

Lean, Agile, and DevSecOps for AFLCMC/WIU Engineering

Brigid O'Hearn
12 May 2022

Software Engineering Institute
Carnegie Mellon University
Pittsburgh, PA 15213

Copyright 2022 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 may be reproduced in its entirety, without modification, and freely distributed in written or electronic form without requesting formal permission. Permission is required for any other use. Requests for permission should be directed to the Software Engineering Institute at permission@sei.cmu.edu.

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

DM22-0428

Class Expectations/Goals for this Overview

**WHAT TO
EXPECT**

You will Learn:

- Basics of Lean, Agile & DevSecOps that will fit into a 2-hour workshop
- Lessons learned from previous and current transformation

You will NOT Learn:

- Detailed Lean, Agile & DevSecOps roles, procedures and techniques such as how to estimate story points, how to execute a Program Increment
- Detailed Scaled Agile Framework (SAFe) methodologies and processes

This is meant to be interactive so please participate via Zoom chat/hand-up, PollEverywhere, etc

Agenda

- Some baseline PollEverywhere questions!
- Product & PMO Context for Lean/Agile/DevSecOps
- What is Agile?
- Culture Aspects of Adopting New Practices
- Summary

Respond at: www.pollev.com/mainsummit799



*What is your level of experience with
Lean?
Agile?
DevSecOps?*

Respond at: www.pollev.com/mainsummit799



How big of a change do you think Lean/Agile/DevSecOps is for your programs?

Respond at: www.pollev.com/mainsummit799



What do you think is your program's biggest challenge in adopting Lean/Agile/DevSecOps?

Respond at: www.pollev.com/mainsummit799



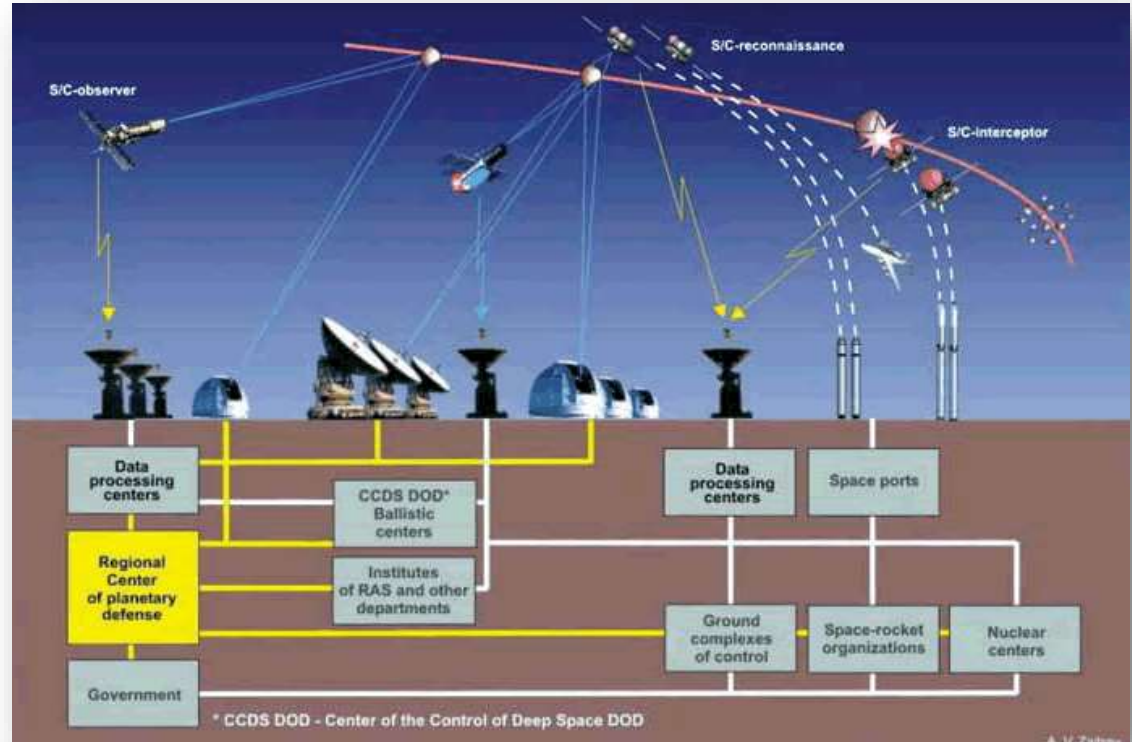
Do you have any specific Lean/Agile/DevSecOps topics or questions that you'd like answered or covered in this workshop?

Product & PMO Context for Lean/Agile

Product & PMO Context for Agile

SOCOM products are integrated into an existing large and complex cyber-physical system.

Our implementation of any approach, including Lean/Agile, *must account for our context*



A “Generic” Complex CyberPhysical System Depiction

Why Agile (and Lean and DevSecOps)?



Direction from OSD across multiple NDAA's reinforces need for fast feedback all along the development path (Lean/Agile)

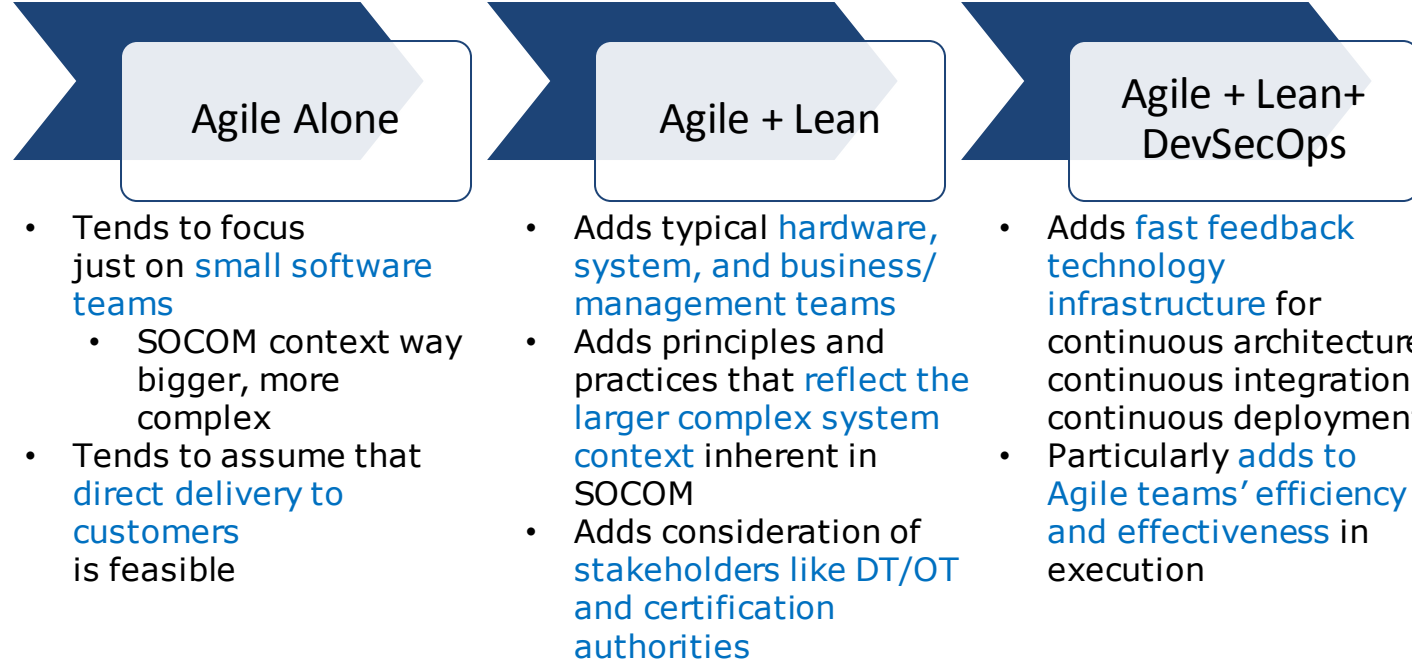


To assure time-certain delivery, parallelization of certification and development activities is needed (DevSecOps or Dev*Ops)



