DRAFT

What All Program Office Staff Should Know about Lean/Agile

For F-35 JSE

Suzanne Miller
Crisanne Nolan

Software Engineering Institute
Carnegie Mellon University
Pittsburgh, PA 15213

Distribution Statement A] Approved for public release and unlimited distribution

DRAFT

Copyright 2021 Carnegie Mellon University.

This material is based upon work funded and supported by the Department of Defense under Contract No. FA8702-15-D-0002 with Carnegie Mellon University for the operation of the Software Engineering Institute, a federally funded research and development center.

The view, opinions, and/or findings contained in this material are those of the author(s) and should not be construed as an official Government position, policy, or decision, unless designated by other documentation.

References herein to any specific commercial product, process, or service by trade name, trade mark, manufacturer, or otherwise, does not necessarily constitute or imply its endorsement, recommendation, or favoring by Carnegie Mellon University or its Software Engineering Institute.

NO WARRANTY. THIS CARNEGIE MELLON UNIVERSITY AND SOFTWARE ENGINEERING INSTITUTE MATERIAL IS FURNISHED ON AN "AS-IS" BASIS. CARNEGIE MELLON UNIVERSITY MAKES NO WARRANTIES OF ANY KIND, EITHER EXPRESSED OR IMPLIED, AS TO ANY MATTER INCLUDING, BUT NOT LIMITED TO, WARRANTY OF FITNESS FOR PURPOSE OR MERCHANTABILITY, EXCLUSIVITY, OR RESULTS OBTAINED FROM USE OF THE MATERIAL. CARNEGIE MELLON UNIVERSITY DOES NOT MAKE ANY WARRANTY OF ANY KIND WITH RESPECT TO FREEDOM FROM PATENT, TRADEMARK, OR COPYRIGHT INFRINGEMENT.

[DISTRIBUTION STATEMENT A] This material has been approved for public release and unlimited distribution. Please see Copyright notice for non-US Government use and distribution.

This material is distributed by the Software Engineering Institute (SEI) only to course attendees for their own individual study.

Except for any U.S. government purposes described herein, this material SHALL NOT be reproduced or used in any other manner without requesting formal permission from the Software Engineering Institute at permission@sei.cmu.edu.

Although the rights granted by contract do not require course attendance to use this material for U.S. Government purposes, the SEI recommends attendance to ensure proper understanding.

Carnegie Mellon® is registered in the U.S. Patent and Trademark Office by Carnegie Mellon University.

DM21-0011

Agenda

DRAFT

F-35 and ODIN Context for Lean/Agile

“Agile” Acquisition – New Pathways

Applying New Programmatic Constructs to ODIN

Agile Roles for ODIN

© 2020 Carnegie Mellon University

[[Distribution Statement A] Approved for public release and unlimited distribution.

Learning Objectives

At the end of this lesson, you should be able to:

- Describe ODIN's Context for Leveraging Agile, Lean and DevSecOps for Program Success
- Describe ODIN's Basic Approach to Agile, Lean, and DevSecOps Adoption

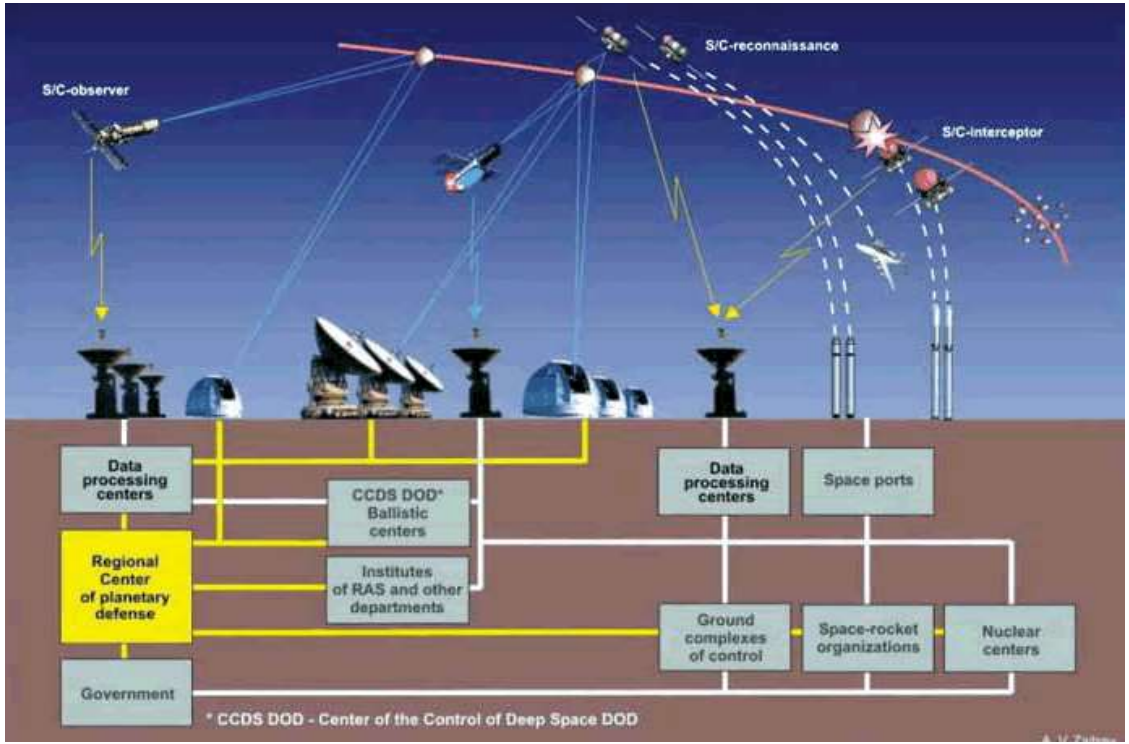
© 2020 Carnegie Mellon University

[[Distribution Statement A] Approved for public release and unlimited distribution.

ODIN System Context for Agile

DRAFT

ODIN is part of F-35 -- a large, complex cyber-physical system. Our implementation of any approach, including Agile, *must account for our context*



© 2020 Carnegie Mellon University

[[Distribution Statement A] Approved for public release and unlimited distribution.

Why Agile (and Lean and DevSecOps) at F-35?

DRAFT

Congressional mandate through NDAA reinforces need for fast feedback all along the development path (Agile)

To assure time-certain delivery, parallelization of certification and development activities is needed (DevSecOps)

Longevity of system in a rapidly evolving threat space (Agile, Lean, DevSecOps)

© 2020 Carnegie Mellon University

[[Distribution Statement A] Approved for public release and unlimited distribution.

Agile Mindset Goes Beyond Software at F-35

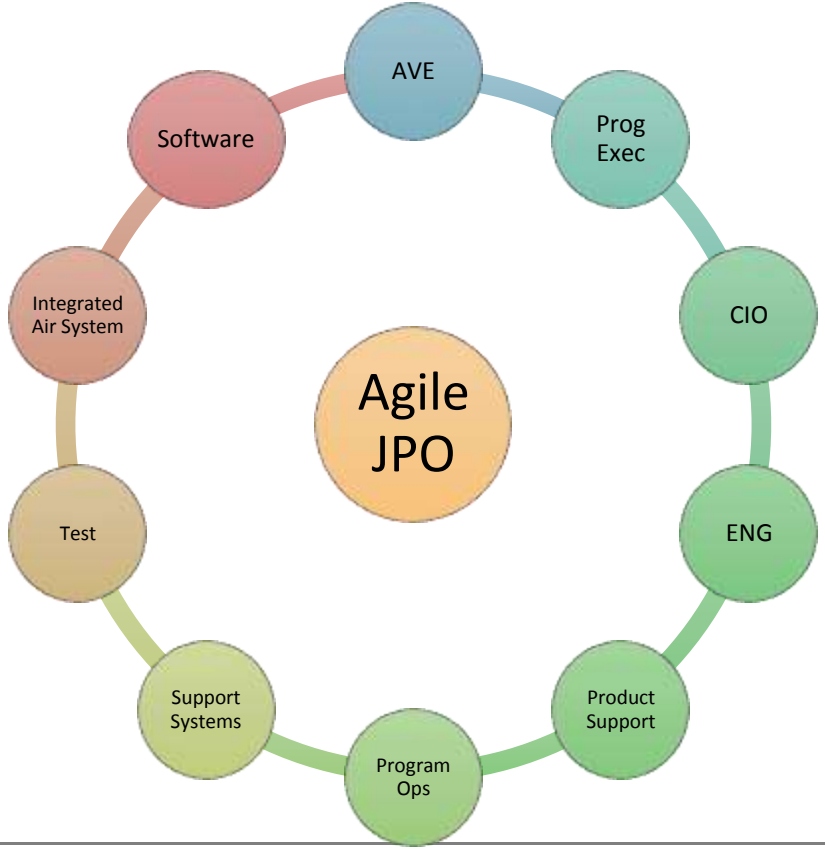
DRAFT

Agile mindsets are being adopted by successful teams across all major lines of business.

An Agile mindset is centered around delivering value by:

- embracing change
- collaborating with all stakeholders
- developing a demonstrable product iteratively
- seeking learning opportunities and pivoting accordingly

Regardless of your product you can reap benefits from an Agile mindset.

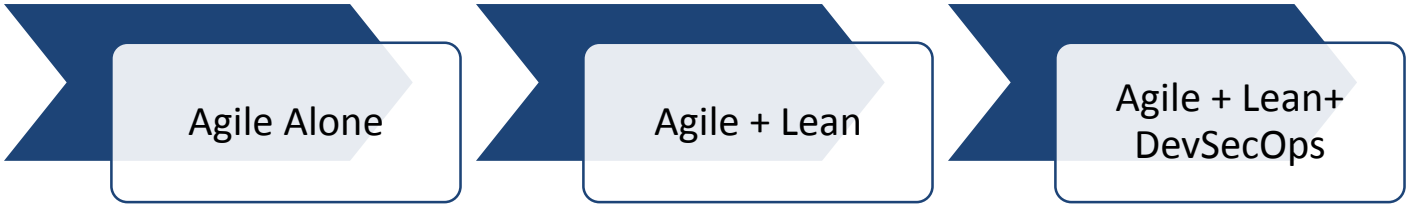


© 2020 Carnegie Mellon University

[[Distribution Statement A] Approved for public release and unlimited distribution.

Why Agile *AND* Lean *AND* DevSecOps at F35?

DRAFT



- Tends to focus just on small software teams
  - F35 context way bigger, more complex
- Tends to assume that direct delivery to customers is feasible

- Adds typical hardware, system, and business/ management teams
- Adds principles and practices that reflect the larger complex system context inherent in F35
- Adds consideration of stakeholders like DT/OT (dev test and opn'l test) and certification

- Adds fast feedback technology infrastructure for continuous architecture, continuous integration, continuous deployment
- Particularly adds to Agile teams' efficiency and effectiveness in execution

© 2020 Carnegie Mellon University

[[Distribution Statement A] Approved for public release and unlimited distribution.

Agile in the Larger DoD Context

DRAFT

DoD continues to track anecdotal success and failures of programs using Agile

2019 Defense Innovation Board Software Acquisition Practices Study (referred to as DIB SWAP Study) frames challenges in the eco-system that lead to Agile/Lean, but also slow down their adoption

Agile/Lean Successes:

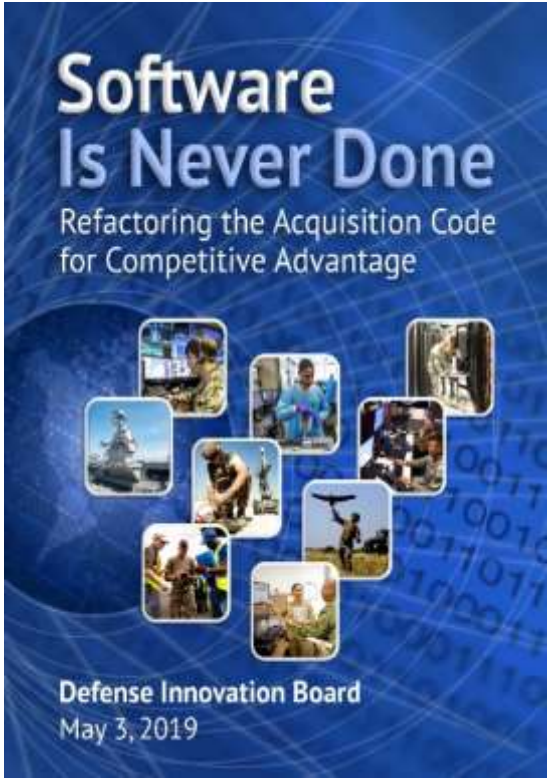
- 2017: JIDO (Joint Improvised-Threat Defense Orgn) reduced time for critical software delivery capability from 6 months to 12 days with a 9X increase in deployment frequency and a 90% reduction in operating costs
- 2018-2019: 4 of 7 programs piloting Agile approaches delivered working capabilities to USERS within three months
  - The other 3 of 7 delivered working capabilities to INTEGRATION TESTING for larger capabilities within three months
- 2019: Space Force Space C2 program developed and fielded two applications within six months of initiation; one of these documented 4 hours per shift time savings for over 1000 users

© 2020 Carnegie Mellon University

[[Distribution Statement A] Approved for public release and unlimited distribution.

