Lean/Agile Support for Multi-Program Product Integration MC-130J CR-2M TLM

Brigid O'Hearn
5 May 2022

Software Engineering Institute
Carnegie Mellon University
Pittsburgh, PA 15213

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.

DM22-0396

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

How big of a change do you think Lean/Agile/DevSecOps is for MC-130J?

Objectives

At the end of this mini-tutorial, you should be able to:

- Describe relevant concepts of Lean/Agile and DevSecOps to apply in your Program and PEO
- Describe relevant concepts from COTS and Lean Hardware Product Engineering/Dev to apply in your Program and PEO
- Begin the discussion of how to move from current state to a productive Integration Eco-System

Product & PMO Context for Lean/Agile

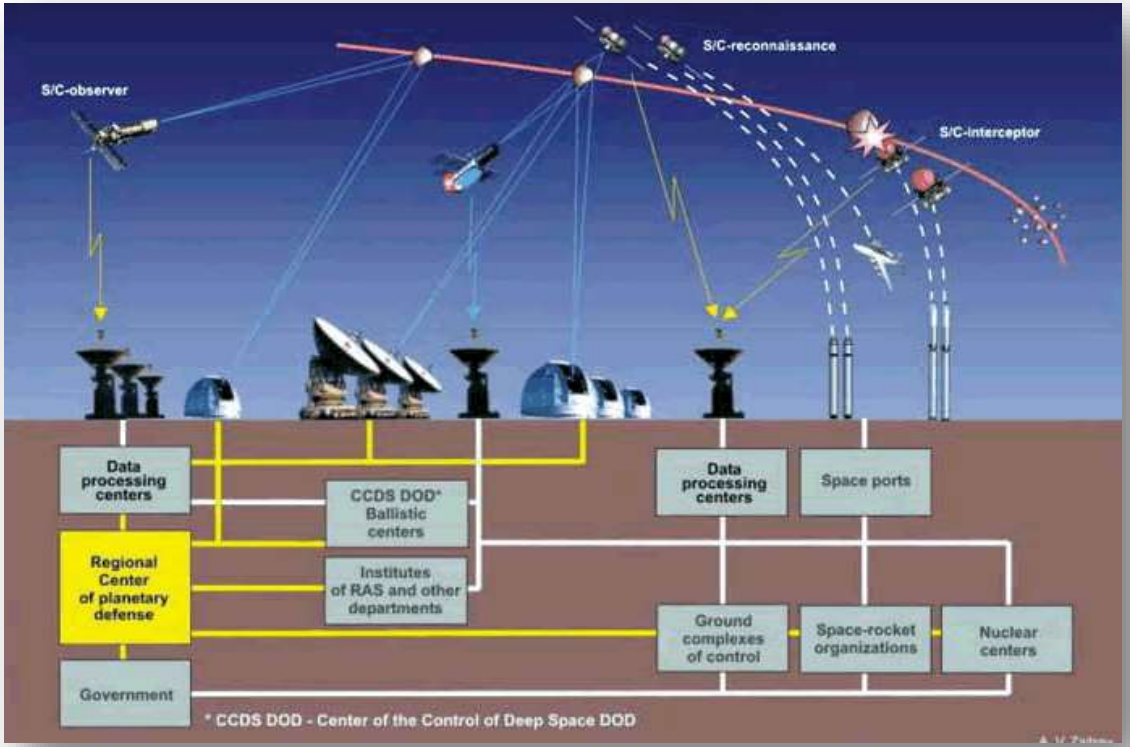
Agenda

- 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

Product & PMO Context for Agile

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

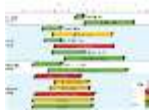
Our implementation of any approach, including Lean/Agile, *must account for our context*



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 for MC-130J?

Agile Alone

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

Agile + Lean

- Adds typical hardware, system, and business/ management teams
- Adds principles and practices that reflect the larger complex system context inherent in SOCOM
- Adds consideration of stakeholders like DT/OT and certification authorities

Agile + Lean+ DevSecOps

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

What word or phrase summarizes MC-130J's reasons for incorporating Agile, Lean, and DevSecOps into our work?

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

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

Relevant Concepts from Lean/Agile for MC-130J

Agile Acquisition Pathways

Agile in the Larger DoD Context

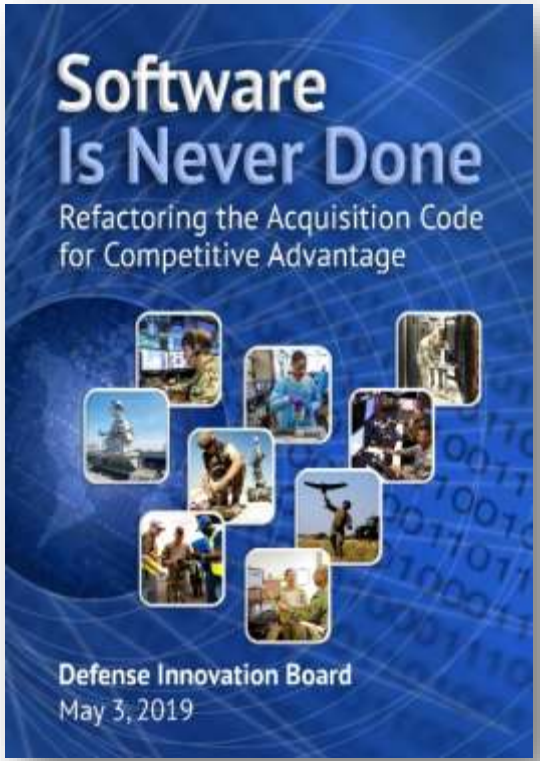
DoD continues to track anecdotal success and failures of programs using Agile