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

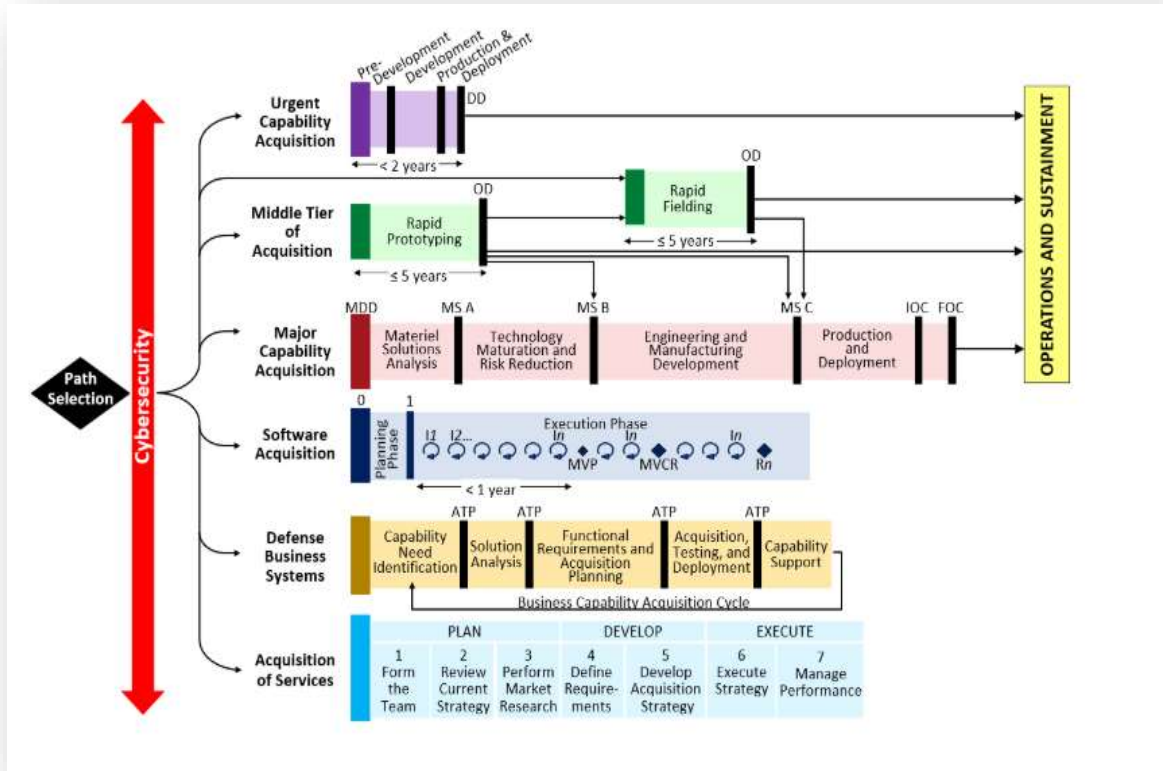
Why Agile *AND* Lean *AND* DevSecOps?



Adaptive Acquisition Framework

New ways of acquiring systems.
Note the Software Acquisition Pathway (SWP) for software-dominant products.

<https://aaf.dau.edu/>



Software Pathway



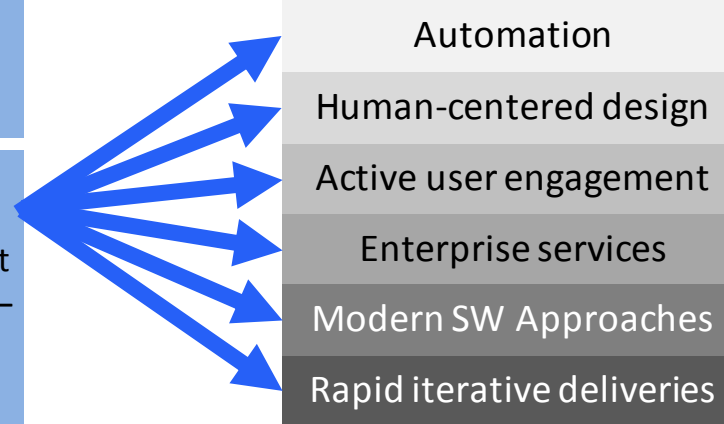
SWP is the **only** acquisition pathway with a Congressional mandate to speed capability delivery.

Achieved via key practices:

SWP is designed to “make it easy to do the right thing” – Practices are **mutually reinforcing** – *It is the preferred pathway for software.*

GAO notes* the SWP is the **only** DoD policy that incorporates leading commercial practices.

*GAO-22-104513, March 2022



Minimum Viable Capability Release (MVCR) at least every 12 months to an “operational environment”

Engagement with other programs of all types supports adoption of practices across the enterprise. SWP is an incubator for DoD!

There are currently 35 programs on SWP with more planning to move.

Respond at: www.pollev.com/mainsummit799



Which Adaptive Acquisition Framework pathways are you familiar with or are you working in today?

Relevant Concepts from Lean/Agile for AFLCMC/WIU Engineering

What is Agile, Anyway?

Reminders and Relevant Interpretations

Working Definition of Agile



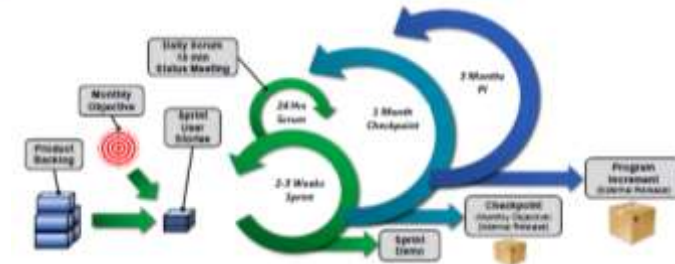
Agile An *iterative and incremental* (evolutionary) approach to software development which is performed in a *highly collaborative manner* by *self-organizing teams* within an *effective governance framework* with “*just enough*” ceremony that produces *high quality software* in a *cost effective and 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>

Typical Agile Tempo

- Tempo: Oriented around calendar months; time-boxed
- Sprints: ~2 weeks / ~2 per month
- Demos: Monthly
- Program Increments (PI): 3 months; PI planning month prior

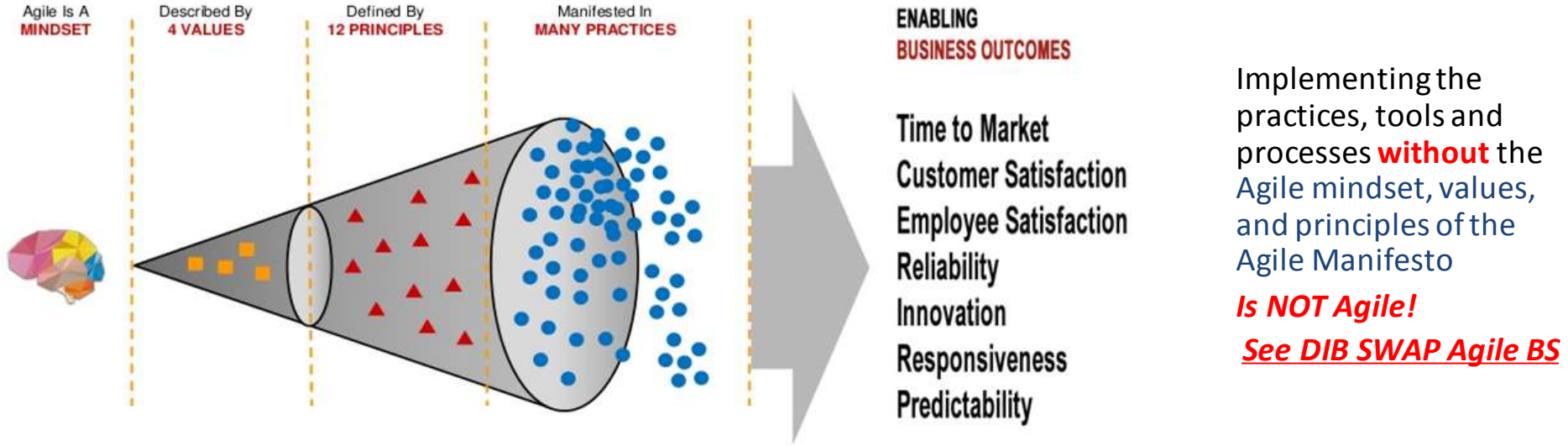


Respond at: www.pollev.com/mainsummit799



Have you participated in Program Increment (PI) planning, daily standups, sprint demos, retrospectives, or other “ceremonies”?

What is Agile?



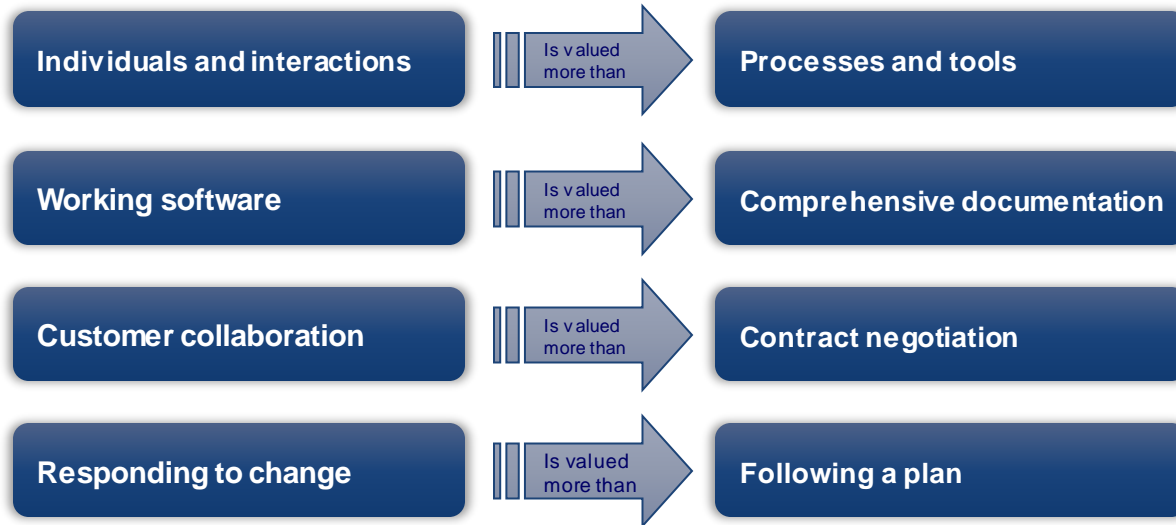
Source: <https://www.slideshare.net/MichaelTarnowski/agile-mindset-for-executives>

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.