2019 DIB SWAP Study

DRAFT



Strong influence on the 2019 and 2020 National Defense Authorization Acts

Also home of the “Agile BS” Appendix many have seen

Main lines of effort:

- **Congress and OSD: Refactor statutes, regulations, and processes for software**, providing increased insight to reduce the risk of slow, costly, and overgrown programs, and enabling rapid deployment and continuous improvement of software to the field.
- **OSD and the Services: Create and maintain cross-program/cross-Service digital infrastructure** that enables rapid deployment, scaling, testing, and optimization of software as an enduring capability; manage them using modern development methods; and eliminate the existing hardware-centric regulations and other barriers.
- **Services and OSD: Create new paths for digital talent (especially *internal* talent)** by establishing software development as a high-visibility, high-priority career track and increasing the level of understanding of modern software within the acquisition workforce.
- **DoD and industry: Change the practice of how software is procured and developed** by adopting modern software development approaches.

Source: https://innovation.defense.gov/software/

© 2020 Carnegie Mellon University

[Distribution Statement A] Approved for public release and unlimited distribution.

A Few Other Notable Recent Events/Links-1

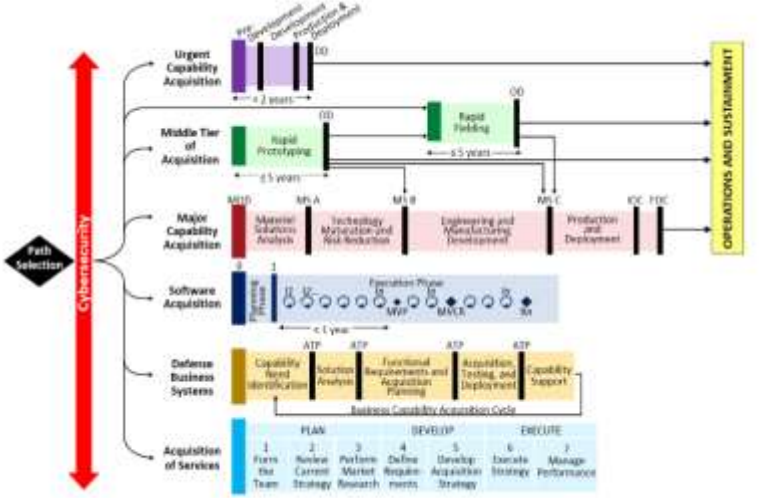
DRAFT

Adaptive Acquisition Pathways: New ways of acquiring software-dominant systems (some of the guidance can also be applied to cyberphysical systems that are software-reliant)

https://aaf.dau.edu/

Section 873/874 Agile Acquisition Pilots: a group of small and large acquisition piloting Agile/Lean approaches to software development in different settings. Lessons learned document published 2020:

https://www.dau.edu/cop/it/DAU%20Sponsored%20Documents/AgilePilotsGuidebook%20V1.0%2027Feb20.pdf



© 2020 Carnegie Mellon University

[[Distribution Statement A] Approved for public release and unlimited distribution.

Respond at: www.pollev.com/mainsummit799

What one word or phrase summarizes Odin's reasons for incorporating Agile, Lean, and DevSecOps into our work?

© 2020 Carnegie Mellon University
[Distribution Statement A] Approved for public release and unlimited distribution.

Working Definition of Agile

DRAFT

Agile An *iterative* and *incremental* (evolutionary) approach to software development which is performed in a *highly collaborative manner* by *self-organizing teams* within an *effective governance framework* with *"just enough" ceremony* that produces *high quality software* in a *cost effective and timely* manner which *meets the changing needs of its stakeholders*. [Ambler 2013]

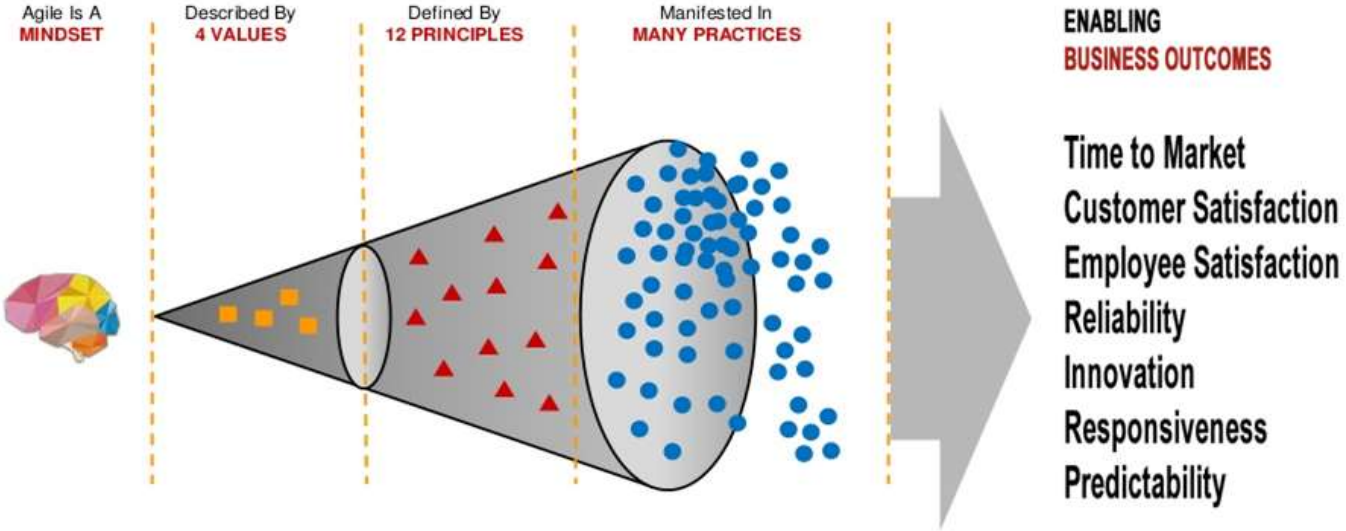
[Ambler 2013] Ambler, Scott. *Disciplined Agile Software Development: Definition*.
http://www.agilemodeling.com/essays/agileSoftwareDevelopment.htm

© 2020 Carnegie Mellon University

[[Distribution Statement A] Approved for public release and unlimited distribution.

What is Agile?

DRAFT



Source: https://www.slideshare.net/MichaelTarnowski/agile-mindset-for-executives

Implementing the practices, tools and processes **without** the Agile mindset, values, and principles of the Agile Manifesto

Is NOT Agile!

It isn't enough to adopt the practices of a successful team. You must adopt attitudes and a mindset for making decisions to adopt practices that will lead to your success.

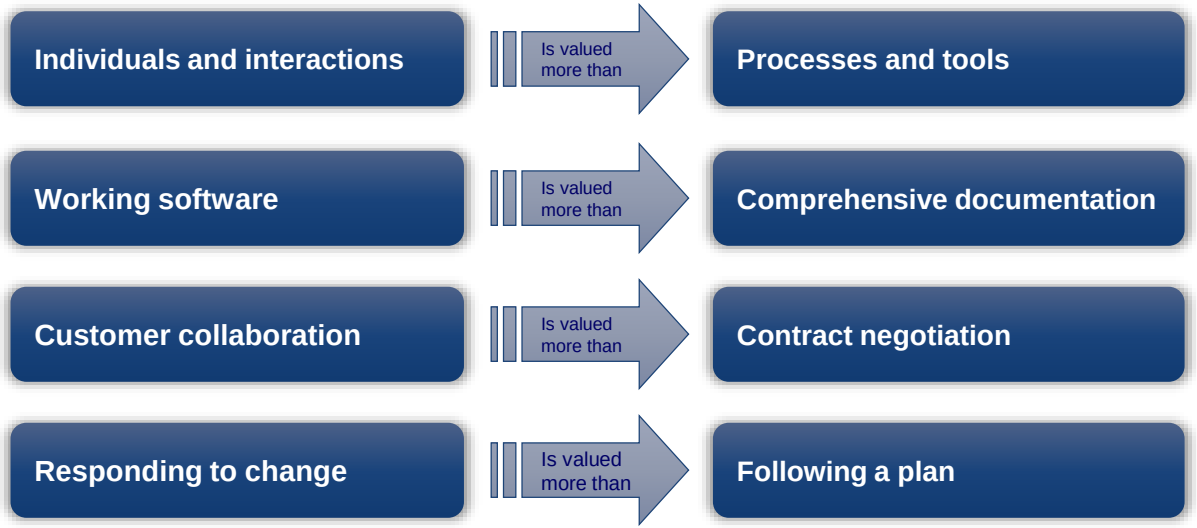
© 2020 Carnegie Mellon University

[[Distribution Statement A] Approved for public release and unlimited distribution.

Agile Values

DRAFT

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



➢ **Zoom Activity: Which side do you think will benefit your customer more?**

Source: http://agilemanifesto.org

Copyright 2020 Northrop Grumman Systems Corporation

© 2020 Carnegie Mellon University

[[Distribution Statement A] Approved for public release and unlimited distribution.

Agile Principles

DRAFT

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

Agile Principles: http://agilemanifesto.org

➢ **Zoom Activity: Which principle do you think will benefit your customer?**

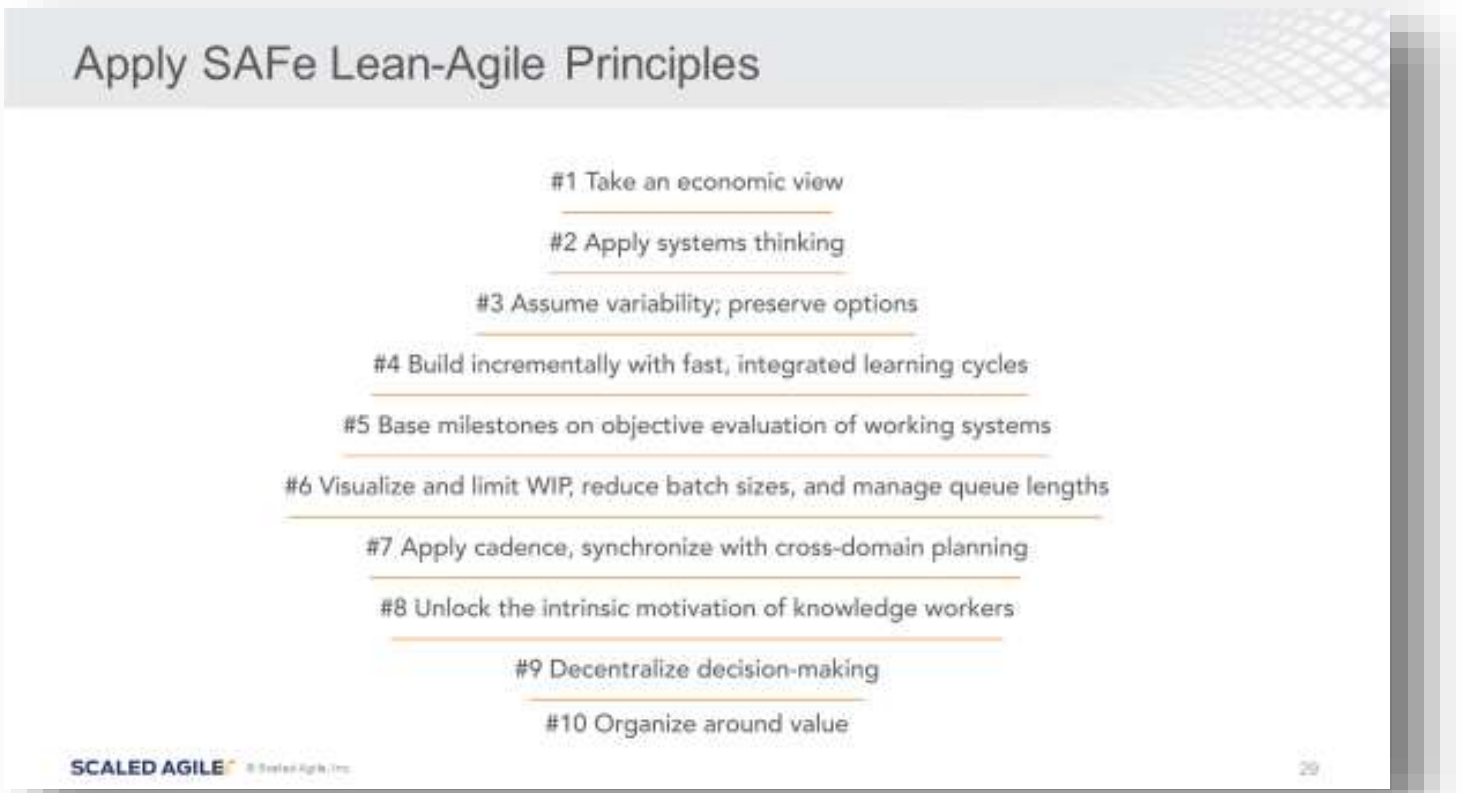
Copyright 2020 Northrop Grumman Systems Corporation

© 2020 Carnegie Mellon University

[[Distribution Statement A] Approved for public release and unlimited distribution.

Don’t Forget the Lean Principles – For the Entire (Govt + Contractor + Stakeholder) Enterprise

DRAFT



© 2020 Carnegie Mellon University

[Distribution Statement A] Approved for public release and unlimited distribution.

