-defined exit criteria, and demonstrated progress.*** *An early decision to select the content for each follow-on increment (2 through N) will permit initiation of activity associated with those increments. Several increments will typically be necessary to achieve the required capability* [DoD 2015, emphasis added].

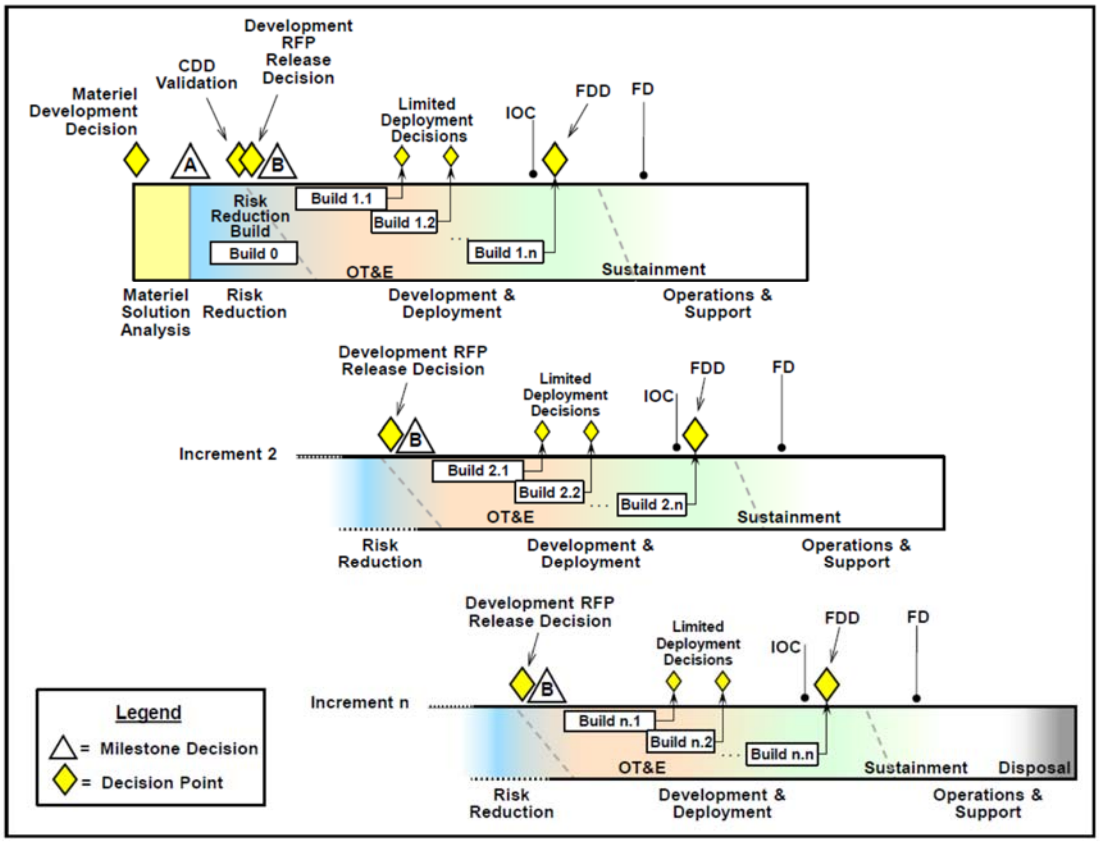


Figure 5: Model 3: Incrementally Deployed Software Intensive Program [DoD 2015]

The 5000.02 also identifies Hybrid model programs A and B, designated respectively as hardware dominant or software dominant, both indicating a reliance on incremental software builds.

4.3 Tailoring is Expected Behavior

The 5000.02 cautions that frequent fielding/deployment have potential to overwhelm programs by creating additional milestones or fielding decisions. This is a common concern expressed by those unfamiliar with Agile methods as well (we discuss whether or not Agile creates additional contracting overhead in Section 5.4). However, the guidance also instructs programs that those increases in overhead can be avoided.

As we emphasized above in Section 4.2, “multiple activities or build phases may be approved at any given milestone or decision point, subject to adequate planning, well-defined exit criteria, and demonstrated progress” [DoD 2015]. Later on, the 5000.02 provides this guidance: “Tailoring is always appropriate when it will produce a more efficient and effective acquisition approach for the specific product” [DoD 2015].

User needs and capabilities are not the same from program to program, so programs and the acquisition strategy will not be the same. The models are an aid to the acquisition team, as they define the acquisition strategy and plans to obtain the best value for the user needs. The tailoring guidance provides significant latitude to program managers and Milestone Decision Authorities (MDAs) to find the best possible development solutions to meet mission needs.

We have learned from our extensive interviews over the last five years that DoD programs adopting Agile/iterative methods have reported great success with tailored incremental milestone reviews that demonstrate progress with working, tested software. (Section 5.5 describes tailoring of program milestones in more detail.)

The next section of this technical note addresses common areas of concern or misunderstanding about Agile methods, real program examples that mitigated those concerns, and ways contracting officers can help to minimize risk and maximize the probability of success when Agile methods are employed.

5 Agile/DoD Contracting: Addressing Common Misconceptions

Now that we have provided a basic overall understanding of Agile and the latitude provided by both the FAR and DoDI 5000.02 to support divergence from traditional waterfall-based approaches, we turn to addressing common concerns expressed about the use of Agile methods on DoD programs.

Over the course of our multi-year research effort into Agile adoption within the DoD and through our Agile Collaboration Group initiative, we have interviewed dozens of participants in varying roles on programs acquiring software-intensive systems throughout the DoD and federal government. This section describes common concerns or misconceptions about Agile methods that have been reported to us by those practitioners and provides examples of solutions employed on successful Agile programs in the past. We also address possible solutions to specific concerns that a contracting officer may have regarding execution of and compliance monitoring under a contract incorporating Agile software development. We also provide recommendations to the contracting officer about what to look for when Agile approaches are applied by the program and the contractor.

5.1 "Agile Doesn't Produce Any/Enough Documentation"

As previously discussed, Agile projects rely primarily on the delivery of working software to demonstrate progress and try to limit documentation produced to that which directly adds value to the development, sustainment, and operational use of the software. More traditional approaches tend to rely on the production of deliverable documentation throughout the lifecycle. Waterfall-style projects typically produce one large requirements document up front, and subsequently large architecture and design documents, to be approved before the development of code begins. By contrast, under Agile each iteration will feature a "definition of done" that describes the contents of the iteration (often as simply as in a brief memorandum), and architecture and design documents will be updated during each iteration, evolving over the lifetime of the software project or software portion of the project.

Thus, the proposed set of documentation deliverables under a contract incorporating Agile software development may seem light. More conservative contracting officers who have not had broad experience with Agile approaches may tend to be a bit nervous about this, interpreting the decrease in documentation as a lack of evidence of progress, quality, or completion.

The emphasis Agile approaches place on automation can help to allay those concerns in practice:

> *Respondents who had the budget and resources to support automation (to the extent practical) of testing, integration, and other activities were able to take advantage of automation tools to streamline the development of documentation deliverables required under contracts or other agreements…. respondents who made extensive use of automation reported that they were able to produce documentation from their Agile workflow that satisfied traceability and other communication requirements, for both program offices and systems engineers. Even if these documents go beyond the minimal required documentation favored under Agile methods, automation supported generating them as much as possible from work-in-progress artifacts in a manner that minimized the amount of additional work required to produce the*

artifacts; in other words, the team was able to "maximize the amount of work not done" in the production of additional documents [Agile Alliance 2001, Wrubel 2014].

If deliverable documentation looks thin or if a contracting officer is concerned that the deliverable documentation proposed will not be sufficient to facilitate the development, sustainment, and operations, the contracting officer should ask the program manager to demonstrate how the documentation relates to the objectives of program execution and documentation requirements. Documentation that does not derive naturally from the development and automation efforts will create additional overhead; when determining if additional documentation deliverables are necessary, *question the value of the additional documentation being requested compared to the effort required to create it.*

As DoDI 5000.02 indicates that tailoring is appropriate to achieve program objectives, ask program managers to verify whether the documentation set proposed by the contractor is in compliance with the information requirements of guiding policy, rather than traditional documentation formats. Additional documentation requirements may be placed on contract to support specific compliance requirements, such as in the case of certification and accreditation processes. Even in these cases, it is possible that a contractor may propose an iterative approach to the development of these documents. Where possible, accepting documentation products generated from automation tools, used by the software team, may be a more effective approach for the government than requiring special customized documentation.

A Contracting Officer Should

- Question whether each document proposed is *necessary* for development, sustainment, or support (eliminating unnecessary documentation to reduce unnecessary cost).
- Question whether the proposed set of documentation deliverables is *sufficient* for development, sustainment, or support.
- Question whether any other specific compliance-related documentation (e.g., specific reports required for Certification and Accreditation purposes) is needed to comply with specific regulatory or statutory needs. Recognize that these documents will likely incur additional cost, and support their delivery in contractor-specified format whenever possible/feasible.

5.2 "Agile Methods Don't Offer Enough Insight"

Opportunities for Insight

Many who don't have experience with Agile are concerned about whether the government can achieve the appropriate level of insight into the progress and quality of software development, given that Agile assumes rather than starting from a fixed state, requirements evolve as understanding of the system evolves. This concern is magnified when Agile development schedules do not align nicely with program milestone events like the Program Design Review (PDR) and the Critical Design Review (CDR). Additionally, the development cadence on Agile software projects generally does not fall into line with monthly reporting intervals, which can cause consternation.

In actuality, well-executed Agile projects can offer far greater opportunities for meaningful insight into program progress than traditional waterfall-oriented projects. As one respondent indicated, “You don’t deliver working code in three weeks without discipline.”[18] Agile methods are also highly collaborative by design, so a program office or empowered user representative should be working shoulder-to-shoulder with the developer on a constant basis.

Agile software projects cannot succeed without consistent collaboration with an empowered end-user representative, or a designated product owner. Just as the government utilizes Key Personnel clauses for contractors as a means of risk mitigation, the government’s responsibility to provide an empowered person operating in a product owner role is critical. The product owner is essential to the ability for Agile software teams to build and deliver software that meets the needs of the warfighter (represented by the product owner). An Agile software developer will insist on government personnel participating in a product owner role, and the responsibilities and commitments of this role should be included in the contract.

Figure 6 demonstrates many opportunities throughout the Agile development lifecycle that offer the opportunity to collect data about the quality of software products. This graphic is not intended to represent every activity in the development cycle. However, it makes clear that when the program office is actively participating in Agile efforts, insight into development progress is nearly constant.

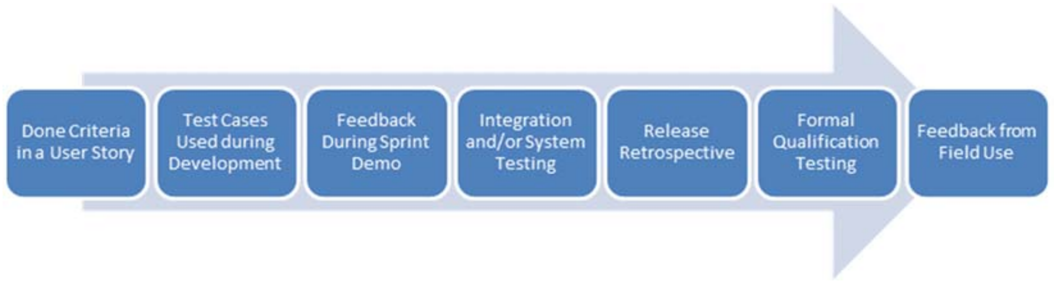

Done Criteria in a User Story
Test Cases Used during Development
Feedback During Sprint Demo
Integration and/or System Testing
Release Retrospective
Formal Qualification Testing
Feedback from Field Use

Figure 6: Many Quality Touch Points in Agile Development [Hayes 2014]

One core element of Agile software development is to “instrument as much as possible”[19] throughout the development process to support test, documentation, analytics, and situational awareness. With the automated development and test environment instrumented appropriately, Agile teams can offer the program office near-constant visibility into most elements of development. As discussed in Hayes, a wide variety of metrics are used by the Agile team to monitor the state of requirements and code [Hayes 2014]. Program offices and contractor/developer teams may choose the level of granularity of this data that program office teams would like to see reported on a regular basis.

Additionally, Agile teams generally perform a customer and/or user demonstration (sprint demo) at the completion of each development iteration. These customer demonstrations review the “definition of done” agreed to at the start of the iteration, and demonstrate that all the items assigned to the iteration have been completed. Thus, the program office or its representative (the product

18 Quote from respondent interview

19 Ibid.

owner) has the ability to observe the technical progress of the software on a regular, predictable basis. It is generally not practical for an entire program office or large user group to participate in the demonstration at the end of an iteration, but user/customer representation in the form of the empowered product owner is a critical part of Agile processes.

Multiple interview respondents indicated that Agile software development teams on their programs provided online user accounts for the development team's collaboration tools to program office personnel. This enabled the program office team to monitor progress in near real time.