Agile Software Development Manifesto

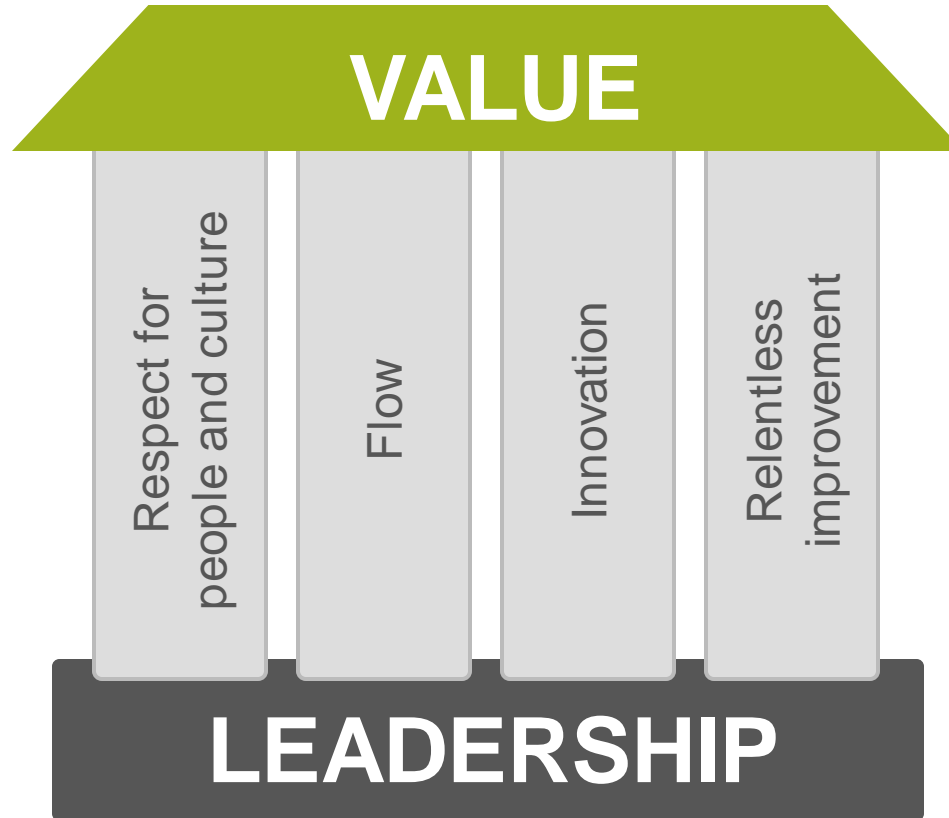
Created in 2001 and focused on small teams

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



<http://agilemanifesto.org>

Work in the House of Lean



Industry isn't the only one to Implement the House of Lean!

Government program offices and stakeholders play a big part in how much can be achieved related to House of Lean:

- **Value**—government is the determiner of the value needed, and evaluation of what is received
- **Respect for people and culture** – understanding how different stakeholders work/need to work is key to alignment
- **Flow** – removing bottlenecks on the government side is often as important as removing them on the contractor side
- **Innovation** – some of the innovation needed on the govt side is process innovation, not just product innovation
- **Relentless improvement** – no program is ever “good enough”; resting on our laurels guarantees threats will surpass our capabilities

Apply SAFe Lean-Agile Principles

#1 Take an economic view

#2 Apply systems thinking

#3 Assume variability; preserve options

#4 Build incrementally with fast, integrated learning cycles

#5 Base milestones on objective evaluation of working systems

#6 Visualize and limit WIP, reduce batch sizes, and manage queue lengths

#7 Apply cadence, synchronize with cross-domain planning

#8 Unlock the intrinsic motivation of knowledge workers

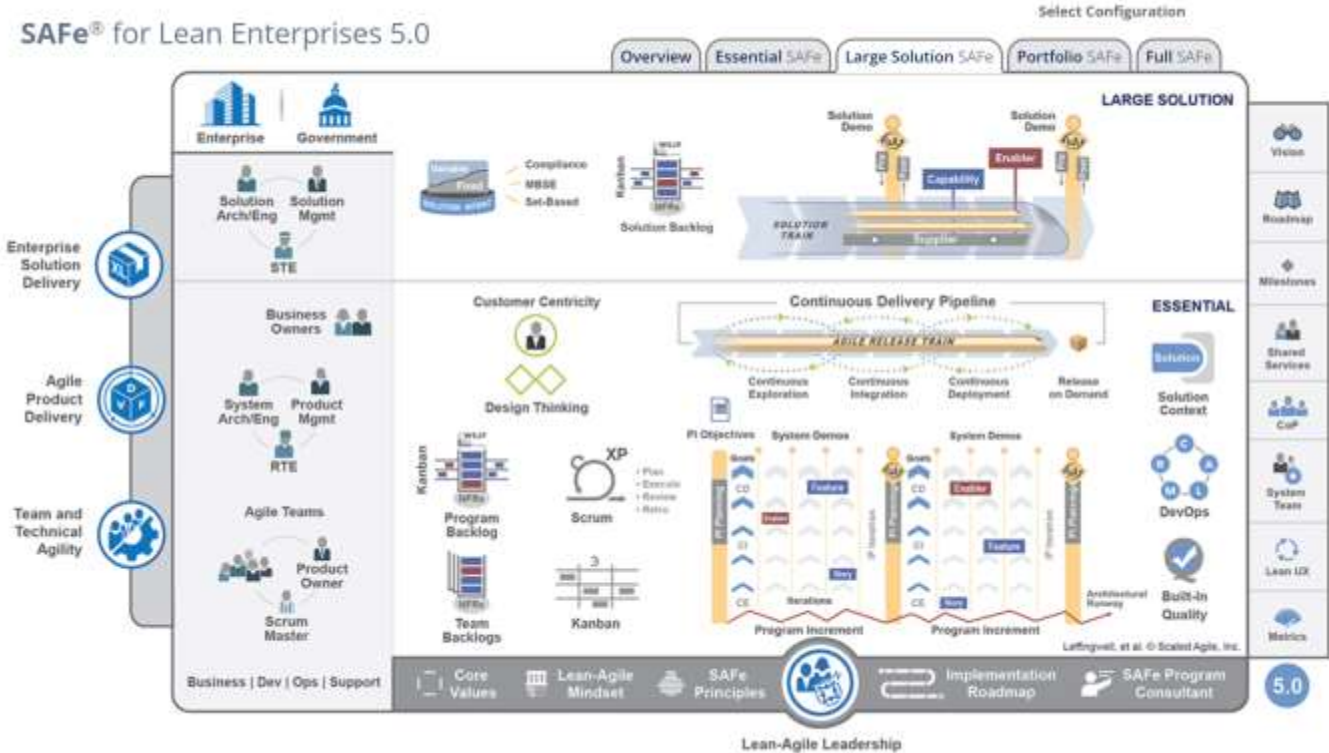
#9 Decentralize decision-making

#10 Organize around value

SAFe Large Solution Configuration

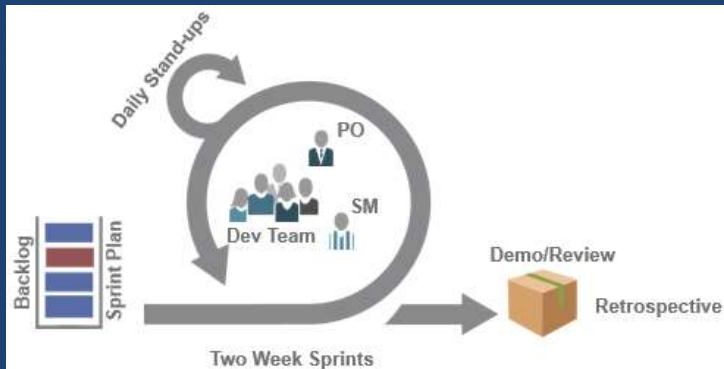
Most large, complex DoD systems use a tailored version of Large Solution SAFe

- Because it is systems engineering-aware
- Because it is MBSE aware
- Because many programs are larger than 10-12 teams