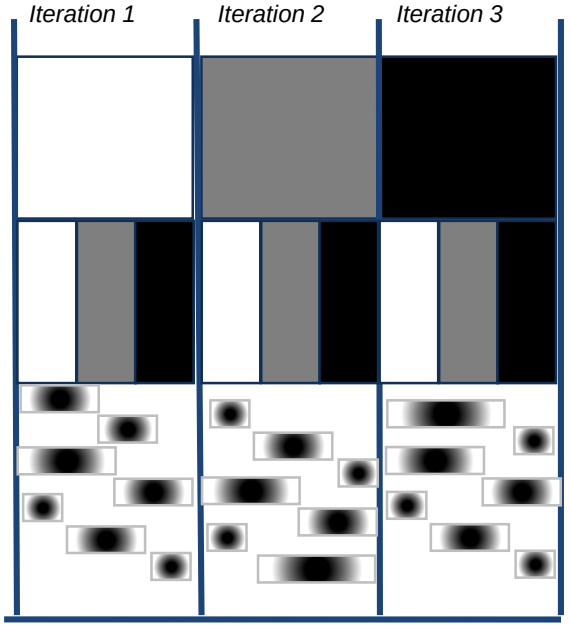
Taking an Iterative Approach

DRAFT

Single batch – one process step per iteration

Multiple batches - complete each batch at the end of an iteration; siloed process steps within each iteration

Multiple batches - decompose each batch into small packages, with multiple start-to-finish cycles in each iteration



© 2020 Carnegie Mellon University

[[Distribution Statement A] Approved for public release and unlimited distribution.

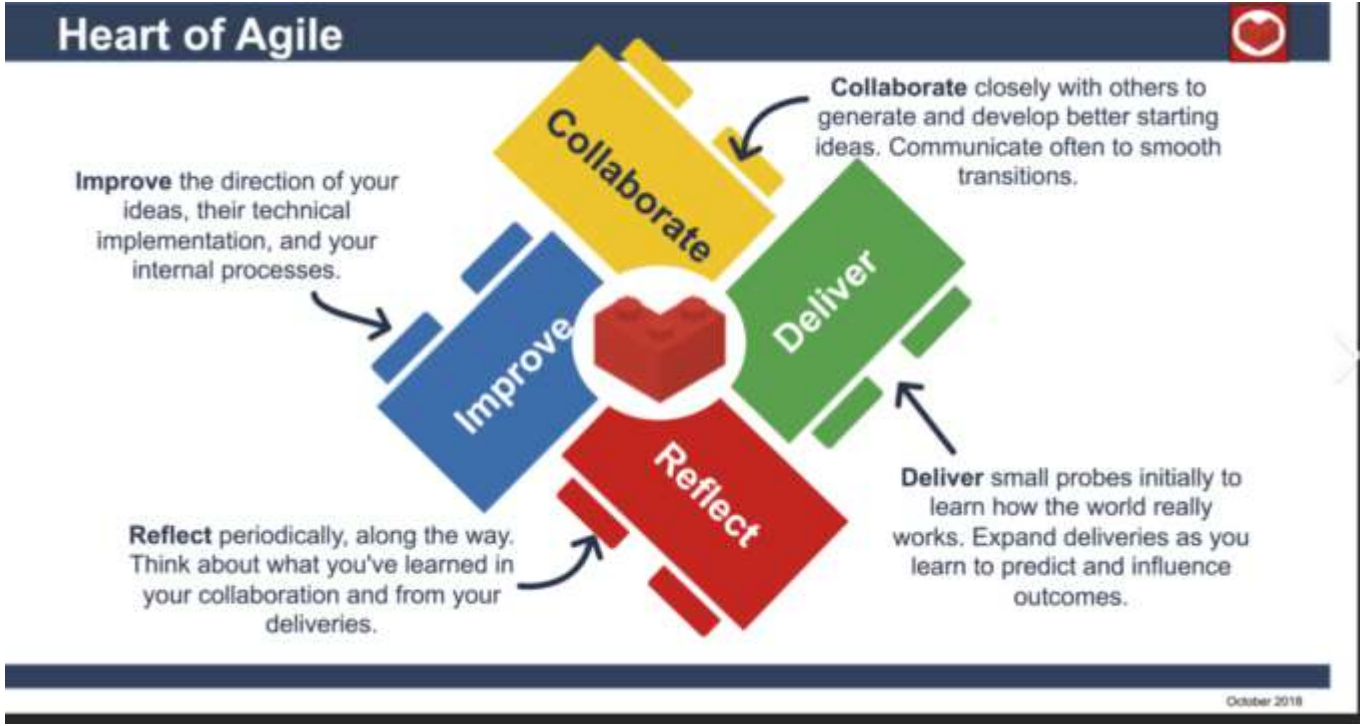
Start with Teams Using Common Concepts But Allow Methods That Suit Their Context

DRAFT

All team-focused Agile approaches have commonalities and differences

Focus on the commonalities whenever possible and choose the best way forward for your team that suits your context

Cockburn's "Heart of Agile" provides a way to look at commonalities across team methods



Source: heartofagile.com

© 2020 Carnegie Mellon University

[[Distribution Statement A] Approved for public release and unlimited distribution.

Agile Team Method Commonalities: Scrum, XP, Kanban Are All Used in F-35

DRAFT

- Collaborate across functions and stakeholders
- Communicate often to smooth handoffs
- Build trust-based relationships
- Use common tools, where feasible
- Establish & evolve prioritized backlogs of work items

- Learn from each iteration's products
- Enable fast feedback

- Work in small batches
- Build quality in
- Design modular work items
- Divide work into short iterations
- Deliver incrementally

- Act on learning, don't just capture it
- Improve direction of ideas, technical implementation, and internal processes

© 2020 Carnegie Mellon University

[[Distribution Statement A] Approved for public release and unlimited distribution.

All Common Agile Team Methods are Based on PDCA Cycle

DRAFT

Plan - a task, change or test, aimed at improvement.

- Analyze what you intend to improve - choose areas with highest rate of return

Do - Carry out the change or test (preferably on a small scale).

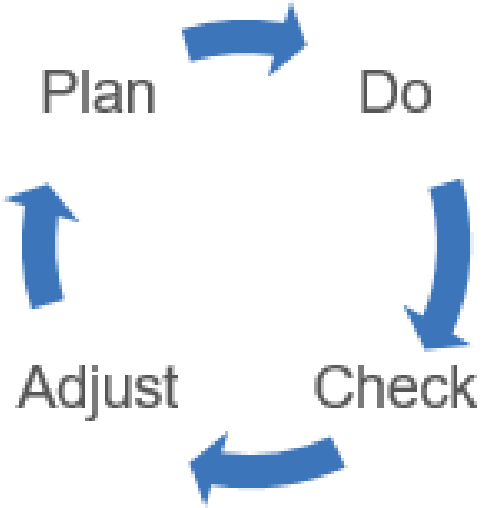
- Implement the change you decided on in the plan phase.

Check - the results. What was learned? What went wrong? (We have received empirical data! We are NOT guessing!)

- Measure / monitor the level of improvement.

Adjust – Make the change based on the empirical data.

- Adopt the change, abandon it, or run through the cycle again.



Plan – Do – Check – Adjust (PDCA) cycle is inherent in all of our work.

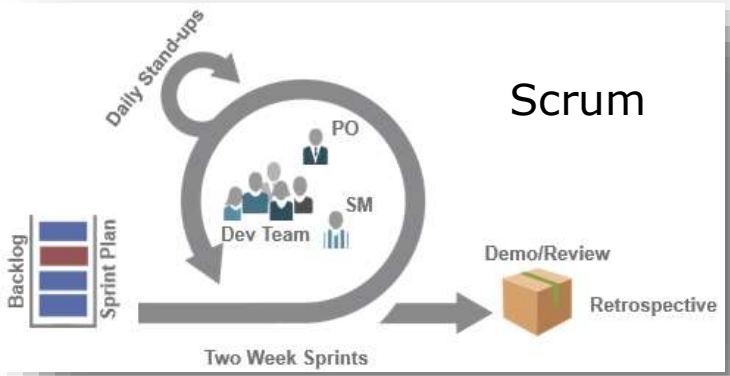
© 2020 Carnegie Mellon University

[[Distribution Statement A] Approved for public release and unlimited distribution.

Choose One or More Methods to Fit Your Team Context

DRAFT

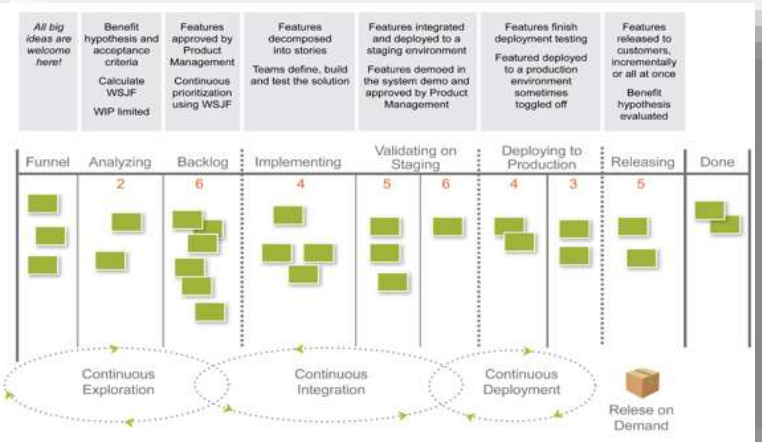
- Most common product dev approach
- Not just sw dev



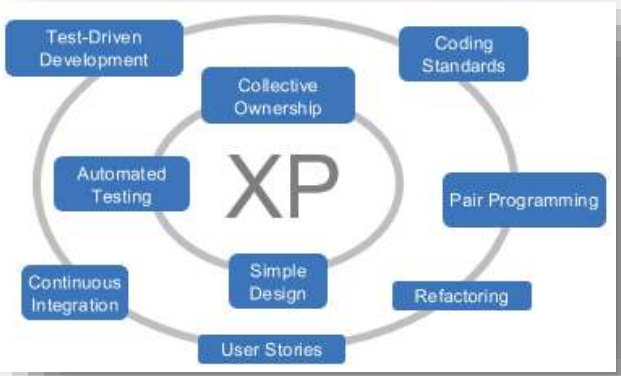
Scrum

- Applies to almost any type of work
- Commonly used for teams not focused on products per se (eg teams who do lots of review/ coordinate tasks)

Kanban



eXtreme Programming



- Most software-specific approach
- Strong focus on technical practices

© 2020 Carnegie Mellon University
[[Distribution Statement A] Approved for public release and unlimited distribution.

Common Tools Will be Used Across Different Team Types at F35

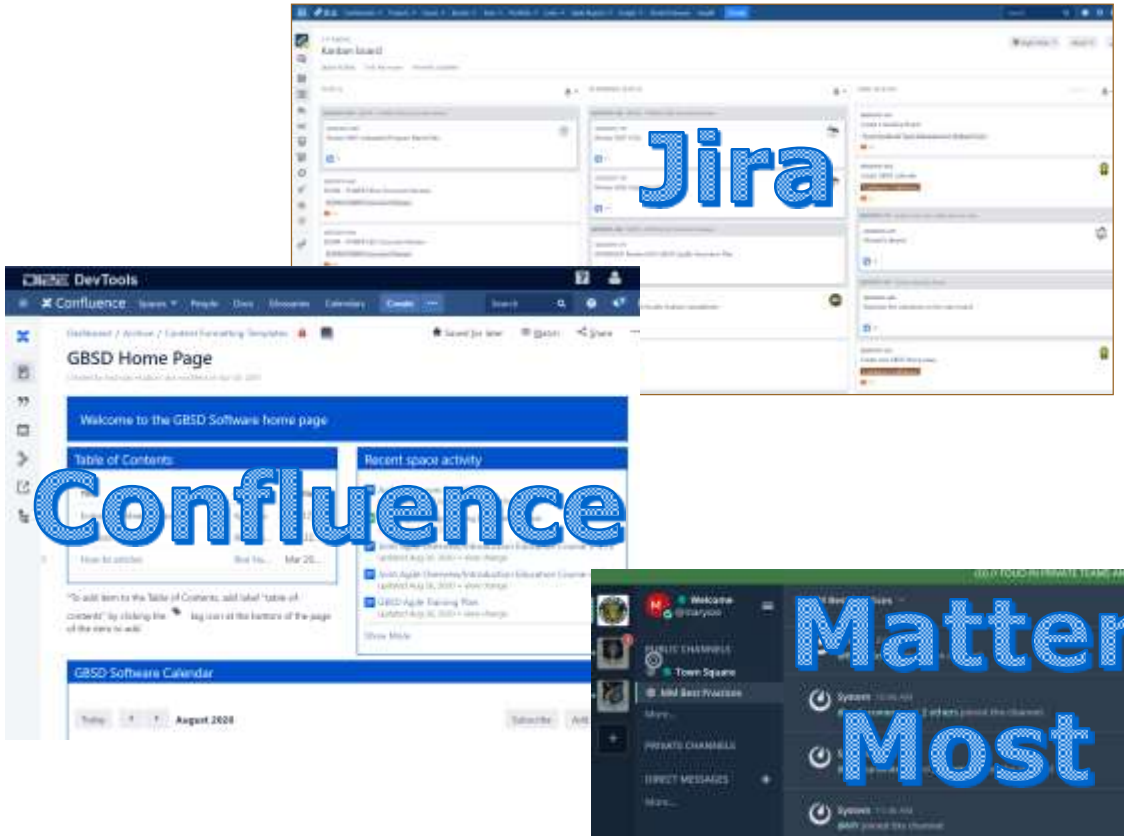
DRAFT

Tool focused on workflow management: JIRA

Tool focused on artifact (document) evolution and sharing: CONFLUENCE

Tool focused on inter/intra-team collaboration: MATTERMOST

Development, integration, and test teams of evolving software and hardware products will have lots of other tools specific to their context



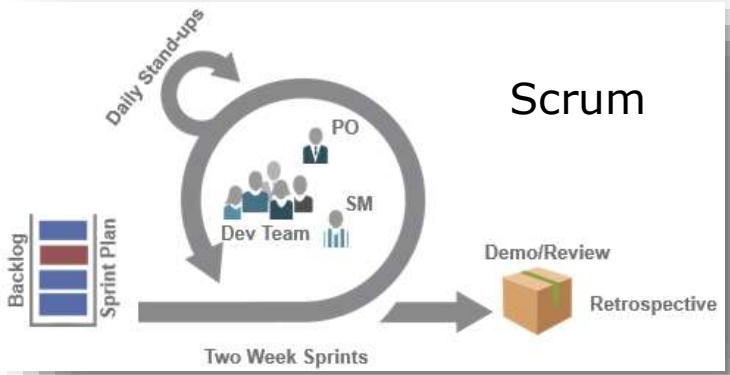
© 2020 Carnegie Mellon University

[[Distribution Statement A] Approved for public release and unlimited distribution.

