DEPARTMENT OF HOMELAND SECURITY

AGILE DEVELOPMENT AND DELIVERY FOR INFORMATION TECHNOLOGY INSTRUCTION MANUAL 102-01-004-01

RCDeyo

Russell C. Deyo
Under Secretary for Management
Chief Acquisition Officer

7/15/16

Date

Instruction Manual # 102-01-004-01
Revision #00

Testimonial

"The Department of Homeland Security is visionary for its embrace of Agile methods. The private sector uses Agile to deliver software faster, better, and cheaper than their competitors. DHS has realized the country can't afford for them to do anything less."

- *Jeff Sutherland, CEO of Scrum Inc. and co-creator of Scrum*

TABLE OF CONTENTS

FIGURES

TABLES

1. INTRODUCTION

1.1 Purpose and Scope

The purpose of this Instruction Manual is to provide Department of Homeland Security (DHS) Program and Project Managers (PM) with guidance to support the implementation of Agile Information Technology (IT) development as outlined by *DHS MD/Instruction 102-01-004 Agile Development and Delivery for Information Technology*.[1] Although Agile practices are expanding into other areas, this Instruction Manual focuses on software development. It helps PMs build upon best practices and experience from industry, federal government, and recent DHS implementations of Agile methodologies. This guidance is intended to enhance understanding of why Agile is the preferred approach to federal IT development, and to provide a starting point for increasing DHS-wide application of and expertise in Agile methodologies. This Instruction Manual is intended to help PMs and other key stakeholders understand how Agile IT development satisfies the requirements of the *Systems Engineering Life Cycle (SELC)* (as established by DHS MD 102-01-103) to support individual IT project needs, while meeting the intent of the *25 Point Implementation Plan to Reform Federal Information Technology Management,*[2] *Contracting Guidance to Support Modular Development,*[3] and other federal guidance on improving IT development and acquisition.

To mitigate some of the challenges associated with delivering usable IT capabilities in a timely manner, the Agile Development and Delivery Instruction identifies Agile as the preferred approach for all IT programs and projects, when appropriate. This conforms to the practice of delivering finalized working software to users in discrete increments within shortened time frames of six months or less for ongoing projects and six to 12 months for initial delivery of new projects. Hence, this guidance is consistent with the concept of modular development as outlined in the *25 Point Implementation Plan* and the *Digital Services Playbook*, which place a relatively short "time box" around a project or project segment in which to plan, design, develop, test, and implement a capability.

As this is a Instruction Manual and not an encyclopedia, it avoids highly detailed instruction and instead presents a practical overview of Agile development methodologies and tools. It also gives the reader directions for applying Agile techniques without dictating a specific "DHS way" for Agile. The state of the art for applying Agile to improve IT development and delivery continues to evolve, as do techniques and guidance for adapting Agile methodologies to the federal enterprise. This document provides an introduction to Agile concepts, techniques, and experiences intended to help

---

[1] Department of Homeland Security, *Agile Development and Delivery for Information Technology,* Revision #0, draft, (Washington: Department of Homeland Security).
[2] Vivek Kundra, *25 Point Implementation Plan to Reform Federal Information Technology Management* (Washington: White House, December 2010), https://www.dhs.gov/sites/default/files/publications/digital-strategy/25-point-implementation-plan-to-reform-federal-it.pdf.
[3] White House, *Contracting Guidance to Support Modular Development* (Washington: White House, June 2012), http://www.whitehouse.gov/sites/default/files/omb/procurement/guidance/modular-approaches-for-information-technology.pdf.

DHS PMs identify and craft program and project management options for applying Agile to achieve greater IT development success. It is based on current best practices, but will continue to evolve as practices change. Appendices in the back of this Instruction Manual , as well as references throughout, provide a more detailed and focused background of proven and emerging Agile practices. They include several resources for learning more about each.

This Instruction Manual complements the DHS Systems Engineering Lifecycle (SELC) Instruction Guidebook (MD 102-01-103-01) by working within its high-level framework without superseding existing federal guidance. Accordingly, this Instruction Manual assists PMs in making informed decisions about applying Agile approaches and tools within the context of DHS acquisition management policies to improve planning efficiency, reduce overall risk, and enable rapid delivery of useful capability. The DHS SELC is the common systems engineering framework used to guide DHS programs and projects in the application of solid systems-engineering discipline. The SELC provides significant detail on systems engineering and its application and execution. Agile is one approach that encompasses the SELC's flexible framework and can be leveraged as appropriate as part of a DHS acquisition approach. The *SELC Tailoring Examples for Selected Types of DHS Acquisition Programs*[4] provides an example of how an Agile IT program could tailor its systems engineering approach (and how it differs from traditional practices) to more effectively encompass user requirements and produce a product in an Agile manner. Readers should refer to the DHS SELC Instruction Manual or one of its supplemental guides for any systems engineering topic not expressly covered in the DHS Agile Development Instruction Manual.

1.2 Why Agile?

Agile development emphasizes inspecting and adapting, rather than following a plan. Teams deliver functionality incrementally, by releasing small system increments or updates frequently, rather than relying on a single, massive deployment. The Agile approach enables programs to deliver prioritized capabilities to users more quickly. This approach also provides more flexibility than many traditional methods, reducing risk by encouraging changing requirements, maximizing learning, and providing the ability to exploit technology advances in a timely fashion. Agile methodologies achieve this flexibility through iterative, incremental development practices that rely on self-organizing, cross-functional teams of developers, stakeholders, and users. The consistent interaction and partnering of individuals with different perspectives encourages development that is highly flexible and lower risk due to frequent planning, development, and delivery checkpoints. The Agile approach to software development incorporates an inherent cultural change in the way professionals develop and deliver software products by integrating users, testers, and developers and enabling them to directly address challenges together. Properly applied, Agile techniques help teams to develop software more efficiently and effectively, providing better solutions to their customers and stakeholders.

[4] Department of Homeland Security, *SELC Tailoring Examples for Selected Types of DHS Acquisition Programs*, version 1.0, draft, (Washington: Department of Homeland Security, March 2014).

Agile development offers the following benefits:

- Requirements and products move at a rapid pace to realistically accommodate technical advances
- Iterations and requirements are informed by discovery as challenges are resolved
- Solutions are produced in a series of smaller increments to reduce risk and meet evolving needs
- PMs select and apply the specific Agile methodologies that best suit the needs of the program, team, customer, and environment

DHS is working on a number of initiatives to promote Agile/modular development in accordance with the referenced Office of Management and Budget (OMB) direction. These include an Agile instruction, this Instruction Manual , and the SELC Guidance.[5] An Agile Center of Excellence and Integrated Product Team are leading these efforts.

An Agile approach provides benefits to minimize risks and increases the probability of acceptable delivery compared to more traditional development processes. Agile development allows for increased flexibility across the life cycle of a project, though it notably does not allow PMs to begin projects without adequate initial project planning. It is important to acknowledge that Agile development projects require appropriate planning both up-front *and* throughout the project life cycle, so long as the planning does not significantly delay the start of the actual execution.

This Instruction Manual provides information on how to achieve success by implementing Agile techniques for software development at DHS.

How to Get Started with Agile at DHS

This Instruction Manual cannot address everything you want to know about Agile. It does, however, point you to resources of use in your learning process:

- For readers new to Agile, Appendices A and B provide further reading references
- Appendix D identifies some Agile training resources
- Appendices E and F provide reference lists of Agile terms and acronyms

All prospective PMs of Agile projects are encouraged to seek advice from DHS PMs who have training in or experience with Agile. DHS best practices for Agile continue to evolve. This and other resources will be updated as DHS processes mature.

1.3 Document Structure

This document is configured both as a basic guide and a reference. It is structured to provide a basic understanding of Agile up-front, with more advanced and detailed reference information organized in the appendices:

- Section 2 provides a brief overview of Agile approaches, along with a brief discussion of the range of Agile methodologies and tools, team roles, and project success factors

[5] Department of Homeland Security, *SELC Tailoring Examples for Selected Types of DHS Acquisition Programs*, version 1.0, draft, (Washington: Department of Homeland Security, March 2014).

- Section 3 provides information on considerations and options for applying Agile within the context of the established DHS acquisition and systems engineering life cycle frameworks
- Section 4 provides guidance and recommendations on developing Agile projects at DHS, along with lessons learned from experienced PMs within Agile projects
- Section 5 outlines tips and practical lessons for implementing Agile development projects based on DHS experience with Agile to date

Readers who are more familiar/experienced in Agile may find the information beginning in Section 3 of particular value, as the latter sections focus on how to implement Agile specifically within DHS. Appendices address Agile methodologies commonly used within DHS, tools that can be used to manage Agile projects, and how PMs can generate Agile metrics and reports to support their projects. A brief glossary of common Agile terms is provided in Appendix E. A checklist, providing a top-level set of actions to help PMs shape their Agile approach, is included as Appendix G.

The *Agile Development Instruction Manual* is a living document. Please submit your feedback, recommendations, and related questions to the Office of Program Accountbility and Risk Management.

2. AGILE: THE BASICS

2.1 Understanding the Agile Approach

Agile is more than a set of techniques, tools, and processes prescribed to improve IT development and address project management challenges. Rather, Agile is a philosophy - a way of thinking about IT projects that is rooted in a basic set of core principles and values.

Although many of the concepts and frameworks relevant to Agile software development have been recognized and discussed for some time, the *Manifesto for Agile Software Development,*[6] published in 2001, is generally recognized as the authoritative collection of tenets that define what is known today as Agile.

The motivation to discover better ways of developing software is rooted in experience with the issues commonly associated with the traditional software development approach known as the "waterfall management model." The waterfall model breaks the software development effort down into distinct phases (e.g., planning, requirements definition, design, development, testing, and implementation), with the rule that ***no phase can begin until the previous phase has been completed***. The traditional waterfall project management model was designed to ensure compliance with applicable standards (e.g., regulatory requirements and demonstrations/reviews) by providing a review opportunity at each phase transition. It is typically applied to efforts where requirements are completely known up-front and are unlikely to change.

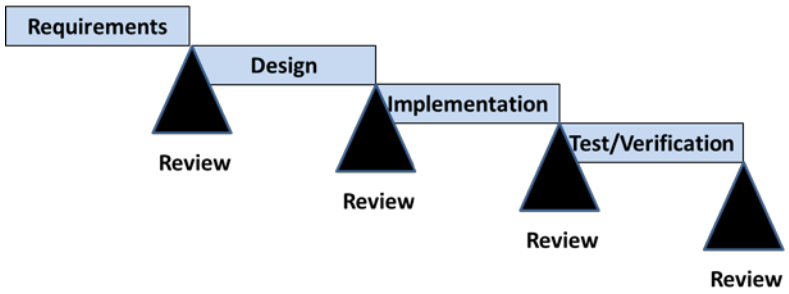

Figure 2-1: Waterfall Management Model

Relying on a waterfall model can unnecessarily hamper progress when applied to software development. In the case of systems where it is not possible to define all requirements completely in advance (which, in reality, is *always*), where some requirements (e.g., data throughput, numbers of users, and data handling rules) may evolve over the course of the development effort, or where the development team seeks to learn from previous practices and activities and incorporate changes into future development, the waterfall model lacks a simple, efficient mechanism for revisiting and revising requirements to meet current needs. Similarly, the waterfall model lacks an inherent mechanism for identifying and taking advantage of technology advances after the design

[6] Kent Beck, et al., "Manifesto for Agile Software Development" (2001), http://Agilemanifesto.org/.

phase has been completed. Historically, these shortcomings have placed PMs at risk of delivering products tied to outdated requirements or obsolete technology.

IT standards and technology advances over the past several decades have facilitated more reliable interfaces and information transfer between independent IT system components. These advances, along with a better understanding of IT project management, have fueled the adoption of industry practices that focus on frequent delivery of enhanced IT capability in a modular, iterative fashion. Agile methodologies and project management techniques exploit the ability to deliver small, frequent IT capability improvements iteratively, focusing on continuous improvement based on increased understanding of requirements, lessons learned from previous iterations, and interaction between the customer and developers throughout the project life cycle.

The Manifesto for Agile Software Development is supplemented by 12 key principles, also posted on the Agile Manifesto website at http://Agilemanifesto.org (also see Appendix A). These principles emphasize frequent, iterative software deliveries and a team approach that integrates business (customer) representatives, developers, and project managers and relies on their frequent interaction. The integrated team executes regular assessments of their progress and adjusts accordingly to enable continuous improvement of both the end product and overall team performance. The iterative approach supports requirements refinement and provides opportunities to recognize and manage changes as information becomes available. Delivery of value remains the highest priority. These 12 principles guide all methodologies and approaches that have come to be known as Agile.

2.2 Agile Methodologies and Frameworks

There are numerous systems development methodologies and frameworks that incorporate characteristics of the Agile philosophy, and variations continue to evolve. There is currently no specific set of DHS-approved methodologies; DHS PMs are encouraged to determine which Agile techniques and tools may be best suited for application to their programs and projects. Many different Agile approaches have been employed to date within DHS.

The following list of some of the more popular Agile methodologies is not intended to be comprehensive. Furthermore, it does not imply any recommendation or requirement to select a methodology from this list. Indeed, as Agile practices continue to evolve, these methodologies may be updated or replaced by emerging techniques. Practitioners selecting specific Agile approaches and tools benefit from considering the characteristics of the project, the project team, the customer, and the project environment. The summaries presented here outline some of the key characteristics of these common methodologies. Section 2.3 (Figure 2-3) provides a hypothetical decision tree with answers to the types of questions regarding project characteristics that PMs need to consider while selecting which methodology is best for their project. There are numerous textbooks, courses, blogs, and other information sources widely available for further study of Agile methodologies and techniques. Appendix B provides a description of these methodologies, as well as a set of initial resources readers may wish to investigate.

Although most Agile approaches are described as standalone methodologies, it is common practice for PMs and development teams to employ a combination of practices from different methodologies and tools based on what best fits their environment. Although it is important to understand the elements of different Agile methodologies in context, it is often best to select the individual aspects to use based on the specific needs of the program, project, and project team at hand.

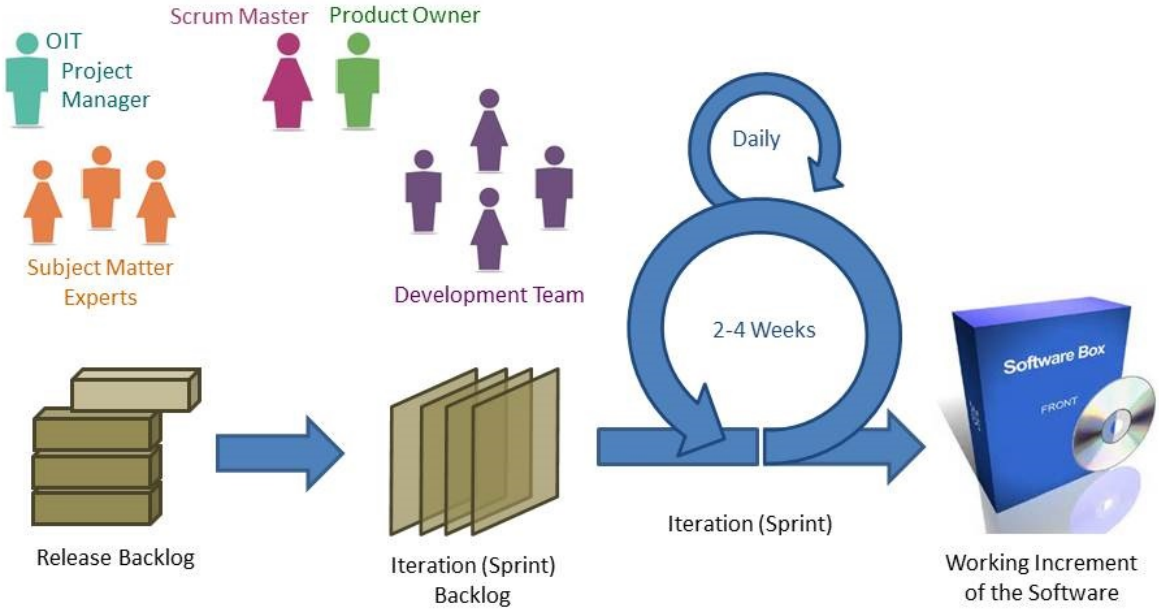

Figure 2-2. The Agile Scrum Framework as a Time-Boxed, Iterative Methodology

Scrum – Scrum (Figure 2-2) is the most widely recognized Agile framework and is often used in conjunction with other methodologies.[7] It emphasizes the use of self-organizing, cross-functional teams, with a product owner responsible for managing user stories (the software project's requirements) that add value to the product or organization. Work is organized into sprints, a time-boxed period for delivery of functionality with defined deadlines and deliverables. Requirements are collectively sorted into project-level and sprint-level backlogs, and overall project activity is monitored, communicated, and managed using frequent team processes including daily stand-ups, sprint planning, sprint reviews, and retrospectives (reviews aimed at incorporating feedback and learning into the next iteration of development and/or testing).

Lean Software Development - Lean is based on manufacturing principles that emphasize the delivery of value while eliminating waste. As with other Agile techniques, there is an emphasis on learning throughout the development process, updating project decisions to incorporate new information, short cycle times, rapid delivery, and a holistic view of product development. As with other approaches, Lean processes are often combined with Scrum and other techniques.

[7] "8th Annual State of Agile Survey," VersionOne.com (2014), http://www.versionone.com/pdf/2013-state-of-Agile-survey.pdf.

Kanban – Kanban is also rooted in the same principles as Lean and is often combined with other approaches. Kanban is a management practice that relies on a visual display of process information and workflows, translating activities into a collection of queues. Kanban manages work in progress by tracking the number of tasks in each work stream at any given time. The visual display of progress within and across these queues helps to identify issues rapidly and optimizes overall project processing. Many Agile teams use these displays ("Kanban boards") as management and communication tools in conjunction with other Agile techniques.

Extreme Programming (XP) – XP is an approach based on using dedicated teams of high-performing individuals working intensively on software development. Key XP values are communication, simplicity, feedback, and courage (i.e., risk-taking). It emphasizes constant end-user involvement and technical best practices, including test-driven development, pair programming (jointly executed coding on a single task between two developers), and constant refactoring (restructuring of code). It is often used together with Scrum to specify code-level development and team processes.

While there are a number of Agile methodologies applicable to individual projects, applying Agile approaches within the context of a large enterprise with existing software development and architectural standards poses additional challenges. There several approaches to scaling Agile. Two of the best-known process frameworks to address the problem of scaling Agile are Disciplined Agile Delivery (DAD) and the Scaled Agile Framework® (SAFe®).

- DAD is a process framework that incorporates many of the core people-first, learning-oriented, solution-focused Agile values, but focuses on applying these in a hybrid manner that recognizes the characteristics of the larger enterprise in which the software development activity takes place
- SAFe® offers an approach to help project managers and executives scale Agile techniques to larger programs and to align project, program, and portfolio efforts with enterprise architecture design

2.2.1 Selecting and Preparing to Use a Methodology

As evidenced by the list above, there is an abundance of named methodologies for applying Agile principles to software development. As the use of Agile has expanded in recent decades, the number of researchers, practitioners, and consultants in this area has also increased. Many of these individuals have developed and/or promoted specific variants, combinations, and adaptations of these and other methodologies. Those new to Agile may benefit more from focusing on the common underlying philosophy and effective practices, such as organizing the project into small work efforts, rather than on the specifics of competing methodologies.

In association with this proliferation of related methodologies, a number of individuals, groups, and firms have developed specific tools (typically, but not exclusively, software based) to support Agile development methods. Similarly, a growing range of training and certification courses have become available. See Section 4.4.1 for a discussion of widely available tools, including tools that have been successfully applied within DHS to date.

At this time, DHS neither specifically endorses nor prohibits the use of any of the known Agile IT software development methodologies or tools. PMs are encouraged to focus on developing a thorough understanding of the specific characteristics and needs of their IT projects and project teams, and to select or tailor the Agile methodologies and tools to apply to the set of challenges faced by each project. This Instruction Manual also contains several illustrative tips (see call-out boxes in Sections 4 and 5) provided by DHS PMs regarding the use of Agile within DHS. These stories and recommendations may prove valuable in making these methodology and tool selections.

2.3 Agile Methodologies and Practices: A Comparative Overview

This section provides an overview of what type of questions PMs may ask to differentiate and select from the various Agile methodologies, as well as to gain a better understanding of the scaling and workflow management techniques available to DHS software developers.

2.3.1 Selecting the Right Agile Practices for the Project

One of the challenges facing PMs seeking to apply Agile is selecting the *right* Agile approach for their project. A PM does not try to find the right methodology for their project based on the predefined ones in this guide. Rather, a PM adopts the practices that make the most sense for their particular situation. The following is a set of questions intended as a resource to provide Agile practitioners, PMs, and executive leadership assistance in beginning to think about which Agile practices would be best suited for their software development projects. This set of questions is not a prescriptive resource, particularly as Agile continues to change and adapt. Rather, it is a way to begin thinking about the types of questions a PM needs to ask during the beginning stages of Agile project planning. These questions ultimately help lead to an informed decision in selecting the proper Agile practices for the project.

As noted throughout this Instruction Manual , it is rare that a single Agile methodology is implemented and carried out fully. Rather, PMs and Agile practitioners select the best practices from different methodologies, based on project characteristics such as the project goals, the development team's environment, and the ability to report and deliver project outcomes. Accordingly, the following questions do not have "right" or "wrong" answers, but rather identify certain practices that can be linked to specific methodologies.

Project Characteristics

- Does the project follow an iteration-based or flow-based cycle?
- How large is the project? Can it be broken into smaller, more manageable subprojects?
- Does the project have guidance for how it fits in within the greater Department-level portfolio or program? If not, does it need to be scaled to fit in with the larger program/portfolio in the future?
- How will transparency be provided to stakeholders (i.e., to appropriate governance, executives, program managers, other oversight bodies, and business users)?

Customer Involvement

- What is the level of business involvement during the development process?
- Is the customer directly involved as a member of the team, or are their requirements/needs represented by a team member (i.e., through the role of the Product Owner)?
- Is the business dedicated to the success of this project?
- Are customers required to be on site as part of the team?

Team Construction

- How often can the team members communicate with one another (daily, weekly, or otherwise)? Are they able to communicate with one another in an ad hoc manner (without any prior planning)?
- Will the team be collocated? Is communication face-to-face or virtual? If face-to-face, is there a team room?
- Are the team members cross-functional in their skill sets? Are any team members specialists who have no one else with that same skill set on the team?
- Do programmers use pair programming in their coding approach?
- If a team member is absent for a period of time, can work still move on uninhibited, or must it be delayed in production and/or testing of working code?
- What is the plan for cross-training on the team?
- What is the expected size of the team?
- How many teams are expected to work on this project?