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

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

2019 DIB SWAP Study



Strong influence on the 2019 and 2020 National Defense Authorization Acts

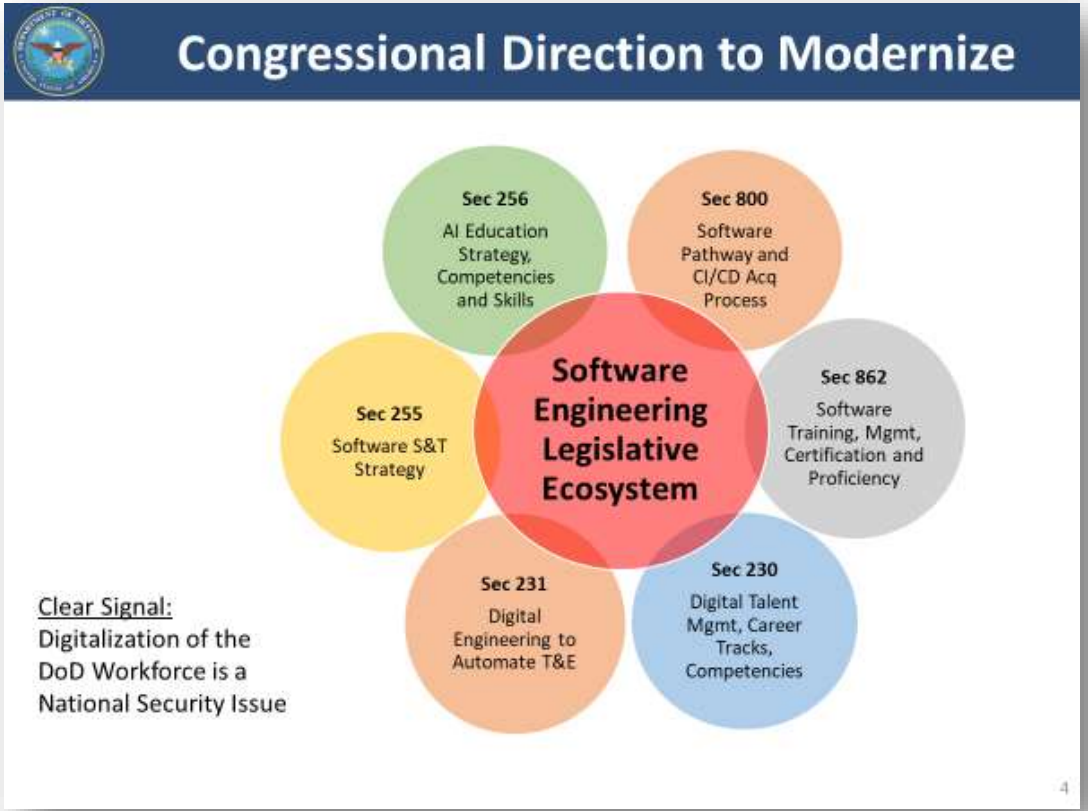
Also home of the “Agile BS” Appendix

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

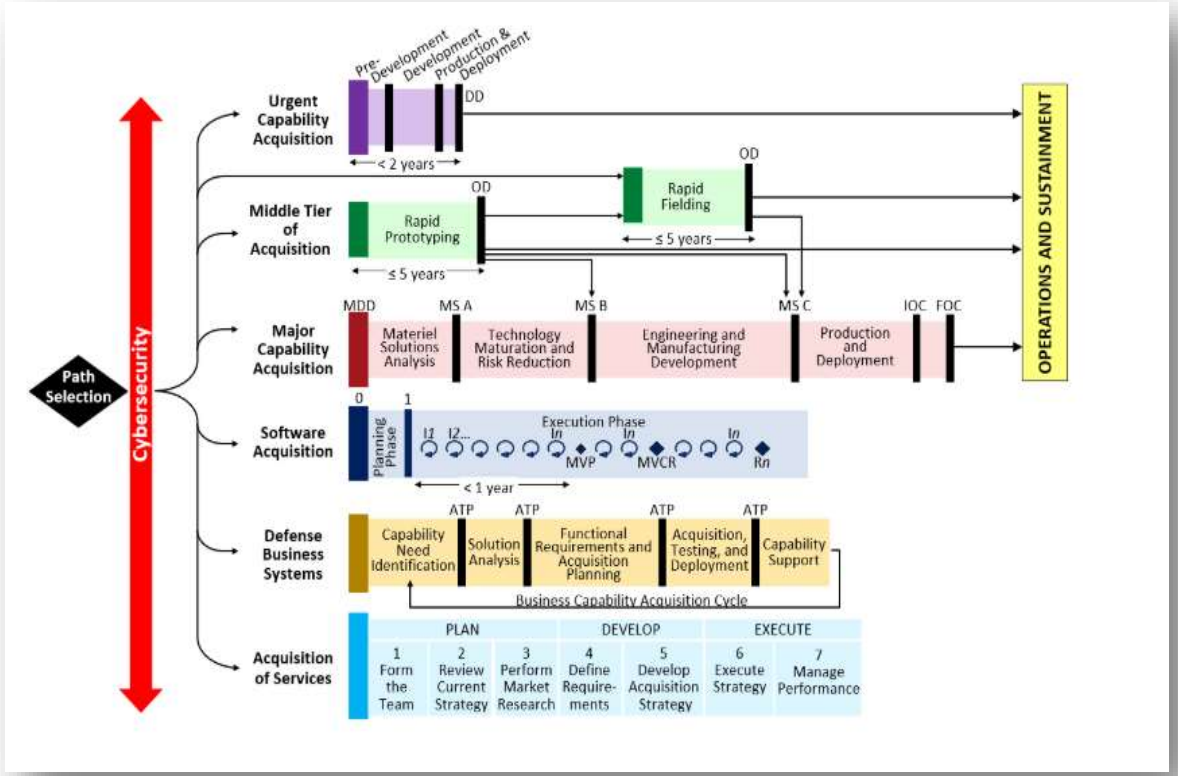


Source: Brady, Sean. *How Software Acquisition & DevSecOps Increase the Lethality of the DoD*, DSO Days, Oct 2020.

Adaptive Acquisition Framework

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

https://aaf.dau.edu/



Key Elements of SW Acquisition Pathway

- Modern software development practices (Agile, DevSecOps, Lean)
- Capitalizing on active user engagement and enterprise services
- Software is rapidly and iteratively delivered to the operational environment to meet the highest priority user needs
- Tightly coupled mission-focused government-industry software teams
- Automated tools for development, integration, testing, certification

Source: DODI 5000.02 Section 4.2

10

If these resonate and more information is desired, we can follow-up with a SWP 101 briefing

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

Relevant Concepts from Lean/Agile for MC-130J

What is Agile, Anyway?

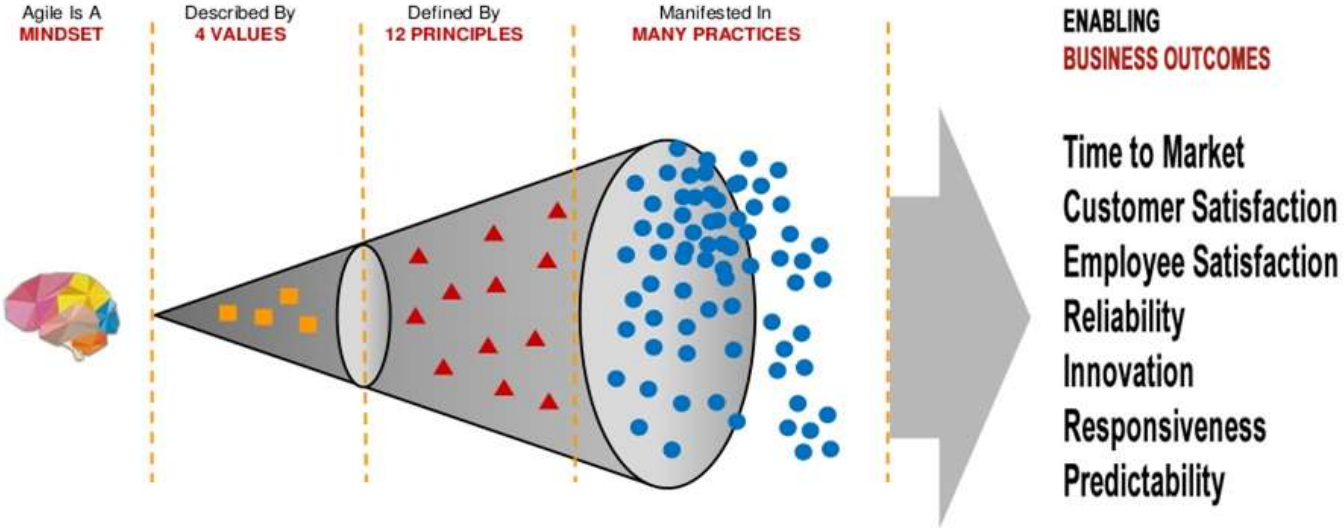
Reminders and Relevant Interpretations

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

What is Agile?



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

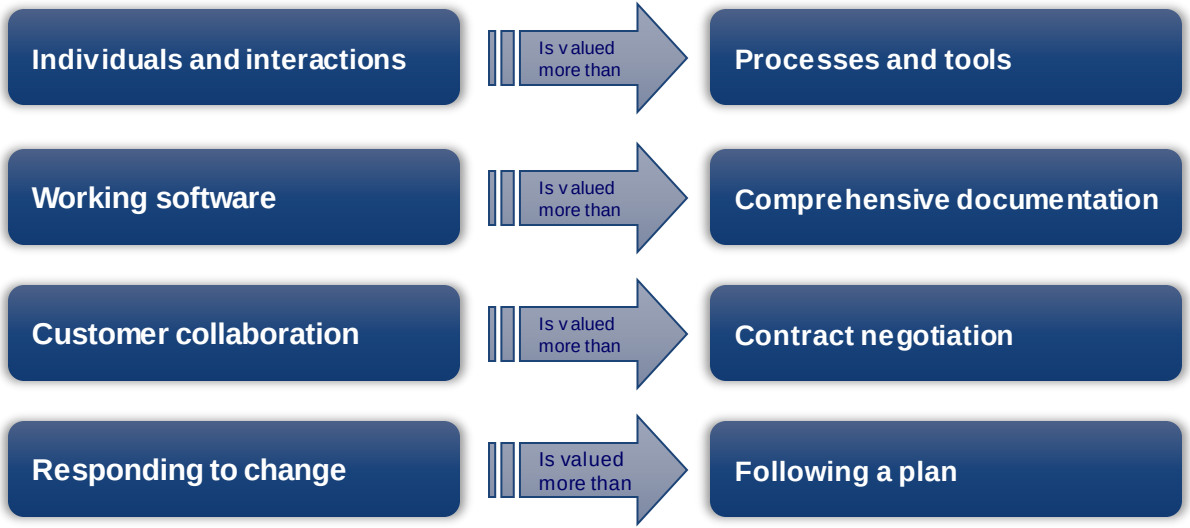
Is NOT Agile!

