Agile in Government: Executive Overview

April 2021

SuZ Miller, Principal Researcher

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.

DM21-0344

Agenda

Today's landscape

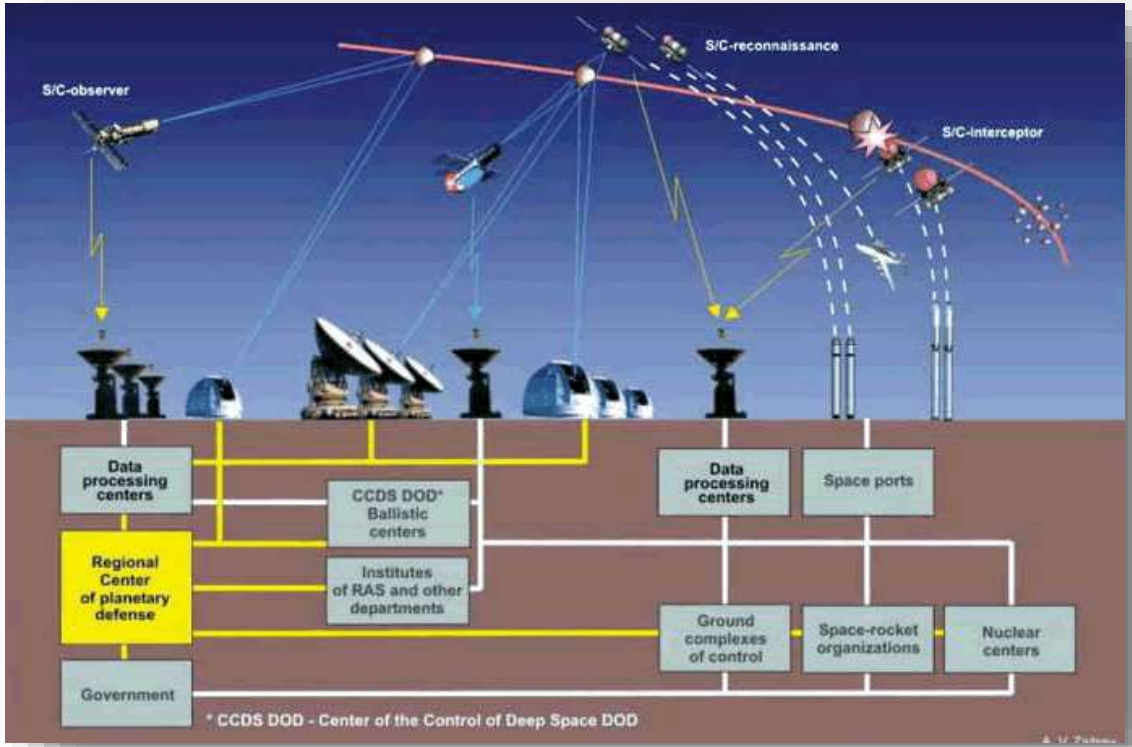
Agile basics: Meaning behind the vocabulary