The answers to these questions help PMs to determine which Agile practices best fit their project's characteristics, and accordingly aid in determining which Agile methodology is the best fit. As seen in Figure 2-3, project characteristics, the level of customer involvement, team construction, and other external factors are likely to influence which processes and practices PMs select for their specific project needs.

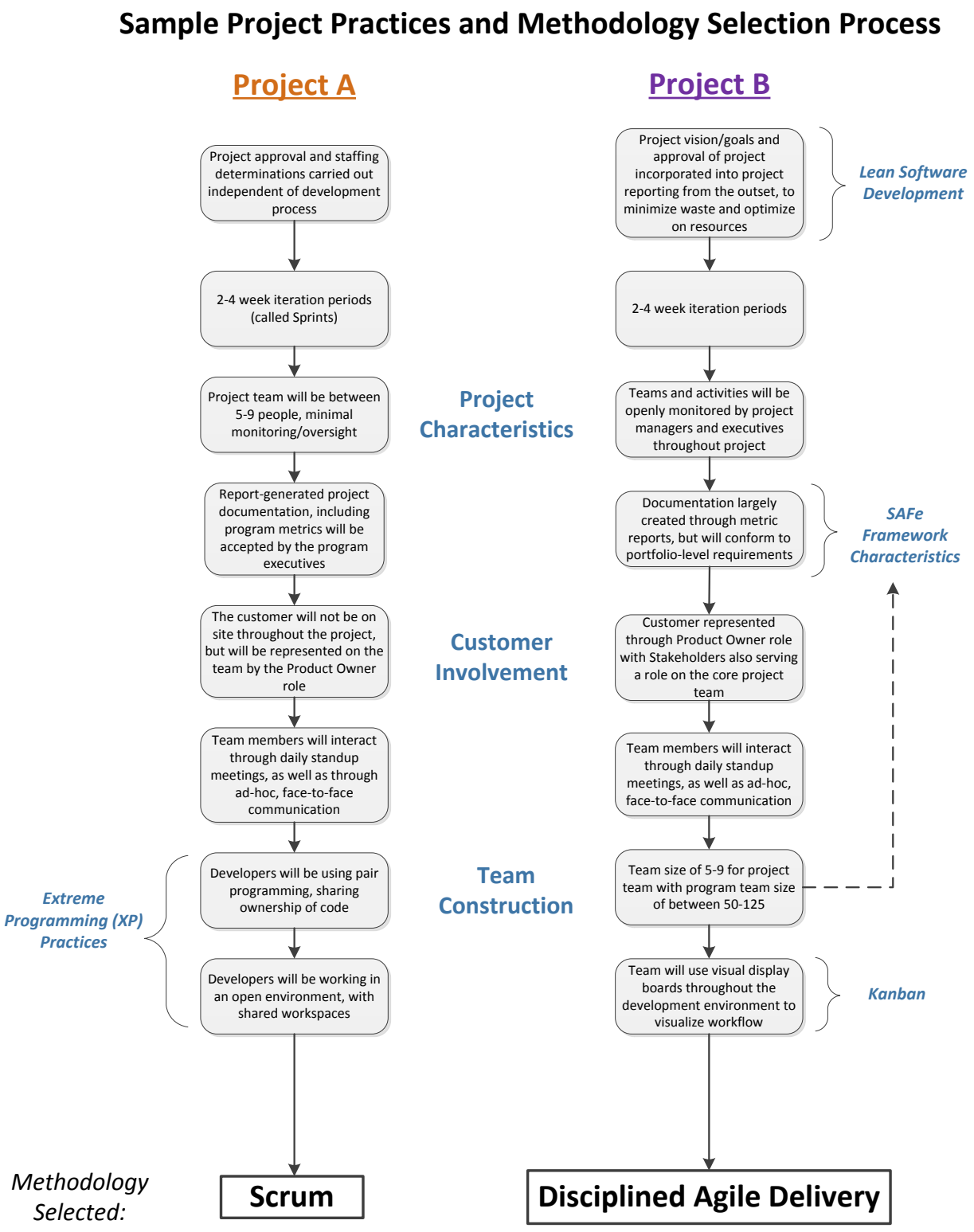

Figure 2-3. Sample Project Practices and Methodology Selection Process

As these visualizations show, selecting the proper practices is not an easy decision to make. Many key questions are asked to create the initial approach. The answers to these questions help Agile project and program managers determine what approach is best to start with for their specific project given its goals, timeline, and team structure. The use of retrospectives allows for the practices to change over time to best fit the project. It is more important to begin work and use retrospectives than to over-analyze to find the perfect approaches for the particular project.

2.4 Agile Teams, Roles, Responsibilities, and Resources

This section outlines some of the possible key roles and responsibilities in an Agile effort. This roles and responsibilities list is not intended to be exhaustive. It includes those roles whose responsibilities differ significantly in support of Agile than in traditional software development. The Agile Development Instruction (MD 102-01-004) provides a detailed list of the roles and responsibilities required to support Agile development at DHS.

2.4.1 Program Manager

As defined in the DHS SELC,[8] a PM is the responsible person who, with significant discretionary authority, is uniquely empowered to plan the program scope of work, life cycle cost, schedule, and performance acceptability levels. The PM is responsible for establishing the project team, and is accountable for accomplishing program objectives or production requirements. PMs need to have a clear understanding of the organization structure they plan to establish in order to clearly delineate responsibility and accountability within their teams. The PM is also responsible for ensuring completion of required artifacts, presenting the business case and status of the project through all phases of review and approval, and managing the performance of the project.

In addition to these reporting, oversight, and governance responsibilities, a PM for an Agile program/project is responsible for fostering an environment that enables the Agile team to succeed. These responsibilities include ensuring proper Agile techniques are employed, obtaining appropriate training and tools, creating a collaborative environment that includes stakeholder involvement, removing impediments, and, in cases where several Agile teams are working on projects within a program, coordinating and integrating the work of these development teams.

Because Agile methodologies rely so heavily on sharing information across the project team, PMs of Agile programs have a strong focus on communications. Two key PM responsibilities are communications with and among the development teams, and transparent communications with the program stakeholders. Agile methodologies often incorporate specific roles focused on communications in various forums. This role may be filled by the PM or by individuals assisting the PM, depending on the scope and needs of the program and development team.

The roles of the Program Manager and Project Manager are both traditional leadership roles within the development environment. In Agile development, however, a shift in the mindset of these leadership roles is needed, as the PMs do not solely serve in a project *control* role, but also in a

[8] Department of Homeland Security, *SELC Instruction Manual* , MD 102-01-103-01, Revision#00, Appendix C (Washington: Department of Homeland Security).

project *support* role. Understanding and carrying out this multifaceted role is key to the successful implementation of Agile development practices and approaches.

2.4.2 Project Manager

If a program is divided into multiple projects, a Project Manager is typically responsible for the day-to-day execution of each project. The responsibilities of Project Managers mirror those of the Program Manager, at a more tactical level, and the terms are used interchangeably in this Instruction Manual.

2.4.3 Product Owner

The Product Owner represents the Government or user community and is responsible for determining what features need to be in the product release. In some cases, the Product Owner relies on a Functional Owner or Value Team (a user or users with specific business expertise or subject matter knowledge) to provide input to determine what functionality-specific features need to be in the product release. The Product Owner is ultimately responsible and is the accountable acceptance authority for the features in a release regardless of the additional input received from the Functional Owner or Value Team.

The Product Owner is responsible for delivering mission value to the agency through the development of the solution. Ideally, in an Agile project environment, the Product Owner serves as the business interface between the end user and the development team, helping to gather and present requirements to the team, and providing insights into those requirements based on the end user's perceived value of the product. Depending on the scope of the project or program, the Product Owner can designate a representative(s) to fulfill this role on a day-to-day basis, taking responsibility for collecting, approving, and prioritizing functional requirements. When using a representative, it is important that the Product Owner delegate the appropriate level of authority and responsibility to make decisions.

The Product Owner (or representative) provides key information on requirements to the development team by discussing, elaborating, and answering questions to ensure that the team fully understands the user stories. The Product Owner/representative ensures that the team products deliver value by fulfilling these requirements. The Product Owner/representative is empowered to make difficult product decisions, including prioritization of user stories in the backlog and which change decisions/options require further refinement before deployment. The Product Owner is responsible for ensuring that requirements prioritization and refinement decisions maximize mission value. Some Agile practices use a Business Value Team instead of a single Product Owner. A Business Value Team is a group of stakeholders who help the Product Owner make decisions about the value of features and functionality for the product. The team is empowered by supporting infrastructure and organizations, is accountable, and is willing to dedicate members to the program as needed.

In some cases, the Product Owner represents the business user community and a separate IT PM represents IT systems ownership; in other cases Product Owners may span both business and IT reporting lines. In all cases, the Product Owner's priority is the delivery of business value.

2.4.4 Development Team

Ideally, Agile development is performed by small teams (approximately five to nine members) of committed, full-time individuals. This cross-functional team is accountable to itself, the PM, and the Product Owner for iteratively and incrementally delivering business value. These team members are responsible for development, testing of the code, configuration management, and other functions. They plan and re-plan work, inspect and adapt products, and deliver "done" products (systems or subsystems that fulfill user stories or requirements and pass operational tests) with each iteration. A key component of the development team's responsibilities is to effectively coordinate and integrate individual or group outputs in order to produce these products, known as potentially shippable increments (PSI).

2.4.5 Test Team

In the Agile context, testers are integrated into the development team. All members of the development team are expected to be generalizing specialists and able to help the team in topics outside their specific areas of expertise. Operating within the development team, testers help to plan and design tests and use cases, develop and maintain acceptance criteria, estimate test execution resource requirements, develop automated test tools and environments if feasible, and participate in post-activity reviews, or retrospectives. In some cases, testers also participate in programming.

2.4.6 Agile Coach

The role of an Agile Coach is part embedded trainer, part consultant, and specifically, the team advisor. The coach works as an advisor to help the team adapt the Agile methodology to their environment, apply Agile and Lean thinking, work through challenges, and remove impediments. Coaches fulfill a supportive rather than a directive role. This distinction is consistent with the Agile tenet that self-governing teams can be more efficient in their operations than those operating in a hierarchical fashion. Thus, the Agile Coach is focused on helping the team to communicate and operate in ways that allow the team to identify and make sound technical decisions, as well as make progress toward product goals. The coach enables, rather than performs, the decision-making.

Identifying and designating an Agile Coach can be a key challenge for the PM. Individuals with prior experience managing Agile programs and projects within DHS are a good source of advice and assistance on this topic. Some PMs and contractors have advised that PMs selecting an Agile Coach to assist with a DHS software development project might consider whether candidates have experience working Agile development within the DHS (or other federal agency) enterprise, as tailored approaches to compliance with federal processes and regulations are keys to success. If a project provides the resources for acquiring an Agile Coach, it is recommended that the PM incorporate one into the project from the outset. Ultimately, as this may be difficult for some projects, one of the most valuable resources available to PMs and Agile practitioners is the knowledge and experience of DHS peers who have managed Agile programs and projects. Many of these individuals can be found on the Agile Center of Excellence and IPT (contact the EBMO SELC/Agile Team at EBMOSELCTeam@HQ.DHS.GOV for information on participating).

2.4.7 Additional Expertise

The overall project team often requires expertise in a variety of technical, subject matter (business area), and regulatory, contracting, and other topic areas. PMs are encouraged to consider the full range of skills and knowledge required, as well as critical points for application of such expertise, in developing the project team(s). In enterprise environments, extended teams may provide additional expertise on an as-needed basis. In the DHS context, this extended team might include documentation specialists, enterprise architects, operational testing specialists, information security and privacy specialists, experts in system accessibility requirements, quality assurance analysts, and compliance validation/verification personnel. Establishing and maintaining open communications and cooperative operations across the extended team is crucial to the Agile approach and improves the success of the program.

2.4.8 Oversight Bodies

The PM must coordinate with the various oversight bodies that govern IT development. These bodies vary depending on the level of investment. For major investments, DHS Executive Steering Committees (ESC) are often established to oversee all aspects of program planning and execution between major acquisition decision events. This authority extends to assisting programs to adopt Agile methodologies and acquisition strategies where appropriate. Many DHS components have their own ESCs consisting of component Chief Information Officer (CIO), Component Acquisition Executive (CAE), and appropriate business and IT representatives that provide oversight for non-major investments.

Specific recommendations on the integration of key functions within DHS Agile product teams are addressed in Sections 3 and 4 of this Instruction Manual.

2.5 Success Factors and Risks

The discussion of the Agile Manifesto in Appendix A provides information on the conditions necessary for successful application of a "true" Agile development approach. At DHS, PMs develop program plans and teams that can successfully meet mission needs under changing conditions. The discussion below highlights a few of the key success factors and risks for which a PM considers and develops an approach for before establishing an Agile team at DHS.

2.5.1 Stakeholder Engagement

Agile efforts place great emphasis on continuous interaction between the development team and product owners. Before establishing an Agile project team, all DHS stakeholders (identified in Section 2.4) need to clearly understand and be committed to their roles in the Agile approach. If stakeholders are not able to be immediately involved, it is possible for someone to represent them during the project if agreed to by both the stakeholders and the PM.

2.5.2 Contracting Environment

Among the issues that may arise is a lack of contracting experience with Agile development approaches, and a resultant lack of flexible, rapid contracting options to keep pace with the desired

Agile development plan. This challenge has been recognized at the federal level,[9] and efforts are underway to improve training and expertise.

Integrating contractor and government staff members within an Agile development team, while maintaining the necessary distinctions and constraints upon decision-making authority, is another challenge that the PM considers in establishing the project team. Similarly, identifying and validating contractors with the required expertise in Agile methods may prove a challenge, as Agile performance standards are still evolving. To help address this, the Office of the Chief Procurement Officer operates the DHS Procurement Innovation Lab, which is aimed at experimenting with innovative techniques for increasing efficiencies in the procurement process and institutionalizing best practices. Under this framework, a project team may volunteer its acquisition project as a test subject and, within a safe and controlled environment, apply creative source selection techniques that can shorten time-to-award and reduce burdens.

2.5.3 Common Reporting Techniques and Metrics

Common reporting techniques and normalized metrics are required to ensure the project can measure and report its status in a way that is easy to maintain and is understandable by management and oversight bodies. See Section 4.4 for a full discussion of Agile tools, processes, and metrics.

2.5.4 Resource Availability

In theory, most Agile methodologies assume the dedicated involvement of all stakeholder, developer, and integration staff throughout the project. Due to resource and staffing limitations, this often is not the case. These realities pose risks for PMs to consider when planning and managing an Agile project. Related topics for consideration as possible risk areas are the impact of Agile development on the supporting IT infrastructure, and on the processes used to validate releases against applicable compliance requirements. The following paragraphs include additional considerations:

Team Maintenance: Human Resource Availability

Agile methodologies benefit from the continued availability and involvement of high-performing teams with expertise in the required topics (technical and/or business). Over the course of a large program, however, some staff turnover is inevitable. This increases the need for cross-training and generalizing specialists so that the team can better adapt to potentially disruptive personnel changes. Although the techniques for managing this risk are not unique to Agile teams, the importance of mitigation actions may be greater, as Agile teams are typically smaller and delivery cycles shorter than with traditionally managed projects. Careful timing of personnel transitions to align with code release cycles may help mitigate this turnover-related risk; Agile's short delivery cycles offer opportunities to ease these staff transitions.

[9] White House, *Contracting Guidance to Support Modular Development* (June 2012).

Infrastructure Requirements: Ability to Support Frequent Deployments

In the enterprise environment, Agile's emphasis on rapid delivery and iterative expansion of capability increases the importance of coordination with enterprise infrastructure providers, who may not be accustomed to sudden surges in demand for resources. Interfacing early and often with enterprise infrastructure providers, specifically in regards to testing, increases the project's likelihood of success. If the Agile development team "outruns" the testing or production environment capacity, progress on the product may suffer. For this reason, the operations and infrastructure teams are often incorporated into the development team to provide one clear vision of the overall ability to deliver functionality to the users. In many Agile projects, automated testing and deployment are integrated into the project lifecycle to improve efficiency and reliability.

Tailoring and Compliance: Adapting to Iterative Releases

The establishment of an Agile program does not absolve that program or end product from meeting overall DHS and federal government IT compliance requirements; however, the DHS SELC framework is explicitly intended to allow PMs the freedom to tailor their approaches for meeting the intent of the governance requirements. Agile satisfies federal guidelines by tailoring standard compliance processes and activities to meet DHS regulatory guidelines as efficiently as possible. The exact parameters of this tailoring are likely to be program specific, and the selection of alternative activities and reviews may be selected in accordance with the specific Agile practices or approach used. Compliance validation processes, such as security and privacy reviews, are in the process of adapting to Agile development methods and support the rapid delivery of value to the business. The appropriate team members/leaders need to be engaged early and often in the development process and continue to be involved in active discussion with the governance, oversight, and compliance bodies throughout the project life cycle.

2.6 Culture Change

The use of Agile methodologies represents a significant shift from traditional development and management approaches to those more suitable to the dynamic nature of technology and software projects. Appropriately applied, Agile practices enable greater transparency in project progress by highlighting potential issues earlier and providing more timely indicators on cost, scope, and schedule to PMs and stakeholder decision makers. Success in applying Agile techniques does depend, however, upon committed participation from all levels of the project governance and implementation structure.

Industry data indicate[10] that resistance to changing the organizational culture remains the single largest barrier to adoption of Agile. Motivating change and project management during periods of organizational and process evolution are likely to remain significant challenges for PMs leading Agile projects, particularly within the federal environment, which relies heavily on organizational and regulatory processes.

[10] "8th Annual State of Agile Survey," VersionOne.com (2014), http://www.versionone.com/pdf/2013-state-of-Agile-survey.pdf.

As noted previously, OMB guidance[11] encourages the use of modular development, and the *DHS Agile Development and Delivery for Information Technology* instruction states that Agile is the preferred IT development approach. This Instruction Manual , along with recently published DHS policies and implementing documentation, such as the SELC Instruction Manual, provides a starting point for DHS PMs to devise their own approaches for using Agile to improve both the performance and efficiency of DHS IT products to meet mission needs.

[11] Kundra, "25 Point Implementation Plan."

3. AGILE AND THE PROJECT LIFE CYCLE

3.1 Integrating Agile Practices into the DHS IT Governance Frameworks

DHS PMs employ Agile within the context of the existing DHS Acquisition Life Cycle Framework (ALF) and SELC. The *SELC Tailoring Examples for Selected Types of DHS Acquisition Programs*[12] is a key reference for PMs to review when determining to develop using Agile techniques.

As stated earlier, the establishment of Agile as the preferred approach for DHS IT programs does not contradict ALF and SELC guidance. Application of Agile provides a means to use the inherently tailorable nature of the SELC to improve program outcomes by increasing speed, efficiency, and collaboration for programs that elect to use this approach.

Although the DHS SELC framework is typically depicted sequentially, it is intended to be neutral with respect to development methodology. The activities required by the SELC may be conducted concurrently, in parallel, or sequentially, with multiple feedback loops and iterations, according to the needs and characteristics of the program. Similarly, SELC technical reviews and associated artifacts may also be combined, modified, or omitted based on a program's specific characteristics and selected development methodology, assuming the tailored path has been approved through the proper governance processes. As noted in the DHS SELC Instruction Manual [13], the philosophy of the SELC is to encourage tailoring for specific engineering needs and to accommodate all development methodologies.

The PM, in collaboration with the Component Acquisition Executive (CAE), Lead Technical Authority (LTA), Lead Business Authority (LBA), and other project stakeholders, is responsible for determining how to best structure a program for success, including tailoring the development approach to satisfy the intent of the SELC framework. This is the case for both traditionally managed projects and those applying Agile practices. However, since options for applying Agile within large, well-defined governance structures are still evolving, PMs planning to adopt Agile practices may need to spend additional time on and pay careful attention to developing and acquiring proper and skilled support for their program plans to achieve successful outcomes.

3.1.1 Assessing the Project

Because DHS has declared Agile to be the preferred approach for software development projects, the nature and circumstances of the project at hand are important considerations for the PM. A PM focuses on developing a plan that enables the project to deliver usable functionality incrementally in accordance with OMB and DHS guidance. The PM brings together information and expertise to evaluate which Agile practices are to be applied for the full scope of the project or for some portions (subprojects) of the overall effort (e.g., in the case of mixed lifecycle projects where the acquisition of commercial-off-the-shelf [COTS] software or hardware constitute a portion of the effort).

[12] Department of Homeland Security, *SELC Tailoring Examples.*
[13] Department of Homeland Security, MD 102-01-103-01.

Consulting an Agile Coach during this assessment phase can also help the PM to identify specific resources and support needed to successfully apply Agile methodologies to the project.

The key assessment the PM makes is what areas of the organization have to be addressed in order to successfully commence Agile IT development to maximize delivering improved capabilities more often with reduced risk. The following questions help the PM to identify areas where altering current organizational practices, tailoring standard processes, or redistributing resources may improve the probability of success. Although it is not feasible to identify all possible factors and project characteristics that may increase or decrease the probability of successfully commencing Agile, these questions are a good starting point:

- As the project planning begins, are the stakeholders (especially the Product Owner and infrastructure owners) supportive of an iterative and incremental approach to delivering functionality, and committed to frequent, active interaction with the development team?
- Is the environment suitable for iterative/incremental product releases? What would need to change to make it more suitable?
- Does the working environment (i.e., conference rooms, information-sharing tools, office layout) support the frequent communications and extensive collaboration key to Agile?
- Will the program provide team training, Agile expertise/coaching, and other required resources, or is there experience on the team in Agile development?
- Can the overall project effort, segmented into smaller, more manageable increments as needed, be effectively managed by a small, collaborative team?
- Do the existing SLAs for maintenance and infrastructure support the flexibility required for Agile deployments? Note that smaller, more frequent, iterative deployments will impact maintenance outage windows, code checkout support, more release transitions to production, and more corresponding infrastructure upgrades to keep environments in synch.
- Do the contract vehicles in place allow for iterative product delivery with many small deployments and the testing, approvals, and documentation required for each?

3.1.2 Identifying Opportunities to Apply Agile Practices

Since there is no single "right" way to apply Agile philosophies and methodologies, PMs can adopt and adapt the Agile practices that best suit their programs and environments. Similarly, a PM need not seek to apply Agile in an all-or-nothing manner. PMs can begin, if appropriate, by identifying subcomponents of the project where Agile methodologies may be most successfully applied, and build on that experience. Those initial subprojects may serve as an opportunity to test new methodologies or develop Agile team skills.

Another opportunity for applying Agile is in refocusing projects that are already in progress. In this case, PMs may assess and identify underperforming subprojects or goals that might benefit from applying an Agile approach. PMs restructuring projects should examine the factors underlying performance issues and identify opportunities to apply Agile to resolve those issues.

Although managing some subprojects traditionally while applying Agile to others may be beneficial in many circumstances, the use of a hybrid approach may also bring complications. Careful planning is required to support successful integration of the subprojects. Having a knowledgeable Agile Coach advising on coordination and integration efforts greatly enhances the team's ability to execute a hybrid approach.