See DIB SWAP Agile BS

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 Values from Manifesto for Agile Software Development

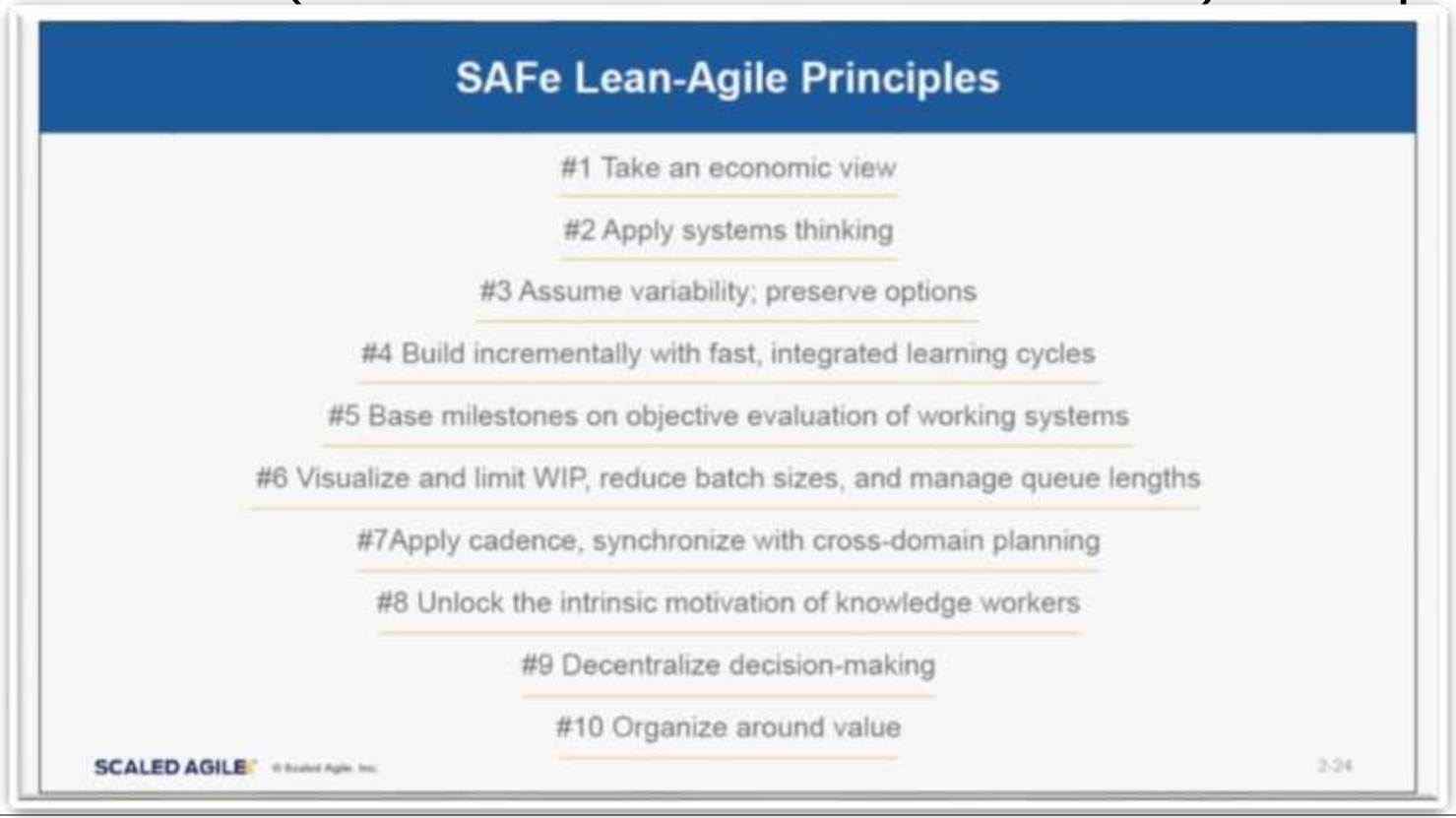
While there is value in the items on the right, **we value the items on the left more.**



Which side do you think will benefit your users more?

http://agilemanifesto.org

SAFe Lean-Agile Principles:
For the Entire (Govt + Contractor + Stakeholder) Enterprise

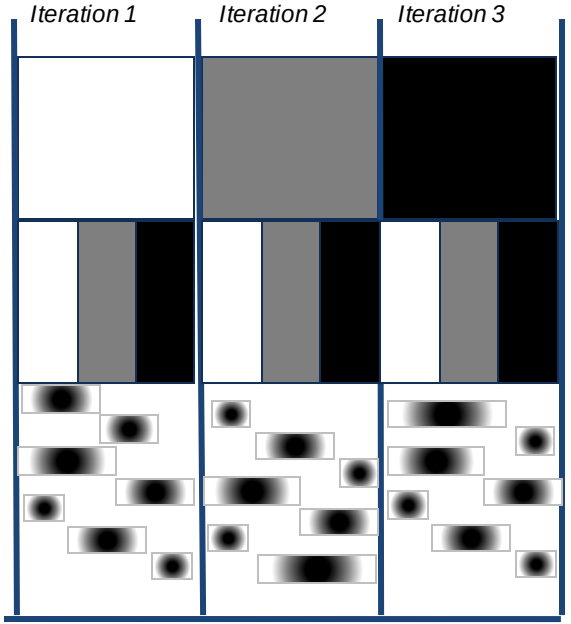


Taking an Iterative Approach

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

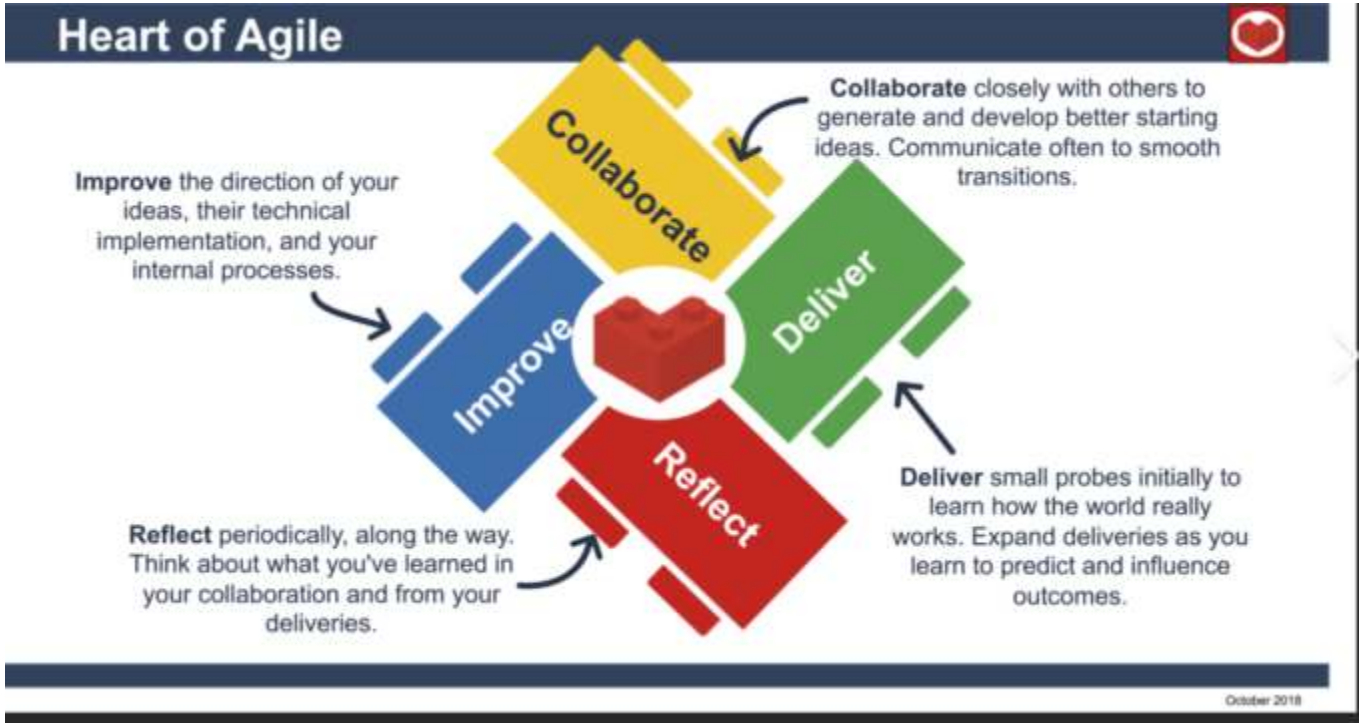


Start with Teams Using Common Concepts But Allow Methods That Suit Their Context

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

Agile Team Method Commonalities

- 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

All Common Agile Team Methods are Based on PDCA Cycle

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).