Beyond the small team: Agile in the larger ecosystem

How do we get there: Enabling Agile culture

Training Systems PMO Context for Agile

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



A “Generic” CyberPhysical System Depiction

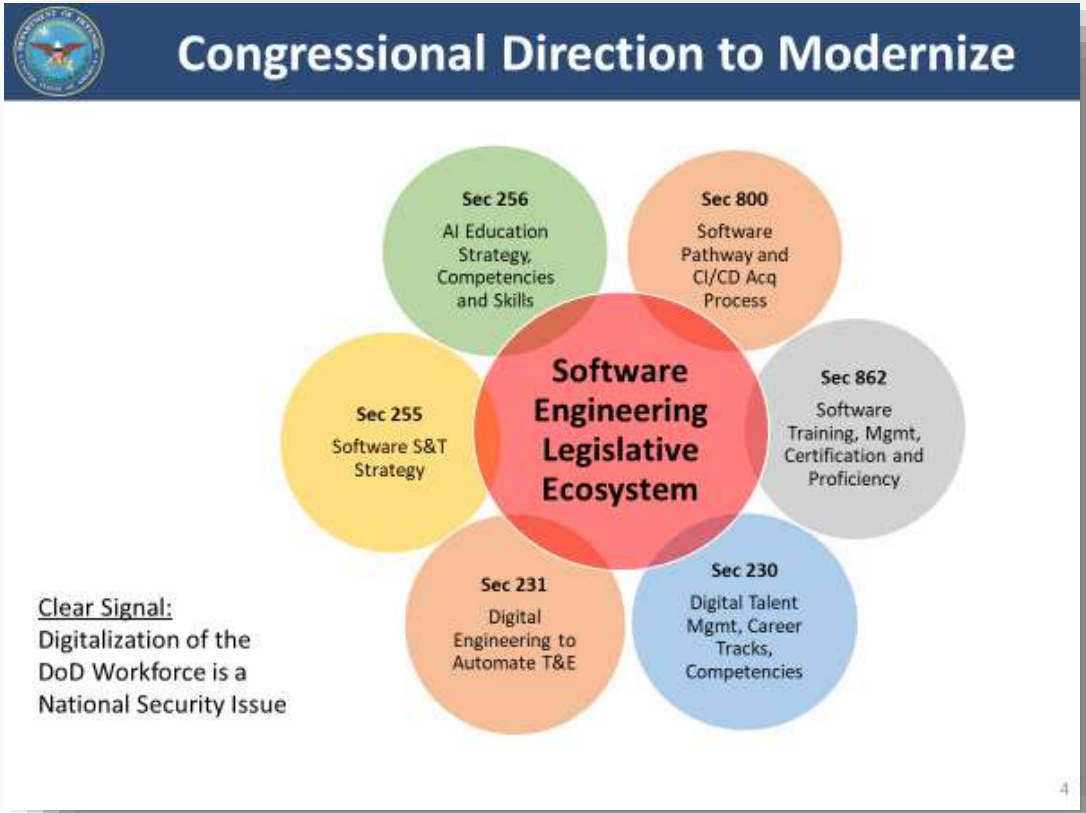
Why Agile (and Lean and DevSecOps) at F-35?

Direction from OSD 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)

Multiple Sources of SW Engineering Legislative Direction



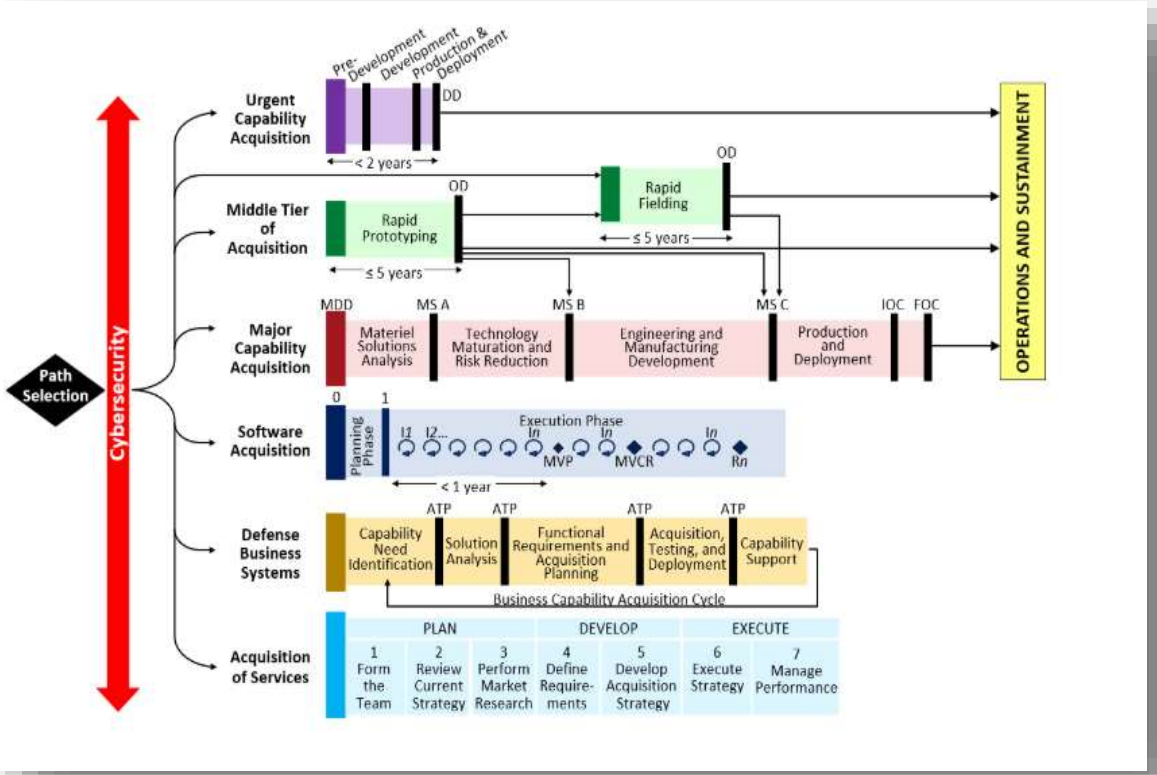
Source: Brady, Sean. *How Software Acquisition & DevSecOps Increase the Lethality of the DoD*, DSO Days, Oct 2020.