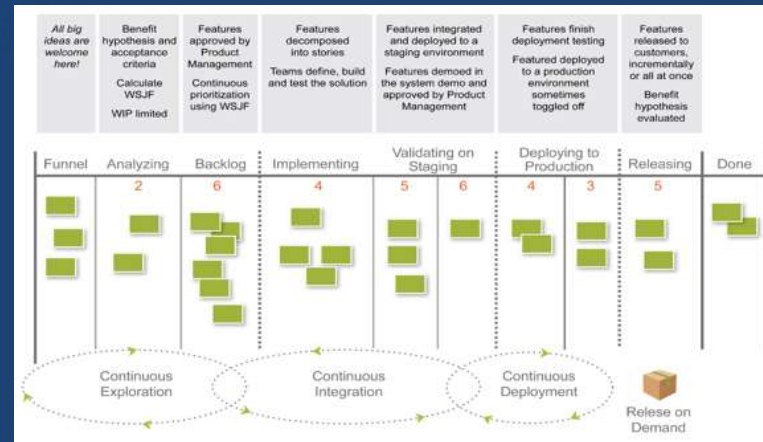
Choose One or More Methods to Fit Your Team Context

Scrum



- Most common product dev approach
- Not just sw dev
- *Use when you have a dedicated team working on a product and timebox is an advantage*

Kanban



- 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)
- Use when services are more in play than product delivery, and when team is not dedicated to single product

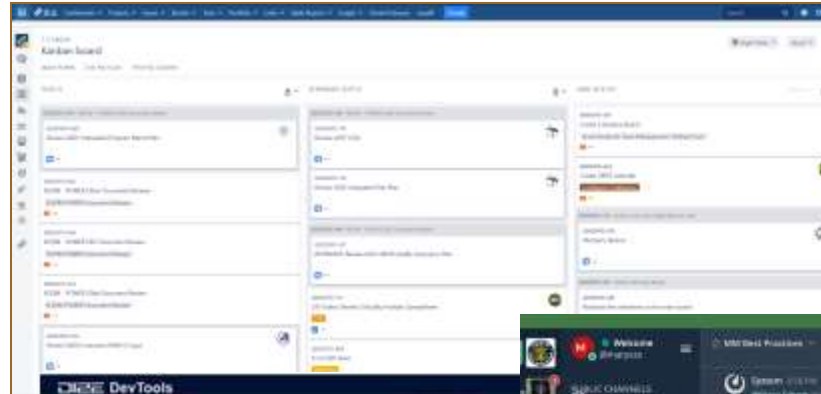
Goal: Common Tools Used Across Different Team Types in DOD and Industry

Tool focused on workflow management: **JIRA**

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

Tool focused on inter/intra-team collaboration: **MATTERMOST** and **MS Teams**

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



Plan, Do, Check, Adjust

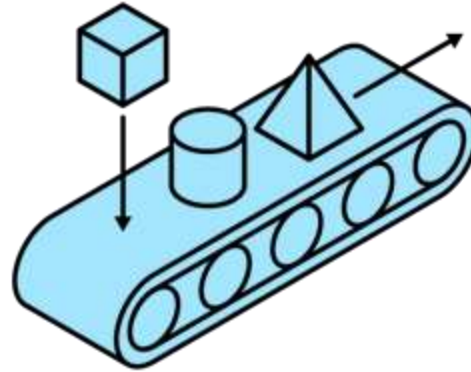


The PDCA problem-solving cycle scales from individual teams to an entire enterprise

Agile Represents a Major Requirements Transition – Different Acquisition Objectives



Buying a
Box



Buying an ongoing
delivery stream

Dev*Ops

Continuous Development
hours - days

$$T_K = (\text{Dev})$$

Continuous Integration
2 weeks - 1 month

$$T_{K+1} = (\text{Dev})(\text{Ops})$$

← Industry culture and policy evolved; blurred lines between dev and ops environments

Continuous ATO
+ hours

$$T_{K+2} = (\text{Dev})(\text{Sec})(\text{Ops})$$

← Security culture and policy evolved; baked elements into automated pipeline

Continuous Safety
+ months

$$T_{K+3} = (\text{Dev})(\text{Sec})(\text{Safety})(\text{Ops})$$

Continuous Airworthiness
+ months

$$T_{K+4} = (\text{Dev})(\text{Sec})(\text{Safety})(\text{AW})(\text{Ops})$$

Continuous Test
+ years

$$T_{K+5} = (\text{Dev})(\text{Sec})(\text{Safety})(\text{AW})(\text{DT/OT})(\text{Ops})$$

Continuous Weapons
+ months

$$T_{K+6} = (\text{Dev})(\text{Sec})(\text{Safety})(\text{AW})(\text{DT/OT})(\text{Seek Eagle})(\text{Ops})$$

Continuous Nuclear
+ months

$$T_{K+7} = (\text{Dev})(\text{Sec})(\text{Safety})(\text{AW})(\text{DT/OT})(\text{Seek Eagle})(\text{Nuc})(\text{Ops})$$

...

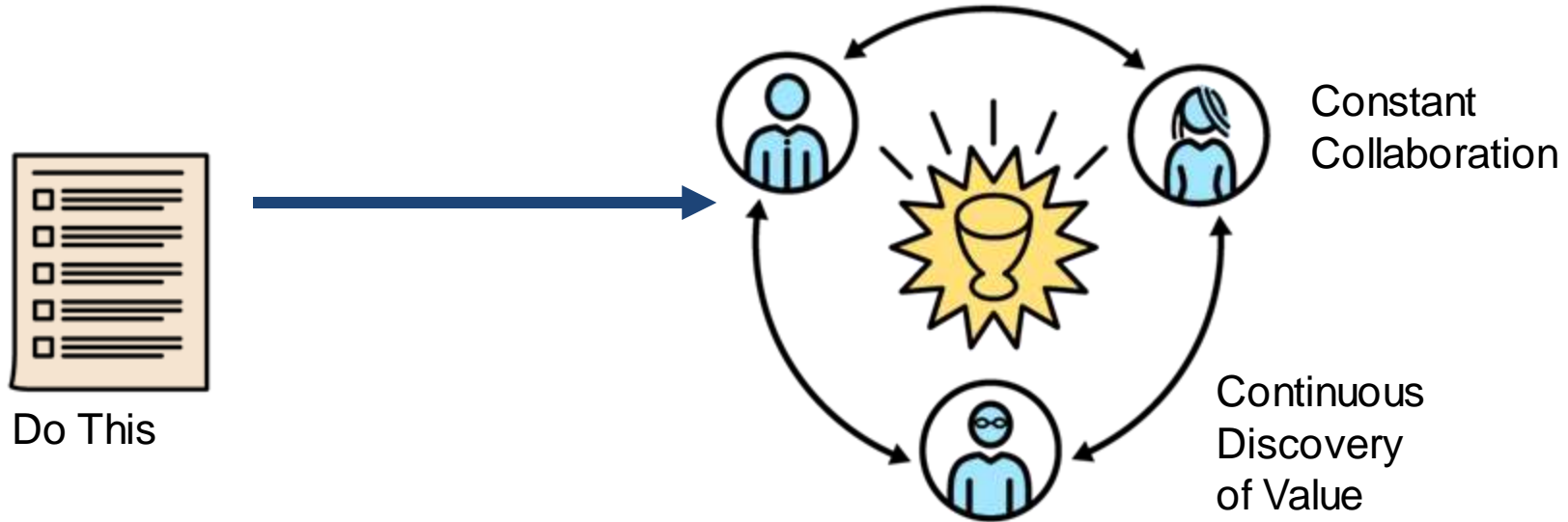
$$T_{K+n} = (\text{Dev})^*(\text{Ops})$$

Must evaluate and evolve these elements (and more) to enable automated pipeline

Keys to Success:

1. Evolve culture and policy
2. Early stakeholder involvement
3. Automated vs. manual processes

Agile Represents a Major Requirements Transition – Different Vendor Interactions



Bottom Line for Programmatics of Complex Systems

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 Framework change some of those requirements

INSIGHT is a necessary *enabler* to effective oversight

- Well-established CDRLs and DIDs may not always be 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***