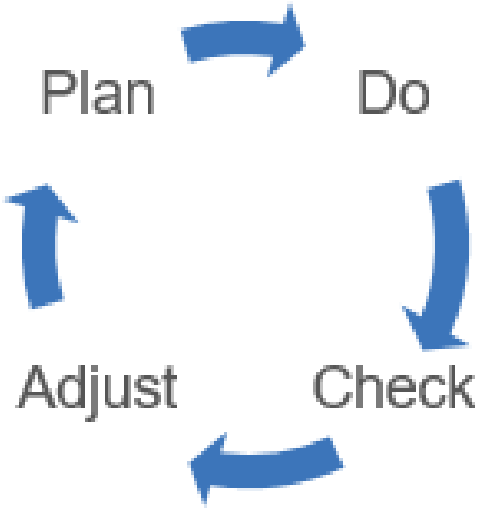
- Implement the change you decided on in the plan phase.

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.

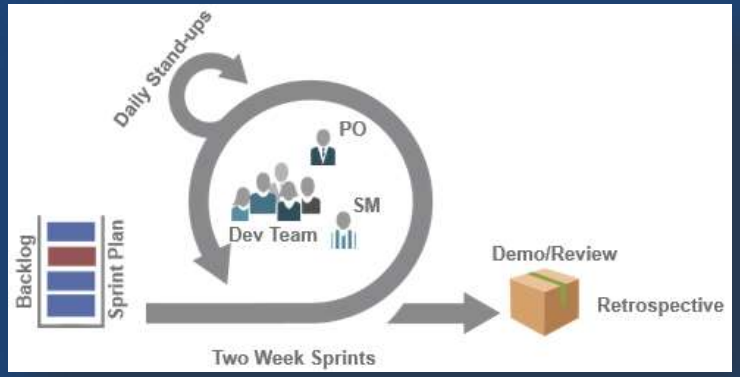
- Adopt the change, abandon it, or run through the cycle again.



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

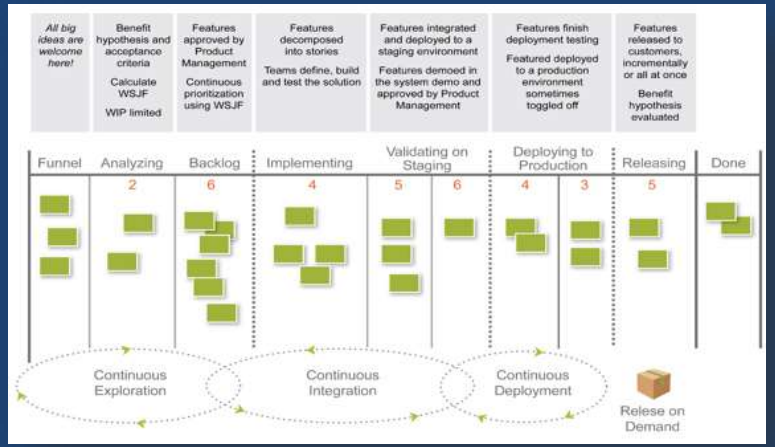
Choose One or More Methods to Fit Your Team Context

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

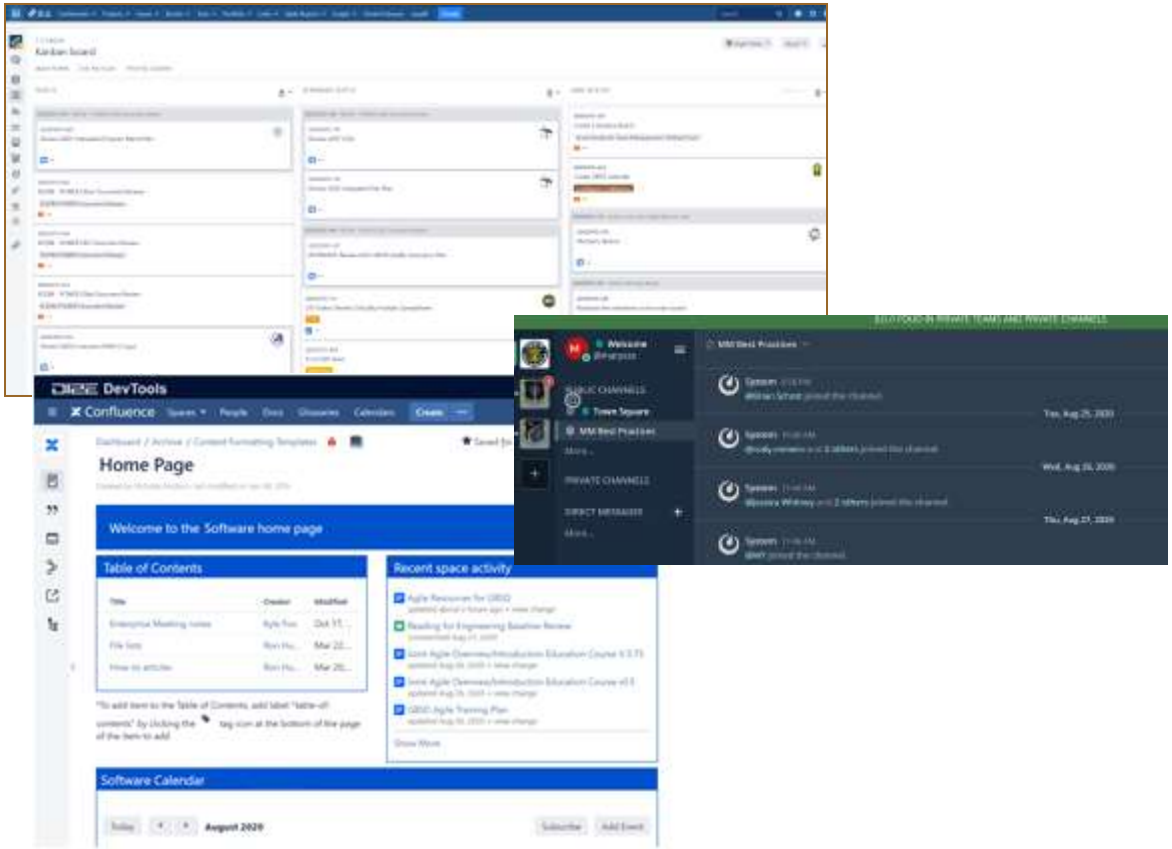
Goal: Common Tools Used Across Different Team Types in DOD and Industry