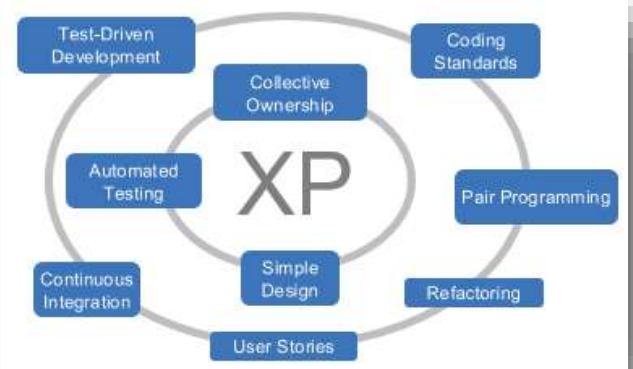
A Quick Tour Through the Three Common Team Methods at F-35

DRAFT

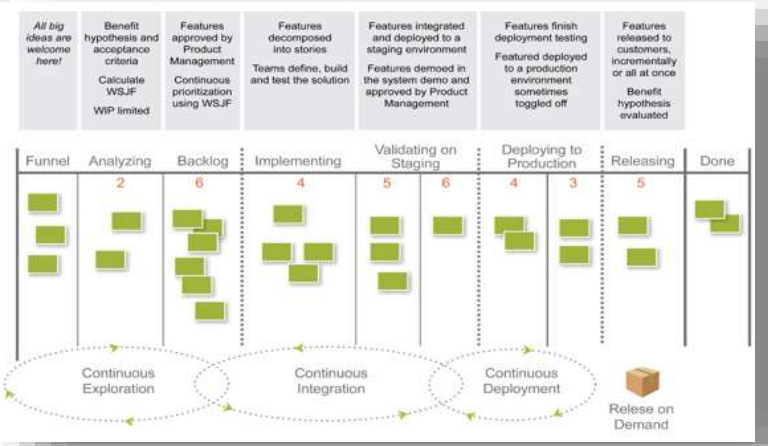
Scrum



eXtreme Programming



Kanban



© 2020 Carnegie Mellon University
[Distribution Statement A] Approved for public release and unlimited distribution.

Scrum's Simple Rules

DRAFT

Scrum is founded on empirical process control theory, or **empiricism**

Empiricism asserts that knowledge comes from real data and experience; and that making decisions is based on what is known.

Scrum employs an iterative, incremental approach **to optimize predictability and control risk (nukes!)**

Optimal Team size is 3-9 members, ("2 pizza rule" (4.6 optimum)) not counting SM and PO

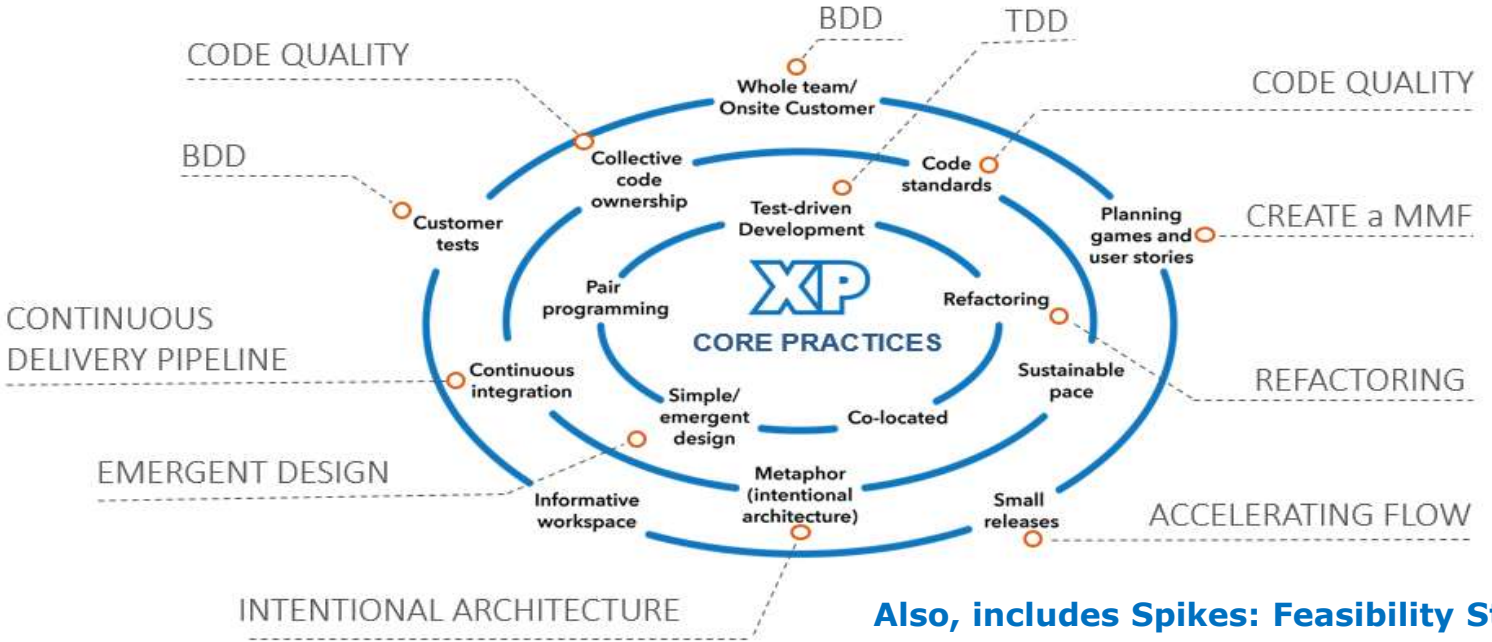
Scrum Teams are cross-functional, self-organizing and self-managing

© 2020 Carnegie Mellon University
[[Distribution Statement A] Approved for public release and unlimited distribution.

XP (eXtreme Programming) Overview

DRAFT

Preferred SW Agile Methodology for several DoD software contexts: Kessel Run, Platform One, SkiCAMP, SoniKube, SpaceCAMP, etc.



Also, includes Spikes: Feasibility Studies

© 2020 Carnegie Mellon University

[[Distribution Statement A] Approved for public release and unlimited distribution.

Kanban

DRAFT

Although Kanban can be used on its own, it is frequently used within the context of Scrum & XP when the team is a product development team

- In that context, Kanban is frequently used in Sprint execution to visualize the work the team is currently committed to

Kanban is the most frequently used team method for non-sw development teams (eg systems engineering, certification, contracting)

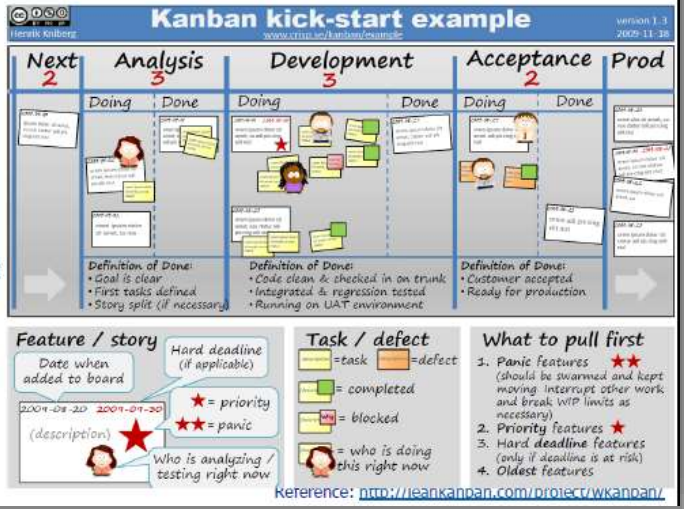
Quick Introduction to Kanban Method

Three Core Principles

- Start with what you do now
- Agree to pursue incremental evolutionary change
- Initially, respect current roles, job titles, and responsibilities

Five Core Practices

- Visualize workflow
- Limit Work In Progress (WIP)
- Manage Flow
- Make Process Policies Explicit
- Improve Collaboratively



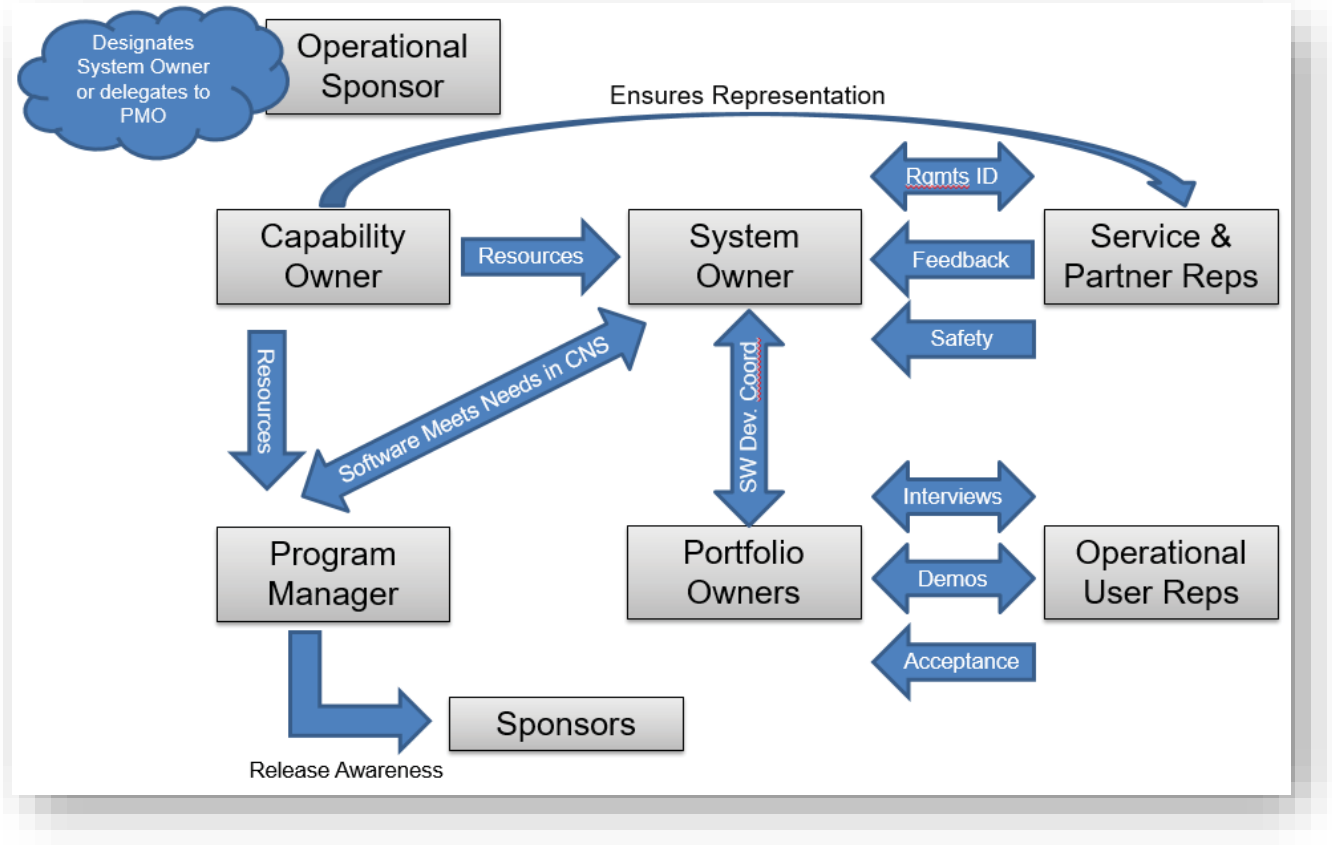
.78

Reference: http://leankanban.com/project/wkanban/

Copyright 2020 Northrop Grumman Systems Corporation

ODIN User Agreement Role Relationships

DRAFT



© 2020 Carnegie Mellon University

[[Distribution Statement A] Approved for public release and unlimited distribution.

ODIN User Agreement Key Roles – 1

DRAFT

Operational Sponsor

- Designates a representative to be the System Owner for the specified capabilities or delegates these roles to the Maintenance Systems PMO.
- Approves future versions of the User Agreement.