Measuring Progress and Quality

Metrics on software development progress and quality must of course also be captured in accordance with both statutory and service-specific requirements. Table 3 demonstrates types of core metrics required by both U.S. Air Force (USAF) and U.S. Army acquisition policy.[20]

Table 3: Sample Regulatory/Policy References [Hayes 2014]

USAF Software Core Metrics	Army Regulation (AR) 70-1 Army Acquisition Policy
Software size Software development effort Software development schedule Software defects Software requirements definition and stability Software development staffing Software progress (design, code and testing) Computer resource utilization	*Section 7-13 Software Metrics: PMs will negotiate a set of software metrics with the software developer to affect the necessary discipline in the software development process and to assess the maturity of the software product. At a minimum, the metrics should address* • schedule and progress regarding work completion • growth and stability regarding delivery of the required capability • funding and personnel resources regarding the work to be performed • product quality regarding delivered products to meet the user's need without failure, as reflected in associated requirements documents • software development performance regarding the capabilities to meet documented program requirements • technical adequacy regarding software reuse, programming languages, and use of standard data elements

As Hayes wrote, "These requirements are written to allow flexibility in implementation—*to fit the scope and nature of the contract at hand"* [Hayes 2014, emphasis added]. In other words, the guidance requires that metrics be provided to characterize the progress of the program in a way that makes sense—makes the metrics useful—given the environment of the program.

For example, different development organizations will offer up different representations of "software size:" SLOC,[21] ESLOC,[22] and function points may all be familiar language to a contracting officer. Estimated software size under a waterfall model will drive schedule and cost estimates, and volume of code produced is used as an indicator of progress.

[20] The GAO also discusses the expectation and method for use of Earned Value Management, in the context of Agile software development programs [GAO 2012].

[21] Source Lines of Code

[22] Equivalent Source Lines of Code

In an Agile project, however, software sizing is not used as a basis for schedule and effort estimation and is considerably less important as a measure of progress; the preference is to time-box iterations with a fixed schedule and cost, and use customer priority and relative *effort* sizing techniques to maximize customer value delivered in each iteration [Anderson 2010, Hayes 2014].[23] Different Agile teams will report on software size in different ways, depending on the complexity of the project, their estimation techniques, and other preferences.

While it is beyond the scope of this technical note to delve deeply into the measurement of Agile projects, it is important for a contracting officer to understand that thanks to an emphasis on automation and frequent delivery, Agile projects will have rich data available to provide insight into both progress and quality of software. When working with a program office team to identify the necessary metrics for required CDRLs,[24] question what metrics are available via the contractor's existing systems and practices to characterize the software to satisfy the intent of service-specific policy or regulation and to help the program manager monitor compliance with the contract terms. As noted previously, cadence of software development may not marry up nicely to a monthly reporting window. Ensure that reporting timeframes (and associated lag, if any) and data access methods are defined in the contract as agreed on by the program office and the contractor. If end-of-sprint and end-of-release reporting conducted by the developer provide all the necessary data to characterize the project, additional documentation may not be required.

A Contracting Officer Should

- Ensure that the program office's commitment to providing representation and timely response is reflected in the contract. This includes
  - providing a representative (typically called a product owner) with authority to prioritize among requirements consistent with the program vision
  - participating in increment planning (establishing requirements for the increment)
  - participating in sprint/iteration and release planning
  - participating in end-of-increment demonstrations
- Ask the program office team to determine what manner of insight and oversight needs to be reflected in the contract—does this include access to collaboration tools used by the development team, copies of completion memos at the end of each iteration, burn down,[25] or other data?
- Ensure that access to automated collaboration tools and reporting/tracking environments, as agreed to by program office and contractor, is reflected in the contract.
- Work with the program manager to ensure that *appropriate* metrics for the specific development effort are included in a measurement plan in the CDRL list. Do metrics/data available within the contractor's automated tool suite, as part of the developer's standard process, meet the objectives for monitoring progress and quality?

[23] Hayes provides more discussion on relative estimation [Hayes 2014].

[24] Contract Data Requirements List

[25] A graphical depiction of a team's progress toward completing their workload, updated daily [Hayes 2014].

5.3 “Requirements Are Too Nebulous with Agile, and That’s too Risky”

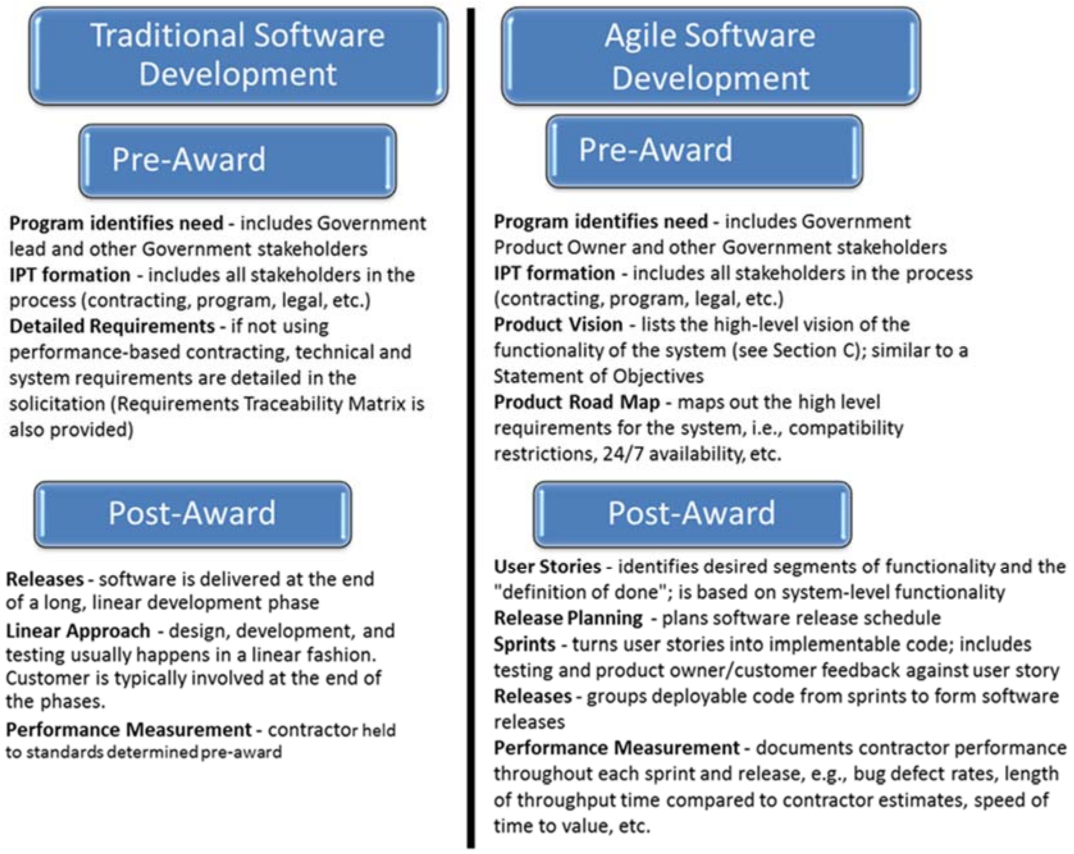
Under Agile methods, we know that requirements are incrementally evolved and refined from a high-level system vision as the system is developed and more is learned about the requirements. It is understandable that this departure from the traditional approach (that of nailing down the requirements to a great level of specificity up front) may at first lead a contracting officer to assume that the lack of specificity introduces risk to into the development cycle. “Traditional acquisition practice relies on certainty in requirements.… Uncertainty is unavoidable but seen as a weakness to be eliminated” [Campbell 2010]. If we don’t specify exactly what we are buying, how will we know when we get it?

We have already discussed that requirements can and *do* change frequently during the course of system development. By expecting change within the scope of the system versus emphasizing rigid up-front specification, DoD programs that use Agile can substantially reduce the overhead associated with complex formal change control processes. These change control processes often result in significant negotiation (which may be contentious) and can create delays in development. The Agile approach of incrementally evolving and refining requirements prevents resource investment dedicated to “developing software for requirements that are not ultimately needed. It also recognizes that money may be better spent for requirements that were *not* recognized at the beginning.” In other words, “*agile principles can protect a client from things they may not know*” [Arbogast 2012, emphasis added].

Steven Van Roekel, the U.S. chief information officer, announced in August 2014 the development of the *TechFAR Handbook*, a publication designed to support federal IT acquisitions in leveraging Agile methods while assuring compliance with the FAR.[26] (The TechFAR is in publicly released draft as of this writing, and subject to changes and enhancements.) While the TechFAR is intended specifically to support IT acquisitions and the delivery of digital services, it clearly demonstrates that articulating detailed software requirements on an incremental basis is not inconsistent with acquisition regulation.

Figure 7 is from the *TechFAR Handbook* and demonstrates the approach of specifying a high-level vision or roadmap before contract award, and evolving requirements over the course of the development effort. These approaches are not exclusive to IT acquisitions, and can be leveraged effectively in other software domains as well.

[26] https://cio.gov/delivering-customer-focused-government-smarter/



Traditional Software Development

Pre-Award

Program identifies need - includes Government lead and other Government stakeholders
IPT formation - includes all stakeholders in the process (contracting, program, legal, etc.)
Detailed Requirements - if not using performance-based contracting, technical and system requirements are detailed in the solicitation (Requirements Traceability Matrix is also provided)

Post-Award

Releases - software is delivered at the end of a long, linear development phase
Linear Approach - design, development, and testing usually happens in a linear fashion. Customer is typically involved at the end of the phases.
Performance Measurement - contractor held to standards determined pre-award

Agile Software Development

Pre-Award

Program identifies need - includes Government Product Owner and other Government stakeholders
IPT formation - includes all stakeholders in the process (contracting, program, legal, etc.)
Product Vision - lists the high-level vision of the functionality of the system (see Section C); similar to a Statement of Objectives
Product Road Map - maps out the high level requirements for the system, i.e., compatibility restrictions, 24/7 availability, etc.

Post-Award

User Stories - identifies desired segments of functionality and the "definition of done"; is based on system-level functionality
Release Planning - plans software release schedule
Sprints - turns user stories into implementable code; includes testing and product owner/customer feedback against user story
Releases - groups deployable code from sprints to form software releases
Performance Measurement - documents contractor performance throughout each sprint and release, e.g., bug defect rates, length of throughput time compared to contractor estimates, speed of time to value, etc.

Figure 7: Requirements and Approach, Traditional Versus Agile Software Development [U.S. CIO 2014]

The FAR (Part 39.103), Modular Contracting, specifically indicates that modular contracting techniques can be used to "reduce program risk and to incentivize contractor performance while meeting the Government's need for timely access to rapidly changing technology" when acquiring IT systems [FAR 2015]. While Part 39 again specifically governs the acquisition of IT systems, the guidance for employing modular contracting techniques is very consistent with Agile principles:

(b) When using modular contracting, an acquisition of a system of information technology may be divided into several smaller acquisition increments that—

> *(1) Are easier to manage individually than would be possible in one comprehensive acquisition;*
>
> *(2) Address complex information technology objectives incrementally in order to enhance 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 is not dependent 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; and*

(5) *Reduce risk of potential adverse consequences on the overall project by isolating and avoiding custom-designed components of the system* [FAR 2015].

Note particularly the emphasis on the evolution of requirements as the system evolves.

A Contracting Officer Should

- Recognize that finely detailed advance requirements specifications are incongruous with Agile approaches. If a program team wishes to place a finely detailed requirement specification on contract for an Agile project, engage in discussions about the appropriateness of the methodology and its emphasis on requirements evolution. (Agile approaches are not appropriate for all software projects—address this disconnect *before* developing the contract further.)
- Ensure that when a program pursues incremental delivery approaches, a clear high-level vision (e.g., a concept of operations, or CONOPS) is placed on contract—one that describes Agile concepts and principles and desired outcomes but does not specifically mandate Agile. (Remember, the program office cannot tell the vendor specifically how to execute the technical work.)
- Ensure that the government's commitment to providing a user representative, empowered to prioritize among system requirements within the scope of the product vision, is documented in the contract. Agile projects cannot succeed without effective prioritization of requirements on an ongoing basis.

5.4 "Frequent Iterations Create Significant Additional Contracting Overhead"

The idea of issuing a new task order or statement of objectives/statement of work (SOO/SOW) for software on a frequent basis (such as for every release, or for a specific timeframe) has led many in the contracting community to object that Agile methods will result in significant overhead associated with the higher number of task orders.[27] However, consider how requirements and requirements change are addressed in waterfall development models:

In a waterfall-based approach on DoD programs (sometimes called "document-centric"):

assigned stakeholders create formal documents as the expression of "what to build" that must be approved prior to use in further design and implementation

verification and validation of the requirements occurs as a (generally) complete set prior to substantive design and implementation

changing the requirements, regardless of source, is a time-consuming and expensive process designed to aggressively control change [Nidiffer 2014].

[27] Interview respondents

The TechFAR also addresses community concerns about administrative overhead on Agile projects: “While the process is highly interactive, the overall amount of work is not greater—just applied differently—to produce quicker results. As the Agile process matures, the amount of administration work should be less” [U.S. CIO 2014].

The key to minimizing unnecessary overhead is to establish a viable contract structure and governance up front. A contracting officer can then work with a program manager to develop templates that bound the scope of a task order, while allowing the program office/contractor team flexibility to operate within that scope. As new task orders are released, the templates can be used to enforce the agreed-on boundaries (e.g., cost, schedule, number of iterations, high-level requirements).