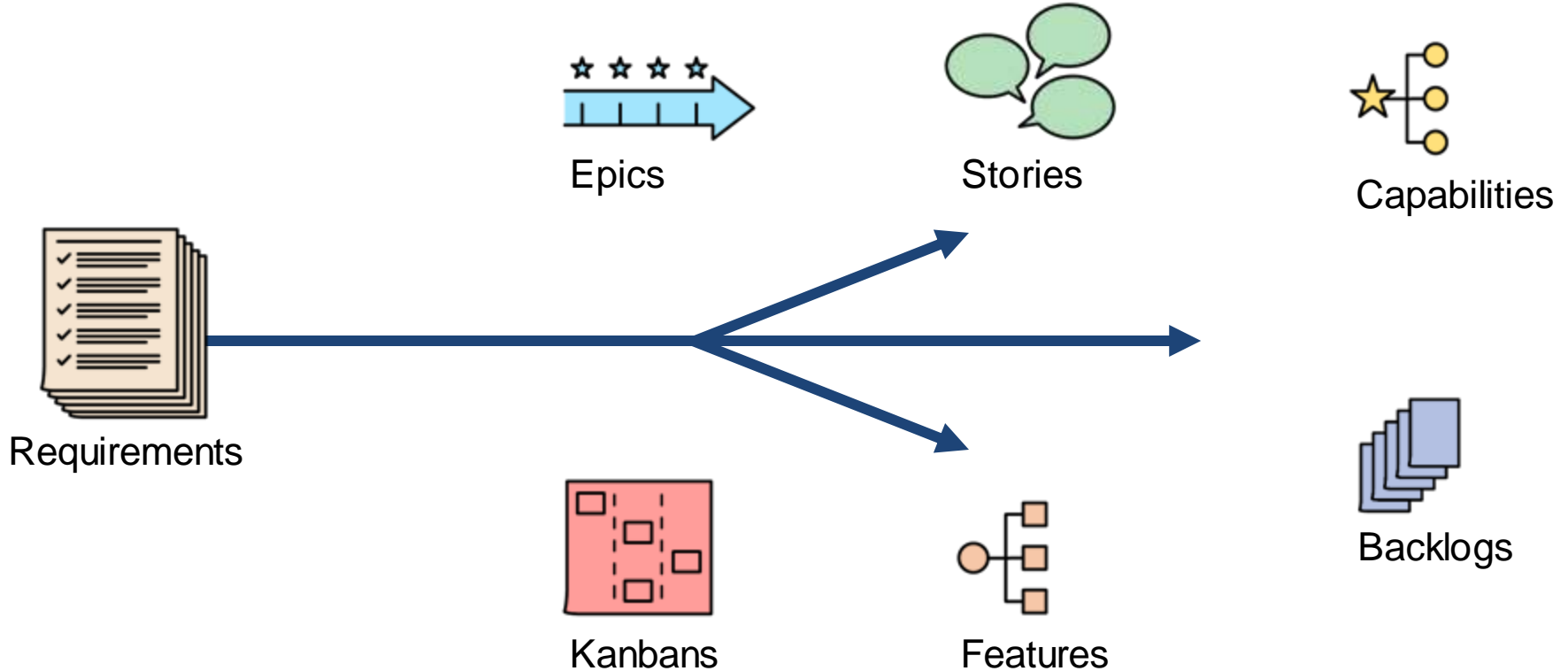


Compliance



Collaboration

Agile Represents a Major Requirements Transition – Different Artifacts

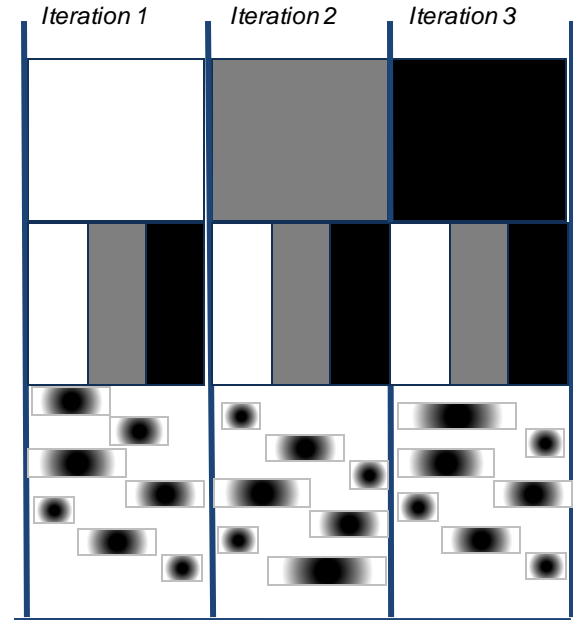


Taking a Small Batch, Iterative Approach

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



Agile Motto for the Backlog Process



*What is a backlog? A container for work.
Who makes the backlog?
And who ensures that what is in the backlog is prioritized? And who ensures that the committed backlog items meet expectations of the customer and stakeholders?*

Product Owners and Managers!

Government Product Owner/Manager is Usually a Full Time Role

Product OWNERS work with teams of 6-10 people

- In addition to refinement and prioritization, Product Owners also help build the Acceptance Criteria and verify that Acceptance Criteria is met

Product MANAGERS work with teams of teams of 50-125 people

- One Product Manager is refining, prioritizing, and coordinating the work of between **50-125 people**



Lots of communication, coordination, and conflict resolution across varied stakeholder communities within the enterprise for both these roles

What Traditional Staff Roles Are Candidates for Product OWNER Role?

Engineers or Program Management staff who are experienced

- Writing and analyzing requirements
- Engendering trust among different stakeholders, especially developers and users
- Prioritizing work in the 2 week to 3 month time horizon
- Analyzing, communicating, and mitigating technical risks
- Communicating with stakeholder communities about their requirements/desirements
- Evaluating and verifying a team's work

What Traditional Staff Roles Are Candidates for Product Manager Role?

Systems engineers or program management staff who are experienced

- Writing and analyzing requirements
- Creating product roadmaps
- Prioritizing work in the 3-6 month horizon
- Analyzing, communicating, and mitigating technical and programmatic risks
- Communicating with important stakeholder communities about their requirements/desirements

Program Managers or Deputy Program Managers who understand that the ProDUCT Manager role is about managing work content, not managing the people doing the work

Stories – For Building Products: Including the “Who” and “Why” of a Requirement

User Stories

Expresses concepts in a way operational user would find useful

Template: *As a “role,” I want to “function” so I can “operational goal”*

Technical Stories

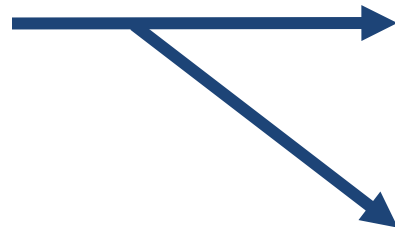
Express quality attributes of a system, subsystem or component that may not be directly seen by the user but are essential to meeting mission goals

Template: *Typically free-form; should incorporate the “why”*

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



Date-based Milestones



Increment-Based Demos

Definition of Done



Team Increment



System Increment



Solution Increment



Release

What is the Definition of Done (DOD)?

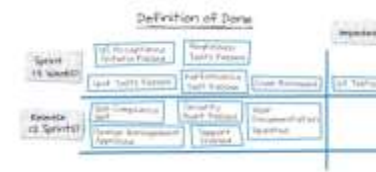
Definition of Done is an explicit declaration of the completion criteria for some aspect of an agile life cycle. DOD can be applied to an individual artifact (e.g., a user story), a sprint or iteration or a Program Increment (PI). It is different from **Acceptance Criteria**, which are completion expectations for a *specific* artifact, as opposed to a completion criteria for all artifacts (DOD).

Why do we need a specific Definition of Done?

One of the ways that Agile methods achieve the speed they are known for is that staff have confidence that when they are “done” with some task or artifact, it’s safe to move on to the next one. The explicit Definition of Done is a key contributor to enabling this confidence.

When is the Definition of Done established?

The DOD is established prior to the work being done (for a User Story, DOD is established before the Product Backlog items are estimated; for a PI DOD is established prior to the completion of PI Planning, etc.)



When do you determine that the Definition of Done has been met?

Verifying that the DOD has been met depends on what DOD is being applied to. If DOD is being applied to an artifact, it is verified before the item is marked as “Done” in whatever **Information Radiator** the team is using to communicate status.

If DOD is being applied to an Agile life cycle phase, like the iteration, it would be determined during the review meeting that occurs at the end of the sprint or PI.

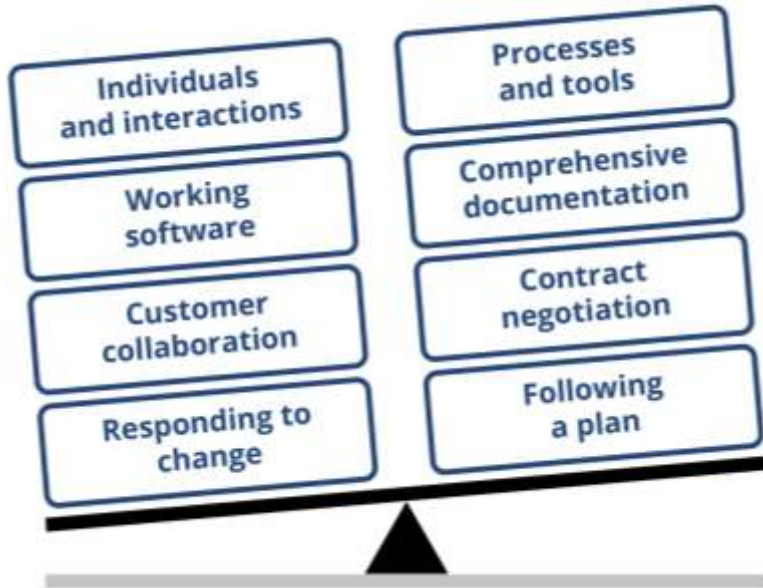
Other Notes on Definition of Done:

There is no “universal” Definition of Done. However, a good definition of done should follow SMART rules: Specific, Measurable, Actionable, Relevant, and Timely. Usually the time is predetermined (e.g. your iteration timebox)

Make the definition public and review it in between iterations, PIs, etc. ---reviewing the DOD as part of the **Retrospective** is a frequent approach, or including a review during the **Sprint Planning Meeting**.