3.2 Options for Tailoring the Systems Engineering Lifecycle Activities

In determining how the SELC process might be tailored to improve project outcomes, the PM assesses not only the project and how Agile might be applied, but also how and which specific Agile practices and tools can produce the information required to support DHS governance processes. Once the PM and stakeholders determine the overall "tool set" of practices, tools, and expertise available for use in managing the project goals and resources, they can work together to develop a project strategy and options for tailoring the SELC to match the strategy.

3.2.1 Reference: Agile Terms as Used in this Instruction Manual

To clearly address options for SELC tailoring to incorporate Agile approaches, it is necessary to introduce some terminology. The information provided below is not intended as a limiting or authoritative set of definitions; some methodologies give slightly different, potentially context-dependent meanings to these and other terms. This terminology is intended to provide a broad understanding of these key Agile concepts for the purpose of clarifying the content of this Instruction Manual.

- ***Epics*** are very large user stories that are eventually broken down into smaller stories. Within the SAFe® framework, ***business epics*** describe top-level business processes. ***Architectural epics*** describe the architecture the system is incorporated into. **Note:** See the *SELC Tailoring Examples for Selected Types of DHS Acquisition Programs* for additional details on business and architectural epics
- ***User stories*** are the basis for a conversation about requirements. They are a feature and/or unit of business value that can be estimated and tested. Stories describe work that must be done to create and deliver a feature for a product. Stories are the basic unit of communication, planning, and negotiation between the Team, Stakeholders, and the Product Owner.[14] They should provide business value at levels just low enough for the development team to design solutions (over-definition is discouraged to enable innovative, efficient design). User stories are usually presented in the format "As a <role>, I want <goal/desire> so that <benefit>." User stories are not a set of detailed requirements, but rather imply a conversation between the Product Owner and the Development team to ensure all involved understand exactly why work is being done and for what purpose. Importantly, the Product Owner should not define *how* a User story should be implemented, only *what* will fully meet the Product Owner's needs

[14] "Agile Glossary," SolutionsIQ (2015), http://www.solutionsiq.com/agile-glossary/

- An ***iteration***, within the context of this Instruction Manual , refers to a time-boxed (fixed length) period of work performed by the Agile team. In Scrum and some other methodologies, ***iterations*** are called ***Sprints***
- The ***backlog*** is the collection of ***stories*** (or ***epics***) remaining to be addressed by the Sprint team. The backlog is generated for the overall product, each individual release (showing the number of user stories remaining for a specific release), and each iteration/sprint (identifying the remaining stories for a specific iteration/sprint)
- A ***release*** is a delivery of software from development into a production environment for use. A ***release*** may incorporate the product(s) of one or more Agile team ***iterations/sprints***
- ***Release planning*** is a set of team activities focused on understanding the scope, constraints, process, practices, and required or prioritized products of a ***release***, as well as developing a plan for successfully completing the ***release***. The ***release plan*** also documents the testing approach for the ***release.*** **Note:** See the *DHS Technical Review Guide* for additional details on release planning and the RRR
- ***Iteration* or *sprint planning*** is a two-part meeting. The purpose of the meeting is to identify, prioritize, and commit to the stories for the next sprint. Part one of the sprint planning meeting is a review of the product backlog. This is when the product owner describes what needs to be built for the next sprint. During this part of the meeting, the team discusses the sprint objectives with the Product Owner, and asks clarifying questions about the stories. During part two of the sprint planning meeting, the team decides how the work will be built and begins decomposing the stories into work tasks and estimating story size
- ***Release planning reviews*** are conducted prior to the execution of each ***release***
- ***Integration testing.*** The vast majority of software defects are discovered during integration of the whole system. It is critical that software code pass both unit and integration tests before being considered done. Optimally the integration testing is automated and can be run multiple times an iteration, even multiple times a day
- An ***Iteration review and demo*** (for Scrum, ***Sprint review*** and ***demo***) is conducted at the end of each ***iteration***, providing a hands-on review of the working software that was produced during that iteration. The iteration review meeting is an informal meeting. It should not become a distraction or a significant detour for the team; rather, it should be a natural result that signals the end of a sprint
- A ***release readiness review (RRR)*** is conducted prior to releasing software to the production environment. The ***RRR*** is focused on ensuring all elements of the release are complete, including testing, documentation, approvals, accreditations, and provisions for sustainment. The ***RRR*** includes the major stakeholders, who also assess the ***release.*** **Note:** See the *DHS Technical Review Guide* for additional details on release planning and the RRR
- A project ***roadmap*** describes a set of planned ***releases***, along with high-level information about the associated scope, goals, and schedules of the ***releases***. The ***releases*** all may be performed by a single project team, or, for a large-scale effort, by a "team of teams." In the

latter case, the ***roadmap*** is used to support ***release*** integration planning. As with most Agile artifacts, the ***roadmap*** is intended to be revised iteratively, as the project plan evolves

- ***Retrospectives*** are time boxed meetings held at the end of an iteration/sprint, or at the end of a release, in which the team reviews and reflects on the previous sprint or release to determine what succeeded and what could be improved. The retrospective is key to an Agile team's ability to inspect and adapt in the pursuit of continuous improvement

The above terminology list is intended to familiarize PMs and Agile practitioners with common Agile terms. Section 3.2.2 discusses common Agile practices in more detail, providing an opportunity to show how specific terminology is linked to various development or data tracking practices in Agile.

3.2.2 Agile Practices and Data Tracking

While face-to-face communication and discussions are preferred, some artifacts are required. These artifacts need to follow Lean principles and minimize waste and repetition, building just enough for the current needs. The artifacts are traditionally updated for specific project assessment and approval reviews, and are focused on the contents of a specific iteration. Accordingly, the overall project artifacts are built iteratively, just as the software is. Agile methodologies, with their emphasis on open communication and collaboration with stakeholders, encourage the continuous review of project activities and achievements. The ready availability of up-to-date, transparent project information helps to minimize the risk of "surprises" or miscommunications between approval reviews.

Since Agile practices rely on regular reviews of plans and results, many key Agile tool sets are focused on maintaining a clear record of project plans, results, rates of progress, and decisions. Although the immediate use of these records is to help the Agile team stay on track (identifying and resolving issues or adapting to changes rapidly), these records naturally focus on many of the same considerations that are assessed in formal governance reviews. Thus, the information gleaned while applying Agile methodologies is readily applicable to supporting formal and informal project reviews.

Figure 3-1 shows many of the most widely recognized Agile practices. As these practices are being phased into Agile development efforts at DHS, PMs are using the outputs of these practices to continuously monitor and report on project health and to identify opportunities for improving project team performance. Each column represents a focus area (i.e., collaboration, planning/adapting) where Agile produces value. Beneath the figure are short descriptions of the most common Agile practices to provide further context.

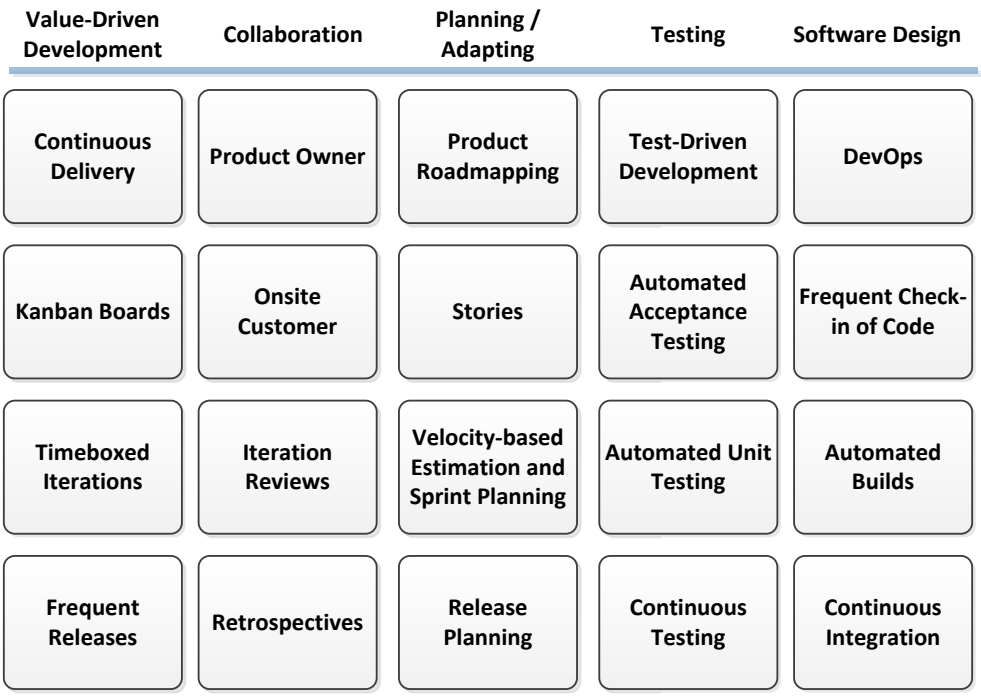

Figure 3-1. Agile Practices Most Commonly Used in DHS

Continuous delivery is the practice of delivering completed pieces of software throughout the development life cycle. These pieces may not be identical in size, complexity, or scope; rather, they serve as a unique part of the final development solution. Continuously delivering software throughout the project timeline also allows PMs to better evaluate project progress and the value of user stories delivered during a given iteration.

Kanban boards are used to help teams visually see the remaining tasks to be completed on boards or walls within their development environment. These boards help teams to envision how much work is being completed at each stage of a sprint/iteration or project in order to eliminate bottlenecks and enhance the progress of development activities.

The practice of *time-boxed iterations* aids in ensuring that project builds have well-defined end points, helping to keep projects on track. An *iteration review* is held at the end of any iteration and records the progress achieved and functionality delivered during the iteration.

Frequent releases not only provide functionality to the Product Owner and users, but also serve as stable building blocks for subsequent deliveries.

The *Product Owner* is the representative of the user community, and serves as the business interface between the end user and the development team. The Product Owner determines which features need to be included in each product release. Similarly, having an *Onsite Customer* gives the development team access to the end user or their representative (in some cases, the Product Owner), who possesses the authority and ability to inform the prioritization of requirements and

make timely decisions. Continuous engagement with the Product Owner is essential to the success of the project.

Iteration reviews are held on the final day of a given iteration to demonstrate the features and functions completed during that iteration, as well as to provide an opportunity for evaluations of the product and the value delivered during that specific iteration. Iteration reviews are sometimes referred to as *Iteration Demonstrations*.

Retrospectives, conducted after iterations or releases, provide opportunities for the project team to identify and record what was successful about the iteration or release, what could be improved, and how to incorporate lessons learned and improvements in future iterations and releases. Information from these retrospectives is often valuable in satisfying governance needs.

Product road mapping represents the creation of a view of the product's business needs/gaps and informs the planning of the development efforts. PMs identify the requirements, sort them by the features they will ultimately support, and prioritize each set of requirements for each feature. The Product Roadmap also includes high-level time frames for increments. These initial time frames are malleable, and PMs need to anticipate that these will fluctuate from original estimates as the product continues to be developed.

Stories, or *user stories*, are the requirements for the solution. These need to be at a low level, but not have too much definition in order to allow the development team to create the best possible solution, and prevent the outcome from being too pre-defined.

Velocity-based estimation and *sprint/iteration planning*, along with *release planning* activities, incorporate information from *user stories* that are to be addressed in the sprint or release and are used to develop plans for the work remaining and informs the successful completion of the sprint/release. These plans, continuously updated by the project team, add to the overall program information store and seed the team's iteration backlog. The velocity-based estimation process plans for the amount of time and effort required to complete the backlog, based on previous backlog sizes and successful completion rates.

Test-driven development (TDD) is a practice in which developers create the test to evaluate their code first, and then build just enough of their code to pass the test before moving on to the next piece of software. The TDD practice motivates developers to write effective, functional code that can be adjusted as needed during later refactoring processes (re-coding without changing function).

Automated acceptance testing is a practice where tests are written typically at the beginning of an iteration and are an executable form of the requirements. They are detailed examples of how the system is supposed to function when the requirement they describe is complete. A user story is not considered complete or successfully addressed until it passes its acceptance test(s).

Automated unit testing evaluates specific fragments of code by evaluating them with control data and operating procedures to determine whether the code is properly written and fit for integration

into the larger development solution. This testing is typically carried out by pre-defined code testing procedures (thus, automated) that can be replicated each time new code units are evaluated.

The plans for and results of the *continuous testing* in Agile projects are also recorded along with each release. This data provides not only key documentation for the overall project, but also information vital to project review and approval events. Testing is implemented as the development occurs within each iteration, rather than solely at the completion of the project.

DevOps, both known as a practice and an approach to Agile development, focuses on the relationship between "development" and IT "operations." DevOps uses common automated testing and development processes in order to help enhance the efficiency and maintainability of existing operational processes. DevOps supports release management by standardizing the environment in which software is developed, allowing for better tracking of progress and resolving of issues.

Developers who practice the *frequent check-in of code* enhance the collaborative environment by allowing other developers to access the code more regularly, which increases the visibility of any individual team member's progress. This practice is common with teams using an online development environment where code is stored, reviewed by team members, and subject to change by more than one person in short periods of time. It also enhances teams' ability to manage different versions of code and reduce any potential issues with version control.

As a software design practice, *automated builds* are a way to more efficiently compile computer code, run tests, deploy software to production, and create documentation. Automated builds specifically link code together in a pre-defined manner, rather than compiling or linking source code manually within build scripts.

Finally, *continuous integration* is the practice where delivery teams frequently integrate their code (e.g., hourly, or at least once daily) into a shared master copy. Each integration is verified by an automated build, which also performs testing to detect any integration errors quickly and automatically. The practice of continuous integration leads to the rapid release of high quality software.

3.2.3 Project Strategy and Structure

As discussed previously, all development, including Agile, requires up-front planning. As such, this subsection addresses:

- Establishing a product vision
- Determining project type
- Development of epics to outline how project aligns with mission needs and requirements
- Determining if project is a single effort, or if it decomposes into multiple projects
- Developing the project roadmap

The SELC framework is typically depicted and discussed sequentially, but SELC phases need not be executed in a strictly sequential, single-pass manner. The nature of the project and the project

environment largely determine the opportunities for tailoring the SELC phases to increase the efficiency and benefits of the project.

During the Solution Engineering phase of the SELC for an IT project (i.e., prior to Acquisition Event (ADE)-2A), the PM focuses on establishing a project vision rooted in the Mission Needs Statement (MNS) and tailored to the relevant business practices and processes of the product owner organization. This project vision also addresses the underlying architecture and related environmental considerations. During this phase, a determination is made as to whether the project consists of a commercial-off-the-shelf (COTS) solution, a modified COTS solution, a non-developmental item (NDI), a purely developmental solution, or a solution that incorporates some combination of COTS (modified/unmodified) and developmental components. The preferred solution type helps to determine the feasible SELC tailoring options.

The preferred solution chosen, along with the business practices and processes, allows the development of higher-level epics. These epics, higher-level user stories that are comprised of the work done in multiple iterations, and the associated operational performance requirements as documented in the Operational Requirements Document (ORD), define the overall vision to meet the needs defined in the MNS. The epics, along with the ORD, constitute the project backlog and represent the portfolio of work that the program is expected to deliver within the program's schedule and budget, which is documented in the Acquisition Program Baseline (APB).

As part of preparing for (or immediately following) ADE-2A, the PM and stakeholders determine if the approved capability is to be divided into discrete and separate projects or if the entire capability is to be managed as one project. The initial Capabilities and Constraints (CAC) document is then developed to describe the capabilities required, as derived from the MNS, ORD, and the business and architectural epics. If the effort is divided into separate projects, each project is responsible for some portion of the overall project backlog. In some cases as part of a larger program, these separate projects require an ADE-2B. Specific APBs may be required for each subproject, or the APB may be updated to reflect that delivery of the approved capability requires completion of all subprojects.

The PM also develops a project roadmap to outline and coordinate capability delivery schedules across all organizations involved in the specific project. The roadmap schedules an initial release, scoped to deliver a "minimum viable product" (MVP) even if some desired features are not included. Subsequent releases build upon the MVP. Each release is described in terms of its objectives and the stories or epics it addresses. The PM works with the Product Owner/user stakeholder to continually reassess and update the project roadmap as the project progresses and release strategy evolves.

3.2.4 SELC Tailoring Plan Development

The SELC Tailoring Plan is developed during the planning phase of the project and documents the technical and management strategy. In addition, the SELC Tailoring Plan identifies the Agile practices to be applied, as well as how the PM applies the Agile approach to satisfy the intent of the SELC activities, artifacts, and reviews. The phases of the SELC most likely to be tailored and met by

Agile approaches are requirements definition, design, development, integration and test, and implementation. The SELC Tailoring Plan is developed, submitted, and approved to support the ADE-2B decision (or multiple ADE-2B decisions at the subproject level, if applicable). In addition to the SELC Tailoring Plan, the project vision, backlog, roadmap, Test and Evaluation Master Plan (TEMP), and the Program Management Plan (PMP) or equivalent also are prepared for and presented at the Project Planning Review (PPR) in support of the ADE-2B decision.

Many Agile processes and artifacts correspond closely to those used in the SELC. Additionally, Agile emphasizes using minimal and incrementally-developed documentation to meet oversight needs. The artifacts produced by an Agile project are often satisfactory in meeting the tailored SELC oversight requirements so long as agreed to by the appropriate authorities at the project outset. For example, each Agile project release features subsets of overall system artifacts, including a release-level CAC, backlog, and System Design Document (SDD), each of which is updated throughout the release cycle and provides traceability to the ORD. At the completion of each release, this collection of information meets the intent of the SELC Functional Requirements Document (FRD) and System Requirements Document (SRD) for that release. See the *SELC Tailoring Examples for Selected Types of DHS Acquisition Programs* for additional information on the requirements, documentation, and accountability that may be required of specific types of Agile projects.[15]

Given the deconstruction of the overall project into Agile project releases, each with its own supporting documentation, each successfully completed RRR may constitute an ADE-2C for the content of that release. When ADE-2C is delegated to the Component Acquisition Executive (CAE)/Executive Steering Committee (ESC), it is combined with the RRR. If, during the RRR, or due to changing priorities, it is determined that the release scope alters significantly from plan or the release metrics raise concern with respect to execution or scope, the ADE-2C is escalated back to the applicable Acquisition Decision Authority (ADA).

The SELC reviews between ADE-2B and ADE-3 may be tailored and satisfied by Agile iteration and release reviews and documentation. See the *SELC Tailoring Examples for Selected Types of DHS Acquisition Programs* for additional information on the Agile SELC Tailoring Options from ADE-2B to ADE-3.

3.3 SELC Tailoring Considerations

Since Agile is intended to enable project evolution over time, the project plan requires continuous assessment and regular updating. The PM revises the SELC tailoring plan to address project changes and schedules regular reviews of the tailoring plan with key project stakeholders and approving authorities. In addition, preparing for and executing the tailored project review events may reveal process or documentation deficiencies that require adjustment to the Agile tools being used by the team, or to project team practices for assessing, recording, and taking action based on iteration and release data. After each SELC review, the PM conducts a brief retrospective with key

[15] Department of Homeland Security, *SELC Tailoring Examples for Selected Types of DHS Acquisition Programs*.

project team members, focused on identifying process improvements or tailoring plan adjustments to streamline future reviews.

Some primary considerations for the PM in tailoring SELC reviews and documentation include the following:

- In most cases, SELC review documentation is updated and revisited incrementally and repeatedly. This practice helps to ensure that current, comprehensive information is readily available for both team use and stakeholder review as needed. For this construct to succeed, PMs need to socialize the concept of repeated updates and iterative reviews, including component-level reviews across different SELC phases, among the stakeholder community.
- PMs need to ensure that the program team has the resources required to track and maintain records of project progress. The information requires continual updating and accessibility in a form that supports development of review artifacts with minimal effort.
- The program's SELC tailoring plan incorporates an explanation of how the intent of the SELC reviews is met by the information recorded during execution of Agile practices. Since information formats for the tailored reviews may differ from those that review and approval officials are accustomed to, stakeholders need to understand the tailored formats and iterative delivery plans to ensure a fair assessment of these products for SELC reviews.

The Agile emphasis on active Product Owner and stakeholder communications and collaboration throughout the program phases helps to familiarize decision makers with the in-progress requirements tracking and refinement tools, and reduces the need to generate large, stand-alone documents for the technical review and approval events. This, in turn, improves the efficiency of review events, and helps to accelerate program decision processes.

3.3.1 SELC Tailoring Communication

As noted in the *SELC Tailoring Examples for Selected Types of DHS Acquisition Programs*,[16] an SELC tailoring plan includes the following:

- Specific proposals for elimination, combination, or addition of review events
- Justification for each of these proposals; for each proposed review elimination or combination, a description of how the modified process satisfies the intent of the standard SELC reviews being eliminated or combined
- Specific proposals for substituting data and reports from team activities (for example, Agile tool data, retrospective documentation, etc.) for standard-format SELC artifacts
- Clear evidence that the objectives of the overall technical review process are still being met

Establishing SELC tailoring examples serves as effective justification for some tailoring options.

[16] Department of Homeland Security, *SELC Tailoring Examples for Selected Types of DHS Acquisition Programs*.

(This page intentionally blank)

4. APPLYING AGILE WITHIN DHS

Expanding the use of Agile development practices within DHS requires educating the entire organization, partly by sharing and taking advantage of the expertise and experience gained in Agile projects to date. This section is rooted in the guidance provided in the DHS Instruction *Agile Development and Delivery for Information Technology* (MD 102-01-004) and incorporates insights derived from interviews with DHS PMs who have used Agile approaches. As reflected in the information that follows, DHS executives and PMs are continuing to explore the best ways to apply Agile tools and methodologies to challenges within DHS.

DHS Instruction *Agile Development and Delivery for Information Technology* states that the Chief Procurement Officer (CPO), DHS and Component CIOs, and CAEs are jointly responsible for encouraging, overseeing, and evaluating progress in applying Agile approaches. The DHS CIO is developing policies and procedures for applying Agile to DHS IT programs and is working with the CPO, CAEs, Director of Test and Evaluation, and Component CIOs to tailor DHS IT development processes to improve outcomes and meet OMB guidance. Since DHS and Component-level policies, procedures, and enabling processes for Agile development are still evolving, IT project PMs need to research and review emerging guidance, best practice assessments, and tools on an ongoing basis.

Advance Planning Provides Benefits for Programs

One DHS PM who successfully applied Agile noted that the planning for a first-time Agile project frequently takes longer than anticipated, but is critical to ensuring that time and energy are not wasted during the development phase.

This Agile project team researched methodologies and tools in advance to determine which approaches, or hybrid approaches, would work best in their environment for their product owners, stakeholders, and developers.