One program contracting for software development under an existing multiple award contract (MAC) indefinite delivery/indefinite quantity (IDIQ) vehicle reported that for ease of contracting and reporting, the program team, software developer, and contracting officer developed a series of templates based on relative sizing of work packages referred to as “epics.” This relative sizing is often referred to as “T-shirt sizing;” the parties agreed to characteristics that defined an epic as small, medium, or large. In development terms, that sizing referred to the number of story points the development team had assigned to the elements of the epic. In contracting terms, the size of the epic represented certain schedule and financial thresholds. When the program office and the software developer prioritized an epic and agreed upon the relative sizing based on discussion of the requirements to be met in the epic, then the contracting officer would issue a new task order for that epic, using the template of the appropriate size. The up-front work that the parties put in to developing the initial template gave the program office and the developer the flexibility to systematically prioritize software requirements and execute the work in small iterations. It also gave the contracting officer predictability and the ability to make straightforward determinations of compliance, even as the developer and the program office worked together to evolve the requirements. The template structure minimizes the overhead associated with the creation of multiple task orders.

A different DoD program leverages a MAC IDIQ vehicle to contract for Agile software development in a different way. This organization contracts on a time and materials (T&M) basis for “software support of [System X].” (System X is a fielded system rather than a new start program.) From a contracting perspective, there is no additional overhead created by the frequent software iterations. The government does assume some risk associated with the delivery of the contractor’s “best effort.” That risk is mitigated by a highly collaborative engagement model between the contractor, the government program office, and the government “customers”—the commands that use the system. In this case, the program office and the contractor collaborate extensively with the using commands and hold regular forums for the customers to identify and prioritize software requirements. The prioritized requirements then form the product backlog for the system, and the contractor works from the backlog to develop a product roadmap, identify target releases, and ultimately develop the sprint backlog that defines the requirements for each sprint. The intensive collaboration is facilitated by a team charter for the government participants, describing the process and the commitments required of all parties. In this case, a high level of trust (fostered by an operating model that has continuously delivered high-quality software with transparency in the development process) supports a reduction in administrative overhead. The program has operated successfully under this model for several years.

A Contracting Officer Should

- Work with the program manager to identify opportunities to streamline future task orders/awards by developing templates appropriate to the scope of the contract.

5.5 "Agile Development Projects Are Not Aligned with Required Technical Reviews Under DoDI 5000.02, so They Can't Be Done"

Technical reviews and evaluations are an important part of the DoD acquisition process. Agile software is developed at a more rapid cadence than seen under traditional waterfall-based development models, as we showed in Figure 1. However, as previously discussed, this means that in Agile software development requirements, architecture, and design specifications are not developed sequentially and "baked" before code is written. Architecture, design, and test documentation are updated as the software is developed. This means that when contracting for Agile software development, some flexibility is required for addressing technical reviews and evaluations in an iterative or progressive manner. This typically means that an incremental, or iteration-based technical review will occur on a regular cadence as part of the software development plan, and the results may then be "rolled up" into a traditional PDR/CDR (or other technical review).

Unlike a traditional PDR, CDR, at these incremental reviews

> *All documentation will not appear at the same level of maturity:*
>
> - *Some documentation will still be in draft condition (such as design documents for the overall system that support requirements that have been allocated to some future increment).*
> - *Some documents will be partially completed (such as those supporting requirements in upcoming increments that are dependent upon the implementation of earlier capabilities).*
> - *Some will be fully complete (perhaps for requirements that are being implemented in the current increment)* [Lapham 2014].

As we discussed in Section 4, the 2015 DoDI 5000.02 guidance specifically allows for tailoring of the acquisition process to achieve more efficient outcomes, which includes technical reviews. A contracting officer should *expect* to see iterative or incremental technical reviews in the plan when Agile methods are proposed, rather than a single PDR/CDR, etc. The contracting officer should verify that the plan being placed on contract clearly illustrates the frequency and tailoring of technical reviews and describes the content of the iterative or incremental reviews, and how these "roll up" into the overall system engineering review process, if applicable.

To illustrate what an iterative/incremental technical review process might look like in a plan, we can use an example from a U.S. Marine Corps Agile pilot program. The program team reported great success employing an iterative model for technical reviews, as illustrated in Figure 8.

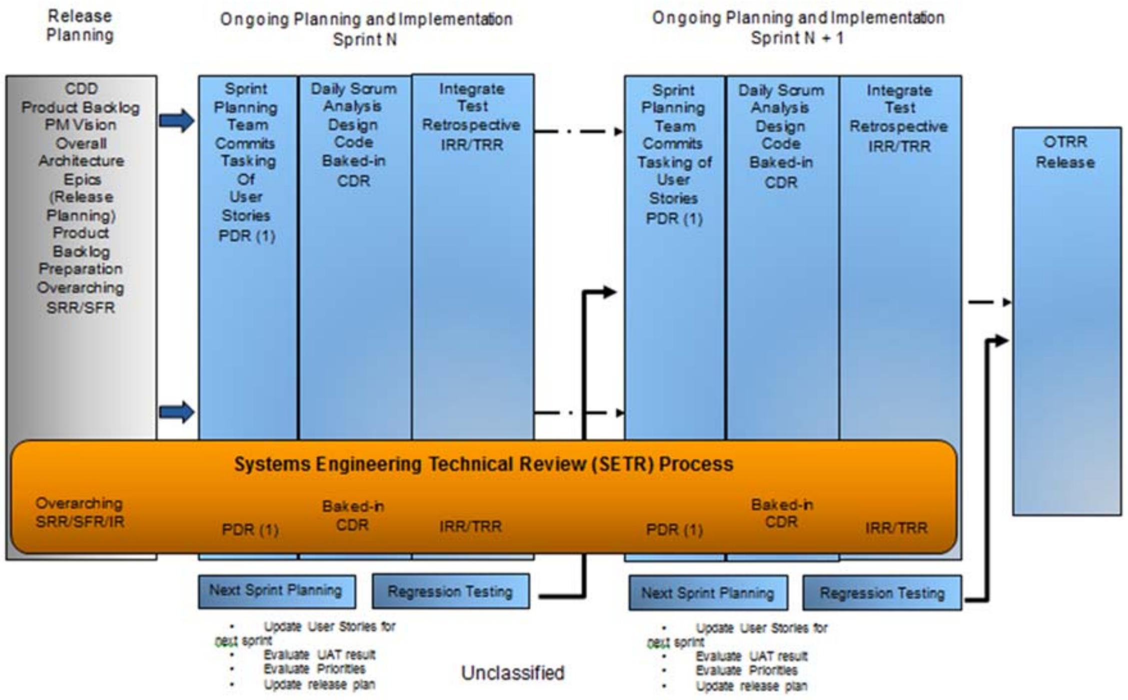

Release Planning
Ongoing Planning and Implementation Sprint N
Ongoing Planning and Implementation Sprint N + 1
CDD Product Backlog PM Vision Overall Architecture Epics (Release Planning) Product Backlog Preparation Overarching SRR/SFR
Sprint Planning Team Commits Tasking Of User Stories PDR (1)
Daily Scrum Analysis Design Code Baked-in CDR
Integrate Test Retrospective IRR/TRR
Sprint Planning Team Commits Tasking of User Stories PDR (1)
Daily Scrum Analysis Design Code Baked-in CDR
Integrate Test Retrospective IRR/TRR
OTRR Release
Systems Engineering Technical Review (SETR) Process
Overarching SRR/SFR/IR
PDR (1)
Baked-in CDR
IRR/TRR
PDR (1)
Baked-in CDR
IRR/TRR
Next Sprint Planning
Regression Testing
Next Sprint Planning
Regression Testing
Update User Stories for next sprint
Evaluate UAT result
Evaluate Priorities
Update release plan
Unclassified
Update User Stories for next sprint
Evaluate UAT result
Evaluate Priorities
Update release plan

Figure 8: Marine Corps (MC)-Agile Increment 1[28, 29]

The figure demonstrates the overarching technical review process, and how incremental technical reviews are incorporated into each iteration (or "sprint"). (The Marine Corps Agile pilot process is described in great detail in *Agile Software Teams: How They Engage with Systems Engineering on DoD Acquisition Programs* [Wrubel 2014].) In the case of this pilot project, the program manager further documented the incremental reviews and mapped them to the corresponding "traditional" reviews for reference, as shown in Table 4.

Table 4: Agile Reviews and Traditional Reviews[30]

Technical Reviews in the Agile Process	**Traditional Analogous Systems Engineering Technical Review**
Initial Release Planning Review (IRPR) • Focused on Initial Release and corresponding sprints Infrastructure Review (IR) • Proposed Hardware Infrastructure • Estimated Virtualized Resource Pool	Systems Requirements Review 2 (SRR2) Systems Functional Review (SFR) (Incremental PDRs will be conducted at the sprint levels)
Release Planning Reviews (RPR) • Oversight will be delegated to the Agile Review Board • Focused on follow-on release and corresponding sprints	Systems Functional Review (SFR) • Subsequent release SFR

[28] Graver, Carmen & Greeley, Les. *United States Marine Corps Agile Pilot Program Lessons Learned (MC-Agile)*. Briefing. February 2013. Unpublished.

[29] Acronyms are expanded in Table 4.

[30] Graver, Carmen & Greeley, Les. *United States Marine Corps Agile Pilot Program Lessons Learned (MC-Agile)*. Briefing. February 2013. Unpublished.

Technical Reviews in the Agile Process	**Traditional Analogous Systems Engineering Technical Review**
Sprint Planning/Reviews[31]	Sprint Preliminary Design Review (S)PDR** **Incrementally conducted with each sprint
Daily Build/Test/Integration Sprint Demonstration Review*** *** Completed products are demonstrated to the product owner	Critical Design Review (CDR) N/A
Release Demonstration	Integration Readiness Review (IRR) Test Readiness Review (TRR)
Sprint and Release Retrospectives • Assessment opportunity to determine what went well and what did not for sprint/releases	Continuous Process Improvement (CPI)
Systems Verification Review (SVR)	Systems Verification Review (SVR)
Operational Test Readiness Review (OTRR)	Operational Test Readiness Review (OTRR)

The incremental reviews are not large, multi-day meetings that many associate with typical PDR and CDR activities on large programs. Rather, the reviews are short meetings that involve the key stakeholders. Small increments are the focus, rather than the entire system, allowing participants to focus carefully on the defined scope.

As the system evolves and more increments are completed, critical documentation is updated over time: "Requirements and design allocated to future iterations should not be expected to be fully matured during early iterations" [Lapham 2014]. Design and architecture documents, the Test and Evaluation Master Plan (TEMP), System Engineering Management Plan (SEMP), etc., will all see periodic updates as the requirements for each iteration are fixed, developed, and tested. Different sections of the TEMP, for example, will be completed at the various reviews at various levels of maturity, as different system requirements are realized. As stakeholders participate in technical reviews, they will notice the evolution of the documentation as the software development progresses.

We have said that Agile provides opportunities to "fail fast" and address problem areas early and within smaller boundaries than "big-bang" development. The implementation of incremental or progressive reviews enables just that—any issues identified at the time of the review can be prioritized and addressed within upcoming iterations. Required technical reviews such as PDR and CDR then present fewer surprises and challenges, as the stakeholders have been engaged in reviewing the incremental progress all along.

A Contracting Officer Should

- Verify that the contractor's Software Development Plan (SDP) addresses *incremental or progressive technical reviews*, including how documentation is updated and how the incremental reviews support system engineering activities and program milestone reviews.

[31] Sprint planning meetings occur at the *beginning* of the sprint for purposes of defining "what done means" for that sprint. Sprint reviews occur at the end of the sprint to assess the progress against the agreed-on parameters of the sprint: "Did it get done?"

- Ensure that reports or documentation called out in the SDP to support these reviews are addressed as CDRLs or documented within collaboration tools to which the program office has access (as previously discussed).
- Ensure that the contract documents the expected participation of the government team in incremental reviews and demonstrations.

The next section discusses using various contracting approaches to support the acquisition of software developed using Agile methods.

6 Contracting Approaches

There is no universal "right way" to approach contracting for software developed under Agile principles: each acquisition is unique. Time constraints, mission needs, and the size of the acquisition (in terms of budget) inform and constrain the types of contract vehicles appropriate for any particular program. Additionally, policy about the contracting process itself, in addition to the business/mission needs, may introduce additional constraints. Arbogast posits that there are three "general areas of concern" for contracting professionals [Arbogast 2012]:

- risk and exposure (liability)
- flexibility to allow for change
- clarity regarding obligations, deliverables, and expectations

Ultimately, however, the contract needs to support the delivery of *deployable* software at defined increments/intervals, rather than incentivizing "big-bang" efforts or the production of compliance documents. Obviously, a program manager and contract officer have the most flexibility when they work together to choose the type of contract vehicle to employ. Both perspectives need to be addressed in the contract. The program manager wants deployable software and the contracting officer wants that also, while protecting the government from "the ramifications of a breakdown of trust and collaboration—and other problems—when framing the contract" [Arbogast 2012]. However, Agile software development can be successfully executed regardless of constraints on contract type or contracting environment. The authors do not endeavor to guide the reader through the rules governing the available contract types for any specific situation, but rather to demonstrate that Agile principles can be supported and applied under any contract type, so long as the contracting "business problem" is properly framed and addressed in the contract.

A new report, *Innovative Contracting Case Studies*, released in August 2014, is considered 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" [OSTP 2014]. This report helps to provide insights and ideas that have been tried by different government organizations. Companion reports include the *U.S. Digital Services Playbook* and *The Tech FAR Handbook*, which provide more examples and ideas about flexibility in the Federal Acquisition Regulation [U.S. CIO 2014]. The Office of Management and Budget in 2012 published guidance on contacting for modular development [OMB 2012]. These various reports help to encourage government organizations to take new approaches. Of utmost importance is that the contract provides incentive for incremental delivery of working software.