Adaptive Acquisition Pathways

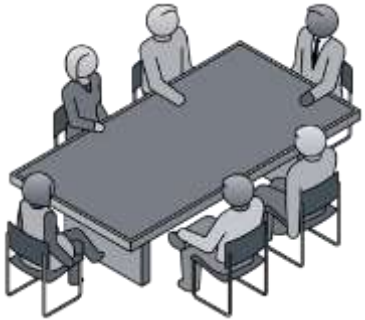
Adaptive Acquisition Pathways: New ways of acquiring software-dominant systems (some of the guidance can also be applied to cyberphysical systems that are software-reliant)

https://aaf.dau.edu/

You may not be using these yet, but it's likely that you will be using one or more in the future!



Attributes of Agile Success in Government Organizations



- **Top Cover**
- **Permission to "fail fast" (learn fast)**
- **Training in Agile**
- Use of Agile coach
- Enough up-front system and software architecture
- Willing to work collaboratively across government/ contractor boundary
- Willing and open to adopt new modes of operation
- Dedicated staff

All of these need support from leadership to "stick"

Agile in Today's Landscape: Summary

(adapted from SEI Testimony to House Ways and Means Social Security Subcommittee)

Agile is an iterative approach to software delivery that builds and delivers software incrementally from the start of the project, instead trying to deliver it all at once near the end.

- Early opportunity for course correction, especially when the environment changes after a program has begun
- Early risk reduction, especially in user-facing areas of the system
- Shorter "idea to realization" cycle resulting in fast user feedback for future increments of functionality

But it's about more than software engineering to do it right: Needs business/acquisition process support

Oversight: Responsibility for oversight and due diligence doesn't change; approach to oversight in an Agile setting does. Some examples:

Flow: Predictable delivery volume, deployment speed	**Engagement:** stakeholder involvement
Quality: Defect backlog	**Risk:** Deferred complexity

The FAR/DFARS encourage bold innovation – the culture has a long way to go

Agile in Today's Landscape: Summary (contd.)

(adapted from SEI Testimony to House Ways and Means Social Security Subcommittee)

Agile will not solve all the complex problems associated with software-dominant systems acquisition and sustainment efforts

- But it has contributed significantly to successful efforts (both in IT and weapons systems)

Benefits from using Agile methods only manifest when the developer and acquisition efforts are aligned

Government obligations in oversight must change when Agile is the focus of development

- SEI has observed negative consequences in organizations that do not address these changes.