The team also began with a brief project startup period, followed by an assessment session that identified considerable modifications and improvements to its initial approach.

The DHS PM credits honest feedback from assessors and the team's desire for improvement with helping to develop a high-functioning Agile software development process.

4.1 Establishing the Project Plan

As with traditional management approaches, each project to be undertaken is rooted in a vision that defines the mission need and capability gaps to be addressed, and ultimately represents the project scope. The vision begins with the MNS and evolves to align with the relevant business practices, methods, processes and supporting infrastructure. The MNS, Capability Development Plan (CDP), and Study Plan Review (SPR) are the current artifacts used to identify the gaps and how a solution will be identified. They also identify the associated business practices, processes, and key players. Using Lean practices, the content of these artifacts needs to be preserved, but within an Agile program, the representation may change or be minimized according to what is most efficient for the project. The development of these project scope descriptors may also be refined using Agile practices and/or tools.

While fleshing out the project scope, the PM also works with key stakeholders and an Agile Coach (if available) to determine what Agile practices can be applied. In answering the project assessment questions outlined in Section 3.1.1, the PM works with the CAE, Lead Technical Authority (LTA), Lead Business Authority (LBA), and other stakeholders to determine how best to structure the project and develop a plan for applying Agile processes across the project lifecycle.

Key topics for the PM and stakeholders to consider when developing the project plan include:

- Agile methodologies and tools best suited for use on this project, given:
  - Project characteristics
  - Project environment
  - Available resources
  - Previous experience with Agile
- Working relationships and partnership practices that need to support an Agile approach, including:
  - Executive support
  - Cross-organizational personnel support
  - Information dissemination
  - Timely decision processes
- Additional resources or support that may be required:
  - Agile coaching (industry expertise and/or experienced DHS personnel)
  - Agile training (at individual and team levels)
  - Agile management tools (internally developed and/or commercial software)
  - Automated testing tools
  - Infrastructure support for Agile software build and test processes
  - Team workspace (if co-located) and space for team meetings
  - Communications support (e.g., web-enabled, for geographically distributed teams)
- Incremental project structure determinations:
  - Operational requirements are decomposed into Agile epics and user stories

Characteristics for Success

DHS Components that have applied Agile development methods with multiple teams had the following similarities:

- They had an Agile champion in the executive level of management
- They began with one Agile team and built on its experience, evolving Agile processes for tailoring the SELC and governance reviews
- They had Business Owner involvement
- They provided Agile training for the entire organization; when domain experts (e.g., architecture, security) were needed, those experts understood their Agile team roles and expectations
- They had participation from enterprise architecture, contracting, and relevant compliance offices
- They were aware of the need to socialize the Agile philosophy early and repeatedly

  - Relative prioritization of epics and user stories
  - Identification of key dependencies
- Project reporting and documentation practices:
  - Accommodates both routine and ad hoc queries
  - Provides appropriate level of information for the various audiences
  - Provide appropriate details to target key decision points
  - Provide ample and opportunities for trend analysis

4.1.1 Emphasizing Flexibility in Agile Project Planning

Agile methodologies are specifically designed to enable the evolution of the specific requirements underlying a given capability. Although this flexibility greatly enhances the probability of successful product outcomes, maintaining project flexibility may require multiple stakeholders and partner organizations to adapt or alter their established business practices. The need for careful, collaborative feedback with all project partners cannot be overemphasized. Feedback from DHS PMs experienced with Agile projects suggests that cross-organizational relationships need to be continuously managed and adjusted in order to support Agile project flexibility.

Preparing the Agile Team

DHS PMs experienced with Agile universally agreed that socializing Agile philosophy and methods is key to team success:

- Ensure that all team members receive compatible core Agile training
- Clearly communicate each team member's Agile team role and associated performance expectations
- Direct communication between users/user representatives and developers is important to "getting it right"
- Track and enforce agreed-upon requirements and standards to restrain requirements expansion ("scope creep")

4.2 Building the Agile Project Team

4.2.1 Agile Team Composition

One of the most important resources the PM plans for is the project team. The Agile methodology selected determines, at least in part, the team roles that need to be filled. A development team with experience in Agile practices is a significant risk mitigator for on-time delivery. Determination of the need for and selection of an appropriate Agile Coach are part of the PM's process of establishing an Agile project team.

At DHS, the project team is likely to consist of a mix of contracted industry developers and internal DHS personnel. The PM works closely with project stakeholders, DHS contracting officials, and the Agile Coach (if available) to identify which project team roles to fill with government rather than contracted staff. An Agile Coach can provide valuable insight and advice regarding the levels of expertise, cross-functional skillsets, and experience required to successfully fill specific team roles.

Ideally, core team members are dedicated to the Agile team or project for the duration of the project. Similarly, industry experts often advise some overlap in team members' skillsets, to mitigate risk associated with a key person becoming temporarily unavailable and to provide the most effective team. In practice, DHS PMs may not be able to field project teams whose members are dedicated full time over the duration of the project. The PM maintains personnel planning projections and reporting tools, including impact assessments, to identify the impacts of potential changes in the availability of key personnel and expertise.

PMs may need to negotiate modified working relationships and schedule coordination processes with supporting organizations. Subject matter expertise in areas such as compliance, privacy, and security may reside in organizations structured to provide part-time consulting support, rather than on-going project team participation. Negotiating the availability and expectations for subject matter expertise is important for the effectiveness of Agile projects.

In establishing an Agile project team, the PM focuses on ensuring that roles are well-defined. While the individuals operate together as a team, role clarity is a potentially major issue for an Agile project team. Failure of any project contributor to understand his or her role and responsibilities within the Agile approach may place the project at risk. Group training for the team may help mitigate this issue.

Agile teams need to have requisite experience not only in implementing Agile from a process point of view, but also need to have experience in requisite technical skills, such as automated testing, automated deployment, test-driven development, etc. Experience helps to ensure successful adoption of Agile.

Managing the Agile Team

DHS PMs experienced with Agile made several recommendations on managing Agile teams:

- Agile is not a magic bullet; there is no substitute for a strong project manager
- Recognize that some individuals are not full-time, project-long team members, and develop mitigation options
- PM needs to actively manage meeting agendas and participation to ensure representation from all areas of expertise needed to make progress
- Recognize that developers have varying skill levels; take this into account in determining levels of detail needed in requirements definition

4.2.2 Agile Team Training

One of the key observations derived from experienced DHS PMs about deciding to apply Agile is that it is vital to understand and accept the Agile philosophy and methods across all participating organizations and players.

All DHS project stakeholders are informed by existing departmental and federal guidance. However, it is likely that individual stakeholders may have differing interpretations of Agile methodologies and practices. PMs may wish to consider providing methodology-specific educational materials and training opportunities not only to members of the development team, but also to project stakeholders and partners, as needed. The success of the Agile project depends in part upon successfully streamlining interactions with stakeholders, and early investment in establishing a common reference framework is advisable.

The PM is responsible for empowering the cross-functional development team to succeed using the Agile approach. The PM, with the assistance of an Agile Coach, if available, assesses the training needs of DHS project team members both individually (in accordance with assigned roles) and collectively. Training that incorporates exercises in which team members apply the methodology while acting in their designated roles may prove particularly valuable. If an Agile management software tool is to be used, then DHS team members responsible for tracking and reporting on project progress need additional, tool-specific training. See Appendix D for information on DHS Agile training resources.

4.2.3 Agile Team Operations and Communications

Many Agile development methods are based on small teams (industry-standard models typically use a single-digit number of participants, including developers, testers, management, and user/product owner representatives). Federal government projects, however, typically require coordination with matrixed, cross-organizational groups and multiple compliance and oversight bodies. It is the PM's responsibility to reach out for assistance when needed outside the team (see Section 5.2.3 for some strategies). PMs communicate team status to the business, infrastructure, and compliance stakeholders for the project, and elevate any issues arising due to lack of communication with or input from key partners. Similarly, if the project team includes contractors, then contract performance is closely monitored and managed.

Open sharing of project information is fundamental to the Agile approach. Tools such as project dashboards provide all team members (including stakeholders) access to real-time information on the work, performance metrics, and user story updates. Transparency also contributes to the success of the retrospectives (review and assessment sessions) that take place at the end of a development iteration. The assessment and performance data addressed in retrospectives may also be reused in the tailored SELC artifacts used for project governance.

For large projects, where multiple small teams are working simultaneously on separate subprojects, additional PM consideration is given to the efficient management of communications among teams and to integrating the activities and outputs of the individual teams.

4.3 Agile Contracting Considerations

4.3.1 Government-wide Contracting Guidance to Support Agile Development

The Federal Acquisition Regulation (FAR) does not expressly speak to Agile concepts such as further refining technical solutions after contract award. This, combined with the common notion that responsible development entails a full detailing of requirements before work start, often causes confusion within the acquisition professionals pursuing Agile development. To overcome this, the Office of Management and Budget has released a suite of guidance and case studies that can aid structuring contracts to support an iterative, customer-driven development process:

- TechFAR, Handbook for Procuring Digital Services Using Agile Processes (https://playbook.cio.gov/techfar/)
- Contracting Guidance to Support Modular Development

- U.S. Digital Services Playbook (https://playbook.cio.gov/)
- Innovative Contracting Case Studies (https://www.whitehouse.gov/sites/default/files/microsites/ostp/innovative_contracting_case_studies_2014_-_august.pdf)

The Homeland Security Acquisition Institute, in partnership with the CPO Procurement Innovation Lab, offers various learning events to explore flexibilities and innovative procurement techniques, including actual case studies and applications within DHS environment.

4.3.2 Partnering with the Contracting Officer

Agile development requires flexibility in contracting; however, accommodating this flexibility can be difficult to achieve with traditional federal contracting approaches that prioritize accountability and risk mitigation. Agile PMs bring the following goals to their contracting officers (CO):

- Rapid contracting processes to keep pace with Agile development
- Contracting to accommodate incompletely defined scope and requirements
- Ability to select the competent vendor to support the highest quality team
- Ability to simultaneously manage award and execution of overlapping release orders
- Ability to respond to requirements changes without requiring extensive change orders

Clearly, DHS Agile PMs need to foster strong partnerships with their COs and staff to achieve these goals. PMs and COs work closely and cooperatively to achieve contracting arrangements that support Agile development while providing appropriate stewardship of federal resources.

A CO experienced with Agile projects can be a strong business advisor and partner to the PM. The PM needs to work with the CO as early as possible in the project planning effort in order to develop appropriate contracting strategies and options. The CO also has a critical role in creating close partnerships between government and industry developers/vendors.

Key contracting considerations for an Agile PM include:

- Vendor experience and demonstrated success with Agile development
- Speed to delivery, including streamlined processes for task award and execution

Federal Acquisition Regulation 1.102-4

"… 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."

- Frequent, iterative deliveries of software
- Modular releases of software to end users
- Ability to monitor changes to maintain contract and project scope
- Potential to award contract vehicle in the most Agile manner possible, without predefined business scope
- Flexibility to accommodate refinement of requirements
- Transparency and collaboration
- Performance to satisfy customers with measurable results
- Risk reduction and mitigation
- Prior experience of staff in the Agile methodology

Among the contracting options that DHS Agile PMs may want to consider are:

- Award an enterprise contract vehicle to a pool of prequalified vendors, and rapidly compete Agile release orders
- Streamline ordering processes to rapidly award Agile task orders
- Streamline contracting processes using ordering guides, streamlined documents, and templates
- Establish contract on- and off-ramps to refresh the vendor pool

4.3.3 Contract Performance Metrics, Vendor Qualifications, and Related Challenges

Although Agile methodologies proceed from the assumption that scope evolves, contractors are still evaluated based on measures of productivity, quality, and related factors. Agile contract metrics may differ from traditional contract metrics, just as Agile project metrics differ somewhat from those of traditionally managed projects. Agile PMs work closely with their COs to define how to appropriately measure contractor performance on their projects.

One of the key challenges identified by DHS PMs experienced with Agile projects is the difficulty of verifying that developers have the Agile experience and expertise required to successfully meet DHS needs. Since no standardized Agile experience measures exist, PMs and COs often emphasize successful past performance on similar projects when identifying and selecting vendors.

In addition, industry certifications such as the PMI ACP-PMP or Scrum Master could be required for team members. Past performance at other government agencies that are actively using Agile processes may provide a pool of qualified contractors. The vendor's history of meeting the metrics listed in Appendix C may also be leveraged in the contractor selection.

Questions for IT Development Vendors

Some recommendations from experienced DHS Agile PMs regarding questions for vendors who claim Agile expertise:

- How do you manage Agile projects?
- What Agile tools and methods do you use, and how much experience does your team have with them?
- What is your methodology for using user stories and determining the relative value of individual stories (in scrum: story points)?
- How does the complexity of the stories influence the cost and schedule, and how do you manage this complexity?
- How do you determine the number of sprints/ development iterations required to produce a deliverable software increment?

Another key challenge identified by DHS PMs is that of applying Earned Value Management (EVM) to Agile projects. Although some Agile tools provide processes for using EVM, they can be challenging to apply in a meaningful way, as it can be difficult to translate progress within an iteration to value contributed to the project as a whole. One successful DHS program captures metrics at the Feature, Feature Point, and User Story level, but its official measure set is via Feature Point. This method of Agile EVM has been approved by DHS Program Accountability and Risk Management (PARM) during oversight of the program.

Competing small increments of work on an Agile project may make it easier to use small businesses and help to support DHS efforts to meet small business utilization goals. The use of small businesses may prove more or less feasible depending on the technologies in use – large businesses may be more likely to have experts available in a range of different technologies, and smaller companies may have a more specialized focus on a few technologies.

4.4 Establishing Tools, Processes, and Metrics

4.4.1 Agile Management Tools

It is important that Agile project team members understand that, although Agile is a flexible approach to IT development, it is not an undisciplined one. Agile requires as much, if not more, discipline than traditional development approaches to execute effectively. Making, tracking, and delivering on commitments are integral to Agile team processes. Similarly, tracking of team performance and product quality is key to Agile's focus on continuous improvement. Agile process tracking and management tools enable planning, retrospective, issue identification, and reporting activities. As discussed, these tools can also record data and produce reports to help satisfy the governance requirements of the tailored SELC process. While there are many tools available for use, it is important to define the project processes first, then find a tool that can meet this processes rather than adapting processes to meet a predefined tool.

Agile tracking tools range from internally developed spreadsheets to sophisticated commercial products with extensive graphical outputs and options for tracking project value against investment. Some tools are focused on specific Agile methodologies or can be adapted for use across multiple methodologies. Kanban boards, for example, are often used in conjunction with other Agile methodologies. Ease of use in the project environment and the ability to produce artifacts in accordance with the

Agile Life Cycle Management Tool Benefits

One DHS PM successfully applied an Agile management process tool to help track work in progress, schedules, and team member commitments. The tool was also used to help identify potential development and release risks.

Information from this tool was regularly used to help guide governance activities such as daily team meetings, weekly team status reports, and updates to executive management.

project's SELC tailoring plan are key decision factors in selecting and/or developing tools. Additional training may be necessary to achieve the full benefit of some of the more comprehensive tools.

There are two key locations through which DHS-approved tools can be acquired:[17]

1. The Data Centers (DC) IT Services Catalog, providing products and services, and located at:

   http://dhsconnect.dhs.gov/org/comp/mgmt/cio/itso/IT_Services_Catalog/index.html

2. GSA Advantage in the Schedule 70 catalog at: https://www.gsaadvantage.gov. Users can also leverage the enterprise licensing for various project management, requirements management, and collaboration tools.

Considerations when assessing candidate Agile tools include the following:

- Project size
- Group communications options and issues
- Project tracking requirements
- Metrics monitoring and reporting capabilities desired
- Any special collaboration support needs

Additional tools supporting Agile team efforts are often devised or customized by the team members themselves and may include daily posting of key status information (either virtually or in a highly visible space). These interactive tools rely upon and reinforce the focus on open information sharing across the team.

4.4.2 Agile Project Management Processes

As with Agile tools, many project and team management processes are somewhat methodology-specific. However, several are common, and many can be adapted for use across different Agile methodologies.

Daily Stand-Up Meetings

The most widely recognized Agile management process is that of holding brief, informal daily team meetings. These sessions, frequently known as "daily stand-ups," are essential collaboration events where all team members are expected to participate. In the daily stand-up, team members are expected to provide information on three topics: (1) work accomplished, (2) current work, and (3) impediments (a.k.a. roadblocks) to their work. Other team members may offer assistance with roadblocks or may advise of interactions with their own work. The discipline of these daily meetings also supports the development and maturation of the project team and helps to foster mentoring and partnering relationships within the team. Although these daily sessions typically work best when all team members are physically co-located, they can also be accomplished remotely.

[17] Homeland Security Systems Engineering and Development Institute, *Recommendations on a Requirements Engineering Tools Suite for DHS Agile Projects* (Department of Homeland Security, April 29, 2014), 9-11.

In Agile, there may be other meetings besides the daily stand-ups; these can be tailored to the needs of the project and team. Goals for meeting rhythms, scope, participation, and outputs are also based on the needs of the specific project, the team, and the project environment. An Agile Coach or other experienced practitioners can help PMs determine whether additional or fewer group sessions are required, define meeting outputs, and refine meeting plans and attendees accordingly.

DHS PMs experienced with Agile found the following processes valuable:

- Maintaining a single, well-understood definition of "done"
- Conducting thorough retrospectives and actively incorporating lessons learned
- Tracking information from retrospectives to evaluate progress and performance
- Enforcing disciplined software build processes
- Maintaining well-ordered, continuously updated code repositories and databases
- Leveraging automated testing to monitor and improve product quality
- Maintaining progress boards and using them to identify potential issues

4.4.3 Agile Project Metrics

Most Agile methodologies and tools incorporate associated performance metrics. Appendix C provides examples of these metrics and associated tool-generated reports. The factors that go into measuring a given Agile project team's performance may be specific to the particular approach and project and may not necessarily be comparable across projects and methodologies; however, these metrics do provide PMs valuable information within the context of a given project.

Over the course of an Agile project, appropriately selected delivery metrics provide indications of whether the team (or vendor) is continuing to learn and improve. Similarly, appropriately selected customer/user satisfaction metrics provide indications of whether the software produced by the project is meeting user needs. The PM, working with the CO and key members of the project team, identifies metrics that are best suited to provide true indicators of the progress and health of the project. The most important metrics used with Agile projects are business metrics, which keep the focus on the delivery of products with business value. Of course, the primary measure of vendor success is the delivery of working code and useable functionality.

As noted in Section 4.3, determining appropriate metrics to apply EVM to Agile development projects typically requires an approach specific to the project and the methodology being used. One technique, known as AgileEVM, is an adaptation of standard EVM based on scrum story points (estimations of the value of user stories). Since EVM typically relies upon clear numerical estimates of project parameters such as cost, backlog volume, numbers of iterations, etc., it is difficult to apply to projects for which incremental progress parameters are not well defined or are difficult to quantify. As a result, DHS programs successfully applying EVM to Agile projects have largely moved from measuring story points to feature points, as features are more concrete than story points (which may vary from team to team). The *GAO Cost Estimating Guide – Best Practices for Estimating and Managing Program Costs* and the *DHS Earned Value Management Guide* are being updated to include EVM guidance for Agile.

(This page intentionally blank)

5. Applying Agile: Tips and Practical Lessons

This section focuses on challenges likely to face DHS PMs of Agile projects and tips for addressing these challenges. The challenges were identified by DHS personnel experienced with Agile projects; the tips (particularly those in the call-out boxes throughout this section) draw upon advice of those experienced DHS personnel and are supplemented with information from Agile industry sources. These are not intended as rules, but rather as informed suggestions based on experience. Note that the selection of the Agile methodologies to be applied affects project planning decisions. See Section 2.3 for an overview of these choices and considerations.

5.1 Tips for Preparing to Run an Agile Project

5.1.1 Get and Maintain Leadership Buy-In

It is important that executive-level stakeholders understand the Agile approach, usage, and expectations–including expectations about the support they need to provide. Applying Agile may affect many organizational activities, and support from the executive level helps ensure that standard processes are tailored to realize the benefits of Agile. Knowing the executive-level stakeholder understanding of Agile prior to starting a project may help to shape educational and communication processes.

5.1.2 Identify Interdependencies and Plan to Manage Them

PMs seek to identify dependencies across project increments during initial planning. Similarly, dependencies across organizational functions such as infrastructure (configuration management, security, etc.) are recognized and incorporated into planning and preparations. If support for these dependencies is not appropriately organized and resourced, the infrastructures may be unable to provide timely support to Agile projects.

5.1.3 Plan for Project Team Resources and Processes

Managing a team that incorporates members of different organizations, possibly working in multiple locations, and/or including both DHS and contractor staff members is a complex challenge. Since Agile project teams rely upon frequent, open information exchanges, PMs have to plan how to encourage collaboration and organize team operations across geographic or organizational boundaries. A thorough assessment of risks associated with applying Agile in the project environment and identification of effective mitigation measures is an essential component of the project initiation process.

Agile Is Not a Cure-All

One DHS PM team running an Agile project noted that as they gained experience with Agile, they came to recognize its limitations, as well as its strengths. They offered the following recommendations:

- Investing time up front in developing a plan, gaining support for the Agile approach, and establishing success factors is a necessity
- Delays in obtaining executive decisions can erode Agile software development timeline improvements
- The entire project management structure needs to support Agile processes in order to realize the benefits
- Agile can help teams deliver, but only within scope of the allotted resources

5.2 Tips for Preparing and Organizing the Agile Project Team

5.2.1 Team Training and Development

Implementing Agile requires discipline. Sometimes Agile teams are perceived to be 'ad hoc' because they deemphasize formal processes, but running an Agile team actually requires a great deal of process and operational discipline. Agile emphasizes the disciplines of transparency, information sharing, and accountability, along with software development skills, in order to progress as efficiently as possible.

DHS PMs running Agile projects emphasize the importance of providing a common base level of training to all team members. All members of the extended project team need sufficient training to understand not only the Agile philosophy but also how Agile roles, processes, and artifacts fit together and where Agile approaches can be scaled across the operational, systems, and software levels to support efficient and effective software development. Contracts for Agile support that specify training requirements for contracted team members are recommended.

Agile Project Teams Are Developed, Not Born

One DHS PM team running an Agile project recommended carefully planning and specifying the "Who" and "How" factors of getting team work accomplished:

- Ensure that both individuals and teams understand what is expected of them, and regularly provide meaningful progress reports
- Make team members accountable to the team, and help one another when needed
- Ensure that everyone on the team uses the same definition of "done"
- Reach across organizational boundaries as needed to address issues with team member availability

5.2.2 Understanding and Fulfilling Roles

One of the challenges identified by DHS PMs experienced in running Agile projects is ensuring that every team member fully understands and fulfills his or her role within the team. There is a learning curve for each team member new to Agile, and definitions of roles vary somewhat depending upon the Agile methodology used. Helping team members understand and learn to satisfy the responsibilities of their roles in the given project team is an area where use of an Agile Coach (see Section 5.2.4) may be very beneficial. Role assignments within the project team may change over time in keeping with the changing needs of the project and demonstrated performance of individual team members. An Agile Coach or other experienced team members may help identify instances where cross-training or role adjustments would be beneficial.