Capability Owner

- Works with Program Manager and System Owner to ensure the ODIN system is developed in concert with the capabilities defined in the ODIN CNS.
- Ensures one representative (backup can be identified) is designated from each of the services and partners, who can meet the commitments specified for Service and Partner Representatives.
- Ensures there are sufficient resources (supplier location, supporting personnel and demonstration coordination, access) allocated to support user engagements.

ODIN Program Manager (PM)

- Works with the System Owner to ensure software development priorities address user priority needs and capabilities.
- Works with other JPO organizations (e.g., Program Management Offices (PMOs) and Hybrid Product Support Integration (HPSI)) to account for F-35 integration and dependencies, and works with the System Owner to account for needs and capabilities.
- Works with System Owner and sponsors to ensure user communities are aware of significant capability releases in a timely fashion.

© 2020 Carnegie Mellon University

[[Distribution Statement A] Approved for public release and unlimited distribution.

ODIN User Agreement Key Roles – 2

DRAFT

System Owner

- The System Owner understands the user organization, the capability being developed, and can advocate for the product. This role is responsible for carrying out the long-term vision of the project and guiding its progress. The System Owner:
- Has authority, within the confines of the approved CNS, to make trades and decisions independently and without additional approval from superiors.
- Participates with the Service and Partner Representatives in requirements identification and prioritization forums and ensures their capability needs and priorities are addressed.
- Provides guidance and expertise about the users, their operational needs, and practical considerations for meeting those needs.
- Identifies unit leadership inside the user organization and advocates for their support.
- Works with the Program Manager and Portfolio Owners to ensure that software development addresses the users' priority needs and validates that content in the CNS is clear and realistic.

Portfolio Owners

- Has the authority to coordinate and execute future engagement, user onboarding, feedback interviews, and testing with the user community.
- Ensures continual interactions occur between product teams and end users to address users' priority needs and capabilities are delivered in timely iterations.
- Balances CNS requirements and users' input to define Minimum Viable Products (MVPs) and Minimum Viable Capability Releases (MVCRs) for user acceptance testing.

© 2020 Carnegie Mellon University

[[Distribution Statement A] Approved for public release and unlimited distribution.

ODIN User Agreement Key Roles – 3

DRAFT

Service and Partner Representatives

- Each Service and Partner nation provides one representative (backup can be identified).
- Empowered to provide feedback directly to the System Owner and Portfolio Owner.
- Participates with the System Owner and Portfolio Owners in requirements identification and prioritization forums and meetings (e.g., ODIN Quarterly Stakeholder Workshops).
- Identify service/partner needs and concerns to include operational and safety impacts to the System Owner with an aim to consolidate varying feedback within their service into a single input, where possible.
- Coordinates with their user community to set expectations about the ODIN roadmap, user base location, and the requirements of user representatives during user-centered design.

Operational User Representatives

The leadership at each location and organization will help manage priorities by appointing a senior operational user representative to be the Portfolio Owner's main contact. The senior operational user representative should:

- Be able to identify further operational user representatives who are reflective of their respective communities (e.g. maintenance, logistics).
- Has the authority to initiate plans for testing and exercises of new capabilities.

These representatives are end users of ODIN and consumers of ODIN data. They provide detailed input on constraints and real world impacts, since they know the mission best. These representatives are expected to support the following:

- User interviews, product demos, and product testing to guide software development via user-centered design.
- User acceptance testing and exercises of new capabilities.
- Roll out of tested capability and continued user feedback to product managers.
- Integrated system demonstrations in preparation for deployment decisions.

© 2020 Carnegie Mellon University

[[Distribution Statement A] Approved for public release and unlimited distribution.

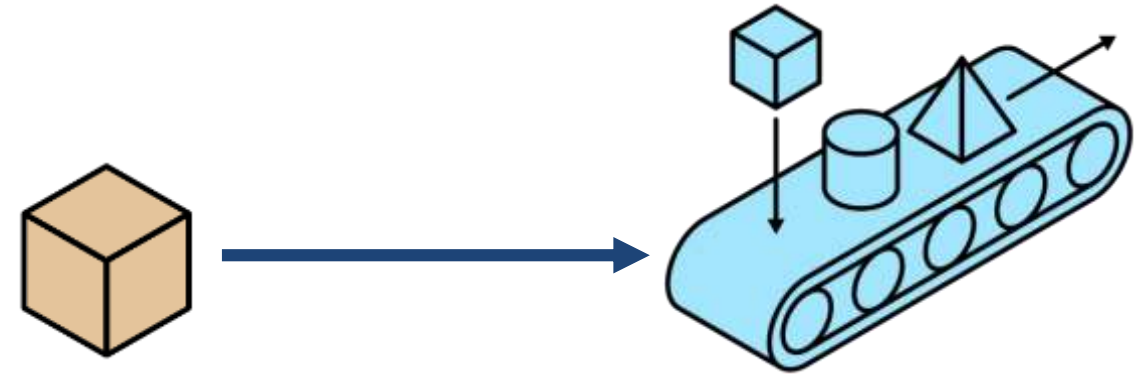
Programmatic Constructs

© 2020 Carnegie Mellon University

[[Distribution Statement A] Approved for public release and unlimited distribution.

Agile Represents a Major Requirements Transition – Different Acquisition Objectives

DRAFT



Buying a Box

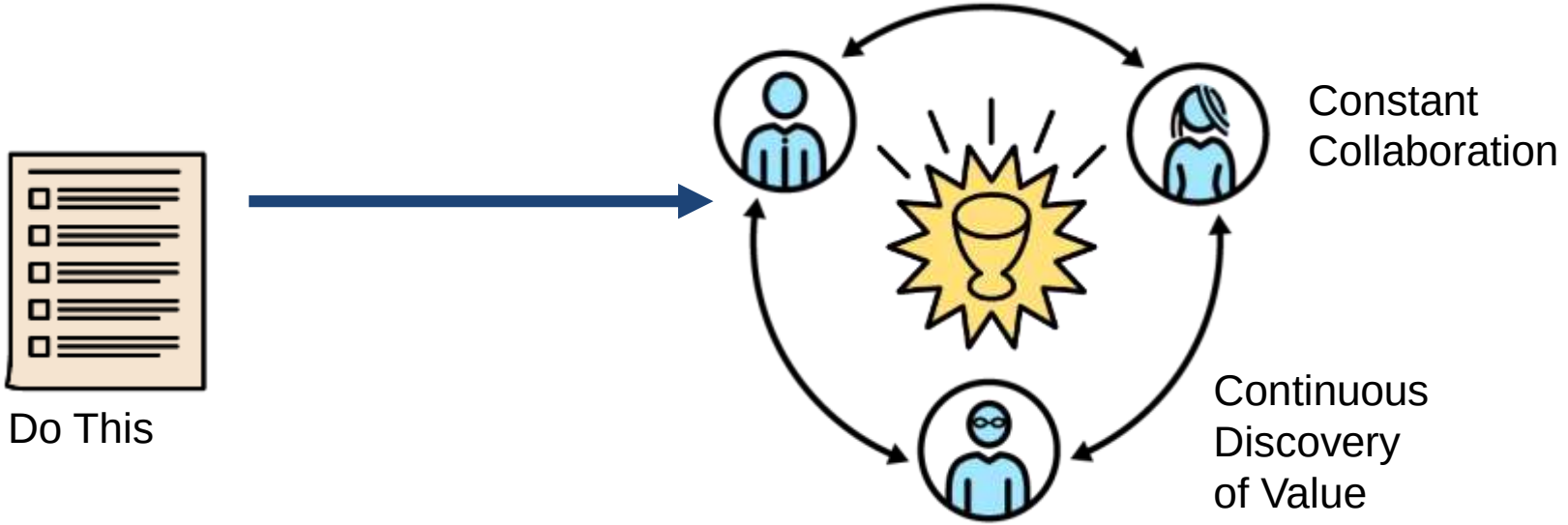
Buying an ongoing delivery stream

© 2020 Carnegie Mellon University

[[Distribution Statement A] Approved for public release and unlimited distribution.

Agile Represents a Major Requirements Transition – Different Vendor Interactions

DRAFT

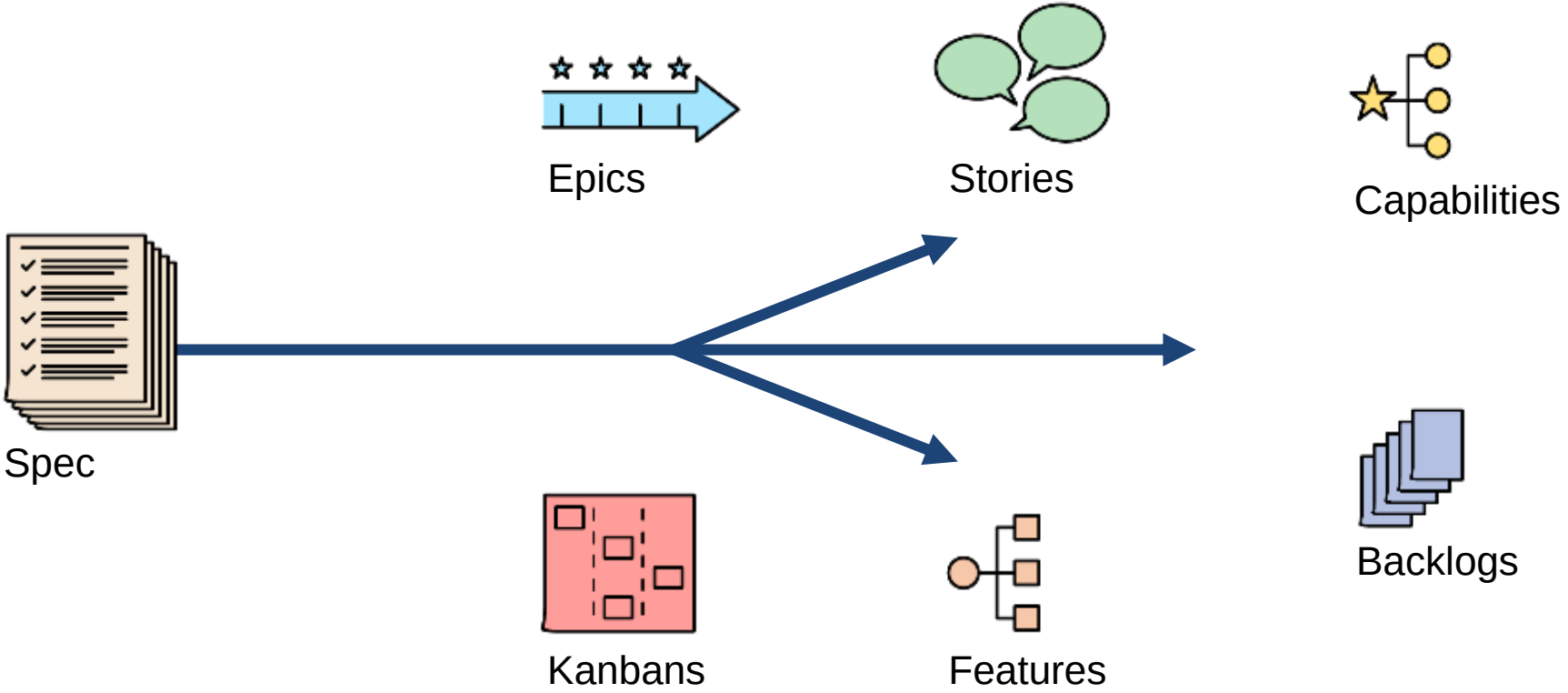


© 2020 Carnegie Mellon University

[Distribution Statement A] Approved for public release and unlimited distribution.

Agile Represents a Major Requirements Transition – Different Artifacts

DRAFT



© 2020 Carnegie Mellon University

[Distribution Statement A] Approved for public release and unlimited distribution.

Agile Represents a Major Requirements Transition – Different Practices for Requirements Management

DRAFT

Date-based Milestones

Increment-Based Demos

Definition of Done

Team Increment

System Increment

Solution Increment

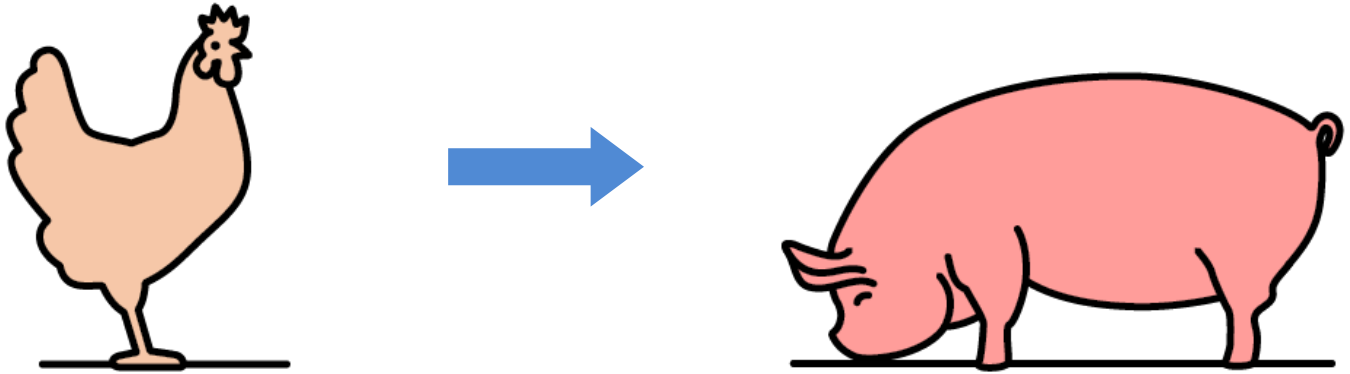
Release

© 2020 Carnegie Mellon University

[[Distribution Statement A] Approved for public release and unlimited distribution.