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

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>

Agile

Many small
batch
interactions

Demos/User
feedback

Continuous
Backlog
Refinement

Seeking Insight

Traditional

Few large
batch
interactions

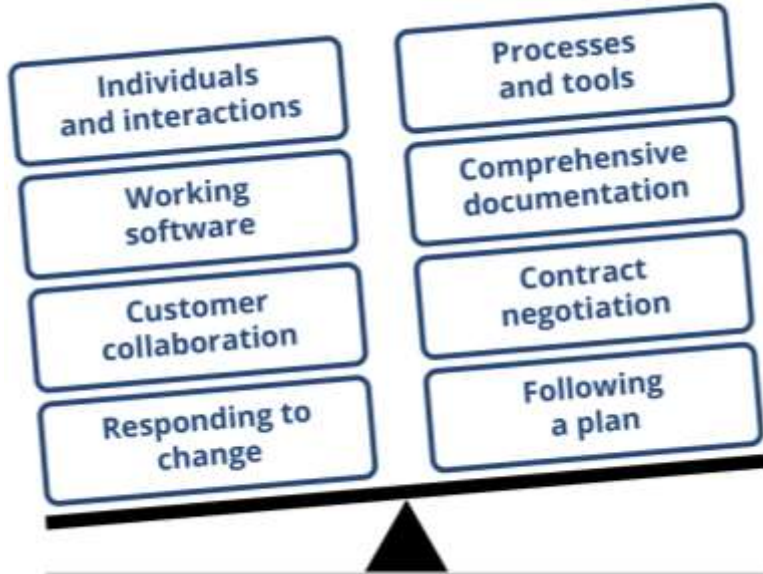
Primarily
Documentation
review

Single Delivery
Requirements
Document

Seeking
Compliance

Reorienting the Manifesto for *Agile Software Development* Toward *Multi-Program System Integration*

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>

Agile

Traditional

Many small batch interactions

Few large batch interactions

Demos/User feedback that explicitly exercise interfaces early and often

Primarily documentation review until late life cycle

Continuous refinement of interface-prioritized backlog across programs

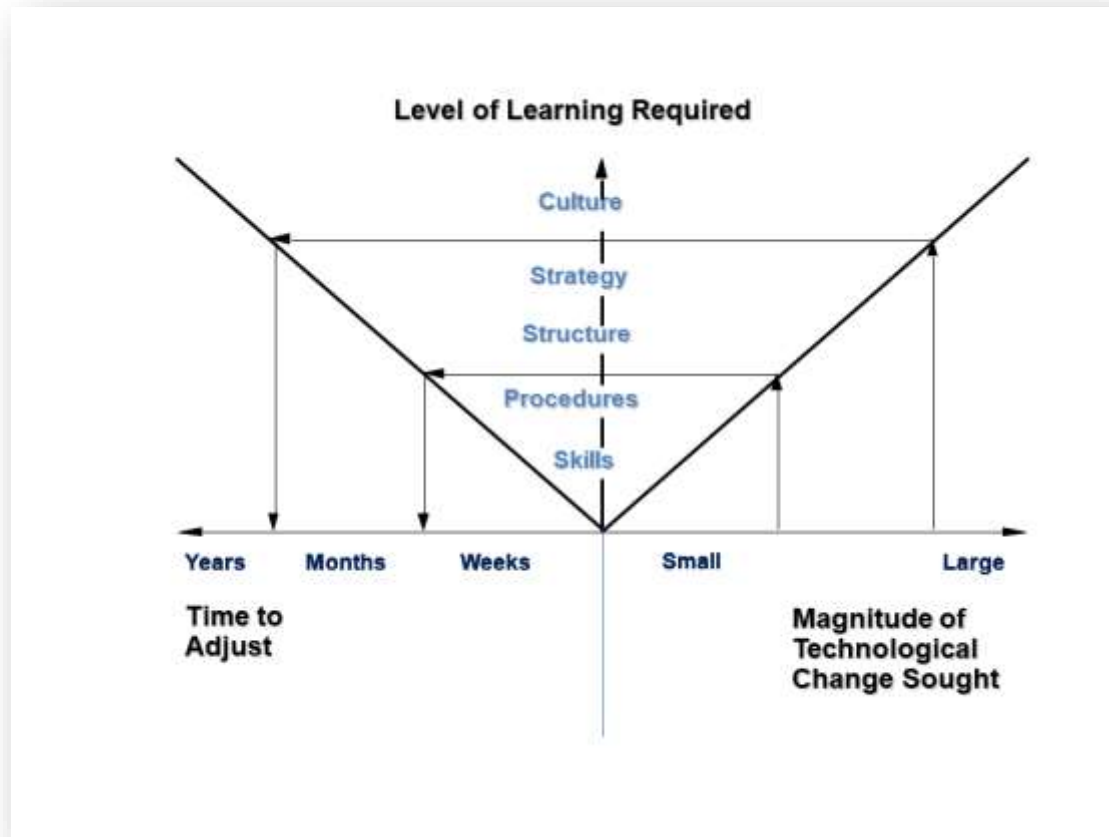
Requirements & interface documents coordinated at infrequent designated acquisition points

Decisions/reqmts constantly validated with data from implementation

Decisions/reqmts periodically validated with analysis until data from implementation available (late)

Culture Aspects of Adopting New Practices

Culture Change will take Time and Effort!



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

Some Advantages We Want to Leverage

Key Enablers for Agile Adoption

Acquisition Processes

- Collaborate: industry, acquirers, and users
- Enabling changes
- Rapid contract action
- Acquiring developer services vs product

Culture and Policies

- Small teams
- Fail fast / Learn fast
- Delegated decisions
- Review SW, not docs
- Continuously improve
- More execution rigor

User Involvement

- Active users involved
- High bandwidth comm
- Demo interim sprints
- Provide ops insights
- Prioritize requirements

Program Structure

- ~6-12 month releases
- Tailor acq processes
- Stakeholder buy-in
- Empowered teams
- Small iterative releases

Aligning Priorities

- Align program docs, processes, contracts
- Leverage loosely coupled architecture
- Rethink reviews

Agile Training

- Requires experienced gov't and contractors
- Invest in training team
- Coaches working with PMO to implement
- When to use Agile

© 2016 The MITRE Corporation and Carnegie Mellon University. All rights reserved.



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

Some Barriers We May Run Into

Barriers to Agile Adoption

Acquisition Processes

- Long timelines
- Fully defined requirements upfront
- Contract mods costly

Culture and Policies

- PMOs struggle to tailor acquisition processes
- Change = risk
- Significant oversight

User Involvement

- Limited engagements
- Few end-users available
- Serial requirements process (ops → tech)
- Limited demos late

Program Structure

- Up-front fixed scope
- Locked requirements
- Too detailed cost est.
- APB, EVM management
- Changes discouraged

Aligning Priorities

- Many stakeholders w/ competing priorities
- Conflicting developer direction, interpretation
- Disrupts team progress