6.1 Contract Types

The FAR, Part 16, defines two broad contract categories: fixed-price contracts (Subpart 16.2) and cost-reimbursement contracts (Subpart 16.3). The contract type is selected and negotiated. In between the two end-point contract types are various approaches to incentive type contracts (Subpart 16.4). The Defense Acquisition University has created a *Comparison of Major Contract Types*, which will help to quickly show the different major contract types. This summary contains other helpful insights on contract category characteristics. We have reproduced these summary materials in Appendix C as material to which the reader may refer when considering the use of Agile software methods.

The fixed-price type contract is the U.S. government's preferred approach, according to the literature and the interviews. It seems counterintuitive at first that the fixed-price contract type could be greater risk for the government. One useful and informative work on contracting in the context of using Agile software development approaches is Opelt's *Agile Contracts: Creating and Managing Successful Projects with Scrum* [Opelt 2013]. (While the book is written based on experience in the European contracting environment, it provides highly relevant ideas and approaches that are readily adaptable to contracting actions under the FAR.)

In *Agile Contracts,* Opelt developed a structure for thinking about contract types relative to the variability of the work scope or the price of the work [Opelt 2013]. Figure 9 shows the grid and contract type that is considered appropriate of the quadrant. The horizontal axis represents the continuum of scope of project requirements from variable/flexible on the left, to a rigid, highly fixed scope on the right. The vertical axis represents the price or budget requirement. The quadrants labeled I through IV represent different combinations of the price/budget and scope variability. A contract to procure commercial hardware would fall into quadrant I. A contract that imposed a fixed budget and time box, while allowing for iterative requirements discovery (as in Agile methods) would fall into quadrant II. (Quadrant III represents contracts such as those for temporary consulting services. Quadrant IV is consistent with projects such as a known hardware design, and variability or unknowns in the manufacturing process.)

Note that while time and material contract type is shown, any variation of cost-plus contract type could be structured. The same range of variation goes for fixed price contract types. So the structure below (Figure 9) will work for the U.S. FAR and DFAR regulations.

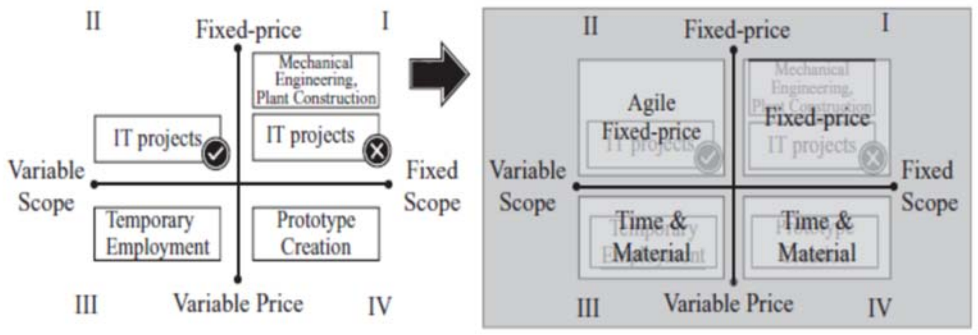


Figure 9: Contract Type Applications

6.2 Cost-Reimbursement Approach for Agile Contracts

Many programs that use Agile software development approaches use cost-reimbursement category type contracts. This category of contracts provides the most flexibility for variation in work performed, within the bounds of the contract work scope, and associated legal limitations. For the Agile software development work scope, remember the business goal is supporting the delivery of *deployable* software that meets the business or operational needs.

Where practical, "Variations of time and materials (T&M) make for good agile-project pricing models: simple, straightforward" [Arbogast 2012, p. 25]. The concerns related to T&M contracts

> *are ameliorated in an agile approach with a usable system each iteration – progress measure in terms of usable software features, high transparency, and termination that can occur at the end of any iteration* [Arbogast 2012, p 26].

Time and materials is just one type in this category of contract. FAR Subpart 16.301-1 describes this:

> *Cost-reimbursement types of contracts provide for payment of allowable incurred costs, to the extent prescribed in the contract. These contracts establish an estimate of total cost for the purpose of obligating funds and establishing a ceiling that the contractor may not exceed (except at its own risk) without the approval of the contracting officer.*

The conditions for application of cost-reimbursement contracts is outlined in FAR Subpart 16.301-2:

> *(a) The contracting officer shall use cost-reimbursement contracts only when—*
>
> *(1) Circumstances do not allow the agency to define its requirements sufficiently to allow for a fixed-price type contract (see 7.105); or*
>
> *(2) Uncertainties involved in contract performance do not permit costs to be estimated with sufficient accuracy to use any type of fixed-price contract.*

Further, the use of a cost-reimbursement contract requires documented rationale and a "written acquisition plan that is approved by at least one level above the contracting officer."[32] Other limitations include the contractor's account system and adequate government capability to manage the cost-reimbursement contract.

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 government capability to manage and oversee the contracted work. Effective government 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 system needs is built into the statement of work or objectives that accompanies the contractual funding constraint.

A number of the organizations that provided insight into their approach to contracting for working software using Agile software development approaches used cost-reimbursable contracts. These

[32] FAR 16.301-2 Application and FAR 16.301-3 Limitations.

contracts were driven more by the uncertainty of the requirements and the prioritizing of the requirements based on changeable operational need. Some of these organizations established fixed work cycles and software release cycles, with constraints on amount of work scheduled for the fixed work cycles. These constraints helped the government to prioritize work. In at least one contract situation, the government contracting officer had become “smart on Agile.”[33] “In deciding how to contract, there is no replacement for knowledge about HOW the work is to be done.”[34]

Some organizations we interviewed described that the cost-reimbursement type contracts allowed faster delivery of working software. Instead of going through the separate phase of documenting requirements for more fixed-price type contracts, the program office was able to work more collaboratively and have, in one case, “blended teams” of customers and Agile developers.

6.3 Firm-Fixed-Price Approach for Agile Contracts

Firm-fixed-price (FFP) contracting ideas are popular in acquisitions: provide a detailed specification of the requirements, and then the winning contractor is obligated to meet the specified requirement at an agreed-on price. In theory, most of the risk under this kind of arrangement is shifted to the contractor: “This contract type places upon the contractor maximum risk and full responsibility for all costs and resulting profit or loss. It provides maximum incentive for the contractor to control costs and perform effectively and imposes a minimum administrative burden upon the contracting parties” [FAR 2015, 16.202-1]. Contracting officers do have leeway to include award-fee incentives based on factors “other than cost,” which include the achievement of specific performance characteristics or schedule reductions, but the fixed-price is generally applied to a firm requirements specification set at the beginning of the program. Change management processes are very rigorous and require contract modifications as previously noted.

With the changing nature of software requirements, a traditional FFP approach can quickly have undesirable unintended outcomes—all changes to the requirements are subject to management overhead of negotiating and securing the changes, and increased cost as they introduce deviations from the original plan. Nailing down every element of schedule, scope, and cost up front creates the opportunity for even minor perturbations in the requirements to ripple throughout the program. In other words, implementing FFP vehicles with detailed software requirements specified up-front can actually put the government at *increased* risk of cost and schedule overruns.

Fixed-cost approaches are not inherently incompatible with Agile, however.

[33] Interview with government program office.

[34] Ibid.

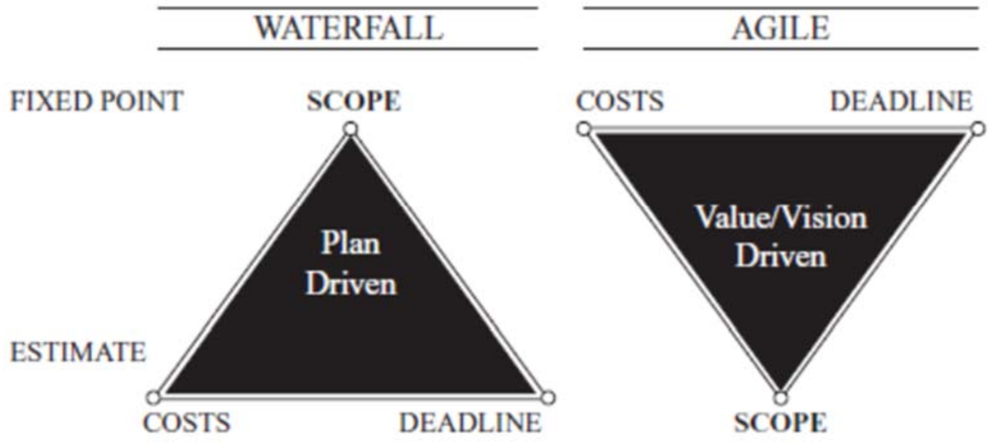


Figure 10: Value-Driven Projects [Opelt 2013]

A traditional waterfall approach nails down requirements and uses those to drive cost and schedule estimates. In an Agile approach, cost and schedule are generally fixed parameters, and these drive the scope of development within the construct of the product vision. The vision bounds the requirements, and scope is determined by developing requirements as prioritized by the customer within the available capacity of the development team. Under the traditional FFP model, once the cost and schedule parameters are agreed to, all three dimensions of the triangle are fixed. Any requirements change breaks the triangle. Opelt suggests an "Agile fixed-price" approach: "The main characteristic of a shift to the agile paradigm is that the scope of an IT project is in contrast to the classic waterfall model, no longer fixed in detail from the start" [Opelt 2013] (see Figure 10).

This Agile fixed-price contracting approach still expects a definition of scope, but the boundaries are established as "values and vision for the project" [Opelt 2013]. A high-level product vision is analogous to the preparation of a statement of objectives (SOO), as undertaken during performance-based contracting in accordance with the FAR [U.S. CIO 2014]. The details of the contract in the Agile fixed-price model come as a result of the interaction between the business/product owner (needs) and development team(s). The contractual arrangement is used to define the appropriate interactions and approved approaches to tradeoff the business needs with the cost (budget) constraints. Opelt focuses on collaborating to come to an understanding of the balance of the risk between the contractor and the program office. (A true firm-fixed-price vehicle would leave the contractor with the entire risk share.)

Some important definitions are helpful in understanding Opelt's approach to arriving at Agile fixed-price contracts:

- *Indicative fixed-price range.*[35] *Before the start of the checkpoint phase, a provisional price is estimated, based on an unformulated rough scope of the subject matter (vision, themes, and epics). This indicative fixed-price range is not yet contractually binding.*

[35] Analogous in DoD settings to Target Cost and Profit, Target Cost and Target Fee, Ceiling Price, Maximum Fee, Minimum Fee. (See Appendix C.)

- *Riskshare.*[36] *The riskshare describes to what extent (percentage) the costs incurred by the supplier will be charged to the customer on failure of the checkpoint phase or when the maximum price range is exceeded. This percentage may, however, vary for the checkpoint phase and the overall project.*
- *Checkpoint phase.*[37] *A period of x sprints or a performance scope of y story points is agreed upon as the test phase of cooperation. The final milestone is a checkpoint whereby the customer and supplier can enter into implementation of the overall project (or maybe not).*
- *Exit points.*[38] *These are clearly defined points in time where the parties may terminate the project in a controlled manner* [Opelt 2013].

Opelt outlines six steps to collaboratively arrive at a contract structure for a fixed-price Agile approach:

1. *Define the contract at the level of product or project vision, topics, and epics from the perspective of the user (i.e., to a level at which the contract is complete but not yet described in detail.)*
2. *Specify the details of an epic, down to the level of the user stories.*
3. *In a joint workshop, an overall estimate is made of the effort required starting from a set of reference user stories from step 2, including the risks of implementation and business value for these user stories.*
4. *Another step is the fixing of the riskshare exit points, and checkpoint phase (also with riskshare for exactly this phase). Neither side is obliged to buy a pig in a poke.*
5. *Agree on the scope and expense management process and, of course, the governance of the decision-making process.*
6. *Agree on a motivational model and a cooperative model, consider a bonus system* [Opelt 2013].

The six steps are presented graphically in Opelt's Figure 3.2, shown below in our Figure 11.

[36] Analogous DoD contract types might include Cost-Sharing, Fixed-Price Incentive, Cost-Plus Incentive Fee. (See Appendix C.)

[37] Follow-on contracts based on performance experience: Typical DoD examples might include: Base Contract with performance expectations and measures (Cost-Plus Incentive Fee). (See Appendix C.)

[38] A typical example would be a Base Contract, with defined Option Periods. If option is not exercised, the contract ends. (See Appendix C.)