Agile Represents a Major Requirements Transition – Different Overall Risk Profile

DRAFT



© 2020 Carnegie Mellon University

[[Distribution Statement A] Approved for public release and unlimited distribution.

Agile Motto for the Backlog Process

DRAFT

© 2020 Carnegie Mellon University

[[Distribution Statement A] Approved for public release and unlimited distribution.

Characteristics of Requirements Models for Large-Scale Agile Systems

DRAFT

Layered - Requirements are divided into layers representing different levels of abstraction / scope

Cascading – Each layer of the Requirements Model feeds the layer below

Categorized – Two general types of requirements expressions – Business/Mission (user-focused) and Enablers (system/infrastructure-focused)

Prioritized—part of what makes scaling feasible is that we're not dealing with 100% of the details of all the requirements at the same time

Adapted from https://www.scaledagileframework.com/epic

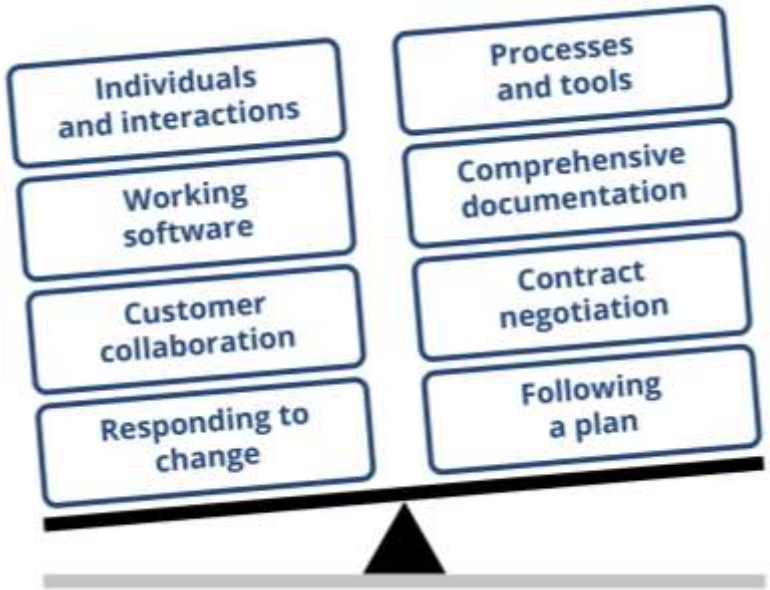
© 2020 Carnegie Mellon University

[[Distribution Statement A] Approved for public release and unlimited distribution.

Reorienting the Manifesto for Agile *Software Development* Toward *System Acquisition*

DRAFT

Through this work we have come to value:



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

https://agilemanifesto.org/history.html

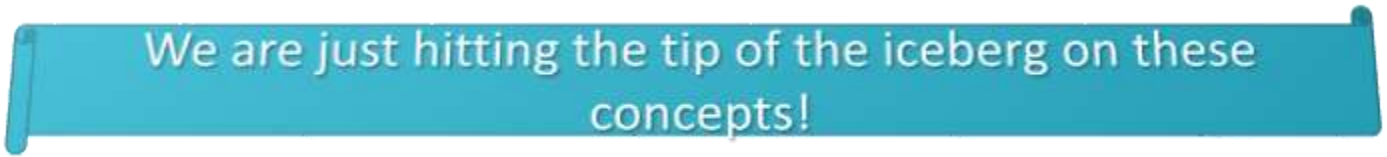
© 2020 Carnegie Mellon University

[[Distribution Statement A] Approved for public release and unlimited distribution.

The Oversight/Insight Challenge for the Lean Program Office

DRAFT

How can the Program Office gain Insight into Contractor Expression Of the Agile & Lean Principles?



© 2020 Carnegie Mellon University

[[Distribution Statement A] Approved for public release and unlimited distribution.

Bottom Line Up Front

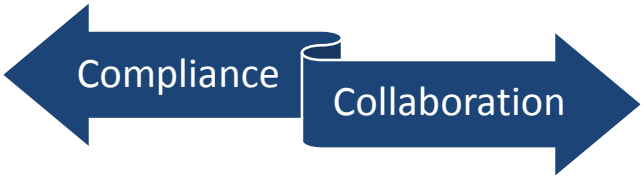
DRAFT

OVERSIGHT is an element of what makes the DoD Acquisition Ecosystem Work

- Oversight mechanisms established by program management determine:
  - Nature of information made available
  - Frequency of communication
  - Urgency/importance
- Well-established procedures & templates convey oversight requirements
  - Recent developments like Adaptive Acquisition Pathways change some of those requirements

INSIGHT is a necessary enabler to effective oversight

- Well-established CDRLs and DIDs may not always the best source of insight
  - Aversion to all off-nominal conditions
  - Conformance to plan becomes the goal
- Agile development settings promote transparency and ongoing insight
  - Available mechanisms, however, require proactive participation from the acquirer to be effective



© 2020 Carnegie Mellon University

[[Distribution Statement A] Approved for public release and unlimited distribution.

Batch Size

DRAFT

Typical Large Batch Realities:

- “Nothing is done until everything is done”
- More Work in Progress is good
- 100% utilization of resources is a goal
- Optimistic reporting of progress in order to “keep the program sold”
- Large scope integration events identify defect levels that strain resources
  - Increases number of potential defects that affect multiple areas of the system
  - Reduces confidence in system robustness
  - Harder for engineers to find sources of defects
- Tendency toward “test quality in”

Aspirations for Small Batch:

- We can learn from even small pieces being implemented/done
- “Stop starting, start finishing”
- Work in Progress is limited to enhance flow through the system
- 100% utilization of resources is recognized as limiting flow, flexibility, and work accomplishment
- Short time between when a defect is found and when it was created
  - Root cause analysis easier with current work, rather than work done in the past
- LOTS of integration happening across entire system, building confidence
- Tendency to “build quality in”

© 2020 Carnegie Mellon University

[[Distribution Statement A] Approved for public release and unlimited distribution.

Feedback

DRAFT

Typical of “Primarily Document” Focus:

- Prefer larger, less frequent demos
- Requirements documents seen as “ground truth” for user needs, even when known to be superseded
- Constraints on opportunities for feedback
- Rushed feedback on documents
- More investment in documenting “to be” state than in documenting “as built”
  - Using documents to “lock down” design,
  - Then struggling to keep them current?

Aspirations for a Broader Aperture:

- Recognition that demo doesn’t EQUAL test, but INFORMS it
- Stakeholder participation in demos of small pieces of functionality
- Open, continuous feedback about both the fact of and the meaning of progress or lack thereof
- Info from demos is fed forward to testing and certification staff to ensure alignment
- Definition of Done that includes certification needs (cyber, DT/OT, ATC, ATO, etc.)
- Participation on continuous integration team by govt staff seen as a high priority

© 2020 Carnegie Mellon University

[[Distribution Statement A] Approved for public release and unlimited distribution.

Requirements

DRAFT

Typical of “Single Delivery” Focus:

- Work must commence early to limit risk
- Narrow time window to set the baseline
- Increasing resistance to requirements change over time, though knowledge of real user need continues to evolve
- Favoring breadth over depth in reviews
  - Hard to take in the large requirements set
  - Time for “digging in” on critical issues is rarely available during the review
- Sometimes we get as far as we can, declare success, and track action items until the next event

Aspirations for Iterative Approach:

- Mix of “push” and “pull” communication across govt/contractor interface as requirements are elaborated/refined
  - Facilitated by workflow and collaboration tools
- Frequent high bandwidth meetings keep the relationship going, not just technical work
- Transparency among stakeholders is an essential ingredient to build trust
- Frequent small batch prioritizations build a solid base of understanding of current state and progress

© 2020 Carnegie Mellon University

[[Distribution Statement A] Approved for public release and unlimited distribution.

Compliance vs. Insight

DRAFT

Typical of Compliance Focus:

- Deadlines and plans are difficult to change, and sticking to the plan becomes it's own objective
  - Product performance, and cost performance often suffer
- Siloed relationships where common goals exist
  - Enterprise perspective gets lost in the informal channels
  - *Isolation* mistaken as hallmark of *Independence*
- Focus on the letter of the spec (verification) can overshadow focus on intent (validation)
- A template to reduce time spent on oversight is valuable – even if it reduces information conveyed
- Work is sometimes designed to pass the audit more than deliver the intended value

Aspirations for a Focus on Insight:

- Off nominal conditions are not always bad outcomes
- Information that invalidates the plan but leads to better enterprise outcomes is not "Bad News"
- Proactively build knowledge with information, rather than retrospectively uncovering facts from data
  - Feed data forward to consumers, rather than batching it up in hopes that we can manage the message it sends
- Focus on sustainable pace, with predictable delivery
- Measures carefully selected and visibly used to solve problems in the process, not punish the contractor
- Collaborative mindset

© 2020 Carnegie Mellon University

[[Distribution Statement A] Approved for public release and unlimited distribution.

Role of Systems Engineering and Modeling in F-35 DRAFT Agile/Lean/DSO Approach

F-35's Digital Engineering Environment brings together hardware, software, and operational eco-systems, enables incremental systems engineering, not just software engineering

- DEE connections into the DevSecOps pipeline is a F35 strategy to ensure that the models stay current

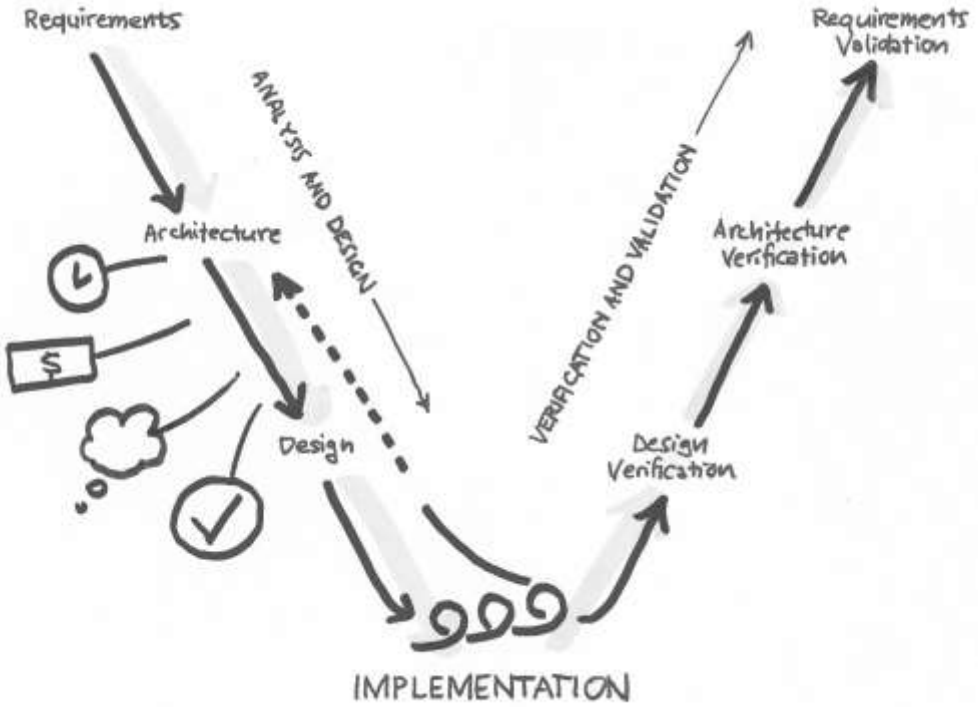
Changing to lean engineering:

- More reliance on modeling
- Less reliance on "complete specifications early"
- Incremental technical reviews based on early implementation
- Small batch system, hardware, and software engineering
- More reliance on road-mapping and continual Capabilities backlog refinement and prioritization

© 2020 Carnegie Mellon University
[Distribution Statement A] Approved for public release and unlimited distribution.

What Happens When “Large Batch” Systems Engineering Meets “Small Batch” Agile SW Development?

DRAFT



© 2020 Carnegie Mellon University

[[Distribution Statement A] Approved for public release and unlimited distribution.