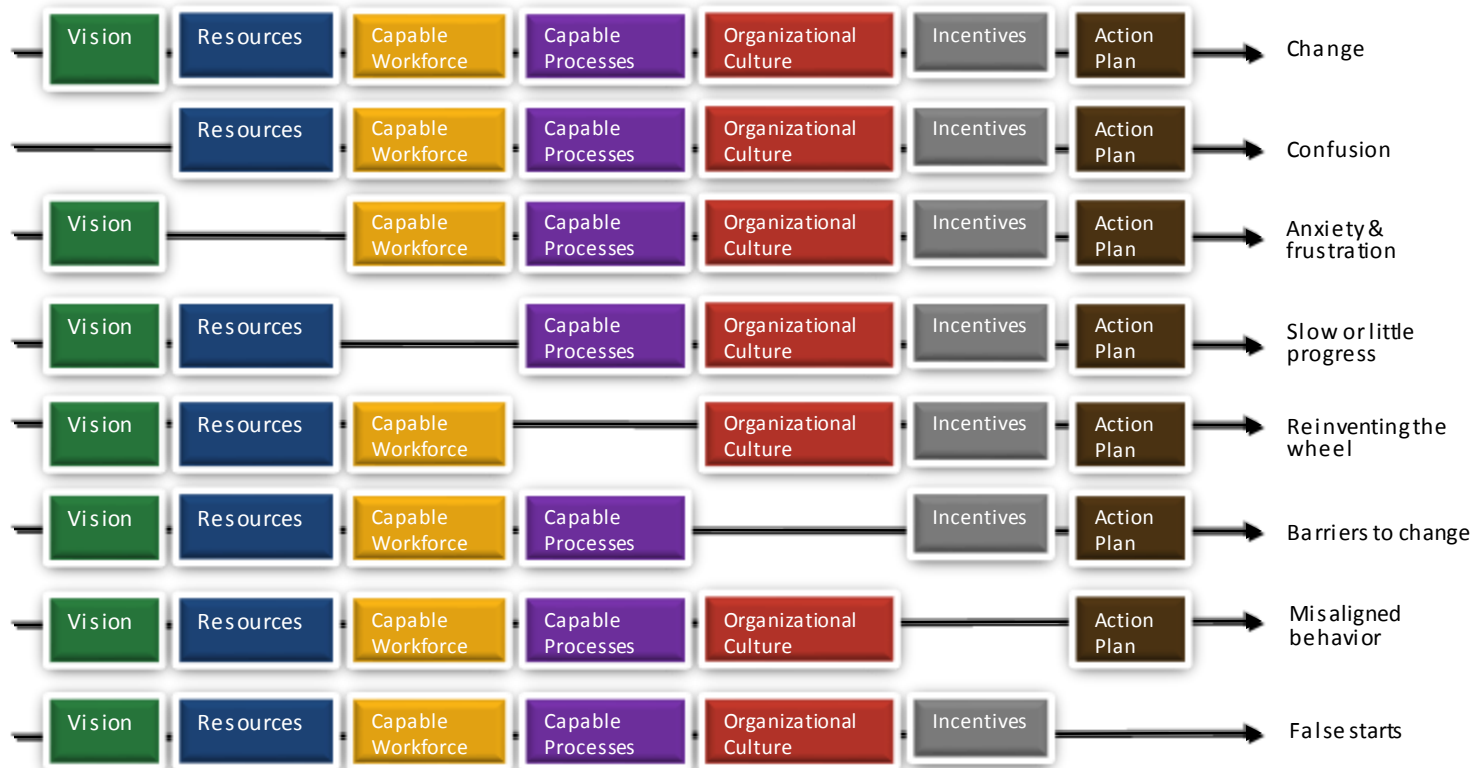
Agile Experience

- Limited insight and experience in Agile in gov't, defense industry
- False claims of Agile
- Need for leadership, culture, process, staff

© 2016 The MITRE Corporation and Carnegie Mellon University. All rights reserved.



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



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

Summary

Agenda

Product & PMO Context for Lean/Agile

Relevant Lean/Agile Concepts for MC-130J's Context

Lean/Agile Alone Won't Solve Your Integration Problems

Programmatic Constructs and Roles in Lean/Agile to Consider Implementing Now

Culture Aspects of Adopting New Practices

Summary



Respond at: www.pollev.com/mainsummit799



What topics/concepts were useful to you in this workshop?

Respond at: www.pollev.com/mainsummit799



What focus areas or topics would you like to learn/discuss in a future workshop?

Welcome to your Lean/Agile Journey!

Incremental, iterative ways of working are here to stay in DoD and in industry

Lots of information today

- But we're just hitting the tip of the iceberg!
- Lots of opportunities for learning!



A Few Relevant Resources

- ❖ USAF Chief Software Officer: <https://software.af.mil/>
- ❖ 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>
- ❖ Agile/Lean Self-Serve Learning - US Space Force Space Systems Center
- ❖ Some of the materials in the learning package came from this site. If you have a CAC, you can access a large set of curated materials on your own
- ❖ [https://www.milsuite.mil/wiki/Portal:AtlasX Agile Self-serve Learning Paths](https://www.milsuite.mil/wiki/Portal:AtlasX_Agile_Self-serve_Learning_Paths)
- ❖ Software Engineering Institute Agile Resources: podcasts, blog posts, Technical Notes on many Agile in Government topics: <https://www.sei.cmu.edu/go/agile>
- ❖ Mik Kersten Project to Product <https://projecttoproduct.org/>
- ❖ Wikispeed <https://wikispeed.com/>

Contact Information

Brigid O'Hearn
Senior Member of the Technical Staff
Software Engineering Institute

bpohearn@sei.cmu.edu

SuZ Miller
Principal Researcher
Software Engineering Institute

smg@sei.cmu.edu

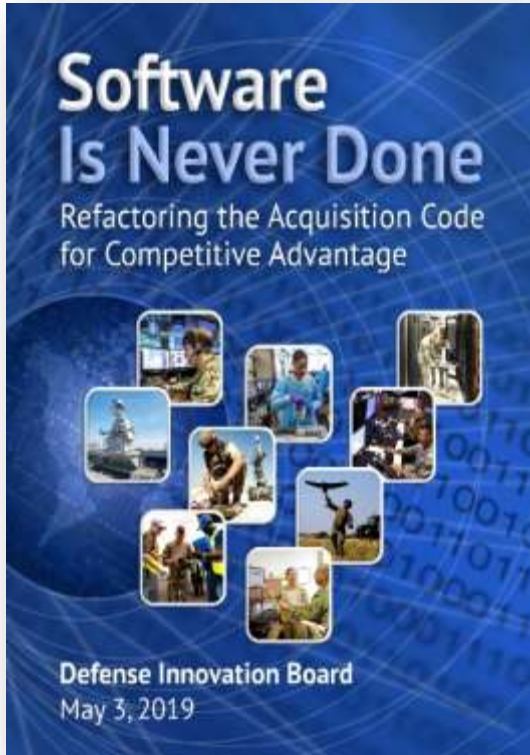
THANK YOU!

BACKUPS

2019 DIB SWAP Study

Strong influence on the 2019 and 2020 National Defense Authorization Acts

Also home of the “Agile BS” Appendix

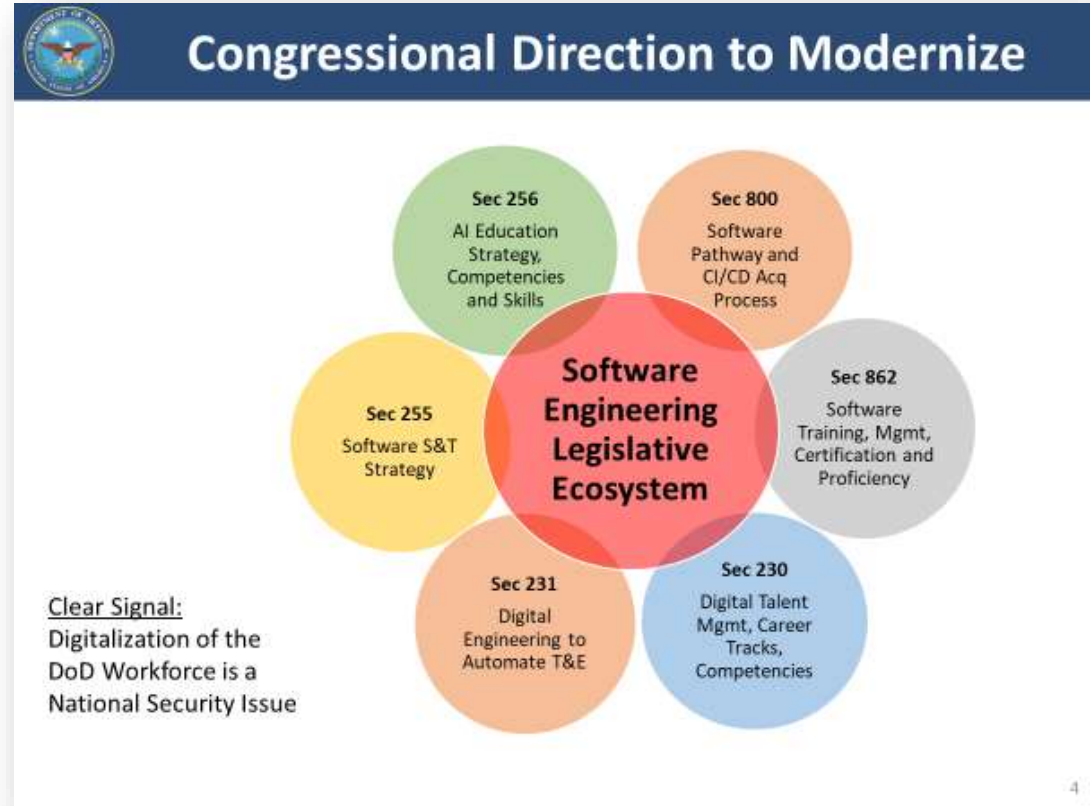


Main lines of effort:

- **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.