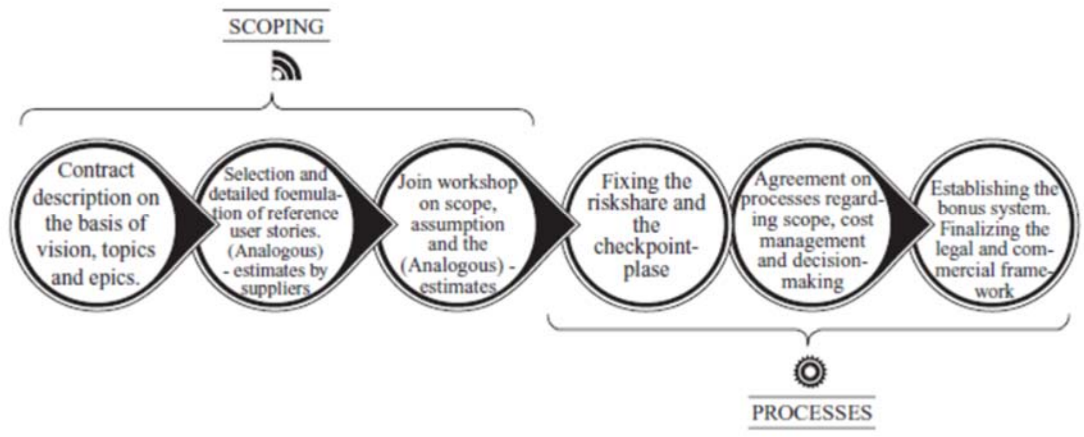

SCOPING
Contract description on the basis of vision, topics and epics.
Selection and detailed foemulation of reference user stories. (Analogous) - estimates by suppliers
Join workshop on scope, assumption and the (Analogous) - estimates
Fixing the riskshare and the checkpoint-plase
Agreement on processes regarding scope, cost management and decision-making
Establishing the bonus system. Finalizing the legal and commercial framework
PROCESSES

Figure 11: Scoping and Process Definition for an Agile Fixed-Price Contract [Opelt 2013]

The approach of fixing costs and letting scope and schedule be variable is an established defense acquisition approach known as "cost as an independent variable" (CAIV). This concept first appeared in DoD Regulation 5000.2-R, Part 3, March 15, 1996. CAIV "is an inherent part of Agile, which starts out with a high-level estimate that can be, and is, refined as the program progresses. Agile allows the developers to provide an incremental total cost estimate at a detailed level as the iterations are performed" [Lapham 2010]. Quadrant II in Figure 9 is consistent with this approach.

As discussed previously, the 2015 DoDI 5000.02 provides example program models to show that variation in program implementation will occur, based on capability being developed and delivered. The Analysis of Alternatives (AoA) and the affordability analysis continue to expect that cost (affordability)

> *constraints for procurement and sustainment will be derived early in program planning processes. These constraints will be used to ensure capability requirements prioritization and cost tradeoffs occur as early as possible and throughout the program's life cycle* [DoD 2015].

The concept of affordability continues to drive tradeoffs:

> *Early in a program, affordability goals are set to inform capability requirements and major design tradeoffs needed to define the product being acquired. Once requirements and the product definition are firm (prior to Milestone B), affordability caps are established to provide fixed cost requirements that are functionally equivalent to Key Performance Parameters* [DoD 2015].

One contracting officer who responded to our interview reported a discussion among team members regarding FFP vehicles noted

> *The benefit is that Agile/Scrum prefers stable teams, which implies a stable burn rate, which works well in FFP. However, the challenge is locking down very specific requirements and priorities in advance that cause issues. The challenge is that executives and programs that don't understand what they are getting in FFP have a hard time justifying the contracts. The*

> *solutions discussed include contracting for work units instead of defining requirements during the project and creating shorter period contracts. The shorter period contracts mean less risks and can allow the COs [contracting officers] comfort in trying Agile.*[39]

The approach described by this respondent is consistent with applying CAIV principles to develop the shorter period "work unit" contracts as contracting officers gained familiarity with Agile.

Support for incremental software development and delivery is contained in DoD Instruction 5000.02. Enclosure 3, Systems Engineering, contains section 11, Software:

> *A phased software development approach using testable software builds and/or fieldable software increments enables the developers to deliver capability in a series of manageable, intermediate products to gain user acceptance and feedback for the next build or increment, and reduce the overall level of risk* [DoD 2015].

Within the context of the Agile fixed-price contract book, the evolution of detailed understanding is continually evolving to the point of delivered software. Figure 12 shows the increasing level of detail that starts with a "vision" for the solution and continually unfolds to more detail, so that software solution can be built, tested, and delivered.

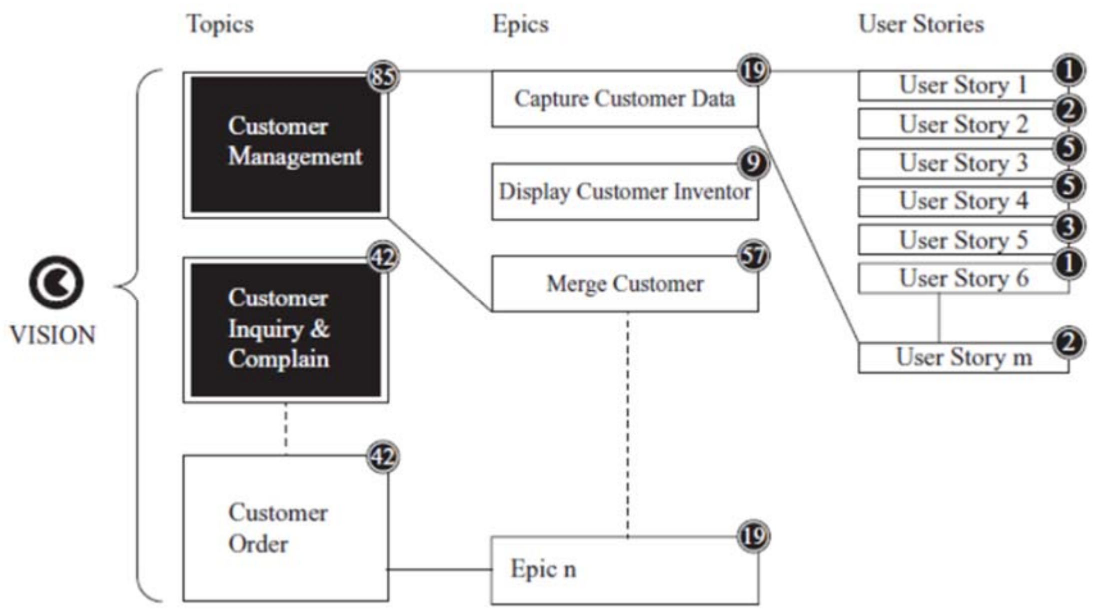


Figure 12: Detailing the Vision [Opelt 2013]

6.4 GAO on Effective Practices for Agile Contracting

The GAO interviewed practitioners on federal Agile projects and identified 32 effective practices in executing Agile projects [GAO 2012]. While the GAO's entire list of practices holds great merit, practices that warrant special note from contracting officers are

[39] Interview respondent

- **Identify measurable outcomes, not outputs, of what you want to achieve using Agile**. An example of this practice is creating a vision statement of project outcomes (such as a decrease in processing time by a specific percent in a set time), rather than outputs (such as the amount of code produced).
- **Negotiate to adjust oversight requirements to a more Agile approach.** This practice notes that teams may be able to adjust oversight requirements by using frequent, tangible demonstrations to gain the trust of reviewers and investors, potentially reducing the need for more formal oversight documents.
- **Make contracts flexible to accommodate your Agile approach.** Contracts requiring waterfall-based artifacts and milestone reviews may not support the frequent changes and product demonstrations or iterations, and may inhibit adoption [GAO 2012].

6.5 Future Work Needed

Government contracting has been addressing the needs of the community through creative and traditional approaches to contracting. Some of these contracting approaches have proven more effective at "deliver[ing] on a timely basis the best value product or service to the customer, while maintaining the public's trust and fulfilling public policy objectives" [FAR 2015, 1.102(a)].

Future work is needed to collect more examples and approaches for effective contracts. This information can help form a body of practice for government organizations to approach software development work, using the Agile values and principles.

7 Summary

Federal and defense acquisition policy increasingly recognizes the promise that Agile or iterative software development methods can bring to programs in terms of timely delivery, improved software quality, and risk reduction. Contracting officers are encouraged by the FAR to support innovative business practices within the bounds of statutory and local agency guidance, but career field education for contracting officers has yet to catch up to provide guidance about effectively adapting contracting to support these development approaches.

This technical note provides a foundation for contracting officers to “hit the ground running” when they collaborate with programs seeking to employ or explore Agile methods. Contracting professionals understandingly tend to adopt conservative approaches in protecting government interests when developing new contracts. By providing a background on Agile and linking it to supporting evidence in the FAR, the DoDI 5000.02, the TechFAR, and other guidance, we hope to demystify Agile and demonstrate that it is in fact an accepted, legal, and encouraged approach to software development. We have provided some guidance to help mitigate common misconceptions about risk associated with Agile software development and provide some specific questions and actions contracting officers can employ, while also emphasizing the support that program office subject matter experts must provide to a contracting officer in the development of an effective contract. While a variety of factors (including program size, the competitive environment, the type of system being acquired, and local agency restrictions) constrain the contracting approach, programs can successfully contract for Agile software development under both fixed-price and cost-reimbursable models.

The authors hope that contracting officers venturing into Agile software efforts will find this a useful primer to enable them to proceed with confidence that typical structures and approaches can be adapted and tailored to produce successful programmatic outcomes well within the bounds of the legal, regulatory, and policy framework in which they operate.

Appendix A: Interview Questions

The notes below served as a template for interviews conducted while researching this technical note, tailored to the experience/background of each participant.

Demographic Data/Contracting Background

- Unique interview identifier (names not collected)
- Role
- Government agency or department
- Experience in Contracting
  - number of years
  - number of contracts
  - types of contracts
  - significance of software in the contracts
  - size of contracts
    - maximum dollar value
    - length of contract

Data/Questions

- Knowledge or training related to Agile software development methods and concepts
- Experience with contracting for work that is expected to use more Agile software development methods and approaches
- Experience with contracting officer representatives overseeing the performance on a contract with agile approaches
- What agency procurement policy (if any) is in place that encourages more agile-oriented approaches in contracts?
  - If policy is in place, are supporting tools/materials provided for the development of contracts?
- Do you or your organization have criteria for selecting the type of contract (FFP,[40] T&M,[41] Award Fee, etc.) vehicle to put in place and what are they? Does the use of Agile influence these criteria?
  - If you make use of award/incentive fees on contracts involving Agile, can you provide an example of your preferred formula/approach, or one that has been successful?
- What specific sections of the FAR have you used to support/justify the contracting approach and the use of Agile?

[40] Firm Fixed-Price

[41] Time & Materials

- How do you or your organization establish a measure of trust with the contractor and program office performance to meet contract?
- A common concern we hear about contracting for Agile is that the requirements lack specificity (by design).
  - How do you ensure that scope of the contract is specific enough to be actionable, but not overly restrictive?
  - How do you use the contract to include the appropriate level of program office/user feedback and collaboration, while avoiding constructive change?
- Progress and Quality
  - Can you provide an example of how software performance/quality is monitored over the course of the contract?
  - Can you discuss how technical progress is monitored over the course of the contract?
  - Can you discuss the technical/documentation deliverables required under Agile contracts?
- Once the contract is awarded, what is your role and level of interaction with the program office?
  - How do you evaluate the level and effectiveness of collaboration between the program office and the contractor?
- How do you or your organization measure success in the contracting process and the final contract? How does this adapt (if at all) when Agile is in play?
- Can you give us an example of the key identified risks on a project using Agile, and tell us how the contract was designed to mitigate those risks?
- Do you envision the guidance in the interim DODI 5000.02 will increase/improve expectations regarding Agile software development?
- What is/are the greatest challenges you have encountered associated specifically with contracting for Agile development?
- Have you had any negative experiences you can share regarding contracting for Agile software development? (Discuss problems/causes, corrective action.)

Appendix B: SEI Publications on Agile Software Development

Previous SEI reports on related Agile topics are available and include

Date	SEI Publication Title
2010	*Considerations for Using Agile in DoD Acquisition* (CMU/SEI 2010-TN-002) [Lapham 2010]
2011	*A Closer Look at 804: A Summary of Considerations for DoD Program Managers* (CMU/SEI-2011-SR-015) [Bellomo 2011]
2011	*Agile Methods: Selected DoD Management and Acquisition Concerns* (CMU/SEI-2011-TN-002) [Lapham 2011]
2014	*Agile Methods and Request for Change (RFC): Observations from DoD Acquisition Programs* (CMU/SEI-2012-023) [Lapham 2014]
2012	*DoD Information Assurance and Agile: Challenges and Recommendations Gathered Through Interviews with Agile Program Managers and DoD Accreditation Reviewers* (CMU/SEI 2012-TN-024) [Bellomo 2012]
2014	*Parallel Worlds: Agile and Waterfall Differences and Similarities* (CMU/SEI-2013-TN-021) [Palmquist 2014]
2014	*Agile Software Teams: How They Engage with Systems Engineering on DoD Acquisition Programs* (CMU/SEI-2014-TN-013) [Wrubel 2014]
2014	*Agile Metrics: Progress Monitoring of Agile Contractors* (CMU/SEI-2013-TN-029) [Hayes 2014]
2014	*Potential Use of Agile Methods in Selected DoD Acquisitions: Requirements Development and Management* (CMU/SEI-2013-TN-006) [Nidiffer 2014]
2014	*Agile Methods in Air Force Sustainment: Status and Outlook* (CMU/SEI-2014-TN-009) [Regan 2014]

In addition, blogs, podcasts, and webinars from the SEI have addressed additional topics related to using Agile software development on acquisition programs. These materials can all be found on the Acquisition Research page of the SEI website.[42]