Particular Architectural Choices Can Enable or Disable Agile/Lean/DSO

DRAFT

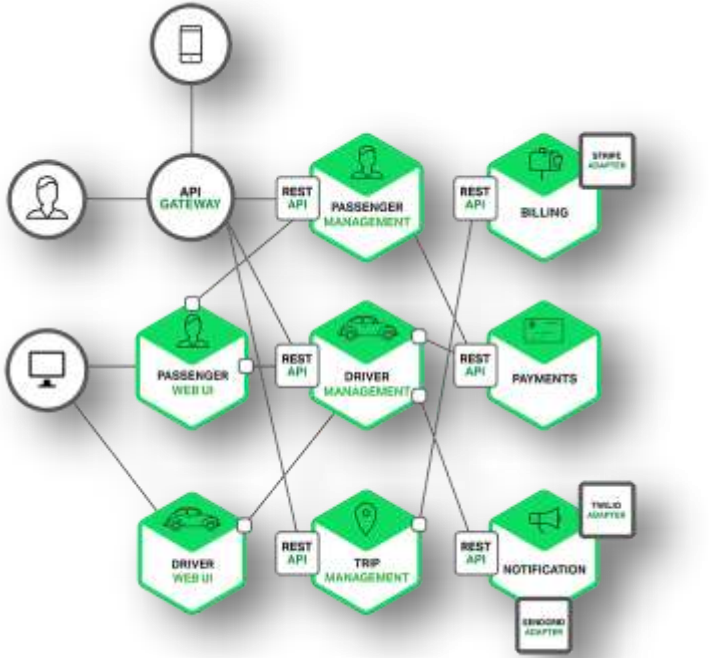
Modular architecture patterns enable

- Teams to work independently
- Integration to occur frequently
- Easier isolation of defect sources

Legacy code being reused often exhibits architecture patterns that cause problems for Agile/Lean/DevSecOps

- Often difficult to refactor to new patterns
- Often built in a time when tight coupling was needed architecturally to get performance outcomes needed

Microservices Architecture Pattern is an Enabler to Agile SW Dev



© 2020 Carnegie Mellon University

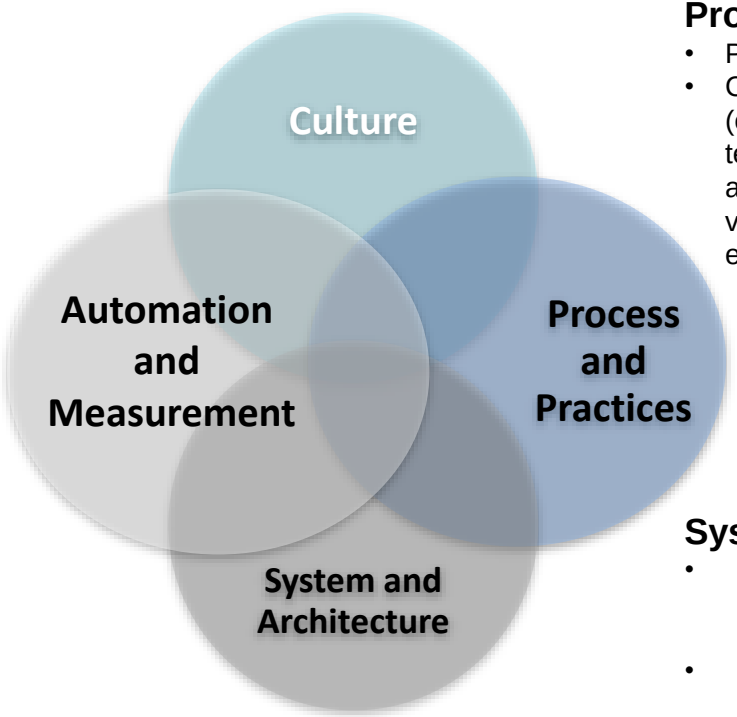
[[Distribution Statement A] Approved for public release and unlimited distribution.

Multiple Dimensions of DevOps

DRAFT

Culture

- Developer and Ops collaborate (Ops includes security)
- *Developers* and Operations support releases beyond deployment
- Dev and Ops have access to stakeholders who understand business and mission goals



Process and Practices

- Pipeline streamlining
- Continuous-delivery practices (e.g., continuous integration; test automation; script-driven, automated deployment; virtualized, self-service environments)

Automation/ Measurement

- Automate repetitive and error-prone tasks (e.g., build, testing, and deployment maintain consistent environments)
- Static analysis automation (architecture health)
- Performance dashboards

System and Architecture

- Architected to support test automation and continuous-integration goals
- Applications that support changes without release (e.g., late binding)
- Scalable, secure, reliable, etc.

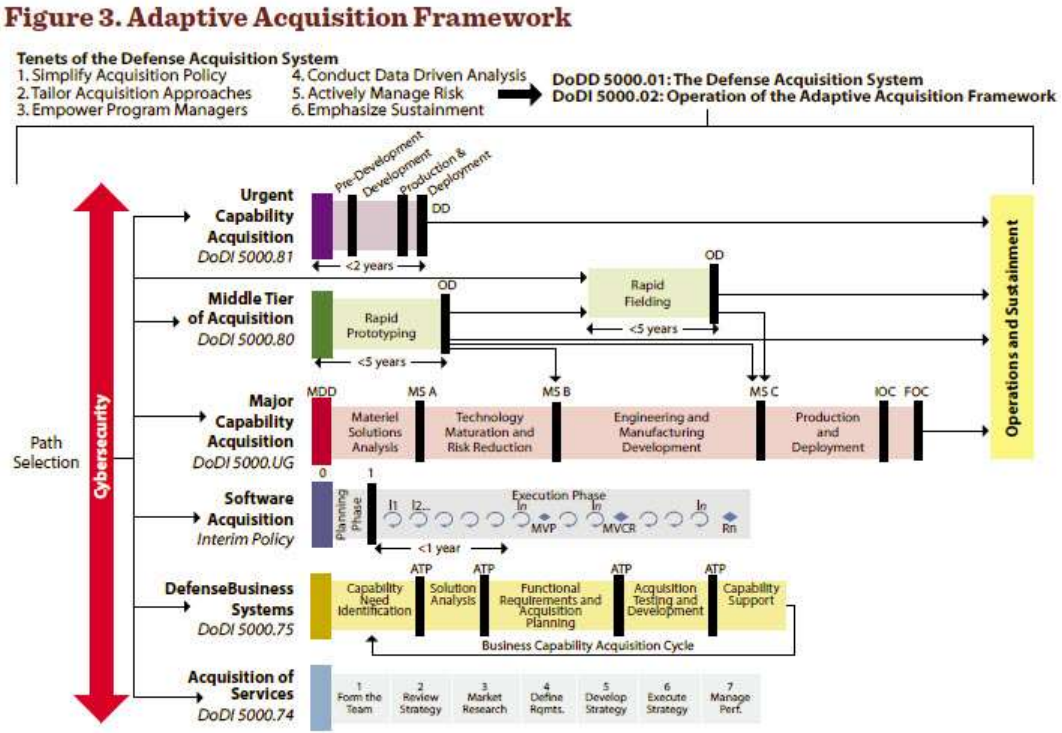
© 2020 Carnegie Mellon University
[Distribution Statement A] Approved for public release and unlimited distribution.

ODIn is Taking Advantage of the New Adaptive Acquisition Pathways

DRAFT

Adaptive Acquisition Pathway being used by Odin:

xxx

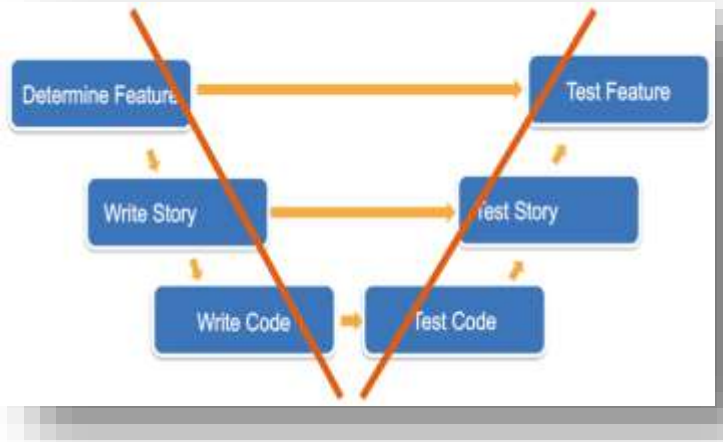


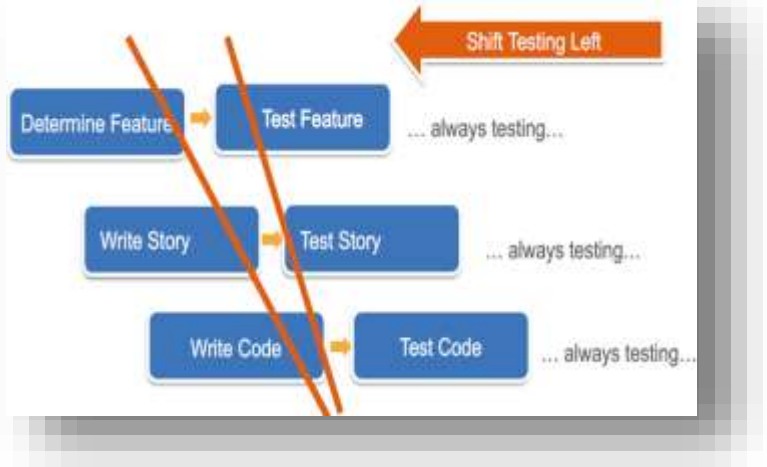
© 2020 Carnegie Mellon University

[[Distribution Statement A] Approved for public release and unlimited distribution.

Role of Testing in Agile, Lean, DevSecOps: Traditional Testing (V-model) Delays Feedback

DRAFT





- **Specific practices enable shifting left for testing and unified certification**
  - **Modeling and simulation (a focus of the F-35 Digital Engineering Environment) and**
  - **Modern design approaches (hypothesis-driven, behavior-driven, test-driven)**
  - **DSO pipeline that accounts for securing the pipeline, not just securing the software**

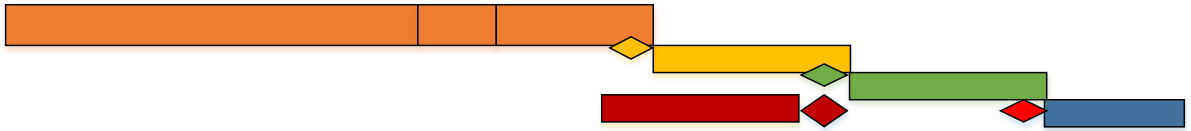
© 2020 Carnegie Mellon University

[[Distribution Statement A] Approved for public release and unlimited distribution.

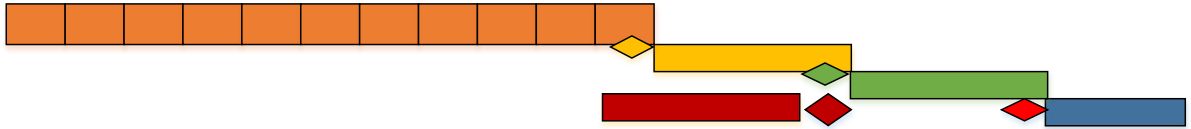
Left-shift with Agile Testing

DRAFT

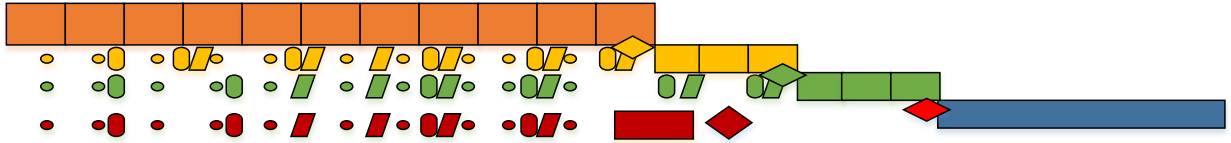
Traditional Vee-process



Agile development with traditional DT and OT (Hybrid)



Agile development with traditional DT and OT, early integration synch points



Moving from phased and siloed testing to Agile testing is the "Big Deal"

Integrating Agile cadence with DT/OT is a key challenge

OEM Dev

Developmental Testing (DT)

Cyber

Operational Testing (OT)

Deployment

© 2020 Carnegie Mellon University

[[Distribution Statement A] Approved for public release and unlimited distribution.

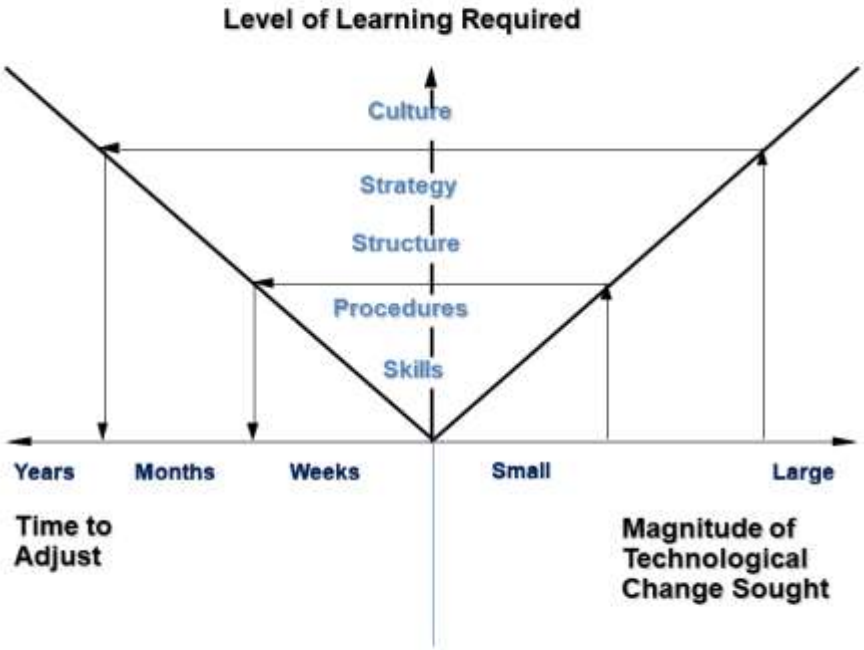
Culture Aspects of Moving to Lean/Agile

© 2020 Carnegie Mellon University

[[Distribution Statement A] Approved for public release and unlimited distribution.

Culture Change will take Time and Effort!