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

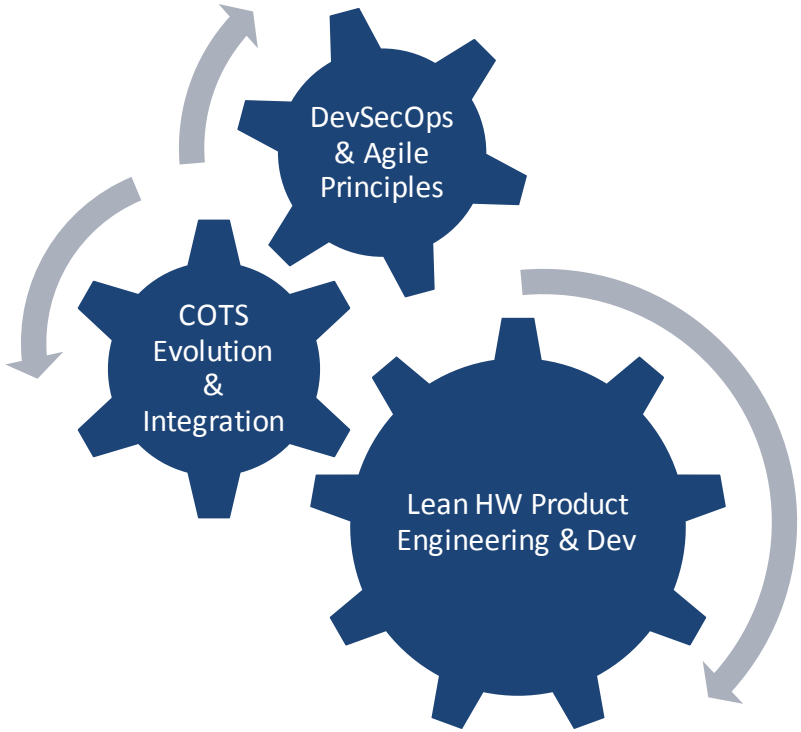


Lean/Agile Alone Won't Solve Your Integration Problems

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

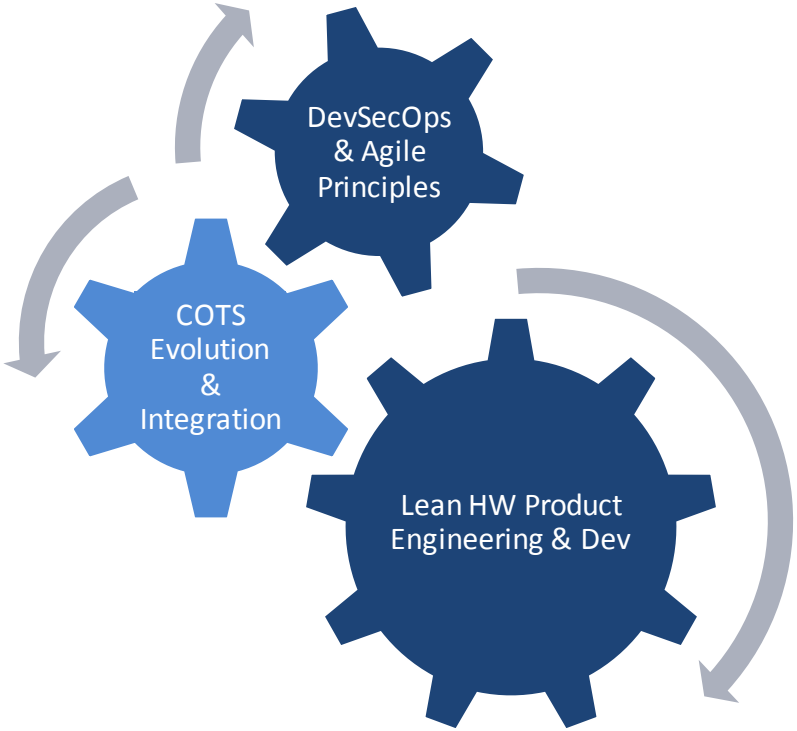
3 Sources of Wisdom for Integration of Complex Cyberphysical Products into a Single Platform



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

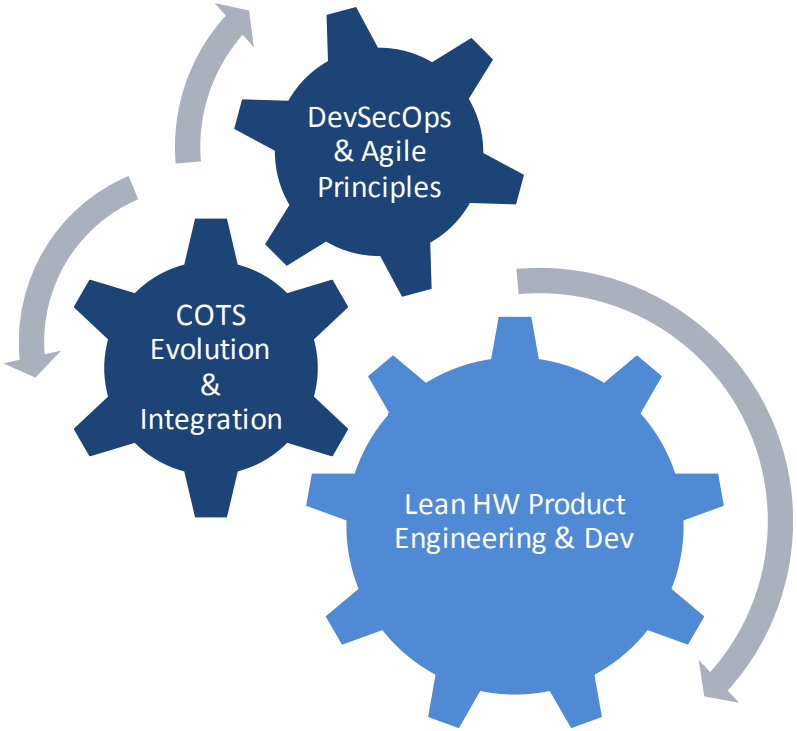
Lean Hardware Product Engineering and Development

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

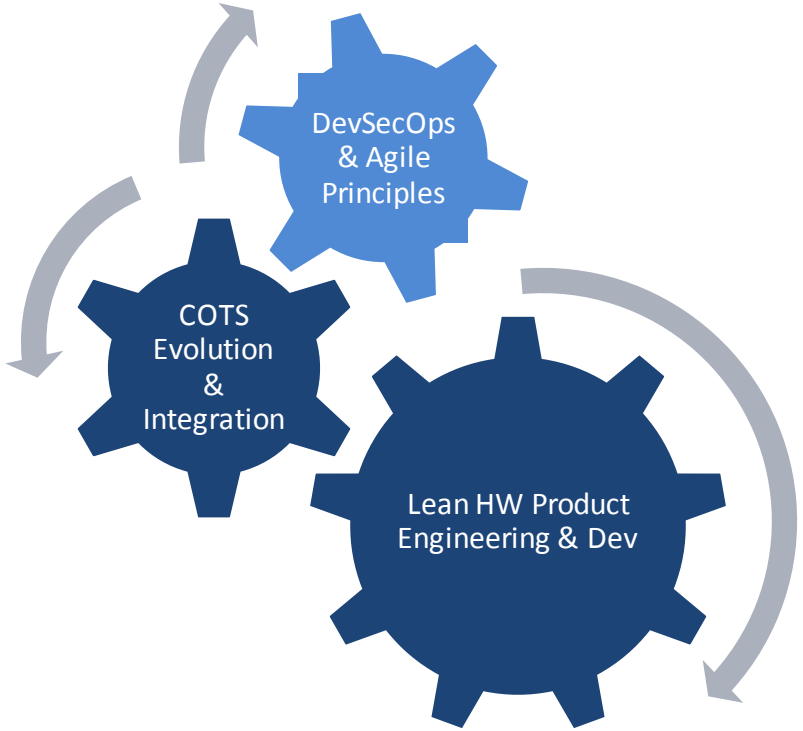


DevSecOps & Agile Principles

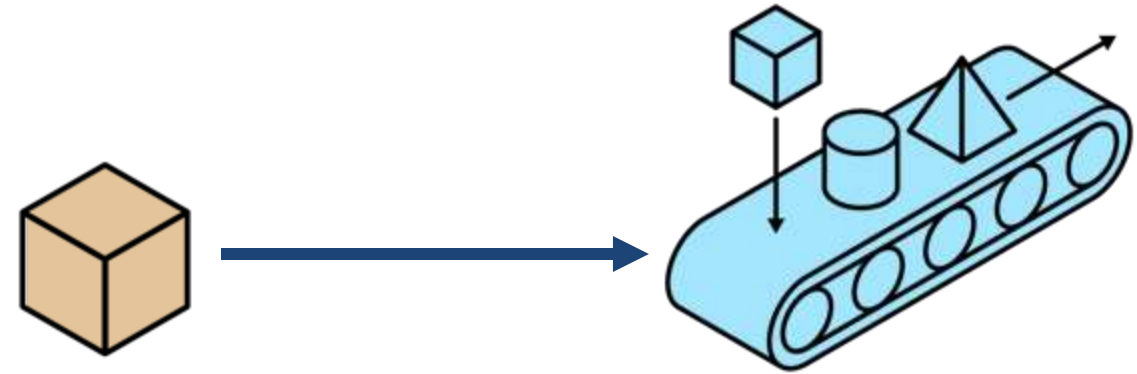
Lightning fast, data driven approach to verification and validation of small increments (back to extreme modularity) of functionality and interfaces

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



Agile Represents a Major Requirements Transition – Different Acquisition Objectives