SEI Blog Entries on Agile in the DoD

- *Readiness and Fit Analysis* (October 8, 2012)
  http://blog.sei.cmu.edu/archives.cfm/author/suzanne-miller
- *Agile Methods: Tools, Techniques, and Practices for the DoD Community* (July 9, 2012)
  http://blog.sei.cmu.edu/post.cfm/agile-methods-tools-techniques-and-practices-for-the-dod-community
- *Using Agile Effectively in DoD Environments* (February 6, 2012)
  http://blog.sei.cmu.edu/archives.cfm/author/mary-ann-lapham

Additional SEI blogs on Agile topics can be found at http://blog.sei.cmu.edu/archives.cfm/category/agilec

Webinar

Agile Research Forum, "Agile Methods: Tools, Techniques, and Practices for the DoD Community," Mary Ann Lapham (August 2012)
http://www.sei.cmu.edu/go/agile-research-forum/

[42] http://www.sei.cmu.edu/acquisition/research

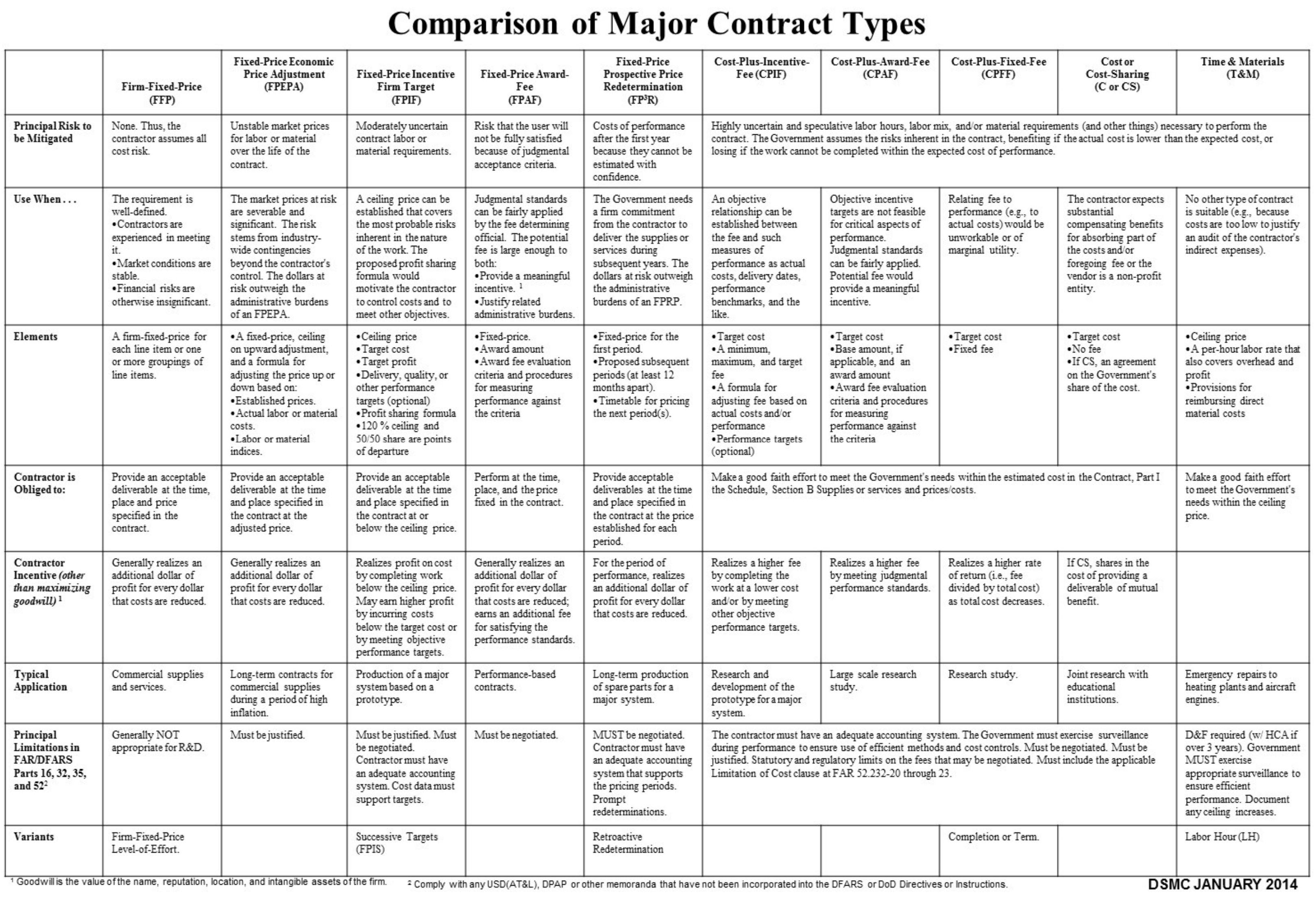
Podcasts

- SEI Agile in the DoD Podcast Series (ongoing series) http://www.sei.cmu.edu/podcasts/agile-in-the-dod/
- *Agile Acquisition* (September 4, 2012) http://www.sei.cmu.edu/podcasts/index.cfm?getRecord=7D03CB1F-9D60-C314-66526F8E8B2864B8&wtPodcast=AgileAcquisition
- *Agile Software Teams: How the Engage with Systems Engineering on Department of Defense Acquisition Programs* (November 24, 2014) http://blog.sei.cmu.edu/post.cfm/agile-software-teams-engage-systems-engineering-328

Appendix C: DAU Guidance on Contract Type Selection

The following two pages are reproduced from the Defense Acquisition University's presentation on contract type selection, *Comparison of Major Contract Types*.

The presentation is available from the Acquisition Community Connection website (https://acc.dau.mil/adl/en-US/214513/file/75692/Comparison%20of%20Major%20 Contract%20Types%20JANUARY%202014%20Final%20Version%20PRINT.ppt).



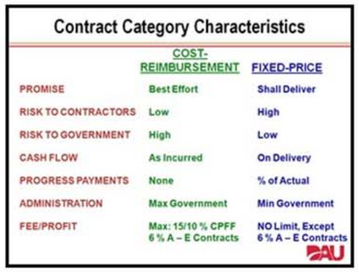
Comparison of Major Contract Types

	Firm-Fixed-Price (FFP)	Fixed-Price Economic Price Adjustment (FPEPA)	Fixed-Price Incentive Firm Target (FPIF)	Fixed-Price Award-Fee (FPAF)	Fixed-Price Prospective Price Redetermination (FP^3R)	Cost-Plus-Incentive-Fee (CPIF)	Cost-Plus-Award-Fee (CPAF)	Cost-Plus-Fixed-Fee (CPFF)	Cost or Cost-Sharing (C or CS)	Time & Materials (T&M)
Principal Risk to be Mitigated	None. Thus, the contractor assumes all cost risk.	Unstable market prices for labor or material over the life of the contract.	Moderately uncertain contract labor or material requirements.	Risk that the user will not be fully satisfied because of judgmental acceptance criteria.	Costs of performance after the first year because they cannot be estimated with confidence.	Highly uncertain and speculative labor hours, labor mix, and/or material requirements (and other things) necessary to perform the contract. The Government assumes the risks inherent in the contract, benefiting if the actual cost is lower than the expected cost, or losing if the work cannot be completed within the expected cost of performance.				
Use When . . .	The requirement is well-defined. •Contractors are experienced in meeting it. •Market conditions are stable. •Financial risks are otherwise insignificant.	The market prices at risk are severable and significant. The risk stems from industry-wide contingencies beyond the contractor's control. The dollars at risk outweigh the administrative burdens of an FPEPA.	A ceiling price can be established that covers the most probable risks inherent in the nature of the work. The proposed profit sharing formula would motivate the contractor to control costs and to meet other objectives.	Judgmental standards can be fairly applied by the fee determining official. The potential fee is large enough to both: •Provide a meaningful incentive. [1] •Justify related administrative burdens.	The Government needs a firm commitment from the contractor to deliver the supplies or services during subsequent years. The dollars at risk outweigh the administrative burdens of an FPRP.	An objective relationship can be established between the fee and such measures of performance as actual costs, delivery dates, performance benchmarks, and the like.	Objective incentive targets are not feasible for critical aspects of performance. Judgmental standards can be fairly applied. Potential fee would provide a meaningful incentive.	Relating fee to performance (e.g., to actual costs) would be unworkable or of marginal utility.	The contractor expects substantial compensating benefits for absorbing part of the costs and/or foregoing fee or the vendor is a non-profit entity.	No other type of contract is suitable (e.g., because costs are too low to justify an audit of the contractor's indirect expenses).
Elements	A firm-fixed-price for each line item or one or more groupings of line items.	•A fixed-price, ceiling on upward adjustment, and a formula for adjusting the price up or down based on: •Established prices. •Actual labor or material costs. •Labor or material indices.	•Ceiling price •Target cost •Target profit •Delivery, quality, or other performance targets (optional) •Profit sharing formula •120 % ceiling and 50/50 share are points of departure	•Fixed-price. •Award amount •Award fee evaluation criteria and procedures for measuring performance against the criteria	•Fixed-price for the first period. •Proposed subsequent periods (at least 12 months apart). •Timetable for pricing the next period(s).	•Target cost •A minimum, maximum, and target fee •A formula for adjusting fee based on actual costs and/or performance •Performance targets (optional)	•Target cost •Base amount, if applicable, and an award amount •Award fee evaluation criteria and procedures for measuring performance against the criteria	•Target cost •Fixed fee	•Target cost •No fee •If CS, an agreement on the Government's share of the cost.	•Ceiling price •A per-hour labor rate that also covers overhead and profit •Provisions for reimbursing direct material costs
Contractor is Obliged to:	Provide an acceptable deliverable at the time, place and price specified in the contract.	Provide an acceptable deliverable at the time and place specified in the contract at the adjusted price.	Provide an acceptable deliverable at the time and place specified in the contract at or below the ceiling price.	Perform at the time, place, and the price fixed in the contract.	Provide acceptable deliverables at the time and place specified in the contract at the price established for each period.	Make a good faith effort to meet the Government's needs within the estimated cost in the Contract, Part I the Schedule, Section B Supplies or services and prices/costs.				Make a good faith effort to meet the Government's needs within the ceiling price.
Contractor Incentive *(other than maximizing goodwill)* [1]	Generally realizes an additional dollar of profit for every dollar that costs are reduced.	Generally realizes an additional dollar of profit for every dollar that costs are reduced.	Realizes profit on cost by completing work below the ceiling price. May earn higher profit by incurring costs below the target cost or by meeting objective performance targets.	Generally realizes an additional dollar of profit for every dollar that costs are reduced; earns an additional fee for satisfying the performance standards.	For the period of performance, realizes an additional dollar of profit for every dollar that costs are reduced.	Realizes a higher fee by completing the work at a lower cost and/or by meeting other objective performance targets.	Realizes a higher fee by meeting judgmental performance standards.	Realizes a higher rate of return (i.e., fee divided by total cost) as total cost decreases.	If CS, shares in the cost of providing a deliverable of mutual benefit.	
Typical Application	Commercial supplies and services.	Long-term contracts for commercial supplies during a period of high inflation.	Production of a major system based on a prototype.	Performance-based contracts.	Long-term production of spare parts for a major system.	Research and development of the prototype for a major system.	Large scale research study.	Research study.	Joint research with educational institutions.	Emergency repairs to heating plants and aircraft engines.
Principal Limitations in FAR/DFARS Parts 16, 32, 35, and 52[2]	Generally NOT appropriate for R&D.	Must be justified.	Must be justified. Must be negotiated. Contractor must have an adequate accounting system. Cost data must support targets.	Must be negotiated.	MUST be negotiated. Contractor must have an adequate accounting system that supports the pricing periods. Prompt redeterminations.	The contractor must have an adequate accounting system. The Government must exercise surveillance during performance to ensure use of efficient methods and cost controls. Must be negotiated. Must be justified. Statutory and regulatory limits on the fees that may be negotiated. Must include the applicable Limitation of Cost clause at FAR 52.232-20 through 23.				D&F required (w/ HCA if over 3 years). Government MUST exercise appropriate surveillance to ensure efficient performance. Document any ceiling increases.
Variants	Firm-Fixed-Price Level-of-Effort.		Successive Targets (FPIS)		Retroactive Redetermination			Completion or Term.		Labor Hour (LH)

[1] Goodwill is the value of the name, reputation, location, and intangible assets of the firm.

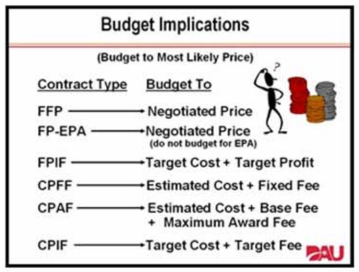
[2] Comply with any USD(AT&L), DPAP or other memoranda that have not been incorporated into the DFARS or DoD Directives or Instructions.