One technique that an experienced Agile consultant noted as valuable for ensuring project team access to key subject matter experts is the concept of shared responsibility for a given specialized role. For example, the DHS organizations tasked with consulting in or approving areas such as security, privacy, technical architecture, or regulatory compliance are typically not staffed at levels to ensure full-time availability of a dedicated staff member for each Agile development team. The Agile approach relies upon rapid access to such expertise whenever needed. PMs may need to

negotiate with leaders of the DHS organization responsible for specific expertise to identify mutually beneficial options for making those resources available. The joint identification of ways to ensure that an informed subject matter expert is always "on call" to the project team benefits the project, and may also increase efficiency of operations in the expertise-owning organization.

5.2.3 Establishing Communication Methods and Standards

Effective communications are vital to the success of the Agile project team. This capability becomes even more important in cases where the PM cannot meet the Agile ideal of having all team members geographically co-located. Engage the team to ensure that everyone involved in the project is able and willing to use the proposed communications methods and technologies. This may involve providing additional training on any Agile management and/or reporting tools selected to support the project.

Agile industry consultants frequently recommend creative use of wall space in team areas as a communication tool. Posting agendas for recurring meetings, active problems in need of team solutions, recent project accomplishments, and next schedule deadlines can help keep these items at the forefront of team discussions. Although some DHS projects consist of team members who are not geographically co-located, equivalent web boards or frequent visual news updates can be used in a similar manner to help transmit valuable information.

One of the key communications standards emphasized by DHS PMs experienced with Agile is establishing and enforcing the definition of "done." Defining what "done" means for an IT development team can be surprisingly challenging. For example, "done" could mean "done with" any of the following: design, coding, unit testing, or deployment. If not all team members use "done" with the same meaning, there can be significant work disconnects that impede project deliveries. One DHS PM noted that undisciplined use of "done" and "almost done" led to a lack of trust in the meaning of either term. It may be necessary to have one or more designated team meetings to discuss and resolve what "done" means for the project team. Often, "done" represents a definition agreed upon by the project team and leadership, and indicates that a product / release / design is ready for deployment. Increasingly, "done" means software has been deployed to production and is working. Ultimately, the definition of "done" pertains to a specific unit of work agreed upon for a given iteration by the project team.

5.2.4 Identifying and Working with an Agile Coach

An Agile Coach is an individual with significant broad experience applying Agile approaches to software development efforts. The Agile Coach role typically incorporates a mix of technical mentor and change agent skills (i.e., identify/implement change through listening, feedback, facilitation, guiding, mentoring, and teaching); the specific skill sets may vary, depending on the complexity and needs of the project. Ideally, an Agile Coach has experience across a range of Agile methodologies and tools as applied to projects similar to the one at hand.

A widely acknowledged Agile best practice is using an Agile Coach to help in all aspects of planning and preparing to execute an Agile project. An experienced Agile Coach can help the PM identify and develop plans for introducing and applying Agile approaches within the context of the project team environment and governance structure. The coach can also help with strategizing and obtaining approval for tailoring the DHS review and approval processes to enable the Agile project team to function most productively.

DHS does not have an internal cadre of Agile coaches. When contracting for Agile coaching expertise, DHS PMs of Agile projects work closely with their contracting organizations to obtain current information on possible coaches' experiences. PMs are encouraged to seek coaches who have documented successful past performance on projects implementing similar technologies and compatible Agile methodologies.

5.2.5 Beginning with a Pilot Project

Starting an Agile team off with a pilot is a recognized industry best practice to decrease the risks associated with an inexperienced team tackling a large, challenging project. Running a small-scale pilot for a set period of time provides an opportunity for the PM to observe team operations, gather detailed feedback, and evaluate and make changes to the approach as required. Lessons learned may result in changes to the team or to the tools and/or methodologies applied, or may be incorporated into a revised vision for the project.

PMs carefully consider and articulate the goals for an Agile pilot project before beginning one. Industry practitioners note that the selection of an appropriate Agile pilot project affects not only team learning but also organizational perceptions of both the project team and the Agile approach. Often cited tips for selecting an Agile pilot project include the following:

- Duration: Project length of eight to 10 weeks is a common recommendation
- Size: A project that can be done by a small (five to nine people) cross-functional team is recommended
- Importance: A project with clear business impact and engagement has more visibility

Pilot Projects Can Yield Benefits

DHS PMs experienced with Agile recommended executing pilot projects if at all feasible:

- Accomplishing a six-month pilot project helped one PM identify significant opportunities for improving the approach
- After the pilot period, the team conducted research on alternative operating options and provided frank feedback on the pilot experience
- The PM made considerable modifications to the approach and credited the experience with helping to develop a high-functioning Agile software development process
- Another experienced DHS PM noted that executing a pilot project can provide a team valuable insight into the challenges of managing technical debt in an Agile software development effort

- Complexity: A project with requirements that are representative of the bulk of the effort is ideal
- Motivated business sponsor: As noted in Section 5.1.1, an engaged executive can motivate the team and help remove organizational barriers
- Leadership and experience: Pilot team members ideally are strongly motivated to adopt Agile practices and/or are experienced on Agile projects

5.3 Tips for Contracting for Agile Projects

5.3.1 Structuring the Contract

A DHS CO experienced with Agile projects noted the importance of basing the contract structure on the product owner's goals for the project, taking into account both the maturity of the system/project plan and the Agile methodology to be used.

There is no specific contract type called Agile contract. Rather, the optimal contract structure that supports an Agile project is a case-by-case determination requiring a certain degree of creativity and teamwork. Consideration of the contract structure should address factors such as appropriate duration of base term and options, scalability, deliverables or level of effort, and pricing with a forward-looking mindset that the awarded contractor may or may not perform well and that everyone needs appropriate incentive.

Also important to keep in mind is that the scope of the contract is established when the contract is solicited. FAR generally requires competing the work that arises after contract award if the work is not within the scope of the original contract. Under case law, a modification is within the scope of the contract only if potential offerors would have reasonably anticipated such a change.

Determining the appropriate type of contract is dependent upon the project plan characteristics. FAR 16.103(a) states that selection of contract type requires exercise of sound judgment that will result in reasonable contractor risk and provide the contractor with the greatest incentive for efficient and economical performance. If the Agile team, requirements, schedules, and

Working with Contractors on Agile Projects

DHS PMs experienced with Agile noted several difficulties in contracting for Agile software development:

- One PM noted that nearly all contractors say they have Agile experience
- Since there are no standardized metrics for Agile experience, certifications may not help in assessing qualifications; successful past performance on relevant projects is the best indicator
- Quantifiable metrics of contract performance are desirable but may not accurately reflect what is truly of value to the performance of the Agile team
- A particular challenge is trying to work with vendors and COs to establish meaningful ranges of expected performance
- Translating project objectives into contract line-item numbers (CLINs) for performance may require significant explanatory language

process are sufficiently detailed and described in concrete terms, then a fixed-price contracting approach may be appropriate. If, however, specific requirements and other project details are incompletely defined at the start of the Agile project, then cost-based contracting (either time-and-materials or cost-reimbursement) is more appropriate. Typically, simpler activities are more suited to fixed-price contracting, while more complex efforts often require a cost-based approach.

Cost-based contracting provides flexibility for the PM and CO to identify and add subject matter expertise to the contract as the specific project needs become clear. To provide for this type of flexibility, the PM and CO need to work closely to establish estimates for both the main contracted activities for the project and any anticipated application of specialized expertise needed during the project.

Project complexity also drives determinations of what levels of expertise are required in the labor categories applicable to the project. This topic is likely to be the subject of some negotiation between the government and the vendor. If an Agile Coach or DHS PMs with Agile experience are available, they may be able to provide significant advice on making these complexity determinations.

5.3.2 Contract Incentives for Agile Projects

A DHS CO experienced with Agile projects cautioned that it is difficult to incentivize contractors on Agile projects. One issue is that determining responsibility for failures is complex, given the Agile approach of refining requirements on an ongoing basis. The use of award fees may not be appropriate for Agile project contracts, since the Agile approach is focused on successful deliveries, rather than on specific cost or schedule goals. For longer-term Agile projects, PMs and COs may find that contract award terms may provide a useful incentive, with satisfactory performance over a specified contract period being either a prerequisite or an automatic qualification for award of follow-on contract periods.

Additional Contracting Challenges

A DHS CO experienced with Agile projects identified the following recommendations:

- Having personnel from contracting and other supporting organizations participate in Agile training alongside core project team members provides a common frame of reference that can streamline project execution
- In situations where it is not possible to determine the work content of specific iterations in advance (as with large and complex projects), consider developing an overarching list of required skill sets/labor categories with quantities and timing of need to be refined during the project
- For complex Agile projects, establishing a competitive range of cost through solicitation and then opening negotiations with vendors may prove a useful approach

5.3.3 Contract Performance Assessment

Given the difficulty in applying contract incentives, tracking and managing vendor performance is vital to Agile project success. As discussed in Section 4 of this Instruction Manual , applying EVM to

Agile projects can be challenging, as it can be difficult to define value metrics (e.g., story points or number of distinct features implemented) in a meaningful manner across the scope of the project.

An experienced Agile CO emphasized that ensuring that the Contracting Officer's Representative (COR) has sufficient resources to perform adequate oversight is vital for Agile projects. Similarly, having qualified monitors assist the COR in determining the technical content and error rates of each iteration is vital to making informed assessments of contractor performance.

5.3.4 Vendor Selection

As noted in Section 4, DHS PMs have found that determining whether candidate vendors truly have the Agile implementation qualifications and expertise claimed can be challenging. Although certifications in various Agile methodologies exist, there is no common standard for these certifications, and their applicability to a specific project may not be clear. Contractor past performance remains the best tool DHS PMs and COs have for evaluating potential vendors and can be weighted accordingly while evaluating proposals. Since the Contractor Performance Assessment Reports System (CPARS) does not specifically request information on vendor performance with Agile approaches, the PM and CO may have to take additional steps to request information on previous clients' experience with vendors on Agile projects.

The TechFAR encourages the use of oral presentations or technical demonstrations (Citation: p25, TechFAR, ibid). This approach enables the Government to directly interact with the technical team that will be working on the project and more accurately determine whether an offeror truly knows Agile development. Under the Procurement Innovation Lab, some DHS project teams have successfully used oral presentations in conjunction with on-the-spot technical consensus panel evaluation and produced significant time savings and program office satisfaction.

Using experienced Agile practitioners (coaches or PMs) for assistance in evaluating proposals is an effective tool for vendor selection.

5.4 Tips for Executing the Agile Project

5.4.1 Determining Focus and Rhythm for Meetings

DHS PMs experienced with Agile projects noted that it is vital not only to keep meetings focused but also to maintain records of progress and issues for project-wide situational awareness. From the outset, consider developing and maintaining a matrix outlining the range of regular meetings in support of the project, noting the purpose, schedule, logistics, participants, lead role(s), inputs, action, and outputs associated with each meeting. This brief overview can provide a key planning reference for the entire team, so that all team members can plan in accordance with the recurring schedule. The records of these meetings also serve as a core reference for the team and the project as a whole.

Although some Agile methodologies rely on specific meeting structures, it is best to adapt these structures to the circumstances and needs of the project. The PM and key stakeholders or deputies (including, potentially, the Agile Coach) periodically review and adjust the overall strategy for project meetings as necessary. Ad hoc interactions between team members and the PM can take place for smaller discussions, while meetings have specific purposes and can result in a relevant decision. While Agile minimizes the amount of documentation throughout the project lifecycle, experienced PMs found that reviewing the documented results of meetings can help identify those that can be discontinued or repurposed, and reviewing backlogged issues or topics requiring ad hoc meetings may help identify needs for scheduling sessions with specific agendas.

5.4.2 Keeping Focused on the Work

One issue noted by both DHS PMs and external Agile experts is that team members can sometimes become so focused on a particular methodology or process that they can lose focus on overall project progress. As one DHS PM noted, a collection of project sub-teams, each developing software based on its own perspectives, is likely to produce disjointed results that cannot be integrated into a useful product. One of Agile's strengths is its adaptability, and it can be counterproductive to focus too strongly on "*doing Agile correctly*." Team members and PMs may research and try multiple approaches and tools before selecting and tailoring the ones most suited to enabling the project goals. Methods and tools are chosen to serve the project, and the "best" applications of Agile are the ones that best help the project team meet its objectives (see Section 4.4.1 for more information on Agile tools).

Agile Projects Run on Information Sharing

DHS PMs experienced with running Agile projects noted the following lessons learned:

- Daily stand-ups are vital
- There are times when you need all team members present to fully address and resolve an issue
- It is particularly important to bring requirements analysts, developers, and end users together to ensure proper interpretation and prioritization of actual needs
- Strict discipline is required to avert meeting overruns
- One team found that holding Scrum meetings at each logical event (Sprint end, delivery, etc.) was more time-consuming than productive, and therefore consolidated meetings to reduce the overall number of such events

5.4.3 Maintaining Discipline in Team Processes

A common observation of experienced Agile PMs within DHS and elsewhere is that Agile, effectively applied, is actually a very rigorous software development approach. Although empowering the project team is at the heart of the Agile approach, the PM still needs to harness the team's effort for project production. Having team members who remain committed to the Agile process is key to the success of the project team.

One potentially major issue for Agile software development teams is the accumulation of technical debt. The term *technical debt* is a relatively recent one that describes the work that remains to be accomplished before a project can be considered complete. Technical debt may arise during the Agile software development process because the iterative code releases are focused on adding value (functionality), not on providing a perfect, finalized product. Increasing technical debt across the project life cycle ends up accumulating due to short cuts, and can threaten the overall functionality of the software. The knowledge gained during the iterative development process enables the team to continue making improvements. Some of those improvements may involve corrections to, or replacement of, previously delivered increments of software through a process known as "refactoring."

Managing technical debt is challenging, requiring the PM to balance potential conflicts between pace and quality goals. Although modifying or improving a previously developed increment of software may force a change in the release schedule, if the team does not make these revisions in a timely manner, the effort required to correct them later tends to increase. In the context of technical debt, the debt is considered to "accrue interest" over time. This increasing technical debt is a risk factor to be addressed as soon as feasible. If the technical debt is allowed to accumulate unchecked, or if the project team loses track of the scope of its technical debt, the project may suffer potentially catastrophic schedule and performance problems.

Process Discipline Is Key

One DHS PM team running an Agile project emphasized the need for strong software-build and code-management processes:

- Iterative software development and frequent builds increase the importance of repeatable, verifiable processes
- Maintaining well-organized, up-to-date code repositories and databases is a key responsibility
- Strong version management and control processes help reduce rework
- Capturing and communicating code review outcomes is key
- The team needs an experienced scheduler to keep track of everything
- Technical debt is tracked and controlled
- Everything is about managing and mitigating risk

5.4.4 Scope of Control

All DHS PMs interviewed for this Instruction Manual agreed that good core project management is a prerequisite for taking advantage of the benefits that Agile has to offer. Planning, evaluating, and adjusting remain essential parts of the PM's daily workflow. Adapting the project plan to both present and emerging realities is a continual challenge.

Interviewees and published Agile experts agree on the importance of measuring project progress and quality. Unfortunately, several DHS PMs experienced in Agile projects noted that identifying consistently meaningful metrics is difficult. Finding the most valuable metrics and gleaning insight from those results is a focus for PMs, who continuously review these results to provide an accurate representation of project progress. As one DHS PM noted, an Agile team's increased velocity (speed of delivery) is not necessarily an improvement if the quality of the deliveries is substandard or if the amount of rework is also increasing. Most Agile tracking tools provide some metrics-reporting capability. The PM works with an Agile Coach, if available, and a team member fully trained on the tracking tool to identify which metrics may be valuable for internal team management actions, stakeholder reporting, or other purposes. When reporting project indicators, make the same metrics available to all stakeholders for review.

Additional Tips

Other considerations noted by DHS PMs experienced with Agile projects include:

- Prioritizing requirements can be particularly challenging if the relationships among the project stakeholders, executives, and end users are not clear and direct
- Not all programmers have both the technical and the business expertise to appropriately interpret and develop from high-level user stories
- Automated testing is a vital tool for rapid identification of issues
- Load testing and other software performance tools can also yield key insights while saving developers time

Another key challenge identified by DHS PMs is that of maintaining focus on the current project status, goals, and plans. In the complex DHS environment, hot-button issues may arise at any time, along with encouragement (sometimes from high levels) to address these issues by modifying an in-work project. The PM guards against allowing hot-button issues to disrupt project progress. When such an issue arises, the PM works with stakeholders to analyze the issue and identify solution options consistent with the overall project planning and requirements prioritization. Stakeholders and business owners can determine where any associated work fits in the existing pipeline and backlog of project priorities, plans, and resources.

One DHS PM noted that an Agile project's success can generate organizational enthusiasm that inadvertantly threatens the success of the project. This PM found that requests for product demonstrations became a signficant drain on project resources. Project team members, including developers, were diverted to create, set up, and execute demonstrations. This diversion of

significant amounts of labor and creative contributions from the team put some project development goals at risk. In addition, once a project gains sufficient publicity within a large organization such as DHS, business owners in other areas may become interested in adding their requirements to the project scope. Resisting the insertion of new, non-prioritized requirements at project demonstration events may become a PM challenge similar to that posed by hot-button issues.

5.4.5 Integration Management

In the case of large projects, where the products of multiple subprojects are integrated into a cohesive whole, PMs work closely with stakeholders, partners, and an Agile Coach (if available) to plan for and track integration activities. Assigning an experienced project scheduler using a comprehensive Agile project management data-tracking tool is recommended. The scheduler, working with team members experienced in IT integration, focuses on clearly identifying interfaces and interdependencies between and among subprojects and on keeping all subproject progress data up to date. A project management tool with the ability to graphically illustrate planned and projected schedules for integration events helps PMs to identify and proactively manage potential integration issues. The use of a configuration management strategy with automated testing can mitigate the risks with integration management across the program and provide a more robust end product.

Integrating the execution of an Agile project with supporting organizational activities is also an important consideration for enabling more widespread adoption of Agile practices. In general, DHS PMs experienced in running Agile projects noted that both PMs and stakeholders need to be aware that improving the DHS software development process is only one piece of the effort to improve DHS IT project deliveries. Although changes are in progress across DHS, the scope and pace of most of them are outside the control of the PM or the project team. In running an Agile project, DHS PMs focus on making process improvements that are feasible in the existing project environment. PMs are also encouraged to record their lessons learned and recommendations for implementing Agile approaches at DHS so that their insights can be integrated with the broader agency business improvement works-in-progress.

5.5 Conclusion

This Instruction Manual provides PMs and Agile practitioners alike with the appropriate information and tools to support the implementation of Agile IT development within DHS. As Agile continues to improve and expand both in the private and public enterprise, the DHS OCIO Enterprise Business Management Office will continue to provide resources on industry best practices and recent DHS implementations of Agile approaches. This helps PMs and practitioners to learn, shape, and use Agile on their projects in the most effective and efficient way possible. As individuals are continually engaged across DHS who have employed Agile practices, it will generate additional resources and information on how practitioners can better deliver projects in an Agile manner, thus increasing the likelihood of IT development success. Embracing the Agile approach, socializing the Agile mindset, and providing additional resources on Agile practices better enables

teams to deliver incrementally, incorporate feedback and learning into their practices, and produce higher levels of software quality in shorter periods of time.

(This page intentionally blank)

APPENDIX A: THE AGILE MANIFESTO

The origins of Agile development can be traced back to the late 1950s, yet in 2001 the creators of Scrum, XP, and many other premier leaders in the software development industry met to produce the *Manifesto for Agile Software Development*. The Manifesto highlights four core values that enable high-performing teams (see Figure A-1):[18]

Manifesto for Agile Software Development

We are uncovering better ways of developing
software by doing it and helping others to 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 still value in the items on
the right, we value the items on the left more.

Figure A-1: Manifesto for Agile Software Development

The four core values speak to multiple aspects of Agile development. *Individuals and interactions* focuses on facilitating effective communication by advocating for daily stand-up meetings, iteration reviews, and retrospectives. These frequent interactions enable team members to voice concerns and limit hurdles to effective development practices, foster respect and truthful communication with one another, and identify lessons learned in preparation for future project work.

Delivering working software pertains to the production of small pieces of working software to the customer, all while running the appropriate acceptance tests when defining and implementing features. Agreeing and adhering to the set time intervals for incremental delivery along the project lifecycle is key in fostering Agile development.

Customer collaboration is enabled in projects by having customers work side-by-side with the development team, whether represented on the team by a designated team member (i.e., Product Owner role in Scrum) or as a team member themselves. They are linked to the team primarily by updating requirements and providing feedback during the testing and demonstration processes.

Finally, *responding to change* is at the heart of Agile. Changing requirements and schedules are common in software development; the Manifesto promotes schedule and design flexibility over concrete determination of requirements and schedule adherence up front. In Agile, built-in items

[18] Jeff Sutherland, "Agile Principles and Values" (2013) Microsoft Developer Network, http://msdn.microsoft.com/en-us/library/dd997578.aspx.

such as iteration reviews and retrospectives alter priorities appropriately throughout the development process.

A.1 12 Principles behind the Agile Manifesto

The Manifesto is supported by key team- and work-focused principles, known as the "Twelve Principles behind the Agile Manifesto" (see A-2).[19] Each principle applies to each and every aspect of the software design process, focusing on everything from customer interaction to team construction, testing, and implementation. The Agile methodologies covered in Appendix B largely seek to adhere to these principles, although they may do so in different ways. Regardless of the methodology selected for the specific development project, DHS projects and programs seek to adhere to each principle as they begin, sustain, and complete software development projects.

[19] Beck, et. al, *Manifesto for Agile Software Development*.

Principles Behind the Agile Manifesto

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

Figure A-2: Twelve Principles Behind the Agile Manifesto[20]

[20] Beck, et. al, *Manifesto for Agile Software Development*.

(This page intentionally blank)

APPENDIX B: AGILE METHODOLOGIES

This section provides greater detail on the most common Agile development methodologies outlined in Section 2.2. Each highlighted methodology includes an overview, structure/principles, and key resources for learning more about the methodology under discussion.

Agile development is an approach built through a collection of methodologies. Figure B-1 gives a brief context for Agile development. Occasional examples of iterative software development in defense and space programs date from the 1950s to the 1980s. Specific methodologies began to arise in the 1990s to address growing challenges of developing rapidly changing software with traditional approaches and to leverage increasing standardization of software modules. Experts in the field and developers of the emerging methodologies got together in 2001 and declared the Agile Manifesto to launch a consolidated movement for Agile.