DRAFT



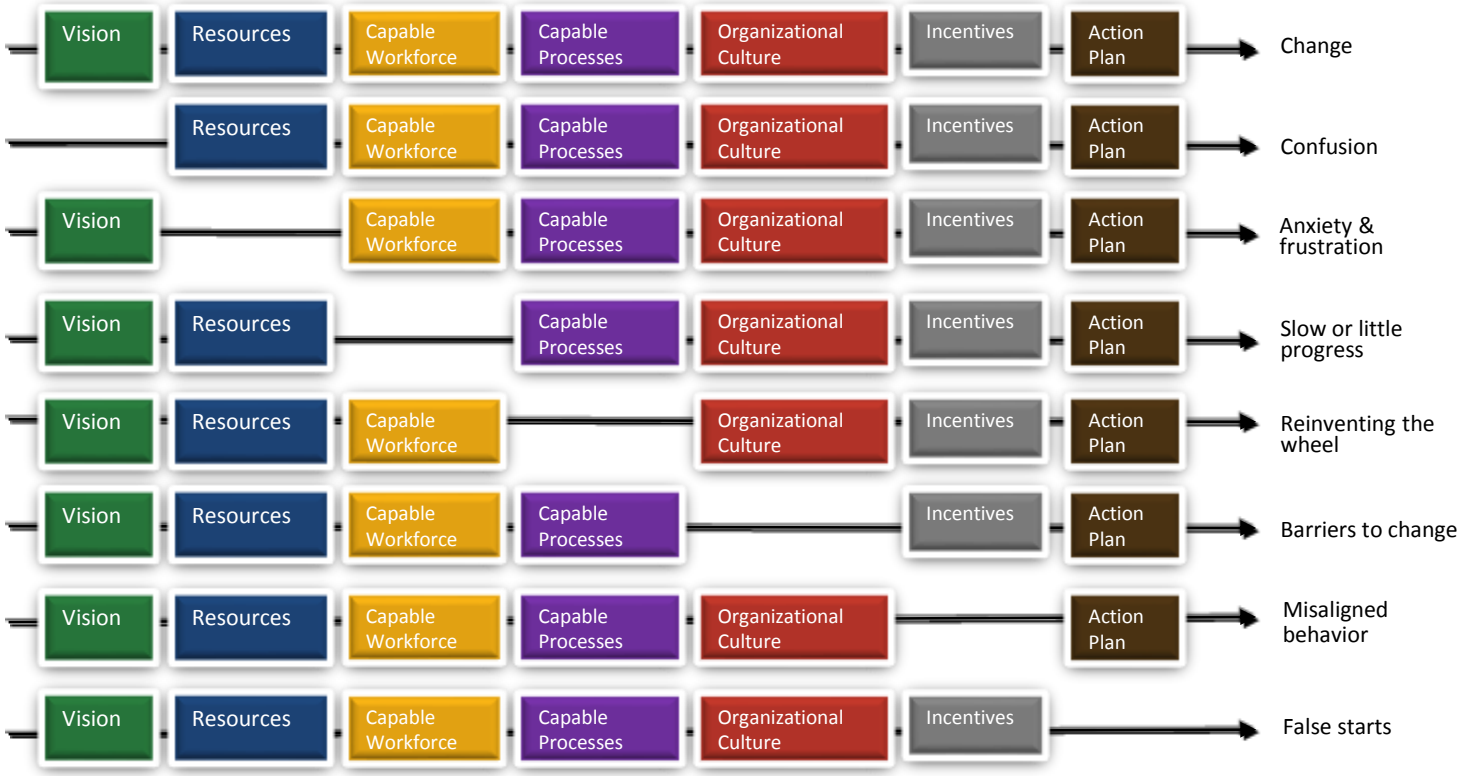
Source: Adler, Paul. "Adapting Your Technological Base: The Organizational Challenge", Sloan Mgmt Review, 1990

© 2020 Carnegie Mellon University

[[Distribution Statement A] Approved for public release and unlimited distribution.

Success Factors in Adopting New Practices: What Can We Learn?

DRAFT



The symptoms on the right can be used to diagnose what might be missing in our adoption support approach

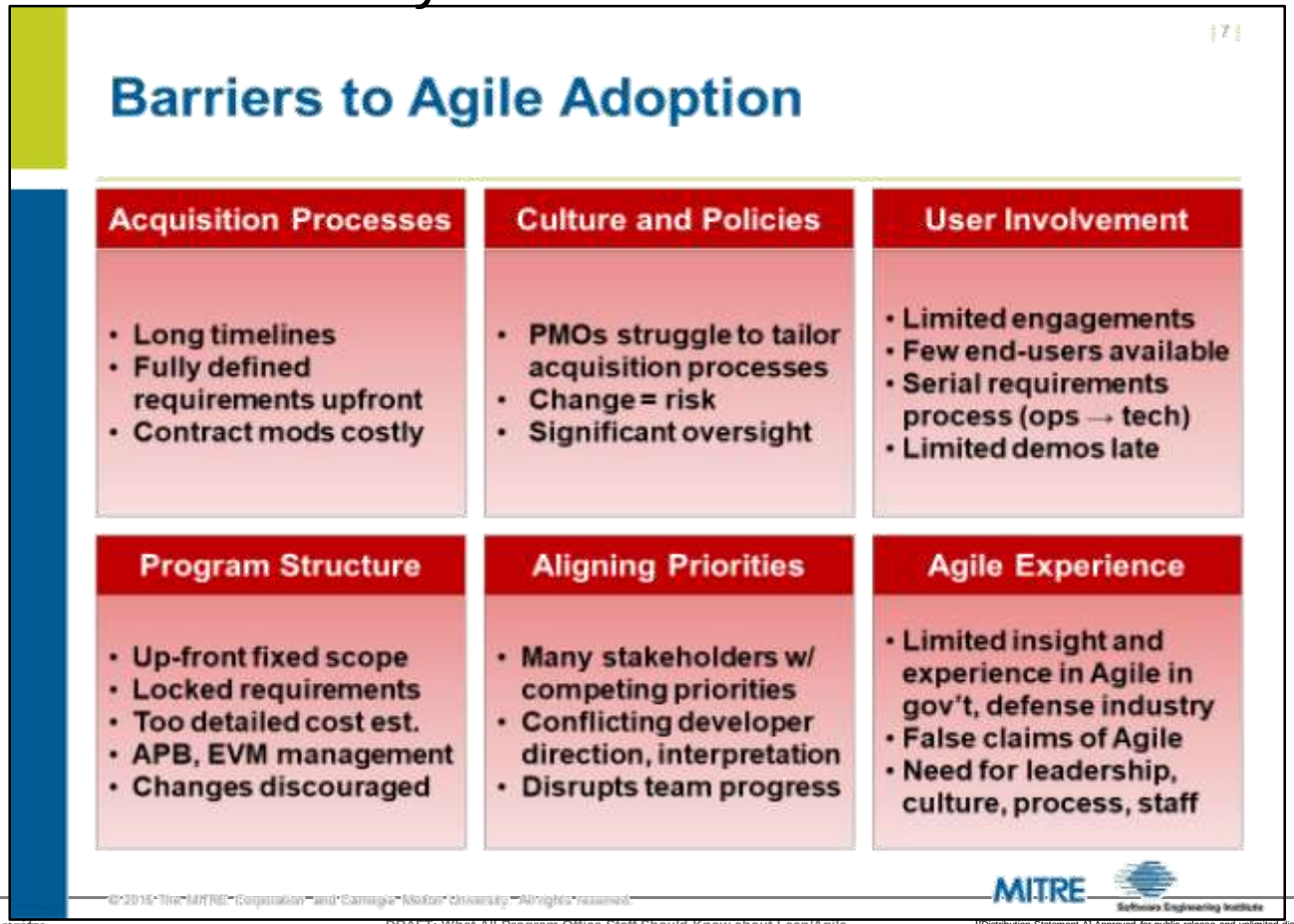
Adapted by Buttles (2010) from: Delorise Ambrose, 1987

© 2020 Carnegie Mellon University

[[Distribution Statement A] Approved for public release and unlimited distribution.

Some Barriers We May Run Into

DRAFT

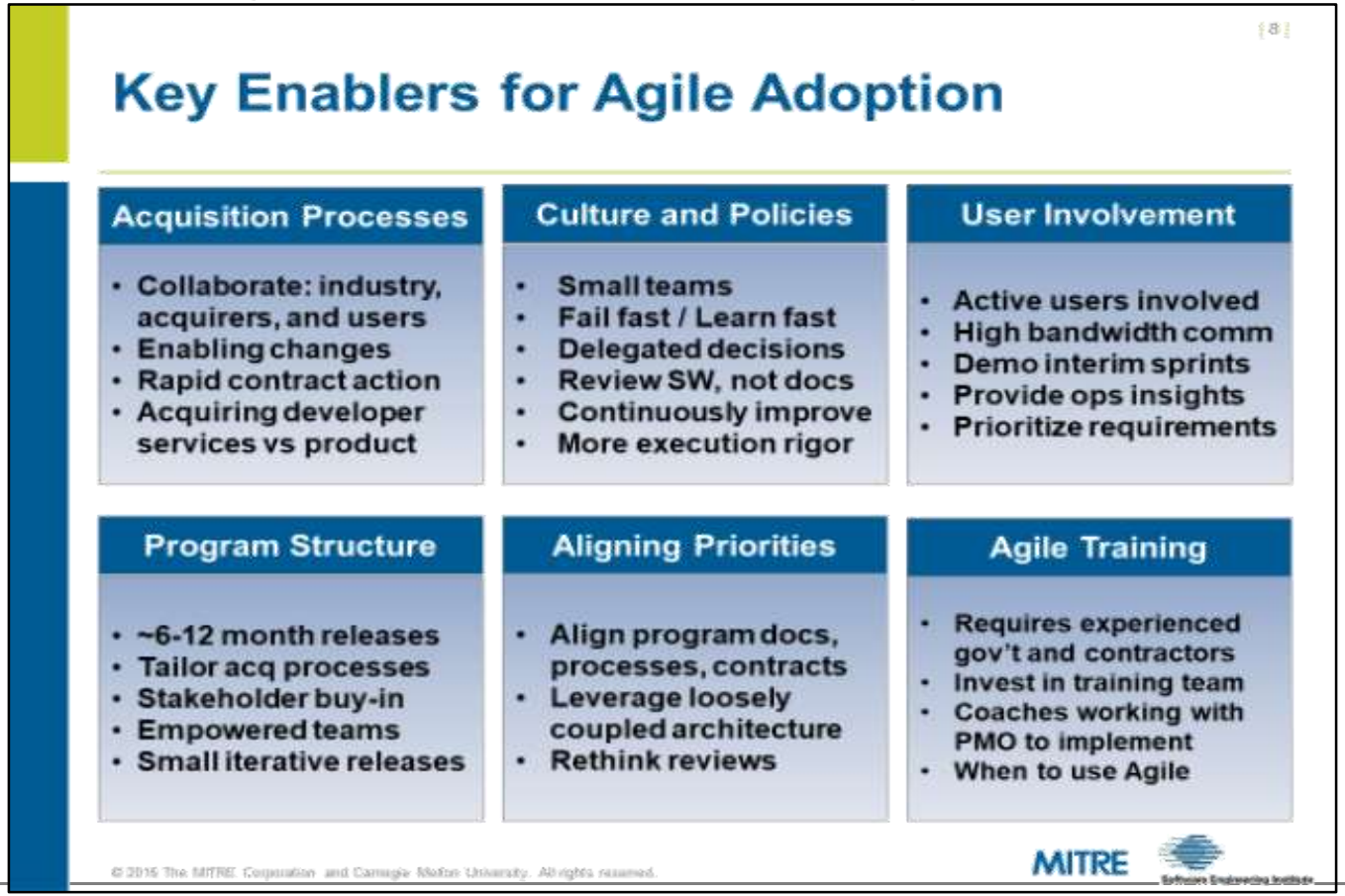


Source: 2016 briefing to General E. Pawlikowski on USAF Agile Adoption, SEI & Mitre

© 2020 Carnegie Mellon University
[Distribution Statement A] Approved for public release and unlimited distribution.

Some Advantages We Want to Leverage

DRAFT



Source: 2016 briefing to General E. Pawlikowski on USAF Agile Adoption, SEI & Mitre

[[Distribution Statement A] Approved for public release and unlimited distribution.

Respond at: www.pollev.com/mainsummit799

What one word/phrase describes what you perceive as the most difficult barrier to adoption of Agile, Lean and DevSecOps in your setting? (Word Cloud)

© 2020 Carnegie Mellon University

[[Distribution Statement A] Approved for public release and unlimited distribution.

Respond at: www.pollev.com/mainsummit799

What was useful to you in Module 2?

© 2020 Carnegie Mellon University

[[Distribution Statement A] Approved for public release and unlimited distribution.

Summary

© 2020 Carnegie Mellon University

[[Distribution Statement A] Approved for public release and unlimited distribution.

A Few Notable Recent Events/Links

DRAFT

USAF Chief Software Officer:

https://software.af.mil/

https://software.af.mil/training/

SMC Agile/Lean Self-Service Learning Paths Catalog (in Beta, available to DI2E users):

Some of the materials in the learning package came from this site. If you have DI2E, you can access a larger set of materials directly:

https://confluence.di2e.net/display/AGILESMC/Self-Serve+Agile+Learning+Paths

Software Engineering Institute Agile Resources: podcasts, blog posts, Technical Notes on many Agile in Government topics:

https://www.sei.cmu.edu/research-capabilities/all-work/display.cfm?customel_datapageid_4050=21345

© 2020 Carnegie Mellon University

[[Distribution Statement A] Approved for public release and unlimited distribution.