DSMC JANUARY 2014



Contract Category Characteristics

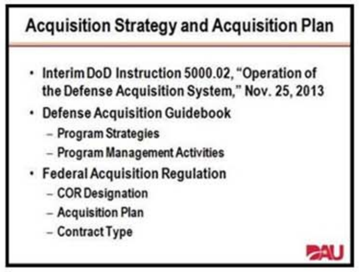
	COST-REIMBURSEMENT	FIXED-PRICE
PROMISE	Best Effort	Shall Deliver
RISK TO CONTRACTORS	Low	High
RISK TO GOVERNMENT	High	Low
CASH FLOW	As Incurred	On Delivery
PROGRESS PAYMENTS	None	% of Actual
ADMINISTRATION	Max Government	Min Government
FEE/PROFIT	Max: 15/10 % CPFF 6 % A – E Contracts	NO Limit, Except 6 % A – E Contracts



Budget Implications

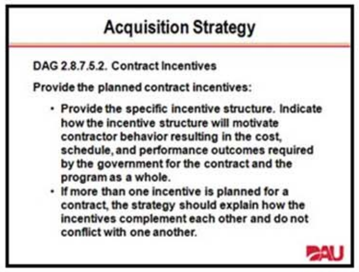
(Budget to Most Likely Price)

Contract Type	Budget To
FFP	Negotiated Price
FP-EPA	Negotiated Price (do not budget for EPA)
FPIF	Target Cost + Target Profit
CPFF	Estimated Cost + Fixed Fee
CPAF	Estimated Cost + Base Fee + Maximum Award Fee
CPIF	Target Cost + Target Fee



Acquisition Strategy and Acquisition Plan

- Interim DoD Instruction 5000.02, "Operation of the Defense Acquisition System," Nov. 25, 2013
- Defense Acquisition Guidebook
  - Program Strategies
  - Program Management Activities
- Federal Acquisition Regulation
  - COR Designation
  - Acquisition Plan
  - Contract Type

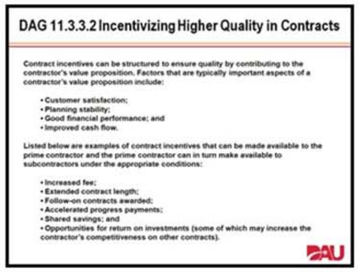


Acquisition Strategy

DAG 2.8.7.5.2. Contract Incentives

Provide the planned contract incentives:

- Provide the specific incentive structure. Indicate how the incentive structure will motivate contractor behavior resulting in the cost, schedule, and performance outcomes required by the government for the contract and the program as a whole.
- If more than one incentive is planned for a contract, the strategy should explain how the incentives complement each other and do not conflict with one another.



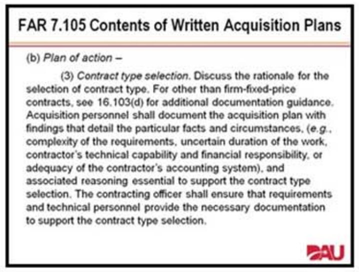
DAG 11.3.3.2 Incentivizing Higher Quality in Contracts

Contract incentives can be structured to ensure quality by contributing to the contractor's value proposition. Factors that are typically important aspects of a contractor's value proposition include:

- Customer satisfaction;
- Planning stability;
- Good financial performance; and
- Improved cash flow.

Listed below are examples of contract incentives that can be made available to the prime contractor and the prime contractor can in turn make available to subcontractors under the appropriate conditions:

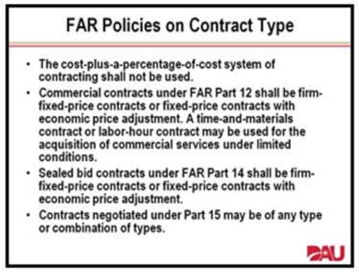
- Increased fee;
- Extended contract length;
- Follow-on contracts awarded;
- Accelerated progress payments;
- Shared savings; and
- Opportunities for return on investments (some of which may increase the contractor's competitiveness on other contracts).



FAR 7.105 Contents of Written Acquisition Plans

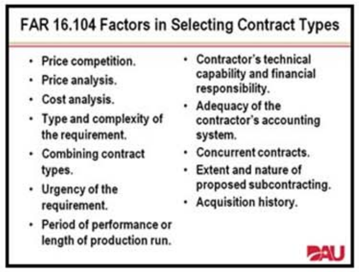
(b) *Plan of action –*

(3) *Contract type selection*. Discuss the rationale for the selection of contract type. For other than firm-fixed-price contracts, see 16.103(d) for additional documentation guidance. Acquisition personnel shall document the acquisition plan with findings that detail the particular facts and circumstances, (*e.g.*, complexity of the requirements, uncertain duration of the work, contractor's technical capability and financial responsibility, or adequacy of the contractor's accounting system), and associated reasoning essential to support the contract type selection. The contracting officer shall ensure that requirements and technical personnel provide the necessary documentation to support the contract type selection.



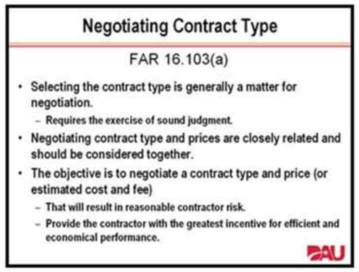
FAR Policies on Contract Type

- The cost-plus-a-percentage-of-cost system of contracting shall not be used.
- Commercial contracts under FAR Part 12 shall be firm-fixed-price contracts or fixed-price contracts with economic price adjustment. A time-and-materials contract or labor-hour contract may be used for the acquisition of commercial services under limited conditions.
- Sealed bid contracts under FAR Part 14 shall be firm-fixed-price contracts or fixed-price contracts with economic price adjustment.
- Contracts negotiated under Part 15 may be of any type or combination of types.



FAR 16.104 Factors in Selecting Contract Types

- Price competition.
- Price analysis.
- Cost analysis.
- Type and complexity of the requirement.
- Combining contract types.
- Urgency of the requirement.
- Period of performance or length of production run.
- Contractor's technical capability and financial responsibility.
- Adequacy of the contractor's accounting system.
- Concurrent contracts.
- Extent and nature of proposed subcontracting.
- Acquisition history.

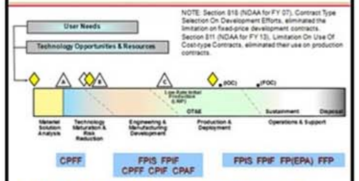


Negotiating Contract Type

FAR 16.103(a)

- Selecting the contract type is generally a matter for negotiation.
  - Requires the exercise of sound judgment.
- Negotiating contract type and prices are closely related and should be considered together.
- The objective is to negotiate a contract type and price (or estimated cost and fee)
  - That will result in reasonable contractor risk.
  - Provide the contractor with the greatest incentive for efficient and economical performance.

"Typical" Contract Types by Phase

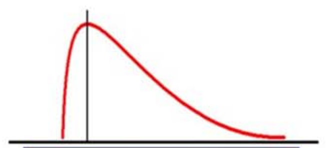

User Needs
Technology Opportunities & Resources
A
B
C
IOC
FOC
Low-Rate Initial Production (LRIP)
OT&E
Sustainment
Disposal
Materiel Solution Analysis
Technology Maturation & Risk Reduction
Engineering & Manufacturing Development
Production & Deployment
Operations & Support

NOTE: Section 818 (NDAA for FY 07), Contract Type Selection On Development Efforts, eliminated the limitation on fixed-price development contracts. Section 811 (NDAA for FY 13), Limitation On Use Of Cost-type Contracts, eliminated their use on production contracts.

CPFF	FPIS FPIF CPFF CPIF CPAF	FPIS FPIF FP(EPA) FFP

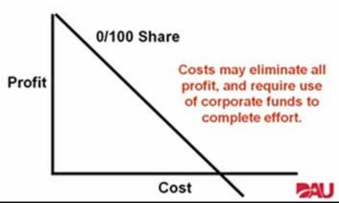
For MDAPs, authorization for cost reimbursement contract for EMD requires an MDA written determination, and for Production a USD(AT&L) written certification to congressional defense committees.

Distribution of Cost Outcomes Does Not Follow a Bell Shaped Curve



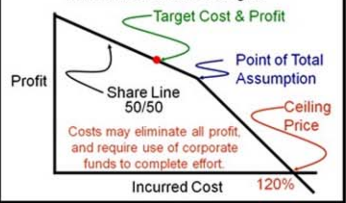
There is limited potential for underrun, but infinite potential for overrun.

Firm-Fixed-Price Contracts

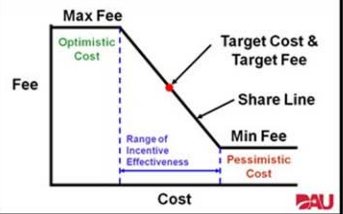


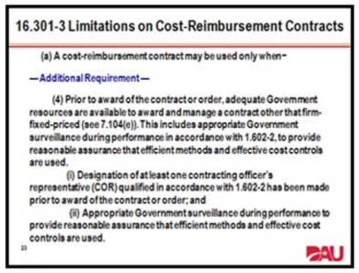
Fixed-Price-Incentive Contracts

Firm and Successive Targets



Cost-Plus-Incentive-Fee Contracts





16.301-3 Limitations on Cost-Reimbursement Contracts

(a) A cost-reimbursement contract may be used only when–

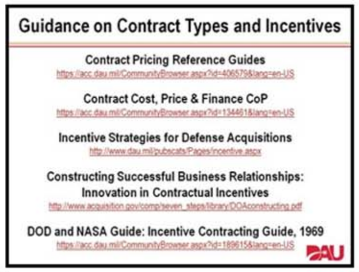
—Additional Requirement—

(4) Prior to award of the contract or order, adequate Government resources are available to award and manage a contract other that firm-fixed-priced (see 7.104(e)). This includes appropriate Government surveillance during performance in accordance with 1.602-2, to provide reasonable assurance that efficient methods and effective cost controls are used.

(i) Designation of at least one contracting officer's representative (COR) qualified in accordance with 1.602-2 has been made prior to award of the contract or order; and

(ii) Appropriate Government surveillance during performance to provide reasonable assurance that efficient methods and effective cost controls are used.

23



Guidance on Contract Types and Incentives

Contract Pricing Reference Guides
https://acc.dau.mil/CommunityBrowser.aspx?id=406579&lang=en-US

Contract Cost, Price & Finance CoP
https://acc.dau.mil/CommunityBrowser.aspx?id=134461&lang=en-US

Incentive Strategies for Defense Acquisitions
http://www.dau.mil/pubscats/Pages/incentive.aspx

Constructing Successful Business Relationships: Innovation in Contractual Incentives
http://www.acquisition.gov/comp/seven_steps/library/DOAconstructing.pdf

DOD and NASA Guide: Incentive Contracting Guide, 1969
https://acc.dau.mil/CommunityBrowser.aspx?id=189615&lang=en-US

The material in this appendix is reproduced from *Agile Methods: Selected DoD Management and Acquisition Concerns* [Lapham 2011].

Appendix D: Agile Glossary

Backlog
An accumulation, especially of unfinished work or unfilled orders.[1]

Done
1. Having been carried out or accomplished; finished.[2] Author's note: In an Agile context, the definition of done can include software, documentation, testing, and certification being complete or any subset of this list being completed. The developer and product owner must agree on what is included in "done." With this in mind, another definition is

2. The useful definition of **doneness** stresses the goal of all Agile iterations: the product must remain shippable.

- All visible features work
  - as advertised
  - within the expected environment
  - in any combination
  - without degradation over time
  - with graceful handling of errors
- Hide all broken or unfinished features

This definition of doneness emphasizes this result: we want a stable app at all times. When we start the app, we know what is expected to work because we can see it and try it. We can prioritize new features by seeing how they must be reconciled with already-visible features.[3]

Epic
A connected or bundled set of stories that result in a definable (in the case of software, desirable) capability or outcome. An epic is a large user story. It is possible to break up an epic into several user stories.[4]

[1] http://www.thefreedictionary.com/backlog

[2] http://www.thefreedictionary.com/done

[3] http://billharlan.com/pub/papers/Agile_Essentials.html

[4] http://www.targetprocess.com/LearnAgile/AgileGlossary/ThemeEpic.aspx

Iteration

In Agile software development,[5] a single development cycle, usually measured as one or two weeks. An iteration may also be defined as the elapsed time between iteration planning sessions.

Just Enough

Combining the two dictionary definitions of "just" and "enough" you get "exactly sufficient." Within the Agile community, this is an appropriate definition. Thus: just enough to be successful, to get started, support the user story queue, accomplish our goal.

Pattern

1. A form of knowledge management. It is a literary form for documenting a common, successful practice. It articulates a recurring problem, as well as the context of the problem and the conditions that contribute to creating it. Likewise, the solution, the rationale for the solution, and consequences of using it are given.

2. A way to capture expertise. Patterns document good ideas—strategies that have been shown to work well for a variety of people in a variety of circumstances.[6]

Product Backlog

The master list of all functionality desired in the product.[7]

Release

The act or an instance of issuing something for publication, use, or distribution. Something thus released: a new release of a software program.[8]

Sprint

A set period of time during which specific work must be completed and made ready for review.[9] Often used as a synonym for iteration.

Story

In Agile software development, a story is a particular business need assigned to the software development team. Stories must be broken down into small enough components that they may be delivered in a single development iteration.[10]

---

5 http://searchsoftwarequality.techtarget.com/definition/iteration

6 *Fearless Change, Patterns for Introducing New Ideas*, Mary Lynn Mann, Linda Rising, Addison-Wesley, 2005, Pearson Education Inc.

7 http://www.mountaingoatsoftware.com/scrum/product-backlog

8 http://www.thefreedictionary.com/release

9 http://searchsoftwarequality.techtarget.com/definition/Scrum-sprint

10 http://searchsoftwarequality.techtarget.com/definition/story

Story Point

According to Cohn, "Story points are a unit of measure for expressing the overall size of a user story, feature, or other piece of work …The number of story points associated with a story represents the overall size of the story. There is no set formula for defining the size of a story. Rather a story-point estimate is an amalgamation of the amount of effort involved in developing the feature, the complexity of developing it, the risk inherent in it and so on."[11]

Technical Debt

Technical debt and *design debt* are synonymous, neologistic metaphors referring to the eventual consequences of slapdash software architecture and hasty software development. Code debt refers to technical debt within a codebase.