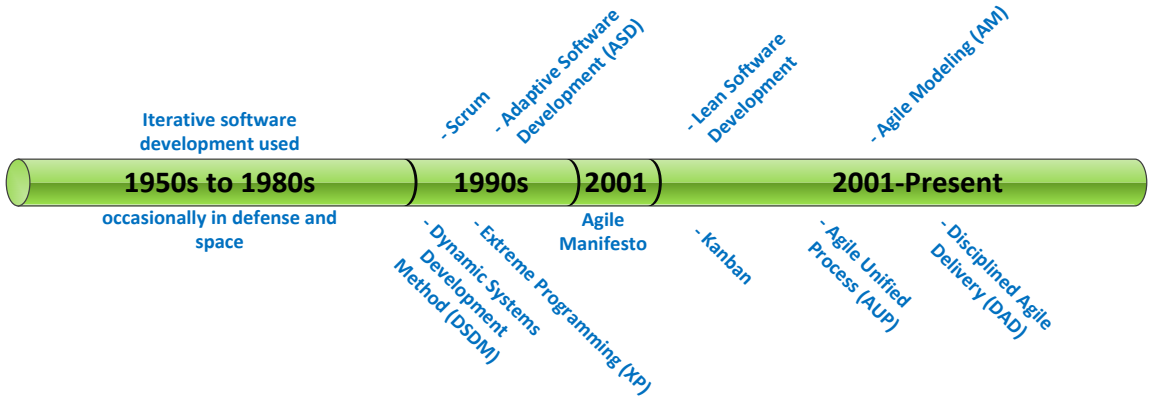

Figure B-1: Agile Development Timeline

Note that Agile methodologies precede, come after, and continue to evolve beyond the Agile Manifesto. Newer methodologies build upon the original Agile concept but do not necessarily supersede earlier methodologies. Accordingly, the Agile movement has built and continues to evolve a collection of methodologies to give developers a wider array of options for tailoring their Agile development. The methodologies in Figure B-1 are by no means comprehensive, but represent a few commonly used methodologies, some of which are discussed in this appendix.

DHS does not advocate for a single approach over any other – each Agile methodology has a set of benefits and potential drawbacks that may impact its applicability to certain projects based on their timeline, team involvement, operating environment, or other reasons. While certain methodologies (notably Scrum and Kanban) are used much more commonly across DHS and private enterprise than others, this appendix seeks to provide information on a number of methodologies to enhance executive-level, PM-level, and operational-level understanding of the various ways Agile can be employed.

B.1 Scrum

Overview:
Scrum, the most widely used framework for Agile software development, addresses complex problems while delivering high-value products frequently and effectively (see Figure B-2). Dr. Jeff Sutherland formed the first Scrum team in 1993, inspired by a 1986 Harvard Business Review paper by Hirotaka Takeuchi and Ikujiro Nonaka titled, "The New New Development Game." The best teams in the world, they wrote, have all the skills required to deliver a project from conception to delivery. These cross-functional teams, in contrast to traditional independent silos of expertise, worked in multiple overlapping phases. This focused group effort reminded them of the Scrum formation in rugby.[21] In 1995, Dr. Sutherland and Ken Schwaber, who went on to found the Scrum Alliance, published the first formal paper on Scrum. [22] To this day they collectively maintain the "Scrum Guide" which is recognized worldwide as the sole canonical description of Scrum. Scrum is most recognized for its architecture based on "Scrum teams," along with their associated roles, responsibilities, artifacts, and rules. Scrum also includes events[23] such as "Sprints," which are time-boxed, executable tasks with distinct, consistent, and incremental progress.

[21] Hirotaka Takeuchi, and Ikujiro Nonaka, "The new product development game," *Harvard Business Review* 64, no. 1 (1986): 137-146.
[22] K. Schwaber, "Scrum Development Process," in *OOPSLA Business Object Design and Implementation Workshop*, J. Sutherland, D. Patel, C. Casanave, J. Miller, and G. Hollowell, Eds. London: Springer, 1997).
[23] Events differ slightly from increments, as increments refer specifically to software development efforts. Events, or Sprints (in Scrum), can be both incremental development activities (which are pieced together to produce the final software product) or can be one-off, focused periods of time where a single task is carried out.

Structure:

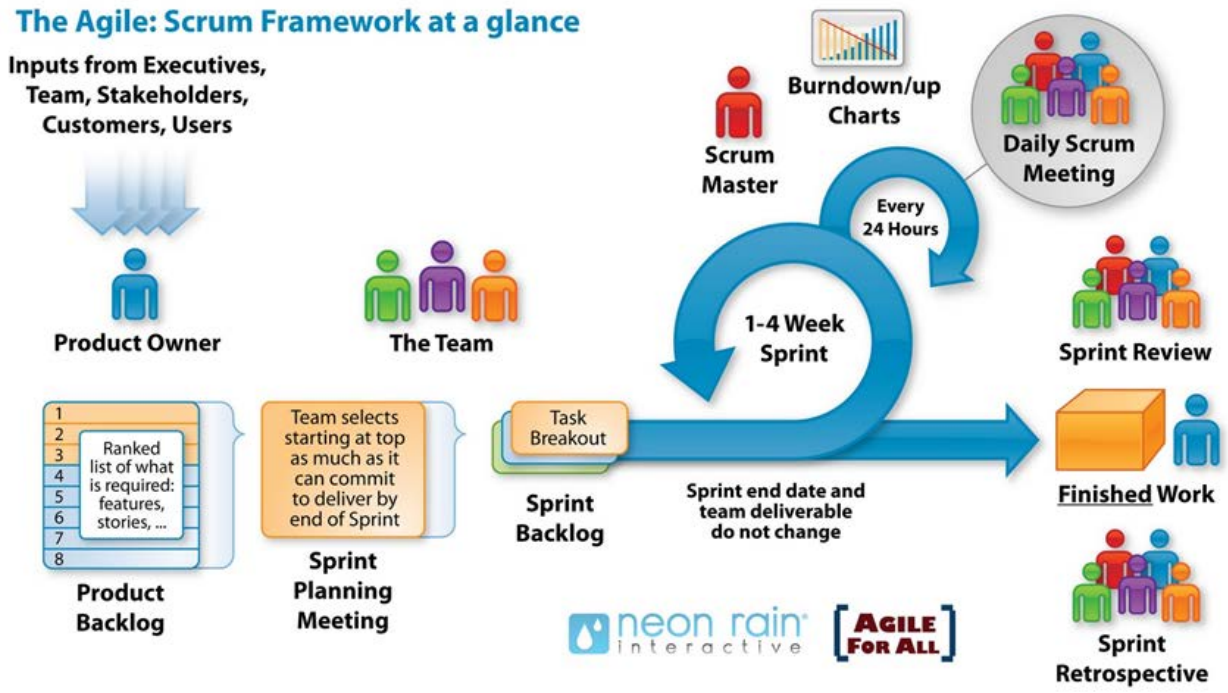

Figure B-2: The Agile Scrum Framework at a Glance[24]

The Scrum framework is based around Scrum teams, where each team consists of three main roles:

1. The **Product Owner** is responsible for representing the stakeholders.
2. The **Development Team** is the group that carries out software coding, implementation, testing, and development.
3. The **Scrum Master** is responsible for removing impediments to the ability of the team to deliver the product goals and deliverables, and ensures that the schedule and program are followed appropriately. The Scum Master differs from a traditional project manager; PMs traditionally have personnel management responsibilities, which are not within the Scrum Master's scope of responsibilities (due to the cross-functional team structure).

In a Scrum project, the teams are self-organizing and choose how best to accomplish their work, rather than being directed by other individuals from outside the team. Furthermore, these teams are cross-functional, including members who have the capabilities to achieve the work without depending on other individuals from outside the team. This model is designed to optimize flexibility, creativity, and productivity.[25]

[24] Kumar Sathees, "What is Agile Scrum?" Art of Project Management Blog, (November 10, 2011), http://satheespractice.blogspot.com/2011/11/what-is-Agile-scrum.html.
[25] U.S. Immigration and Customs Enforcement, *Agile Software Development at U.S. Immigration and Customs Enforcement*" (Washington: Department of Homeland Security), 35.

Sprints are the basic time unit in Scrum, with each sprint restricted to a specific duration (i.e., each sprint is "time-boxed"). Sprints may be no longer than four weeks, with most teams choosing to work in two week iterations. Sprints have clearly defined deadlines and deliverables, with the Scrum Master tracking progress and ensuring completion. These sprints are planned by holding "Sprint planning meetings" to determine the type of work to be done, preparing the Sprint "Backlog" (or prioritized list of tasks to be done during the sprint), and communicating expected responsibilities between team members. Teams meet daily during sprints, at a "Daily Scrum" or "Daily Standup," lasting no more than fifteen minutes. Each sprint is intended to produce, among other things, potentially shippable increments of code that are ultimately built into the final product solution.

The Sprint backlog is drawn from the overall product backlog, which is the ordered and prioritized list of items that collectively describe all the features needed for a software product. These requirements are decomposed into user stories and specific story points, which describe what to deliver and in what order.

Finally, a *burndown chart* is a publicly displayed chart that shows the remaining work/tasks in the sprint backlog. It shows where the team stands regarding completing the tasks that comprise the backlog items that achieve the goals of the sprint. The X-axis represents days in the sprint, while the Y-axis is effort remaining. To motivate the team, the sprint burndown chart should be displayed prominently. Ideally, the chart burns down to zero by the end of the sprint.

Scrum is the most common Agile approach[26] in today's IT environment. It does, however, require a level of discipline and rigor. Successful Scrum projects also require an informed and strong Scrum Master who can facilitate the Scrum meetings and coach the team to maximum performance. While Scrum is based on a cross-functional, non-hierarchical team structure, it is important to note the importance of a well-versed project manager in the DHS enterprise when evaluating whether Scrum would be a successful approach for project teams. The most critical role in Scrum is the Product Owner, who is empowered to make, change, and develop priorities for the project.

Key Resources:

As Scrum is the most widely used Agile development methodology, there are several key references for learning more about the approach:

1. Scrum Alliance®

   Scrum Alliance is a membership organization that encourages and supports the adoption and effective practice of Scrum.[27] A nonprofit organization that promotes the value and importance

[26] "State of Agile Survey: Agile Methods and Practices: Agile Methodology Used," VersionOne (2014), http://www.versionone.com/pdf/2013-state-of-Agile-survey.pdf.
[27] "Who is Scrum Alliance?" Scrum Alliance, updated 2014, http://www.scrumalliance.org/about-us.

of Scrum adoption and utilization, Scrum Alliance is a community resource for learning more about Scrum's applications and benefits, as well as training resources and certification courses.

http://www.scrumalliance.org/

2. The Scrum Guide™ – "The Definitive Guide to Scrum: The Rules of the Game"

Scrum.org serves as the source for Ken Schwaber and Jeff Sutherland's "The Scrum Guide," a 2013 document providing a comprehensive outline and discussion of Scrum's framework construction, the roles and responsibilities of project members, and Scrum's intended purpose and uses.

http://www.scrumguides.org/

3. U.S. Immigration and Customs Enforcement - *Agile Software Development at U.S. Immigration and Customs Enforcement* (2013)

This ICE document provides a structured overview of how software is implemented at ICE using Agile development methodologies. It identifies roles and responsibilities, best practices in Agile development, and a list of integrated tools available within the ICE Integrated Development Environment (IDE). This document also explains guidelines for documentation, integration to support legal processes, and resources for core/other critical documentation regarding Agile within DHS.

B.2 Kanban

Overview:
Kanban seeks to alleviate the bottlenecks in traditional development by not overburdening the development team and limiting "in-progress" work in order to efficiently and effectively design and deliver products to users. Kanban is based on three basic principles: visualize what you do today (workflows), limit the amount of work-in-progress (WIP), and enhance flow (backlog prioritization). These principles allow Kanban to be responsive to change and allows for priorities to change frequently. It requires short cycle times that deliver features to end users faster.[28]

Structure:
Kanban, while often mentioned in the same breath as Scrum, differs in some key areas. **Error! Reference source not found.** provides detail on how these similarly collaborative yet structurally varying methods differ:

Table B-1: Kanban – Scrum Characteristics Comparison[29]

Characteristics	Kanban	Scrum
Roles	No prescribed roles	Pre-defined roles of Scrum Master, Product Owner, and Team Member
Product delivery	Continuous Delivery	Time-boxed sprints
Work flow	Work is pulled through system (single process flow)	Work is pulled through the system in batches (via sprint backlog)
Change allowance	Changes to scope of backlog can be made at any time	No changes to Sprint scope of work allowed mid-Sprint
Measurable	Cycle time	Velocity
Appropriate developmental environment	More appropriate in operational environments with a high degree of variability in priority	More appropriate in situations where work can be prioritized in batches

Most notably, Kanban is structured to reduce bottlenecks that reduce a pipeline's flow. It does so by maintaining a clear, visual representation of the work items as they flow through the development process through the use of a **Kanban board** (represented in Figure B-3 with columns indicating specific processes). The Kanban board represents the "pull model" emphasized by Kanban, exposing bottlenecks in work flow so the team can address them quickly. Teams do not have prescribed roles, allowing for flexibility and allowing members to work on each other's artifacts easily.[30]

[28] "What is Kanban?" VersionOne, updated 2014, http://www.versionone.com/what-is-kanban/.
[29] Mike Bria, "Comparing Kanban to Scrum," InfoQ (May 13, 2009), http://www.infoq.com/news/2009/05/kniberg-kanban-v-scrum.
[30] "What is Kanban?" Kanban Blog (2014), http://kanbanblog.com/explained/.

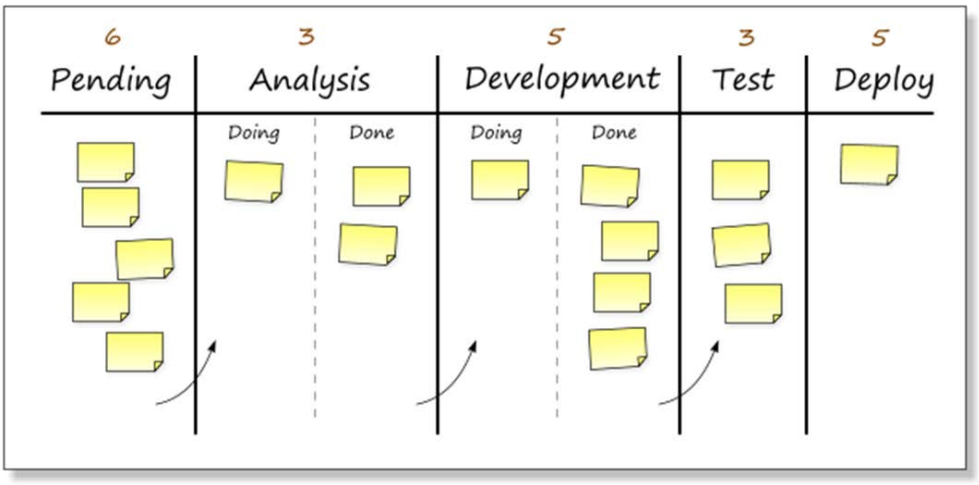

Figure B-3: Kanban Work Item Board Sample[31]

The numbers at the top of each column are the limits on the number of work items allowed per column (represented by sticky notes and indicative of team capabilities/available resources). As tasks are completed in each process, the notes are physically moved into the next process and keep team members aware of work to be done, completed tasks, and where focus may need to shift one way or another.

Key Resources:

1. VersionOne®: Agile Made Easier

   VersionOne is a tool for increasing continuous delivery capabilities while not overburdening the development team. Though a software/project management solution at its core, VersionOne also presents significant online resources that highlight the key differences between Kanban and Scrum, explains how Kanban is structured to promote effective development team work, and discusses the software solution more in depth itself.

   http://www.versionone.com/what-is-kanban/

2. Kanban Blog

   This resource examines the benefits and drawbacks of Kanban, including the presentation of several key visual representations of work flow processes and how they relate to development activities for Agile environments and team members.

   http://www.kanbanblog.com

[31] "What is Kanban?" Kanban Blog (2014), http://kanbanblog.com/explained/.

B.3 Extreme Programming (XP)

Overview:
Extreme Programming (XP) advocates for frequent releases in short, iterative development cycles. This is intended to promote team productivity and introduce checkpoints at which various customer/stakeholder requirements can be introduced, refined, and/or adopted. XP was originally developed by Kent Beck. While working for Chrysler Corporation in 1996, he published and expanded upon the method in *Extreme Programming Explained (2000).*[32]

XP originally relied on four key values to achieving program success: communication, simplicity, feedback, and courage. A fifth value, respect, was added at a later date by Beck. In terms of structure, XP draws upon industry best practices (such as programming in pairs or doing iterative and redundant code testing by all team members) and ties them together in an effort to further more effective project output.

Structure:
As Beck explains, XP attempts to reduce the cost of changes in requirements by having multiple short development cycles/iterations, rather than a single long cycle. XP presumes that changes are a natural, inherent aspect of software development projects to be planned/accounted for rather than avoided. The approach focuses on coding and ensuring that team members have a complete understanding of business requirements early in the process in order to best satisfy stakeholders/users.[33] XP is structured around iterative planning and feedback loops, as illustrated in Figure B-4, which also provides the timeline that the feedback process takes during the project life cycle.

[32] Kent Beck and Cynthia Andres, *Extreme Programming Explained: Embrace Change* (Boston: Addison-Wesley Professional, 2004).
[33] Python syntax coding graphic, Wikipedia Media Commons (used by permission), http://en.wikipedia.org/wiki/File:Python_add5_syntax.png.

Figure B-4: Extreme Programming Planning and Feedback Loops[34]

XP employs four basic activities during the software development process: Coding, Testing, Listening, and Designing. Each of these activities is described in Table B-2.

Table B-2: Extreme Programming Activities

Activity	Description
Coding	The emphasis of XP programming is the importance of having working code for a product to be of any value to the customer. Without the code, the product cannot be used. Accordingly, XP sees coding as an integral part of working through complex problems and allowing the programmer to identify clear and concise thoughts in a way that others can understand and, in turn, attempt to solve together.
Testing	Includes *unit tests,* where the programmer thinks of every possible test to "break" the code, as well as *acceptance tests* (which verify that programmer's perceived requirements fulfill the customer's requirements) and *integration testing* to evaluate the project component within the overall system environment.
Listening	Posits that programmers focus on understanding what the customer needs the system to do, and focuses on the "business logic" or business rules that are needed for it to be successful. XP advocates that programmers understand these needs in order to relate the technical aspects of a requirement or, ultimately, a product in order to fulfill these needs and uses.
Designing	Refers to the logic of structuring the system to avoid dependencies within the system such that changes in one part of the system do not adversely affect other parts of the system.

[34] Beck and Andres, *Extreme Programming Explained.*

The XP five key values guide how team members, project managers, and stakeholders can better interact and collaborate to ensure product quality (see Table). These values, when employed by project teams, are aimed at allowing teams to benefit from clear coordination and feedback throughout the development process.

Table B-3: Five Key Values of XP

Value	Explanation
Communication	System requirements are effectively communicated from the customer to the team. XP rapidly builds and passes along institutional knowledge amongst members of the development team in an effort to give one another consistent information. XP advocates that if the designers of the system share the view of the users of a system, it can be designed and constructed more effectively.
Simplicity	XP emphasizes starting with the simplest possible solution, upon which further functionality can be built later. This approach deviates from traditional programming, as XP programmers attempt to anticipate the needs of today rather the needs of the future. XP proponents note that the disadvantage is the increase of work at later dates in certain cases, though they argue that the disadvantage is significantly outweighed by the advantage of not investing in more task work up front in the project.
Feedback	Feedback within XP refers to the different dimensions of system development: feedback from the system (periodic integration tests and unit tests to allow for fixes before product is released), feedback from the customer (acceptance tests by the customer to directly ensure it meets their needs), and feedback from the team (estimates of completion time required are provided by the team to the customer).
Courage	Encourages programmers not to hesitate to throw away portions of code if they have worked on it; improved code can lead to better results and remove impediments to effective development.
Respect	Encourages team members to give and feel the level of respect they deserve as team members. Value comes from participation, and developers are encouraged to respect the expertise of customers and vice versa. PMs and executives respect team members' responsibility and appropriate authority over their work.

Key Resources:

1. *Extreme Programming Explained: Embrace Change* (Beck and Andres 2004)

   Beck and Andres' text provides the foundations for XP understanding and use in today's programming environment. They explain how XP improves software development practices by integrating the above concepts into daily processes. The 2nd edition (2004) expands upon the original text by emphasizing how XP can be used on teams of larger than 6-12 (as noted in the 2000 version).

   Available in print or PDF format at http://dl.acm.org/citation.cfm?id=1076267

2. XP123

   XP123.com is a site exploring topics relating to Agile and Lean software methods, specifically emphasizing XP. It includes a number of posts on how XP can be used, including a large number of articles sorted by the role of the Agile developer/team members. One can also find a set of visuals (charts, pictures, team rooms, etc.) to help indicate what a motivating "Agile environment" can look like.

   http://xp123.com/

B.4 Lean Software Development

Overview:
Adapted from the Toyota Production System, Lean software development is a meld of lean manufacturing and lean IT principles and practices to form a more streamlined software development domain. "Lean development" stems from a book by Mary and Tom Poppendieck, which discusses the original lean principles alongside a set of 22 tools.[35] These tools are compared to Agile practices and expanded upon to show the parallel to Agile activities and how Lean principles can be incorporated into Agile projects to increase effectiveness and efficiency.

Structure:
Lean software development is structured around seven key principles, aligned closely with those found in the aforementioned lean manufacturing domain (see Figure B-5):

Figure B-5: Lean Software Development Principles

1. **Eliminate Waste**: Provide market and technical leadership; create nothing but value; write less code.
2. **Amplify Learning**: Create design-build teams; maintain a culture of constant improvement; teach problem solving methods.
3. **Deliver Fast:** Deliver solutions in small iterations; limit work to known/pre-established capacity; focus on cycle time, not utilization of the product; release product(s) early and often.
4. **Defer Commitment**: Schedule irreversible decisions for the last responsible moment; break any possible dependencies between components; maintain options for solutions/decisions.

[35] Mary Poppendieck and Tom Poppendieck, *Lean Software Development: An Agile Toolkit* (Boston: Addison-Wesley Professional, 2003).

5. **Empower the Team**: Train team leaders and supervisors; move responsibility and decision-making to lowest possible level; motivate team to support continuous process improvement.
6. **Build Integrity In:** Synchronize effort; automate testing/routines; refactor to avoid code duplication.
7. **Optimize the Whole**: Focus on value to the customer; deliver a complete product with input from all designers/stakeholders.

These principles guide Lean software development by emphasizing limiting any "waste" that project teams create (i.e., duplicate code, re-iteration of working components, extensive documentation of activities beyond what is required) to achieve a more streamlined, efficient project outcome.