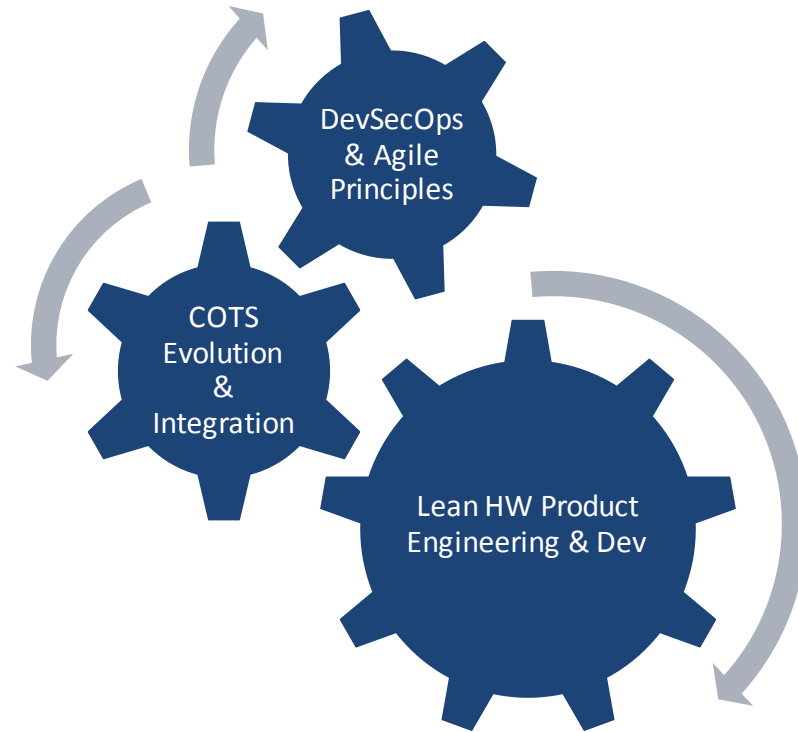
Source: <https://innovation.defense.gov/software/>

Multiple Sources of SW Engineering Legislative Direction



Source: Brady, Sean. *How Software Acquisition & DevSecOps Increase the Lethality of the DoD*, DSO Days, Oct 2020.

3 Sources of Wisdom for Integration of Complex Cyberphysical Products into a Single Platform



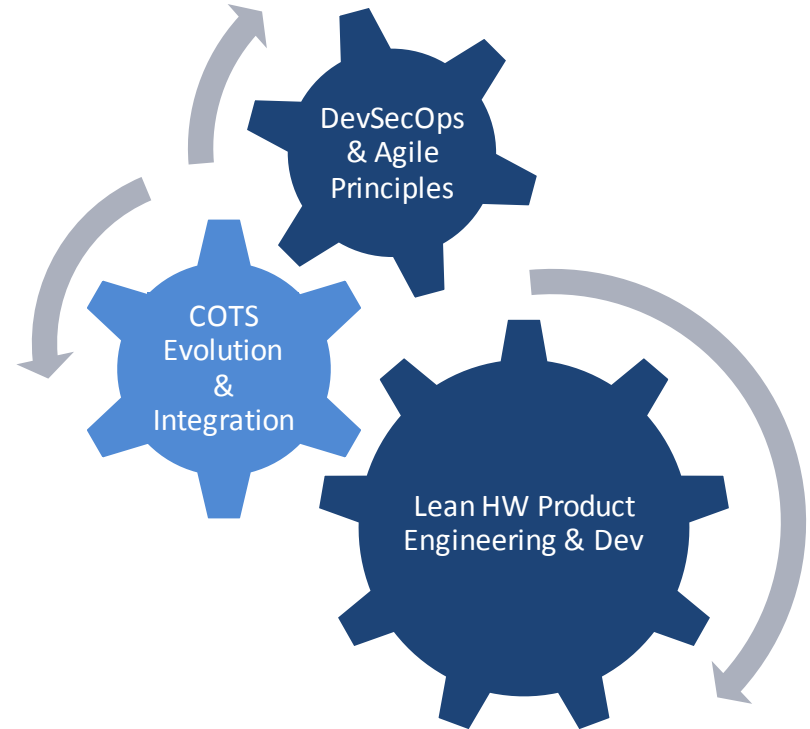
COTS Evolution and Integration

COTS integration into existing platforms are similar to a market eco-system of COTS vendors who all need to integrate with each other on known commodity hardware

Interface standards by accepted external organizations that all are governed by in some way evolve with the participation of all the interested vendors

Examples/Publications:

- Overview of SEI COTS-Based Systems
<https://resources.sei.cmu.edu/library/asset-view.cfm?assetid=485899>
- Evolutionary Process for Integrating COTS-Based Systems (EPIC)
<https://resources.sei.cmu.edu/library/asset-view.cfm?assetid=6053>



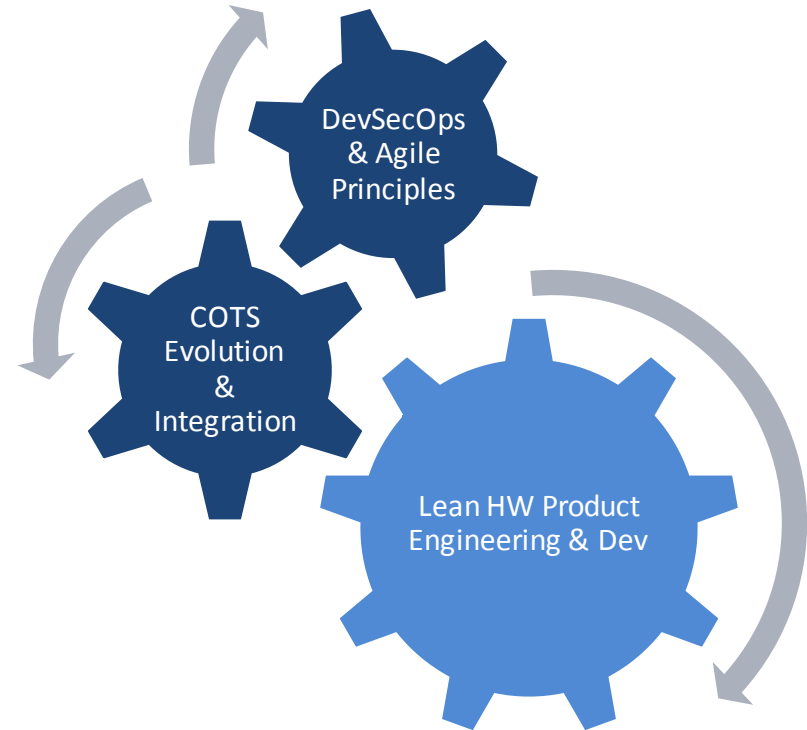
Lean Hardware Product Engineering and Development

Integration of “contracted” interfaces/components into existing platforms

Extreme modularity and/or use of interface standards (MOSA/OMS)

Examples:

- Iterative developed automobiles: Wikispeed car, BMW Group Leipzig factory
- German shipbuilding cruise ships with military ships being built side-by-side

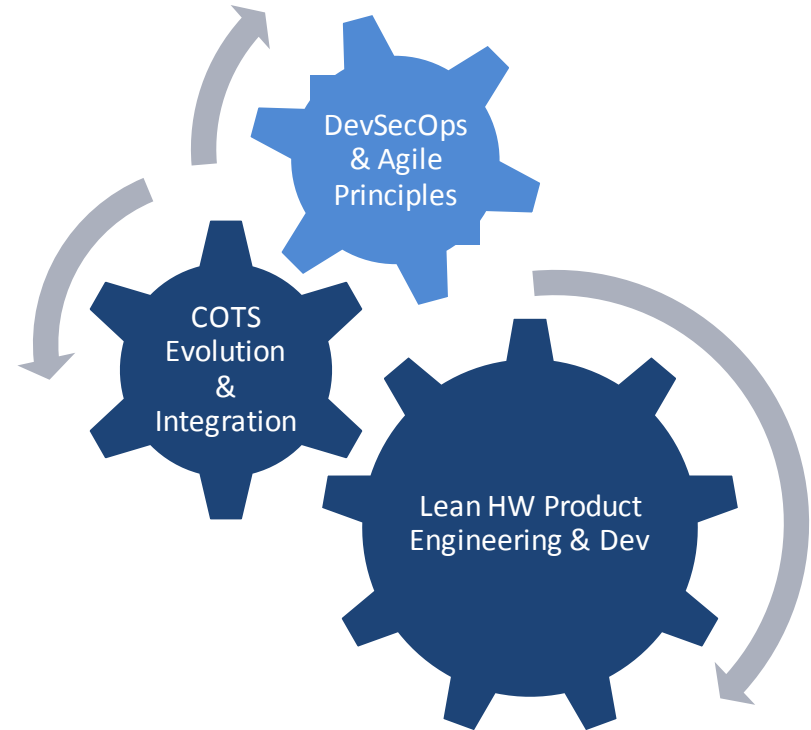


DevSecOps & Agile Principles

Lightning fast, data driven approach to verification and validation of small increments (back to extreme modularity) of functionality and interfaces

Collaborative small team environments with scaled “shared consciousness” through constant, transparent information exchange

- Across Program Office and Contractor boundaries
- Recognizing, honoring, and evolving shared interface standards as data from iterative implementation is acquired



Batch Size

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”

Feedback

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

Requirements

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