Ward Cunningham first drew the comparison between technical complexity and debt in a 1992 experience report:

> *Shipping first time code is like going into debt. A little debt speeds development so long as it is paid back promptly with a rewrite... The danger occurs when the debt is not repaid. Every minute spent on not-quite-right code counts as interest on that debt. Entire engineering organizations can be brought to a stand-still under the debt load of an unconsolidated implementation, object-oriented or otherwise* [Ozkaya 2011].

Time Box

A fixed amount of hours or days in which to accomplish something.[12]

Time Boxing

A planning technique common in planning projects (typically for software development), where the schedule is divided into a number of separate time periods (**time boxes**, normally two- to six-weeks long), with each part having its own deliverables, deadline, and budget.[13]

User Story

Descriptions of discrete functionality known to be needed by a particular user segment that is part of the project's audience, and other stories that address infrastructure and quality attributes that are pervasive to the product (e.g., security or usability).

Velocity

Velocity is a measure of a team's rate of progress. It is calculated by summing the number of story points assigned to each user story that the team completed during the iteration. If the team completes three stories each estimated at five story points, its velocity is 15. If the team completes two five-point stories, its velocity is 10.[14] Velocity, in the Agile community, refers to the amount

[11] Cohn, M., *Agile Estimating and Planning*, p.36

[12] http://www.agileadvice.com/archives/2006/02/timeboxing_a_cr.html

[13] http://en.wikipedia.org/wiki/Timeboxing

[14] Cohn, M. *Agile Estimating and Planning*, p. 38.

of capacity of a *particular* team to produce working software. It does not have a general analog in traditional DoD projects.

References

URLs are valid as of the publication date of this document.

[AFEI 2012]
The Association for Enterprise Information. *AFEI 2012 Agile in Defense Fall Workshop*. http://www.afei.org/Working-Groups/ADAPT/Documents/Agile%20in%20Defense%20Fall%20Workshop%202012.pdf (2012)

[Agile Alliance 2001]
Agile Alliance. *The Agile Manifesto*. http://www.agilealliance.org/the-alliance/the-agile-Manifesto/ (2001)

[Agile Alliance 2001b]
Agile Alliance. *The Twelve Principles of Agile Software*. http://www.agilealliance.org/the-alliance/the-agile-Manifesto/the-twelve-principles-of-agile-software/ (2001)

[Agile Alliance 2001c]
Agile Alliance. *What is Agile Software Development?* http://www.agilealliance.org/the-alliance/what-is-agile/ (2001)

[Ambler 2004]
Ambler, Scott W. *Disciplined Agile Software Development: Definition.* http://www.agilemodeling.com/essays/agileSoftwareDevelopment.htm (2004)

[Anderson 2010]
Anderson, David J. & Reinertsen, Donald G. *Kanban: Successful Evolutionary Change for Your Technology Business.* Blue Hole Press, 2010.

[Arbogast 2012]
Arbogast, Tom, Larman, Craig, & Vodde, Bas. *Agile Contracts Primer.* (Derived from the book *Practices for Scaling Lean & Agile Development.* Addison-Wesley. 2010) http://www.agilecontracts.org (2012)

[Bellomo 2011]
Bellomo, Stephany. *A Closer Look at 804: A Summary of Considerations for DoD Program Managers*. Software Engineering Institute, Carnegie Mellon University. http://resources.sei.cmu.edu/library/asset-view.cfm?assetid=9751 (2011)

[Bellomo 2012]
Bellomo, Stephany & Woody, Carol. *DoD Information Assurance and Agile: Challenges and Recommendations Gathered Through Interviews with Agile Program Managers and DoD Accreditation Reviewers* (CMU/SEI-2012-TN-024). Software Engineering Institute, Carnegie Mellon University. http://resources.sei.cmu.edu/library/asset-view.cfm?AssetID=34083 (2012)

[Campbell 2010]
Campbell, Grady. *The Illusion of Certainty*. Software Engineering Institute, Carnegie Mellon University. http://resources.sei.cmu.edu/library/asset-view.cfm?assetid=20918 (2010)

[DAU 2014]
Defense Acquisition University. *Comparison of Major Contract Types*. https://acc.dau.mil/adl/en-US/214513/file/75692/Comparison%20of%20Major%20 Contract%20Types%20JANUARY%202014%20Final%20Version%20PRINT.ppt (2014)

[Defense Science Board 2009]
Defense Science Board. *Report of the Defense Science Board Task Force on Department of Defense Policies and Procedures for the Acquisition of Information Technology*. Office of the Under Secretary of Defense for Acquisition, Technology, and Logistics. http://www.acq.osd.mil/dsb/reports/ADA498375.pdf (2009)

[DoD 2015]
Department of Defense. *Department of Defense Instruction 5000.02*. http://www.dtic.mil/whs/directives/corres/pdf/500002p.pdf (2015)

[FAR 2015]
U.S. General Services Administration. *Federal Acquisition Regulation*. https://acquisition.gov/far/index.html (2015)

[GAO 2012]
U.S. Government Accountability Office. *Effective Practices and Federal Challenges in Applying Agile Methods*. GAO-12-681. http://www.gao.gov/products/gao-12-681 (July 2012)

[Hayes 2014]
Hayes, Will, Miller, Suzanne, Lapham, Mary Ann, Wrubel, Eileen, & Chick, Timothy A. *Agile Metrics: Progress Monitoring of Agile Contractors* (CMU/SEI-2013-TN-029). Software Engineering Institute, Carnegie Mellon University, 2014. http://resources.sei.cmu.edu/library/asset-view.cfm?assetid=77747

[Highsmith 2000]
Highsmith, J. *Agile Project Management: Creating Innovative Products, 2nd ed*. Addison-Wesley, 2009.

[Kennedy 2011]
Kennedy, Matthew. "An Agile Systems Engineering Process: The Missing Link?" *CrossTalk* (May-June 2011): 16-20. http://www.crosstalkonline.org/storage/issue-archives/2011/201105/201105-Kennedy.pdf

[Lapham 2010]
Lapham, M.A., Williams, R., Hammons, C., Burton, D., & Schenker, A. *Considerations for Using Agile in DoD Acquisition* (CMU/SEI-2010-TN-002). Software Engineering Institute, Carnegie Mellon University, 2010. http://www.sei.cmu.edu/library/abstracts/reports/10tn002.cfm

[Lapham 2011]
Lapham, Mary Ann, Garcia-Miller, Suzanne, Adams, Lorraine, Brown, Nanette, Hackemack, Bart, Hammons, Charles (Bud), Levine, Linda, & Schenker, Alfred. *Agile Methods: Selected DoD Management and Acquisition Concerns* (CMU/SEI-2011-TN-002). Software Engineering Institute, Carnegie Mellon University, 2011. http://resources.sei.cmu.edu/library/asset-view.cfm?AssetID=9769

[Lapham 2014]
Lapham, Mary Ann, Bandor, Michael, & Wrubel, Eileen. *Agile Methods and Request for Change (RFC): Observations from DoD Acquisition Programs* (CMU/SEI-2013-TN-031). Software Engineering Institute, Carnegie Mellon University, 2014. http://resources.sei.cmu.edu/library/asset-view.cfm?AssetID=77732

[Nidiffer 2014]
Nidiffer, Kenneth, Miller, Suzanne, & Carney, David. *Potential Use of Agile Methods in Selected DoD Acquisitions: Requirements Development and Management* (CMU/SEI-2013-TN-006). Software Engineering Institute, Carnegie Mellon University, 2014. http://resources.sei.cmu.edu/library/asset-view.cfm?AssetID=89158

[OMB 2012]
Office of Management and Budget. *Contracting Guidance to Support Modular Development.* http://www.whitehouse.gov/sites/default/files/omb/procurement/memo/contracting-guidance-to-support-modular-development.pdf (2012)

[Opelt 2013]
Opelt, Andreas, Gloger, Boris, Pfarl, Wolfgang, & Mittermayr, Ralf. *Agile Contracts: Creating and Managing Successful Projects with Scrum*. John Wiley & Sons, Inc., 2013. http://onlinelibrary.wiley.com/book/10.1002/9781118640067

[OSD 2010]
Office of the Secretary of Defense. *A New Approach for Delivering Information Technology Capabilities in the Department of Defense, Report to Congress,* November 2010, Pursuant to Section 804 of the National Defense Authorization Act for Fiscal Year 2010. United States Department of Defense, 2010. http://dcmo.defense.gov/documents/OSD%2013744-10%20-%20804%20Report%20to%20Congress%20.pdf

[OSTP 2014]
White House Office of Science and Technology Policy (OSTP). *Innovative Contracting Case Studies*. http://www.whitehouse.gov/sites/default/files/microsites/ostp/innovative_contracting_case_studies_2014_-_august.pdf (2014)

[Ozkaya 2011]
Ozkaya, I., Brown, N., & Nord, R. Ch. 3, “Communicating the Value of Architecting within Agile Development,” 11-22. *Results of SEI Independent Research and Development Projects (FY 2010)* (CMU/SEI-2011-TR-002). Software Engineering Institute, Carnegie Mellon University, 2011. http://resources.sei.cmu.edu/library/asset-view.cfm?assetID=9895

[Palmquist 2014]
Palmquist, Steven, Lapham, Mary Ann, Garcia-Miller, Suzanne, Chick, Timothy, & Ozkaya, Ipek. *Parallel Worlds: Agile and Waterfall Differences and Similarities* (CMU/SEI-2013-TN-021). Software Engineering Institute, Carnegie Mellon University, 2013. http://resources.sei.cmu.edu/library/asset-view.cfm?AssetID=62901

[Regan 2014]
Regan, Colleen, Lapham, Mary Ann, Wrubel, Eileen, Beck, Stephen, & Bandor, Michael. *Agile Methods in Air Force Sustainment: Status and Outlook* (CMU/SEI-2014-TN-009). Software Engineering Institute, Carnegie Mellon University. http://resources.sei.cmu.edu/library/asset-view.cfm?AssetID=312754 (2014)

[Sliger 2008]
Sliger, Michele & Broderick, Stacia. *The Software Project Manager's Bridge to Agility*. Pearson Education, 2008.

[U.S. CIO 2014]
United States Chief Information Officer. *TechFAR Handbook for Procuring Digital Services Using Agile Processes*. https://playbook.cio.gov/techfar/ (2014)

[Wrubel 2014]
Wrubel, Eileen, Miller, Suzanne, Lapham, Mary Ann, & Chick, Timothy. *Agile Software Teams: How They Engage with Systems Engineering on DoD Acquisition Programs* (CMU/SEI-2014-TN-013). Software Engineering Institute, Carnegie Mellon University, 2014. http://resources.sei.cmu.edu/library/asset-view.cfm?AssetID=295943

REPORT DOCUMENTATION PAGE	*Form Approved OMB No. 0704-0188*

1. AGENCY USE ONLY	(Leave Blank)
2. REPORT DATE	August 2015
3. REPORT TYPE AND DATES COVERED	Final
4. TITLE AND SUBTITLE	Contracting for Agile Software Development in the Department of Defense: An Introduction
5. FUNDING NUMBERS	FA8721-05-C-0003
6. AUTHOR(S)	Eileen Wrubel, John Gross
7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)	Software Engineering Institute Carnegie Mellon University Pittsburgh, PA 15213
8. PERFORMING ORGANIZATION REPORT NUMBER	CMU/SEI-2015-TN-006
9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)	AFLCMC/PZE/Hanscom Enterprise Acquisition Division 20 Schilling Circle Building 1305 Hanscom AFB, MA 01731-2116
10. SPONSORING/MONITORING AGENCY REPORT NUMBER	n/a
11. SUPPLEMENTARY NOTES	
12A DISTRIBUTION/AVAILABILITY STATEMENT	Unclassified/Unlimited, DTIC, NTIS
12B DISTRIBUTION CODE	

13. ABSTRACT (MAXIMUM 200 WORDS)

This technical note (TN), part of an ongoing Software Engineering Institute (SEI) series on Agile in the Department of Defense (DoD), addresses effective contracting for Agile software development. Contracting officers do not receive career field education targeted at achieving successful outcomes with Agile software development methods. For the purposes of this TN, the SEI gathered data from program office team members, contractors, and contracting officers about the state of contracting activities involving Agile development. The authors conducted a series of interviews and mined past interviews and survey data on Agile software development to understand common questions and concerns and provide some real-world examples to address them. This TN offers a primer on Agile based on a contracting officer's goals, describes how program office teams need to support contracting efforts, and addresses common concerns about Agile and how those concerns can be mitigated in the contracting process.

14. SUBJECT TERMS	Agile, contracting, FAR
15. NUMBER OF PAGES	75
16. PRICE CODE	

17. SECURITY CLASSIFICATION OF REPORT	18. SECURITY CLASSIFICATION OF THIS PAGE	19. SECURITY CLASSIFICATION OF ABSTRACT	20. LIMITATION OF ABSTRACT
Unclassified	Unclassified	Unclassified	UL

NSN 7540-01-280-5500

Standard Form 298 (Rev. 2-89) Prescribed by ANSI Std. Z39-18
298-102