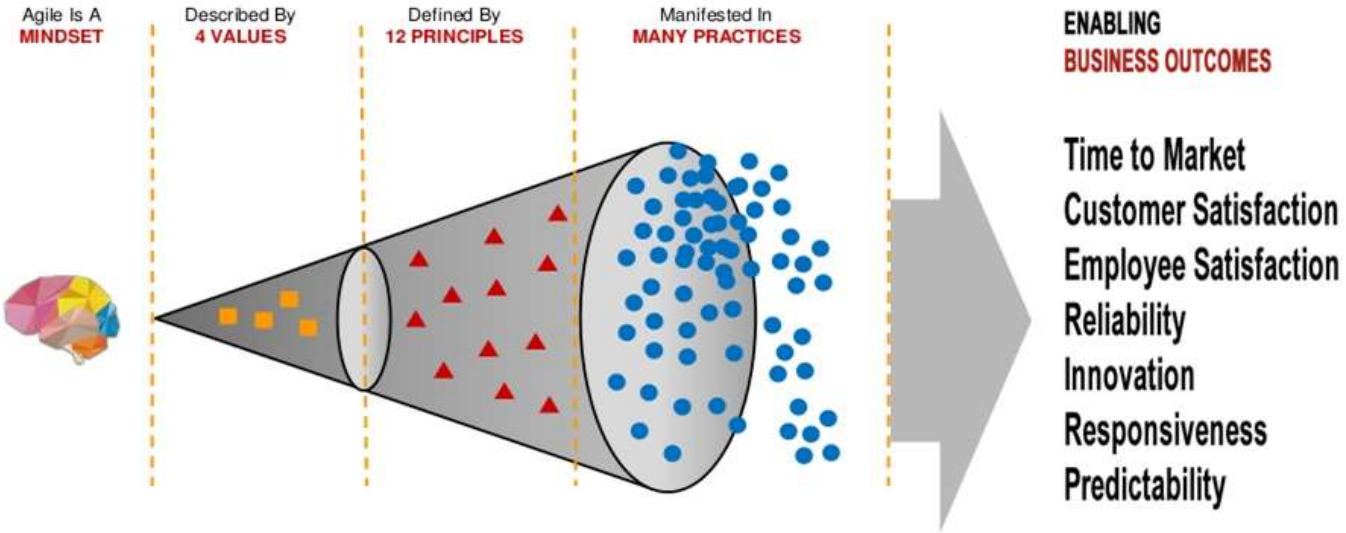
Changing the oversight approach in Agile settings means asking different questions on a new cadence

- Leads to different measurement and reporting approaches as well.

A focused government workforce development effort is required to enable the knowledge, skills, and abilities needed for effective oversight and interaction in Agile settings.

[1]July 14, 2016, Link: https://insights.sei.cmu.edu/blog/sei-researchers-provide-congressional-testimony-on-social-security/

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

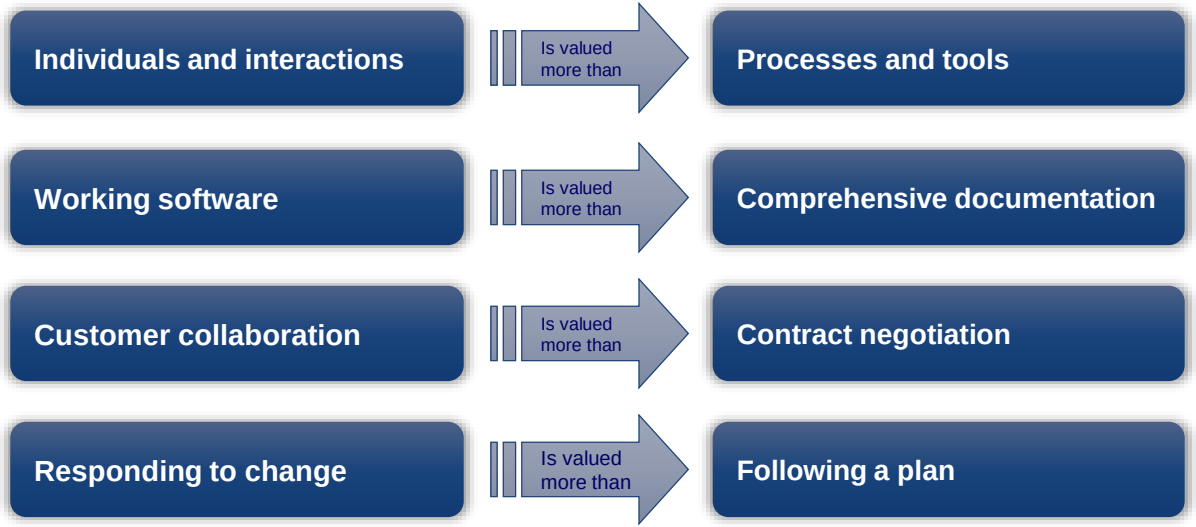
Is NOT Agile!