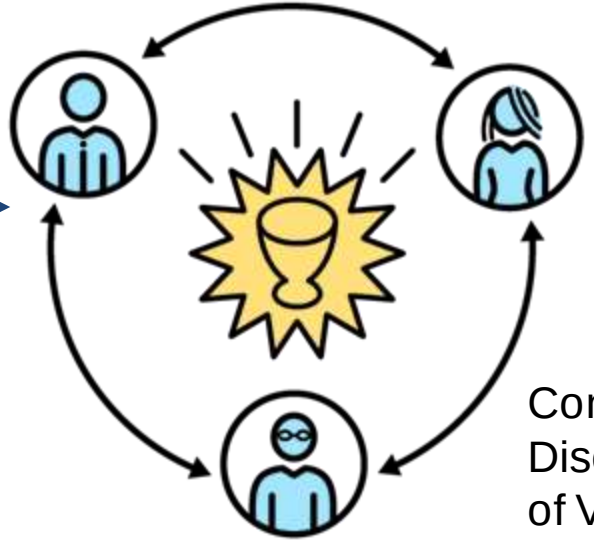


Buying a Box

Buying an ongoing delivery stream

Agile Represents a Major Requirements Transition – Different Vendor Interactions

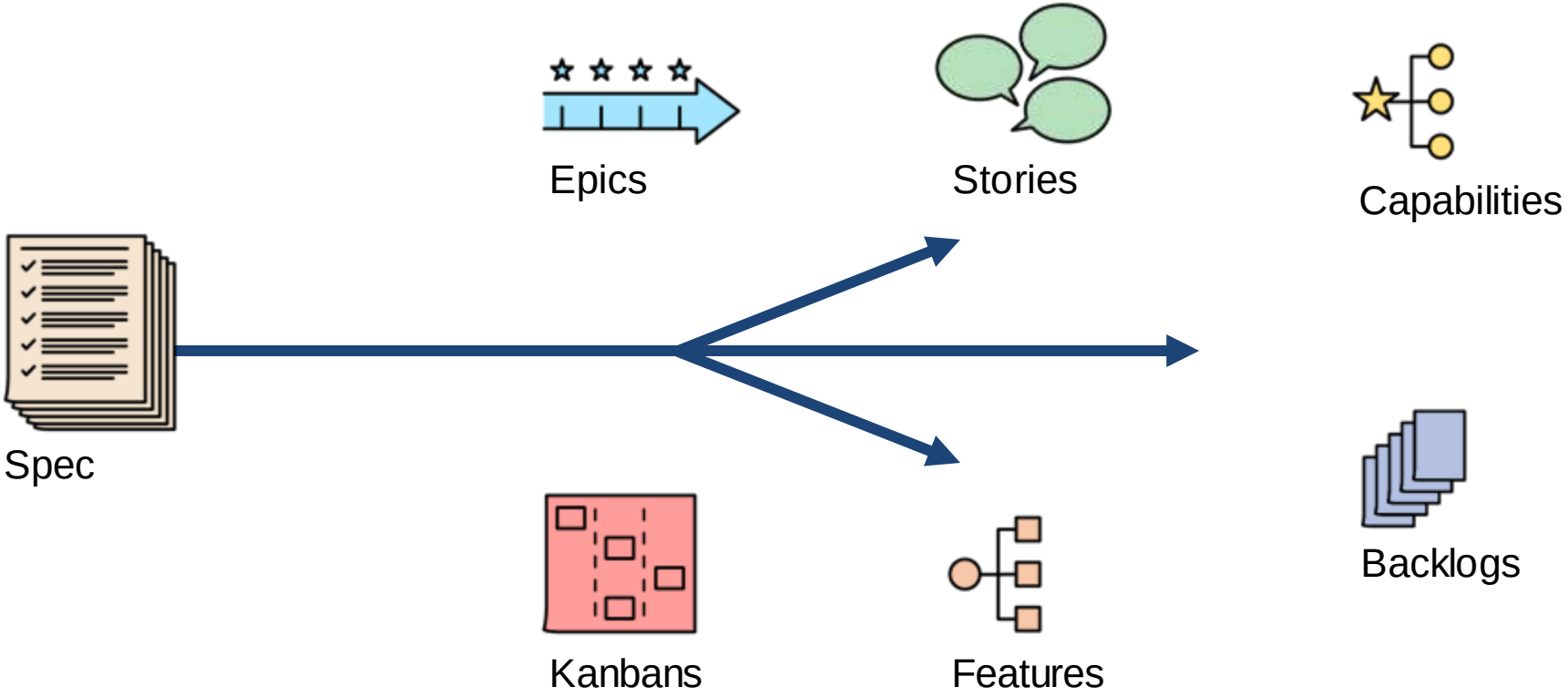
Do This



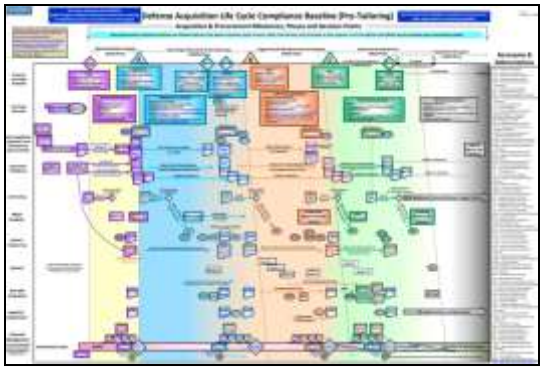
Constant Collaboration

Continuous Discovery of Value

Agile Represents a Major Requirements Transition – Different Artifacts



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

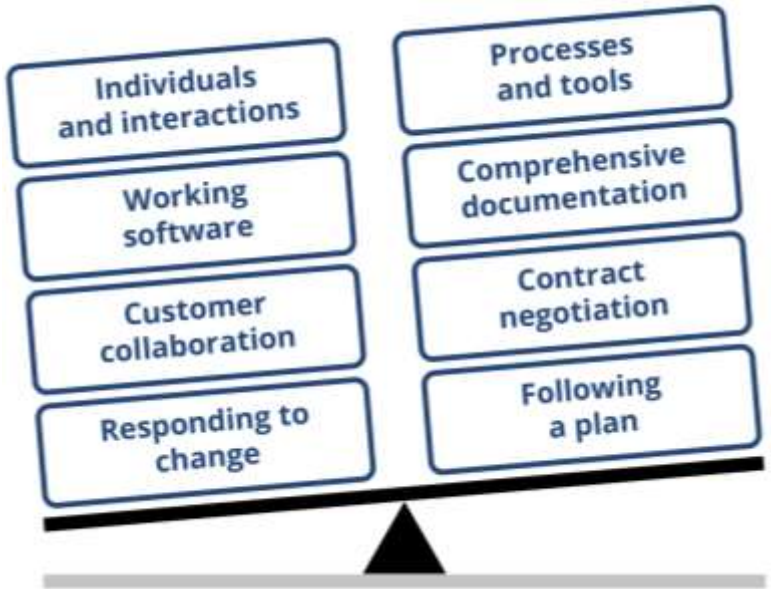
Programmatic Constructs and Roles in Lean/Agile to Consider Implementing Now

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

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

Through this work we have come to value:



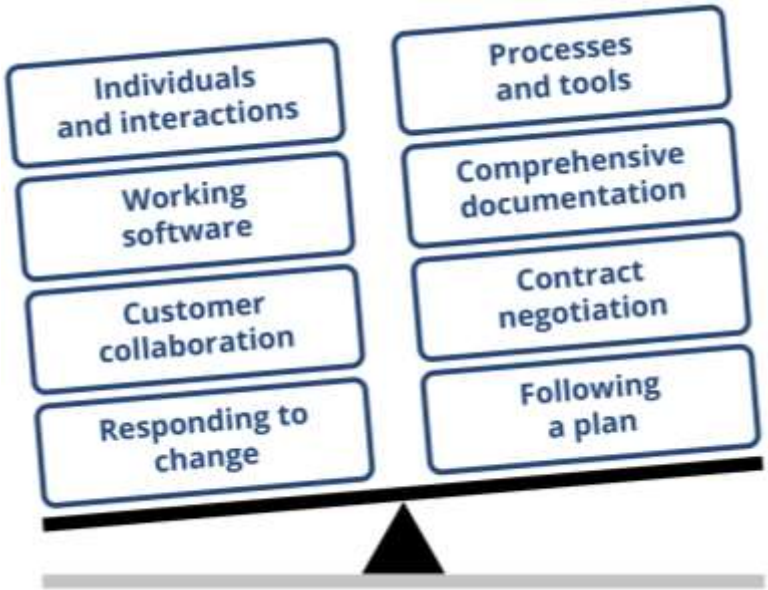
That is, while there is value in the items on the right, we value the items on the left more.