It is also important to note that Lean software development focuses on the front portion of development processes, areas that Agile and traditional development activities tend not to directly address. Emphasizing the front (and final) steps helps to ensure that the project development and deployment, including Agile practices and test-driven development (TDD) efforts/patterns, run as smoothly and efficiently as possible. These patterns and TDD efforts seek to set the level of expectation using automated testing. They only develop the necessary amount of work needed to pass those tests/expectations (known as "just good enough" development).

One of the benefits of a Lean approach is that it ties the Agile principles into the part of the project lifecycle that is not addressed by other principles: the visioning, approval, contracting, and staffing of Agile projects (as noted in Figure B-6). Approaching projects with this mindset is key in ensuring that all components of a project's lifecycle incorporate these Lean principles rather than simply having *portions* of the process be Lean in nature.

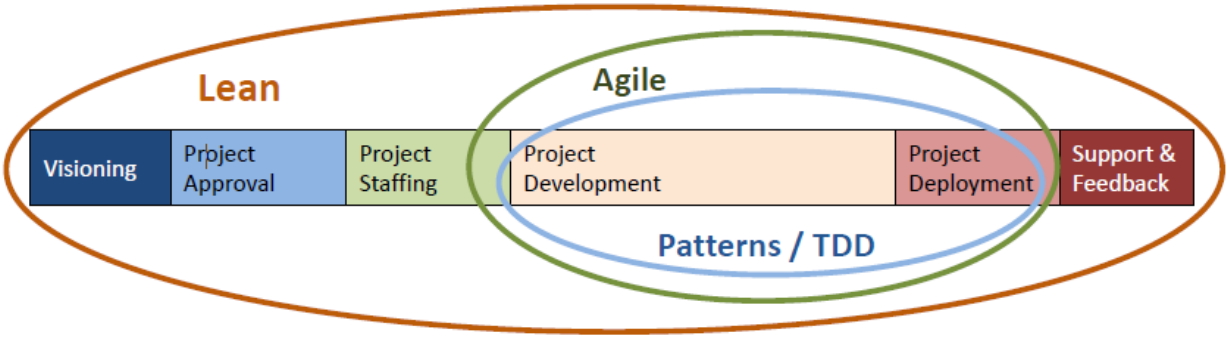

Figure B-6: Lean Software Development Along Product Lifecycle[36]

[36] Abel Avram, "Presentation: Principles and Practices of Lean-Agile Software Development," InfoQ (November 20, 2008), http://www.infoq.com/news/2008/11/Lean-Agile-Alan-Shalloway.

Key Resources:

1. IBM's "Agility @ Scale: Strategies for Scaling Agile Software Development"[37]

Provides a high-level understanding of how to scale Agile development to fit within the enterprise perspective. Posts discuss the differences between traditional approaches and Agile, how to institute the necessary culture changes to effectively adopt Agile, and how Agile effectiveness is evaluated. Furthermore, discussion of the seven principles of Lean point to *why* these principles are valuable for development teams, and how they can increase efficiency and product quality.

https://www.ibm.com/developerworks/community/blogs/ambler/?lang=en

2. Mary and Tom Poppendieck – "Lean Software Development: An Agile Toolkit"[38]

Reviews the Lean Software Development principles, practices, and implementation to development efforts across multiple domains. The book goes in depth into the 22 thinking tools that "aid software development leaders as they develop the Agile practices that best fit in their particular domain."[39]

[37] Scott Ambler, "Agility@Scale: Strategies for Scaling Agile Development Software: How to Scale Agile Software Development," IBM developerWorks, updated April 2014, https://www.ibm.com/developerworks/community/blogs/ambler/?lang=en.
[38] Poppendieck and Poppendieck, *Lean Software Development Agile*.
[39] Ibid., xxiv.

B.5 Disciplined Agile Delivery (DAD)

Overview:
"The DAD process framework is a people-first, learning-oriented hybrid Agile approach to IT solution delivery. It has a risk-value life cycle, is goal driven, and is enterprise aware."[40]

In other terms, the DAD framework scales Agile strategies to address the full IT product delivery process from project initiation to deployment into production. Based on the Agile Scaling Model, the framework adds more lean governance techniques to balance self-organization and adds a risk-driven viewpoint to the value-drive approach that, in turn, increases the chance of project success.[41] DAD incorporates the Agile Unified Process (AUP), which was subsumed into DAD after 2012.

Structure:
DAD is a hybrid process, adopting and tailoring strategies from a number of resources. Specifically, DAD adopts strategies from Scrum, Extreme Programming (XP), Agile Modeling, AUP, and Kanban. DAD is goal driven, emphasizing the delivery life cycle and how products really serve to be solutions (rather than simply products alone).

DAD avoids the traditional "handoffs" of work products (primarily documents) that often contribute to work bottlenecks and are wastes of time and money. DAD leverages the cross-functional nature of teams and team members to encourage one another to perform work related to disciplines rather than their specialty. Accordingly, the more effective use of resources and the reduced reliance on formal documentation and sign-offs leads to more effective, more efficient development and improved productivity. Table explains the five primary roles in DAD.

Table B-4: Five Primary Roles in DAD

Role	Description	Responsibilities
Stakeholder	Someone who is materially impacted by the solution.	Provide requirements, either as part of the project team or through a team representative, in order to inform user stories. Responsible for ensuring that developed products satisfy all appropriate requirements following iterations and tests, thus preparing them for release. Stakeholders include four distinct sub-groups: End Users (who actually use the system), Principals (decision-makers who pay for the system), Partners (who make the system work in the environment with other existing systems), and Insiders (members of the development team).

[40] Scott Amber and Mark Lines, "Disciplined Agile Delivery: An Introduction," *Thought Leadership White Paper* (IBM Software Design and Development, 2011).
[41] Amber & Lines, 2011.

Role	Description	Responsibilities
Product Owner	Individual who speaks as the "one voice of the customer."	Clarifies details and maintains list of work items that team needs to implement. Represents work of Agile team to stakeholder community.
Team Member	Focused on producing the actual solution for the stakeholder(s).	Performs analysis, testing, evaluation, design, programming, planning, estimation, and many more activities throughout the project. *Note: DAD recognizes that not all software programmers[42] on the project team write code; hence they use the term "team member" rather than "developer."*
Team Lead	Agile Coach, helping team to focus on delivering work items and fulfilling iteration goals/commitments made to product owner.	Acts as leader, facilitating communication and empowering team members to self-optimize their processes. Ensures that the team has the resources it needs.
Architecture Owner	Owns the architecture decisions for the team and facilitates the creation and evolution of the overall solution design.	Makes system architecture decisions for the team and ensures that the solution is integrated and tested on a regular basis. The role of Architecture Owner may be filled by the individual in the Team Lead role on smaller Agile teams.

Key Resources:

1. "Disciplined Agile Delivery: An Introduction" (Amber and Lines, 2013)

Provides a detailed overview of DAD, including visual representations of the delivery life cycle (including phase breakdowns), explains how to leverage enterprise assets, and provides resources for individuals looking to understand more about Agile and DAD specifically.

http://public.dhe.ibm.com/common/ssi/ecm/en/raw14261usen/RAW14261USEN.PDF

2. Disciplined Agile Delivery community webpage

Explains the DAD Agile process framework in both high- and low-level styles, providing resources for understanding what DAD is, why it is used, and a community for interaction between DAD users to help others understand and solve challenges to Agile implementation.

http://disciplinedAgiledelivery.com/

[42] Scott Amber and Mark Lines, "Disciplined Agile Delivery: An Introduction," *Thought Leadership White Paper* (IBM Software Design and Development, 2011), pp. 6.

B.6 Scaled Agile Framework® (SAFe®)

Overview:
The Scaled Agile Framework® is a framework for applying Lean and Agile principles at an enterprise scale.[43] Unlike the previous Agile methodologies discussed in this section, SAFe® is not intended to provide a way to carry out an Agile project at the individual level. Rather, SAFe® is a framework for applying the principles of Lean and Agile methodologies to projects within a larger program-level and portfolio-level enterprise. Specifically, it enhances PMs' and executives' ability to scale their development portfolios and programs to provide the value-add of specific Agile development projects. These projects, within the larger enterprise, are then scaled based on the SAFe® architecture to deliver software in accordance with specific tasks (team-level) user stories and epics (program level), and investment areas (portfolio-level).

Structure:
SAFe® is structured along three levels: Teams, Programs, and Portfolios. These levels allow for the iterative processes done by individual development teams, such as Scrum, XP, Lean, or others to feed into a larger programs and the overarching portfolio vision for development within an enterprise. In simple terms, SAFe® is an approach for *scaling* an Agile program to fit into an enterprise environment.

These levels connect to one another through a multitude of ways, notably via specific **tasking processes** (i.e., a Portfolio backlog is comprised of Program-level releases), **planning and "value streams"** (placing specific value on individual programs and projects to enhance the larger portfolio), and **development releases**.

Similar to traditional Agile methodologies, SAFe® has the main ingredients of Agile:

- Values: Requirements and backlogs
- Teams: Development teams (three to five members) up through the portfolio teams (up to several hundred or thousands of members)
- Time-boxes: Iterations, potentially shippable increments, budget cycles

1. Values

   Values are drawn from the requirements and product backlog for Agile teams. To structure these to an enterprise scale, the architecture for values is based on a hierarchy. Stories are based on stakeholder requirements, and subsequently feed into features, the services that the system provides to fulfill stakeholder needs. These features are tied into higher-level epics, which are referred to as either architectural epics supporting technology initiatives

[43] "About the Framework," Scaled Agile Framework® (2014), http://ScaledAgileFramework.com.

across the portfolio, or business epics, which are customer-facing initiatives that come from a business's strategy or are driven by strategic/market events.[44]

2. Teams

SAFe® project teams align with the selected methodology, yet are advised to align with the traditional "cross-functional team" mentality and have between five and nine members. These teams carry out iterations, or Sprints, with two-week average timelines, producing potentially shippable increments (PSI) after each iteration.

At the program level, five to 10 of these Agile teams come together with specific roles and responsibilities (see Table).

Table B-5: SAFe® Program-Level Team Members

Role	Responsibilities
Program Manager	Responsible for identifying items to be added to the Program Backlog, prioritizing the Backlog, and interfacing with Product Owners to confirm alignment between the software components and goals of the enterprise.[45]
System Architect	Focuses on the stakeholder needs and ensuring that the solution is designed in order to cater to these needs while delivering functionality across various features, components, and the larger solution.[46]
User Experience Designer	Ensures that the user experience is consistent across each aspect of the solution (specifically the user interface), and is responsible for relaying the requirements for the user-system interaction across the solution to the program teams.[47]
Release Train Engineer	Considered to be the "Chief Scrum Master," responsible for organizing and conducting program-level meetings, managing impediments, and informing team members (at higher and lower levels) of any major issues or updates.[48]
System Team	Builds and uses the infrastructure to support development, such as testing platforms and automated testing frameworks. Also demonstrates the software products to stakeholders, and relays stakeholder feedback to the development team in order to refine and integrate these PSIs into the final software product(s).[49]

[44] "Business Epic," Scaled Agile Framework (2014), http://scaledAgileframework.com/business-epic.
[45] "Product Management," Scaled Agile Framework (2014), http://www.scaledagileframework.com/product-management.
[46] "System Architect," Scaled Agile Framework (2014), http://scaledAgileframework.com/system-architect/.
[47] "UX," Scaled Agile Framework (2014), http://scaledAgileframework.com/ux/ .
[48] "Release Train Engineer," Scaled Agile Framework (2014), http://scaledAgileframework.com/rte/ .
[49] "System Team," Scaled Agile Framework (2014), http://www.scaledagileframework.com/system-team/

Role	Responsibilities
Release Management Team	A cross-functional team made up of various representatives who serve as the governing authority over releases. Responsible for meeting both regularly and in surge capacity (as content is prepared for release) in order to evaluate progress and the overall quality of software before releasing it.[50]

3. Time-boxes

   SAFe® time-boxes all major activities across the framework. From iterations to releases to PSI demonstrations, all activities are intended to occur over certain planned timeframes. While allowing for adjustments due to changing requirements or technical reviews, the time-boxing approach seeks to keep all aspects of the development project and associated progress visible across its lifecycle. It also promotes frequent testing and demonstration of software in order to obtain timely and valuable customer feedback.

What differentiates SAFe® from a traditional methodology is that it allows for teams/projects/PMs to select their own methodology depending on their development plan, and then subsequently scale it to better fit into a larger enterprise environment. At DHS, PMs need to consistently demonstrate project progress and value to their respective leadership and stakeholders. SAFe® provides a user interface known as the "Big Picture," which serves as an interactive graphic to highlight individual roles, teams, activities, and artifacts required to scale Agile to the team, program, and portfolio levels.[51] The "Big Picture," found in Figure B-7 in its original and unaltered form, can help everyone from teams to portfolio managers understand how their roles and responsibilities fit into their appropriate development environment.

More information, including the user interface that describes roles, responsibilities, and processes within the SAFe® framework, can be found at http://scaledAgileframework.com/.

[50] "Release Management Team," Scaled Agile Framework (2014), http://www.scaledagileframework.com/release-management/

[51] "About the Framework," Scaled Agile Framework (2014), http://www.scaledagileframework.com/about/.

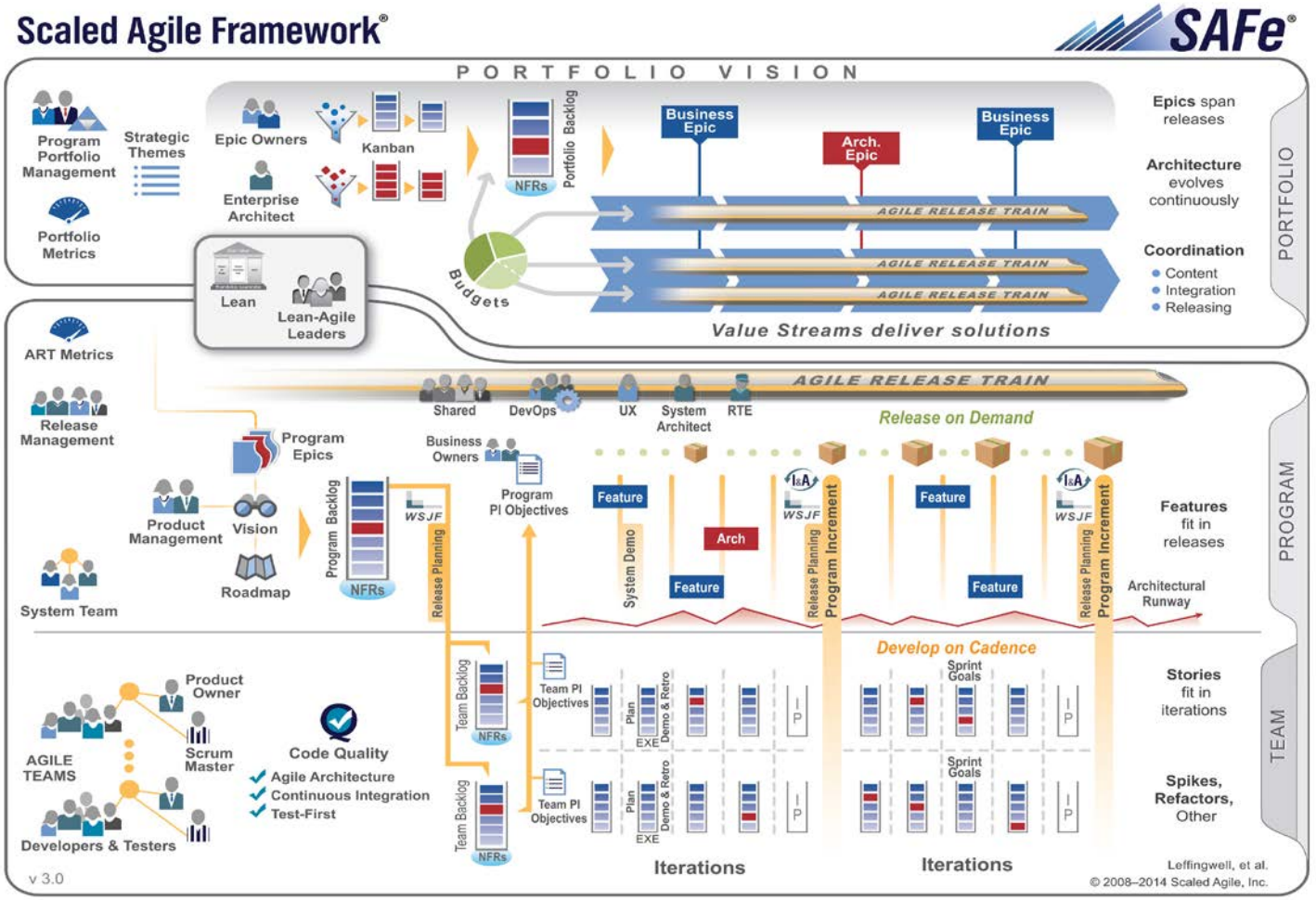

Figure B-7: Scaled Agile Framework® Big Picture[52]

[52] "Scaled Agile Framework," Scaled Agile Framework (2014), http://ScaledAgileFramework.com/.

(This page intentionally blank)

APPENDIX C: METRICS AND REPORTING

An integral component of Agile projects is the ability to examine and report project status, as well as the metrics and status associated with individual tasks, iterations, and projects as a whole. These metrics show PMs and executives alike how their project is progressing, how Agile is optimizing the use of team members and resources, and where the project stands in terms of key Agile measures such as velocity and burn rate.

C.1 Agile Metrics

It is important to note that there are different types of metrics; they may include contract/deliverable metrics, performance metrics, or more Agile metrics such as velocity and burn rate. As with many aspects of Agile, the factors that go into measuring an Agile project team's performance or progress are specific to the particular approach and project, and are not necessarily comparable *across* projects (particularly those of differing scale and impact) and methodologies. However, they are useful on *all* projects in informing the PM of valuable project context and performance on specific tasks, as well as the overall project. Table C-1 provides an overview of common Agile metrics.

Table C-1: Common Agile Metrics

Metric	Description	Calculation
Burndown rate	Show the rate at which work (i.e., features being developed, PSIs being released) is being completed. These are completed on both the iteration level and release level. Ideally, the burndown charts indicate that the amount of remaining work approaches zero and remaining work is depleted upon the completion of a specific iteration or release period.	The burndown rate is an estimate of how much a team can complete during an iteration. It shows how quickly a team is completing planned work, and how much this rate varies between days or iterations. The team burn rate can be calculated either through • Hours completed per day • Work items completed per day Hours are both *actual* (the number of hours a team completed during a specified interval, divided by the number of days in the interval) and *required* (the total number of hours estimated for an interval divided by the total number of days in the interval). Work items per day are calculated the same way, just with work items being substituted as the main variable instead of hours.

Metric	Description	Calculation
Burndown Items Remaining	Shows the number of remaining work items to be completed during a given iteration or project timeline.	To visualize/calculate the work items remaining graphically, the "ideal work remaining line" is a straight line that connects the start point to the end point of an iteration or project. The actual tasks remaining is a simple count of the number of remaining tasks. The project/iteration timeline is displayed on the x-axis, and the remaining number of story/task points (representing the remaining work to be completed) is displayed on the y-axis. For hours remaining, it is a calculation of the number of *hours remaining* (cumulative across all tasks) and the number of *hours completed*, which is then cross-referenced against the planned/forecast number of available work hours.
On Time Delivery	Measures user stories anticipated to be delivered during releases versus the number actually delivered in that sprint.	Number of stories forecast for an upcoming sprint divided by the number of stories delivered by that sprint.

Metric	Description	Calculation
Running Tested Features	Measures how many working features were delivered for deployment during a sprint in relation to the monetary investment.[53] The desired software is broken down into features/stories, which are part of what ultimately has to be developed. These features/stories then go through automated tests, and if successful, are implemented/released. The measure takes into account how many of these stories are identified, pass testing, and then are released. RTF is measured weekly, and broken down as follows: • *Running*: Features that are running as a single software product, and includes all necessary components; if components are left out or dysfunctional at the time of measurement, it is not considered running • *Tested*: Did the software pass the automated tests or not, and did it function as the customer required/intended? • *Features*: End-user features, not technical features; these are components/aspects of the customer-provided requirements, and are captured as user stories RTF ideally shows, when visually represented, linear growth throughout the project. Stagnant or decreased RTF values are indications of products encountering testing troubles or issues such as changing requirements or failing to meet customer requirements in the final "Features" stage.	The visual graph is produced by lining "features completed and ready for deployment" on the y-axis and "number of days through a sprint (aka Sprint Day # ___)" on the x-axis.
Team Member Load/Effort	Indicates individual contribution to projects to provide transparency across the project and allow the PM (and team members) to balance the workload as necessary.	User-reported effort (integrated into many Agile project and performance management tools) by iteration, user story, or desired unit of measurement. Is reported as a standard for all individuals.

[53] Ron Jeffries, "A Metric Leading to Agility," XProgramming.com (June 14, 2004), http://xprogramming.com/articles/jatrtsmetric/.

Metric	Description	Calculation
Velocity	Rate at which a team produces working software. Velocity measures the number of units (builds, releases, etc.) completed during a certain defined time interval. From a planning perspective, this time interval is defined at the outset of the project, using velocity to determine how the team anticipates producing software increments and releases. The final velocity rates from multiple iterations are often reported together in order to more accurately provide context for and predict subsequent iterations. Velocity is, in some cases, an estimate. It estimates the level of effort used within a given iteration, and identifies the number of user stories that were completed during the same iteration. Once the velocity is calculated, the team can then revise estimates on how long a project takes to complete, based on estimates of the remaining user stories and assuming team effort and velocity remain constant throughout the remaining iterations.	Velocity is the Units of Effort Completed / Sprint. Calculated following iterations and applied to work planning for following iterations.
Work Item Count	Visualization of product releases/components of release development efforts in progress.	

Earned Value Management:

One common approach for determining the proper metrics to apply to an Agile project is AgileEVM, which takes the unit of measurement down from the project level (as is typical with standard EVM practices) to the scrum story-point level in order to measure the value of specific user stories. These metrics help to report items that traditional Agile measurement (e.g., velocity, burndown charts) may not specifically address. NOTE: One successful DHS program captures metrics at the Feature, Feature Point, and User Story level, but its official measure set is via Feature Point. This method of Agile EVM has been approved by DHS Program Accountability and Risk Management (PARM) during oversight of the program.

To measure EVM, there are specific calculations that are combined or adjusted to get the desired metrics. As explained by Saji Pillai on ScrumAlliance.org,[54] the calculations require first identifying

[54] Saji Pillai, "Agile Project Reporting and Metrics," ScrumAlliance (July 14, 2013), http://www.scrumalliance.org/community/articles/2013/july/Agile-project-reporting-and-metrics.

the Budget at Complete (BAC) (total budget assigned to complete the work) and the Actual Cost (AC) (dollar value for the cost incurred by a given increment of work).