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: Agile Manifesto for SW Development

Where it all started….

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



http://agilemanifesto.org

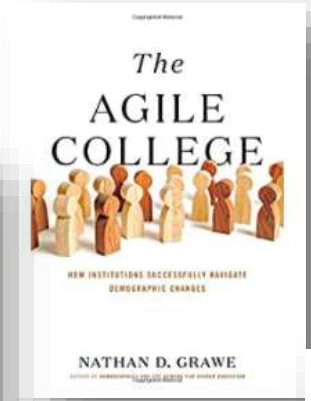
Agile Principles

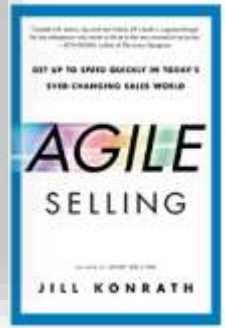
For most of these, substituting “working product” allows translation into non-software environments.

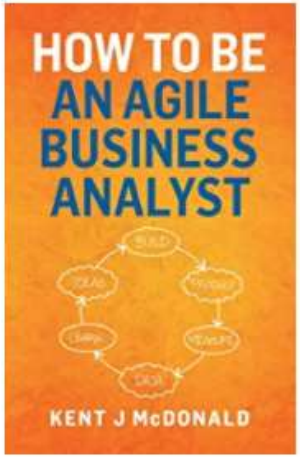
1. Our highest priority is to **satisfy the customer** through early and continuous delivery of valuable **software.**
2. Welcome **changing requirements**, even late in development. Agile processes harness change for the customer's competitive advantage.
3. **Deliver working software frequently**, from 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.

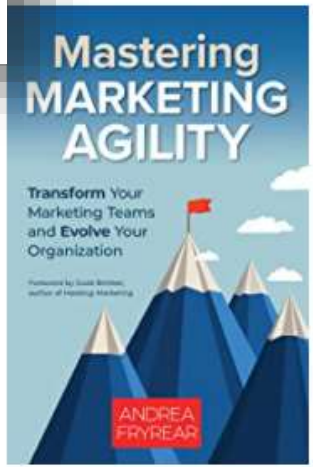
Agile Principles: http://agilemanifesto.org

Agile is Everywhere









Working Definition of Agile

Substituting “working product” allows translation into non-software environments.

Agile (*adj.*): 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]