https://agilemanifesto.org/history.html

Agile	Traditional
Many small batch interactions	Few large batch interactions
Demos/User feedback	Primarily Documentation review
Continuous Backlog Refinement	Single Delivery Requirements Document
Seeking Insight	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)

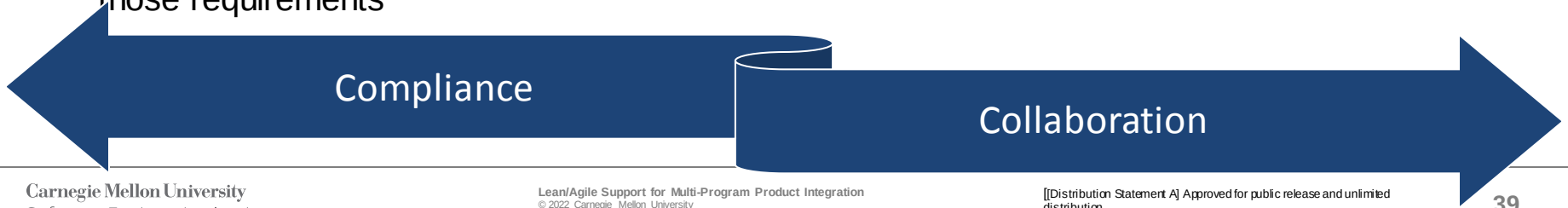
Bottom Line for Programmatics of Complex Systems

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 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***



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:

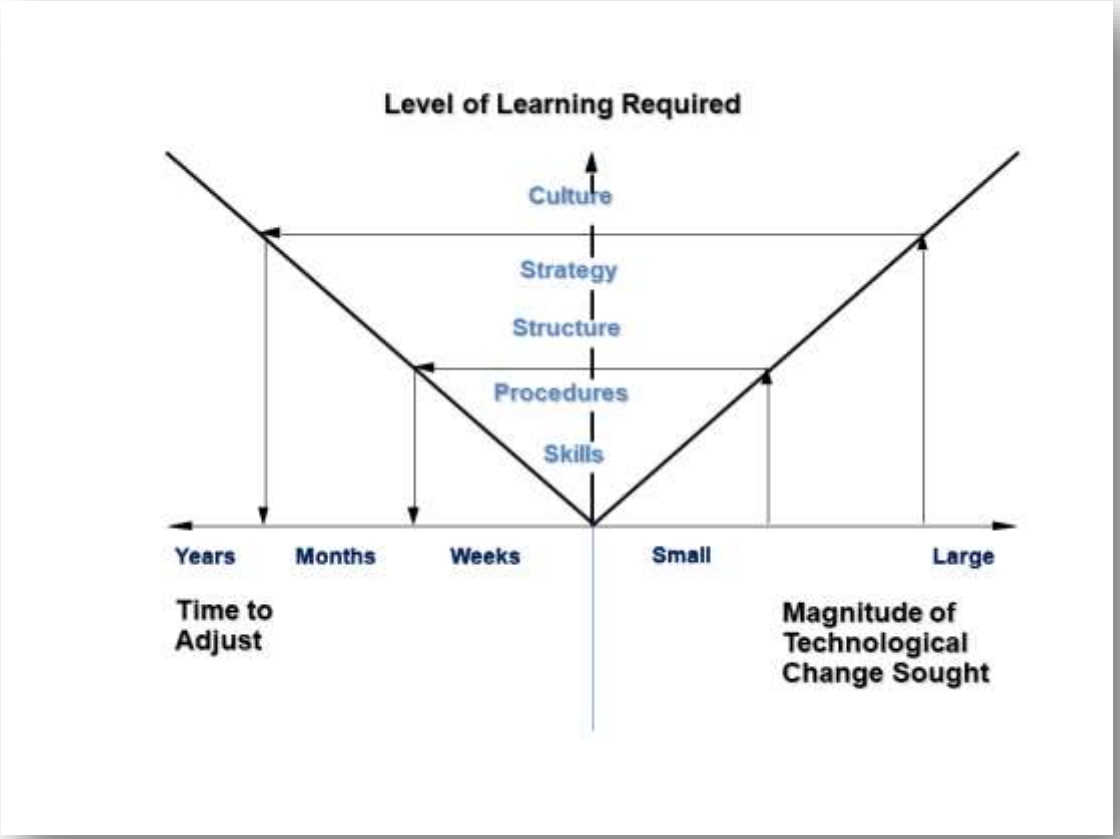
- Mix of "push" and "pull" communication across govt/contractor interface as requirements are elaborated/refined
  - Facilitated by workflow and collaboration tools
- Frequent high bandwidth meetings keep the relationship going, not just technical work
- Transparency among stakeholders is an essential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

Culture Aspects of Adopting New Practices

Agenda

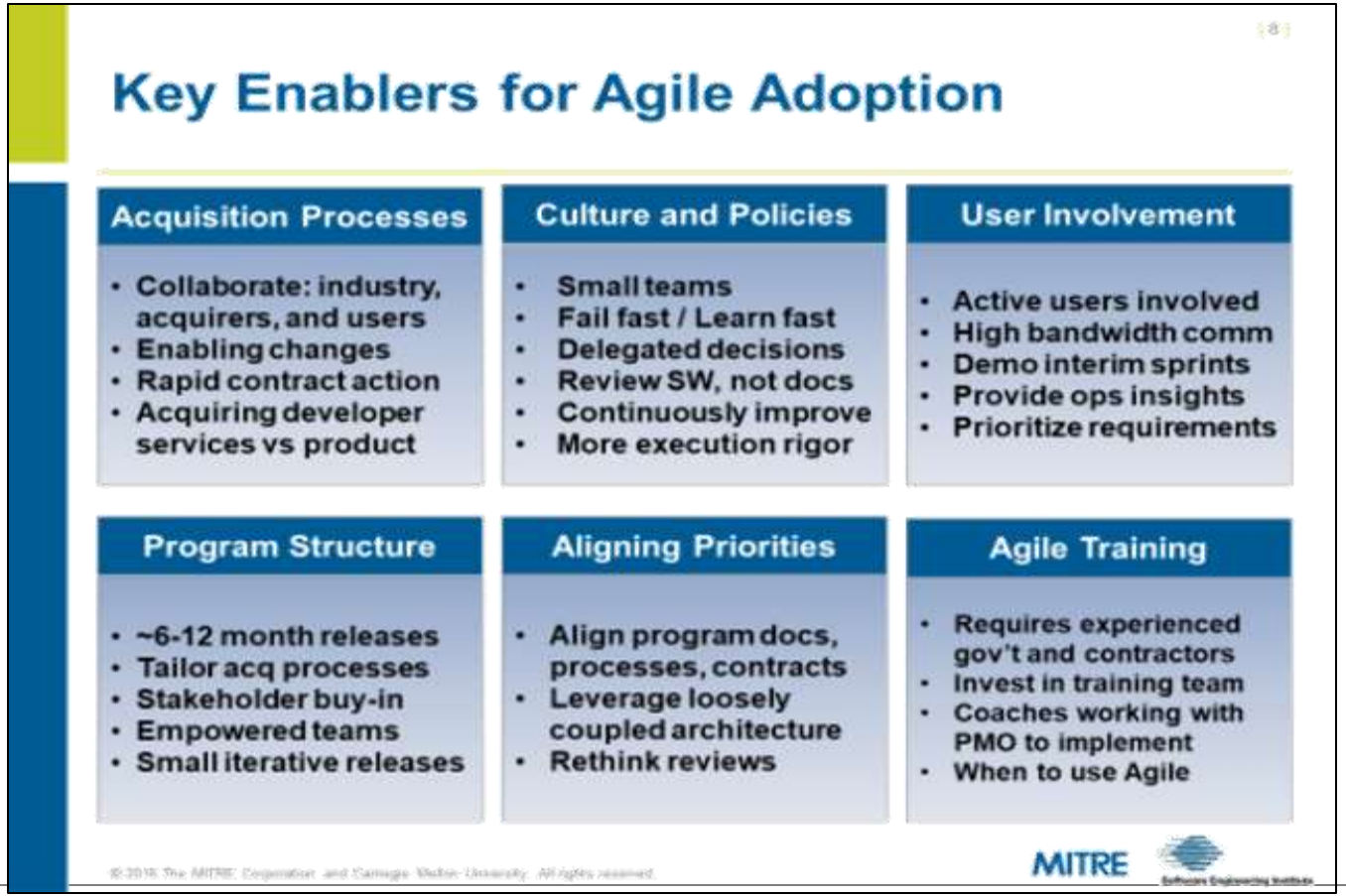
Product & PMO Context for Lean/Agile
Relevant Lean/Agile Concepts for MC-130J's Context
Lean/Agile Alone Won't Solve Your Integration Problems
Programmatic Constructs and Roles in Lean/Agile to Consider Implementing Now
Culture Aspects of Adopting New Practices
Summary

Culture Change will take Time and Effort!