The metrics to indicate EVM are then calculated, as indicated in Table C-2:

Table C-2: Earned Value Management Metrics[55]

Metrics	Formula	Metric Analysis
Planned Value (PV)	BAC * Planned % Complete	Planned value of the work generated by a particular milestone or point in time. Measured in dollars or hours.
Earned Value (EV)	BAC * Actual % Complete	Indicates actual value of the work generated by project at a particular milestone or point in time. Measured in dollars or hours.
Cost Performance Index (CPI)	EV/AC	Indicates how much monetary value has been "earned" out of every dollar spent (cost efficiency).
Schedule Performance Index (SPI)	EV/PV	Measures schedule efficiency, indicating how fast a project is progressing against the rate of planned progress.
Estimate to Complete (ETC)	(BAC-EV)/CPI	Forecasted amount to complete the remaining work, based on past performance. Measured in dollars or hours.
Estimate at Complete (EAC)	BAC/CPI or AC + ETC	Forecasted cost for the total work in the project plan, based on past performance.
Planned Release Story Points (PRSP)	Number of story points (n) at the product backlog level	Story points are defined at the product backlog level. Compared against actual story points released upon sprint/project completion.

C.2 Agile Reports

The types of reports produced and examined by Agile team members, project and program managers, and portfolio-level decision-makers are vital in visualizing/documenting project progress, enhancing the ability to track the contributions from individual team members, and providing the perceived value for specific developments efforts. Different Agile roles require different reports, which are most commonly created and provided at the end of each iteration. An introduction to the types of reports that project teams and PMs generate is described in the following subsections.

C.2.1 Backlog Reports

Backlog reporting is key in any Agile environment. Specifically, Agile methodologies recommend reporting the Product Backlog and Sprint Backlog at the end of each iteration, thus respectively highlighting the features that made up the just-completed iteration product and the features

[55] Ibid.

committed to delivering in a subsequent iteration. These include the user stories and, commonly, a visual representation of how the recently-included stories map back to the original requirements.

Complementary to the Product and Sprint Backlog reports is a *Changes* report, which highlights any within-iteration changes to any added, deleted, or modified features while a project is in progress. This serves to help provide the Product Owner with information and accountability, as it acts as a traceability document for budget retrospectives; in other words, it give the decision maker the information needed to ensure not only that they received the features they agreed upon, but that they can see why some may have been added, adjusted, or removed during the project lifecycle.

C.2.2 Release Plan and Progress Report

A release report highlights the current burn-up rates for a specific release, an update on the status of a storyboard (which includes the number of user stories both completed and remaining for the given iteration and full release), and a summary of how this release fits within the larger project timeline. The release plan can also tie into the original project plan by highlighting progress on the mapping of user stories to specific iterations (completed and remaining) as well as delivered features.

C.2.3 Burndown Charts and Burn Rate Reports

Burndown reports include the burndown chart, visually indicating the amount of work completed and how much work there is remaining for a certain iteration or release, along with the burn rate, and the assignment of remaining tasks and work items to individual team members. The burndown reports are more often used internally but may be tailored to provide overview information on a given iteration for PMs and program-level executives when required.

C.2.4 Test Reports

Reporting on testing is often one of the most valuable pieces of information for customers and stakeholders, as it shows how well the project is holding up against the testing that ultimately determines the software's usability and how it interacts with existing systems. Testing occurs both internally and externally for projects, with automated acceptance testing feeding into the integration with the larger software project and testing against the user interface.

Test reports include an overview of the events and activities that occurred during a given development iteration, any alterations of pre-existing artifacts (or refactoring of code), an assessment of the testing procedures that were in place and occurred, and a presentation of the results and evaluation of the software tested. In many cases, test reports can also include previous test results in order to provide context and enhance a stakeholder's understanding of how the most recent iteration fits within the project lifecycle.

In sum, there are a number of key reports common to Agile development. While no project requires the exact same documentation as another, there are valuable resources and tools that can help PMs to stay informed throughout the project and its iterations while providing valuable information to their executives. Refer to Section 4.4.1 to learn more about which tools may be able to help document this information and produce these types of reports.

APPENDIX D: TRAINING AND ADDITIONAL DHS RESOURCES

The Homeland Security Acquisition Institute (HSAI) offers the following Agile training courses to DHS staff:

- AQN AGI: Agile Acquisition (2 days)
- APM MIT: Managing IT Projects (4 days)
- AQN REQ: Fundamentals of Requirements Writing (3 days)

Full course descriptions and schedules are available at the HSAI web site.

http://dhsconnect.dhs.gov/ORG/COMP/MGMT/CPO/PAW/Pages/AcquisitionWorkforceTrainingPrograms.aspx

The U.S. Citizenship and Immigration Services/Office of Information and Technology (USCIS/OIT) Applied Technology Division offers the following Agile training courses to DHS staff:

- Introduction to Agile Using Scrum
- User Story Workshop
- Product Owner Training
- Introduction to Kanban
- Introduction to Automated Testing
- Test Planning for Automated Testing
- Introduction to Performance Testing Web Applications

The course schedule can be viewed here:

http://ecn.uscis.dhs.gov/team/mgtoit/Offices/oit/ATD/Training_Registration/SitePages/default.aspx

The Customs and Border Protection Office of Information and Technology CBP/OIT offers its staff the following vendor-provided training courses in its OIT Agile Program/Project Management training program:

- Certified Scrum Master (CSM) Training
- Product Owner Workshop
- Effective Agile Requirements Gathering (Tailored for CBP/OIT)

The DHS Enterprise Business Management Office SELC/Agile Team maintains the DHS Agile Center of Excellence website, which contains following:

- Articles - The latest Agile news and developments from DHS, Government, and Industry
- Governance - Agile policy, guidance, and directives
- Templates - Reusable Agile artifacts and forms
- Events - Upcoming Agile conferences and training
- Links - Agile links from inside and outside DHS

- Tools - Job aids, tutorials, and information decks

The DHS Agile Center of Excellence website can be found here:

http://dhsconnect.dhs.gov/org/comp/mgmt/cio/ebmo/Pages/Agile.aspx

DHS Components offering or planning to offer additional Agile training courses or other resources are encouraged to send that information to the DHS CIO Enterprise Business Management Office (EBMOSELCTeam@HQ.DHS.GOV) for inclusion in the DHS Agile Center of Excellence website, future versions of this Instruction Manual , and related DHS Agile guidance publications.

APPENDIX E: LEXICON

Term	Definition/Explanation
Artifacts	Artifacts refer to the by-products (physical pieces of information) of software development, such as enterprise architecture models, design documents, use cases, or user stories. Other artifacts can include project plans, data objects (applications), or testing plan outcomes.
Backlog	Collection of stories or epics remaining to be addressed. The backlog is generated for the overall product, each individual release (showing the number of user stories remaining for a specific release), and each iteration/sprint (identifying the remaining stories for a specific iteration/sprint).
Burnup/down chart	A visual aid to show the number of user stories being addressed during a given iteration (burnup) or the number of user stories remaining to be addressed within a given iteration (burndown).
Epics	Describe very large user stories that are eventually broken down into smaller stories. Within the SAFe® framework, ***business epics*** describe top-level business processes. ***Architectural epics*** describe the architecture the system is incorporated into.
Iteration	A time-boxed (fixed length) period of work performed by an Agile team. In Scrum and some other methodologies, iterations are called Sprints. Each iteration ideally reduces the backlog by completing work to satisfy one or more user stories.
Iteration/Sprint planning	A two-part planning meeting. The purpose of the meeting is to identify, prioritize, and commit to the stories for the next sprint. Part one of the sprint planning meeting is a review of the product backlog. This is when the product owner describes what needs to be built for the next sprint. During this part of the meeting, the team discusses the sprint objectives with the product owner, and asks clarifying questions about the stories. During part two of the sprint planning meeting, the team decides how the work will be built and begins decomposing the stories into work tasks and estimating story size.
Iteration/Sprint Review	Conducted at the end of each iteration to assess progress, requirements/user stories addressed, and software capabilities provided during the iteration period.
Lean Documentation (Artifacts)	The content of documentation or artifacts needs to be presented, but with an Agile program, the representation is minimized according to what is most efficient for the project.

Term	Definition/Explanation
Pair programming	A software development technique in which two programmers work simultaneously on a single task, with one (the driver) writing the code and the other (the observer) reviewing each line of code as it is written. The observer is also tasked with considering improvements and future problems within the work, and evaluates the progress to ensure that it meets any higher-level initiatives for the development. The two programmers involved switch roles frequently.
Refactoring	The process of re-configuring/restructuring existing code without changing the external behavior. The goals of refactoring are to improve code readability and complexity.
Release	Delivery of software from development into a production environment for use. A release may incorporate the product(s) of one or more Agile team iterations/sprints.
Release planning	Team activities focused on understanding the scope, constraints, process, practices, and required or prioritized products of a release, as well as developing a plan for successfully completing the release.
Release Readiness Review (RRR)	Conducted prior to releasing software to the product environment; focused on ensuring all elements of the release are complete including testing, documentation, approvals, accreditations, and provisions for sustainment. The RRR includes all major stakeholders, who also assess the release.
Roadmap	A project roadmap describes a set of planned releases, along with the high-level information about the associated scope, goals, schedules, etc., of the releases. When a "team of teams" approach is used, the roadmap is used to support Release integration planning. With most Agile artifacts, the roadmap is intended to be revised iteratively as the project plan evolves.
Sprint	Sprints (used in Scrum) are another term for iterative development periods. Sprints are typically between one week and one month in length with an average of two weeks. Sprints begin with a planning meeting, where tasks are identified and effort commitment for the sprint goals are made. The sprint ends with a Sprint Review/Retrospective, identifying lessons learned for the next review and evaluating progress achieved during that particular sprint.
Stories	Requirements, features, and/or units of business value that can be estimated and tested. Stories describe work that must be done to create and deliver a feature for a product. Stories are the basic unit of communication, planning, and negotiation between the Team, Stakeholders, and the Product Owner. They should provide business value at levels just low enough for the development team to design solutions (over-definition is discouraged to enable innovative, efficient design)
Tailoring	Within DHS, the SELC is the framework that is adjusted (tailored) for the extracting of a set of processes, tasks, and artifacts to best suit the program's objectives. What is tailored within the project, and how it is tailored, is

Term	Definition/Explanation
	based on the project characteristics, such as cost, schedule, risk, and mitigation strategy.
Technical review	Project/program reviews based on exit criteria that track progress in technical, schedule and/or management risk areas. In order for a technical review to be successfully passed or exited, the criteria demonstrate that the project is on track to achieve its final goals and can continue with additional activities within an acquisition phase or be considered for continuation into the next acquisition phase.[56]
Waterfall	A sequential software development process broken down into distinct phases (e.g., planning, requirements definition, design, development, testing, and implementation), with the rule that no phase can begin until the previous phase has been completed.

[56] Department of Homeland Security, Office of the Chief Information Officer, *Systems Engineering Lifecycle*: *Appendix B*, Version 2.0, Part II, (Washington: Department of Homeland Security, September 2010), 209.

(This page intentionally blank)

APPENDIX F: ACRONYMS AND ABBREVIATIONS

AD	Agile Data
ADA	Acquisition Decision Authority
ADE	Acquisition Decision Event
ALF	Acquisition Lifecycle Framework
APB	Acquisition Program Baseline
ASD	Adaptive Systems Development
AUP	Agile Unified Process
CAC	Capabilities and Constraints
CAE	Component Acquisition Executive
CDP	Capability Development Plan
CDR	Critical Design Review
CIO	Chief Information Officer
COR	Contracting Officer's Representative
COTS	Commercial-Off-The-Shelf
CPARS	Contractor Performance Assessment Reporting System
CPO	Chief Procurement Officer
DAD	Disciplined Agile Delivery
DCAM	DHS Collaborative Architecture Methodology
DHS	Department of Homeland Security
DSDM	Dynamic Systems Development Methodology
FRD	Functional Requirements Document
ICE	Immigration and Customs Enforcement
ILSP	Integrated Logistic Support Plan
IT	Information Technology
IRR	Integration Readiness Review
LBA	Lead Business Authority
LCCE	Life Cycle Cost Estimate
LSD	Lean Software Development
LTA	Lead Technical Authority
MD	Management Directive
MNS	Mission Needs Statement
NDI	Non-Developmental Item
ORD	Operational Requirements Document
OMB	Office of Management and Budget
OTRR	Operational Test Readiness Review
PARM	Program Accountability and Risk Management, Management
PDR	Preliminary Design Review
PM	Program Manager
PMP	Project Management Plan
PRR	Production Readiness Review
PSI	Potentially Shippable Increment
RRR	Release Readiness Review
RUP	Rational Unified Process

SAFe®	Scaled Agile Framework ®
SAR	Security Assessment Report
SDD	System Design Document
SDR	System Definition Review
SE	Systems Engineering
SELC	Systems Engineering Life Cycle
SER	Solution Engineering Review
SP	Security Plan
SPR	Study Plan Review
SRD	System Requirements Document
TEMP	Test and Evaluation Master Plan
TFS	Microsoft Team Foundation Server
XP	Extreme Programming

APPENDIX G: PM CHECKLIST

When tasked to take on an Agile IT project, or to migrate an existing IT project from waterfall to Agile, a DHS PM's first action is likely to be developing a top-level checklist for how to begin. The list of actions below may help the PM shape an approach:

1. **Know the Product Owner**: As noted in Section 2, the Product Owner/representative identifies the vision, and has overall responsibility for the DHS IT project. The PM's ability to successfully structure and manage the project plan is rooted in understanding the perspective of this key stakeholder *(see Sections 2.4.3 and 2.5.1).*

    - Identify who the users of system will be
    - Engage and spend time with current and prospective users of the system
    - Understand the scope of Product Owner/representative's project responsibilities for both business and IT communities

2. **Understand the Vision**: The product vision is a definition of what the project produces, who uses the IT system/services, and how it meets requirements and DHS strategy. DHS PMs of Agile projects are encouraged to revisit the product vision with the Product Owner periodically, jointly maturing the vision and adjusting project plans as the users, capabilities, and benefits become more completely understood *(see Sections 2.4.1, 3.2.3, 3.2.4, and 4.1)*.

    - Engage leadership early and often to relay these users' needs and how they fit in with DHS strategy
    - Specify the desired high-level functionality of the software/system, ensuring that the desired outcome is clear and defined
    - As the project matures, ensure that the requirements within the MNS or associated requirements are portrayed through user stories and story points, and are fulfilled in the final product/solution
    - If migrating from a waterfall development effort, consider a "phased implementation" to increase adoption of agile across programs

3. **Enhance Agile Knowledge:** If the PM is not already a highly experienced Agile practitioner, it is recommended that he or she seek out and take advantage of all available educational resources: read texts, join online communities of interest, review the applicable federal, DHS, and Component guidance, and attend training. DHS and Component CIO organizations may be able to identify experienced DHS PM Peers who may act as mentors (see Sections 2.4.7 and Appendix D).

    - Encourage Agile PMs and appropriate peers to share their Agile experiences, helping to identify best practices and resources for further learning
    - If possible, attend courses as listed in Appendix D of the Agile Development Instruction Manual to learn more about Agile activities and best practices

4. **Find an Agile Coach:** If no qualified Agile coaches are available within the organization, peer PMs or the CIO organization may be able to help identify suitable candidates whose services can be made available to the PM *(see Sections 2.4.6 and 5.2.4)*.

    - If possible, identify an Agile Coach with relevant training and qualifications
    - Through the appropriate contracting resources and other contacts, evaluate potential Coaches' experience and capabilities thoroughly, finding the best fit for your team

- Work with a coach/resource to clearly identify and establish project goals, desired team structure, and the required project environment so that everyone begins on the same page

5. **Assess the Project Environment:** Work with the Product Owner and Agile Coach to identify and understand the critical components of the project environment: project stakeholders (along with their perspectives and engagement), project resources and limitations (primarily in terms of budgets and schedules), existing project, technology and contracting options, and physical locations of and interrelationships among the key project entities (users, infrastructure architects, stakeholders) *(see Sections 2.3.1, 2.4.3, 3.1.1, 3.1.2, 3.2.2, and 3.2.3)*.

   - Identify project stakeholders and organizations
   - Engage relevant DHS/Component programs with similar existing projects, technologies, or contracting methods
   - Identify facilities and relevant project management tools for Agile team activities

6. **Develop a Project Plan**: At the highest level, this plan is likely to address options for incrementally developing and delivering the product. Working with the Product Owner and Agile Coach to outline the overarching project plan may produce valuable insights about the technologies, skills, infrastructure and other resources required to fulfill the plan *(see Sections 3.1.1, 3.3, 4.1).*

   - Identify at what points stakeholders/leadership interact directly with the PM (i.e., after each release or only at technical reviews)
   - Establish a clear understanding of executive involvement

7. **Develop a Project Team**: The Agile Coach and/or experienced peers may help the PM to identify DHS, contract, and other project support requirements. Capabilities of the project team include contracting officer support, as well as oversight, compliance, infrastructure, and resource management representation *(see Sections 2.4.1, 2.4.4, 4.2, and 5.1.3)*.

   - Consider a Team Process Agreement (TPA), outlining team make-up, roles and responsibilities, and establishing shared understanding of practices within the scope of the specific development project

8. **Develop an SELC Tailoring Plan**: Consider how to best meet the SELC requirements within the context of the incremental project plan and Agile approach. As the project plan and SELC tailoring plan mature, the content of those plans may help to inform choices about the Agile practices, methodologies, and tools best suited to support the project. As with the product vision, the project plan and SELC tailoring plans are likely to evolve. The PM is encouraged to review them jointly with the Product Owner, Agile Coach, and other stakeholders on a regular basis, and update them as needed *(see Sections 3.1, 3.2.3, 3.2.4, and 3.3)*.

   - Work with Product Owner, stakeholder(s), and appropriate oversight bodies to establish agreement on which Agile practices need to be applied, and how the project follows lean practices while being tailored to satisfy the SELC activities, artifacts, and reviews
   - Review SELC Tailoring Examples and schedule regular reviews of the tailoring plan with key stakeholders and approving authorities
   - Provide tools and appropriate training to team members to ensure that in-progress information is captured in accordance with the tailoring plan

9. **Obtain Agile Training, Tools, and other Resources to Enable the Project Team:** As discussed in the main body of this Instruction Manual , a key part of the PM's job is providing the training, tools, communications, and supporting infrastructure the team requires to be productive *(see Sections 4.2.2, 4.2.3, 5.2.1, 5.2.3, and 5.2.4)*.

    - Train team on Agile practices, tools, and techniques
    - Provide project environment to enhance team communication, awareness of project and work status, and promote ad hoc interaction and feedback

10. **Select, Practice, and Maintain Agile Practices:** As noted by PMs with Agile experience both internal and external to DHS, disciplined focus on the core Agile practices and philosophy are key to project success *(see Sections 2.3.1, 2.4, 3.1.2, and 3.2.2)*.

    - At the project outset, identify Agile practices and methodologies that best fit the needs of the project, team members, and environment
    - Keep iterations small and focused
    - Create a prioritized list of features and bug fixes to be delivered, into specific backlogs to be maintained throughout each iteration; make the list visible to all team members throughout the project life cycle
    - Allow the entire team to have access to information on bugs, features, and version control
    - Release features and improvements no less frequently than every four weeks

11. **Create a Backlog of Requirements (User Stories)**: The list of features and requirements for the new system, rooted in the product vision, and matured by the users, helps specify the work to be done *(see Sections 2.4.3, 3.2.1, 3.2.2, and 4.4.3).*

    - Provide team members with access to a visual representation of all user stories, including those to be addressed in future iterations, currently in progress, and completed
    - Use the ESDO Carwash suite of tools (JIRA, Atlassian Suite) or obtain alternate automated tools
    - If migrating from a waterfall development effort, upload the existing requirements into the automated tool for easier tracking and product owner review
    - For each iteration, decompose user stories into collections of specific story points to be addressed by the software code and show how the product features ultimately meet the users requirements
    - Contact the DHS EBMO Requirements Engineering Center of Excellence for assistance with converting existing requirements into user stories (http://dhsconnect.dhs.gov/org/comp/mgmt/cio/ebmo/Pages/default.aspx)

12. **Develop a Project Roadmap and Release Plans**: Releases are focused on delivering working, fully-tested increments of the system. A release plan details the work (elements of the backlog) to be accomplished in a given release. An experienced Agile Coach or peer can help the PM develop a challenging but feasible roadmap for the Agile team *(see Sections 3.2.1 and 3.2.4)*.

    - Identify user stories and features to be delivered during each iteration; update as the project matures to accurately reflect work completed and work remaining

13. **Develop Metrics and Establish Tracking Tools/Procedures**: The Agile Coach can help the PM develop metrics and tracking mechanisms that accurately reflect team progress and productivity, and meet stakeholder's needs *(see Sections 4.2.3, 4.3.2, 4.4.3, and Appendix C).*

- Identify metrics and mechanisms to best track project progress and features delivered to stakeholder

14. **Plan and Execute Iterations**: With each iteration, the team delivers an increment of tested product functionality. Each iteration concludes with a demonstration that confirms the value of the increment produced *(see Sections 2.4.4, 3.2.2, and 3.2.4)*.

    - Hold iteration planning meeting
    - Identify user stories/features to be delivered during the specific period of work
    - Carry out iteration review meeting, successfully demonstrating software developed during the iteration for the stakeholder
    - Update burndown chart (if used) to reflect updated status of backlog and overall project status (velocity and burndown rate)
    - Update burnup chart (if used) to reflected updated status on the amount of work remaining to completion

15. **Track Progress**: Recording and reporting team progress both satisfies project oversight needs and provides information the PM and Agile Coach may use to refine the project plan *(see Section 4.2.3)*.

    - Use available Agile tools and techniques to record project progress and generate the minimum number of reports necessary to satisfy governance and oversight requirements
    - If employed, ensure tools are constantly updated to reflect most accurate information on project status, bugs, and remaining backlog to ensure that information is made available to entire team

16. **Review Progress; Learn and Apply Improvements**: At the end of each iteration and release, the PM and team reviews progress using Agile tools and processes such as burndown charts, burnup charts, and retrospectives to discuss what went well, what didn't go well, and to identify actions to correct problems. The PM, Agile Coach, Product Owner, and other stakeholders also use this information to assess progress against the overall project plan, and to make adjustments as necessary.

    - Carry out retrospective meeting with team to allow for feedback and to immediately incorporate learning into future iterations
    - Update project plan to reflect most recent status and information on project status (including remaining backlog, upcoming SELC artifacts and activities, and project velocity)

17. **Start delivering immediately**! If the PM waits to do all the things listed above, then delivery may not start until a long time into the project. Emphasize moving to delivery as soon as possible.

Note that these actions, although numbered, are not necessarily executed sequentially. Most of them, in fact, are done iteratively, throughout the project lifecycle.