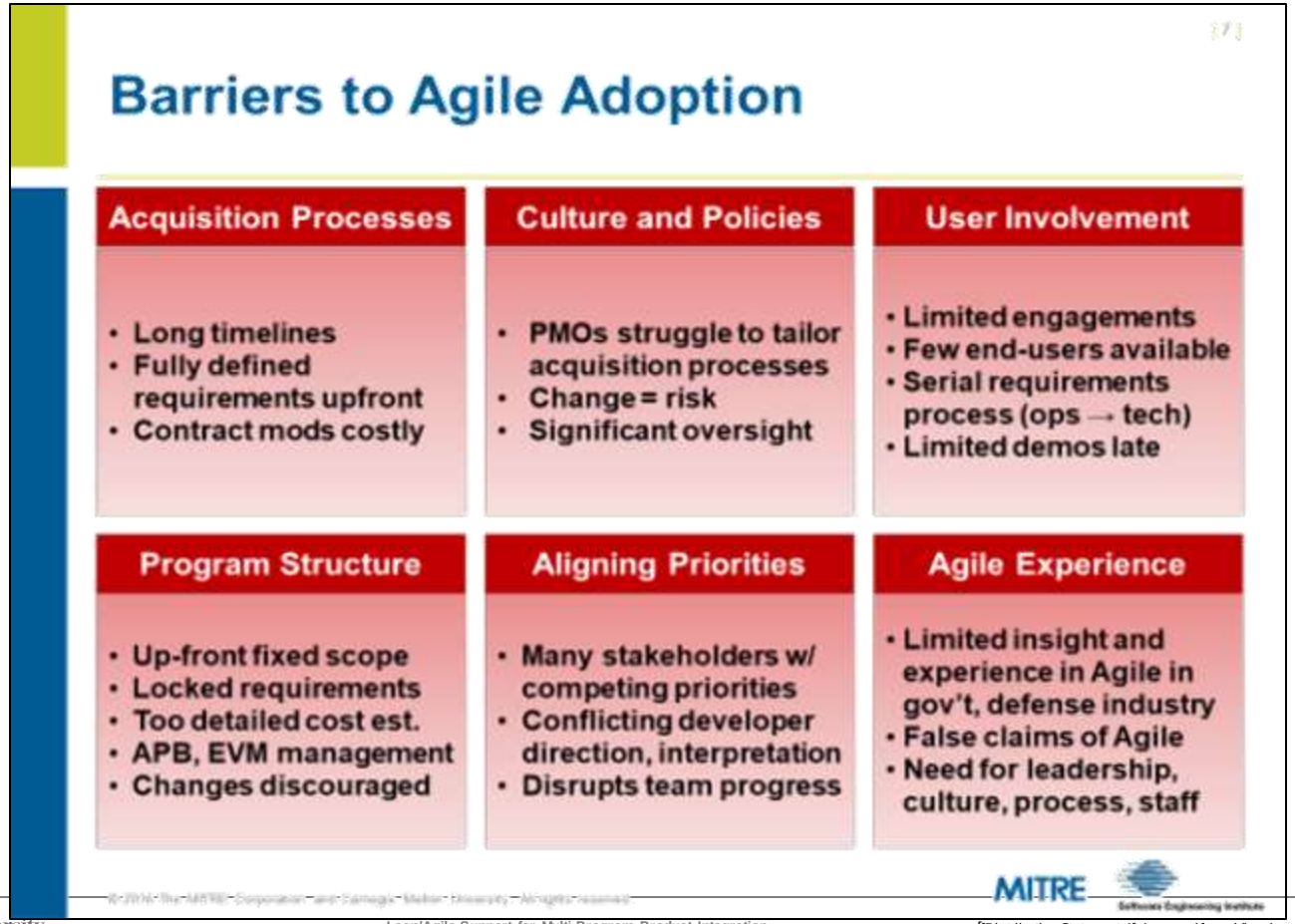
Source: Adler, Paul. "Adapting Your Technological Base: The Organizational Challenge", Sloan Mgmt Review, 1990.

Some Advantages We Want to Leverage



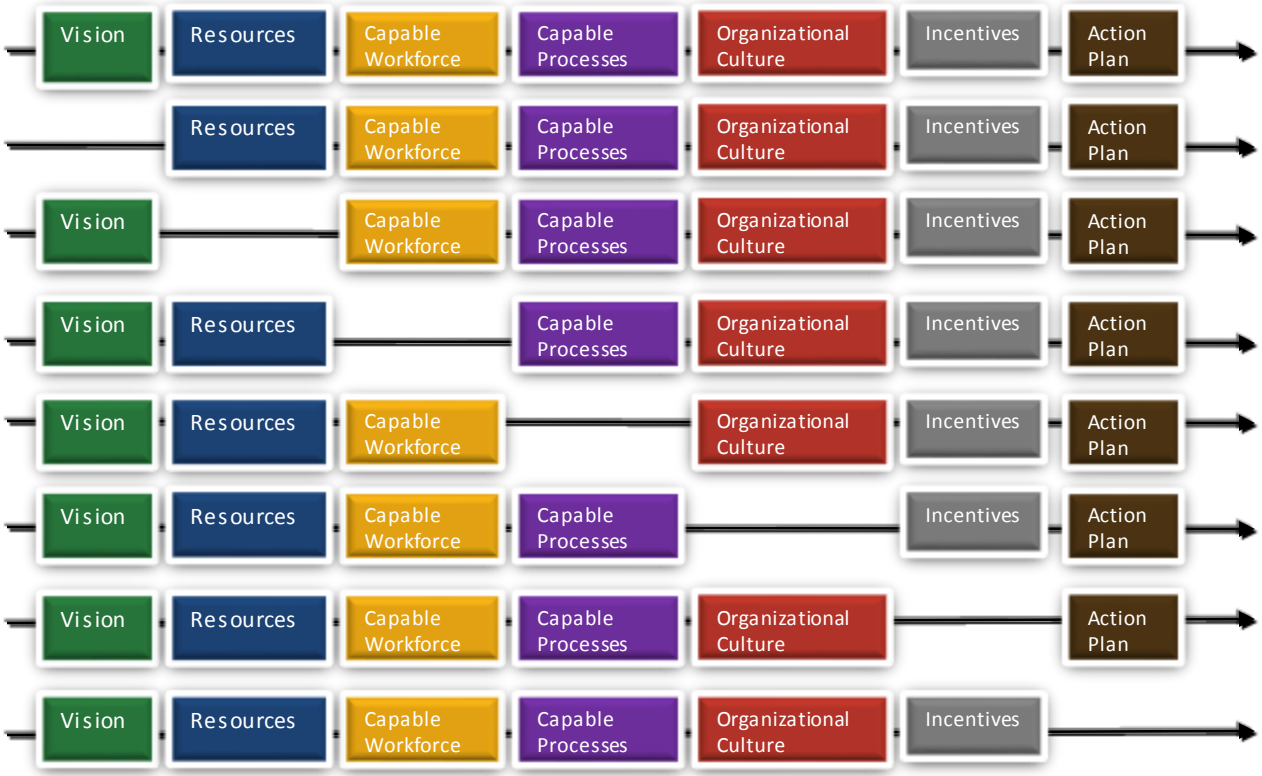
Source: 2016 briefing to General E. Pawlikowski on USAF Agile Adoption, SEI & Mitre

Some Barriers We May Run Into



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

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

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!

How big of a change do you think Lean/Agile/DevSecOps is for MC-130J?

What do you perceive as MC-130J's most difficult barrier to adoption of Agile, Lean and DevSecOps?

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
- 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