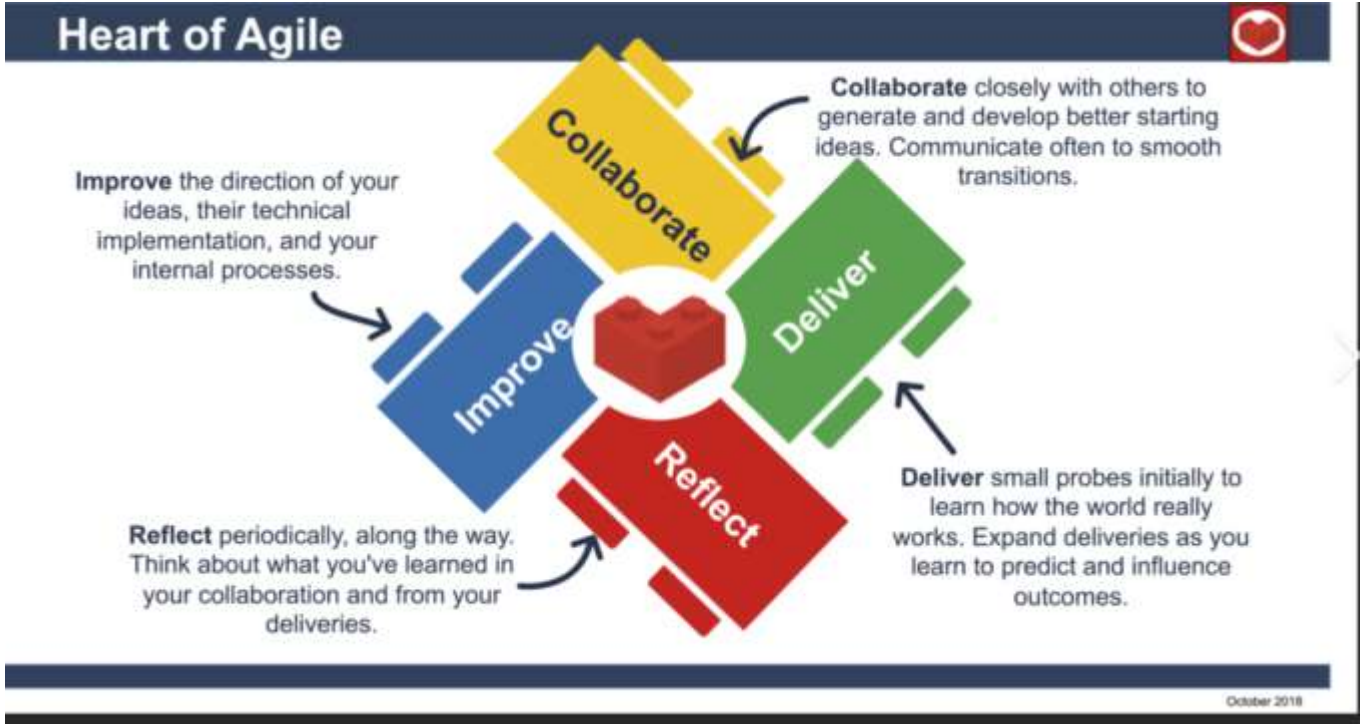
[Ambler 2013] Ambler, Scott. *Disciplined Agile Software Development: Definition*.
http://www.agilemodeling.com/essays/agileSoftwareDevelopment.htm

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: Scrum, XP, Kanban Are All Used in F-35

- 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

Some Observable Characteristics of Agile Implementations

Iterative—elements are expected to move from skeletal to completely fleshed out over time, not all in one step

Incremental—delivery doesn't occur all at once

Collaborative—progress is expected to be made by stakeholders and the development team working collaboratively throughout the development timeframe

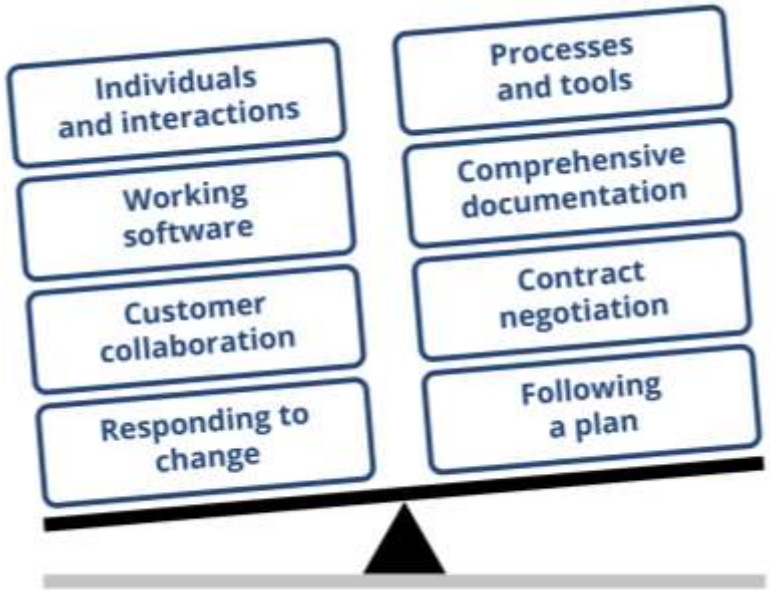
Loosely-coupled Architecture—multiple self-organizing, cross-functional teams work concurrently on multiple product elements (e.g., requirements, architecture, design, and the like) for multiple loosely coupled product components

Dedicated—team members are allowed to focus on the tasks within an iteration/release as opposed to multi-tasking across multiple projects

Time-boxed or Flow-based—relatively short-duration development cycles that permit changes in scope rather than changes in delivery time frame

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

https://youtu.be/LtDmoL1XQe0

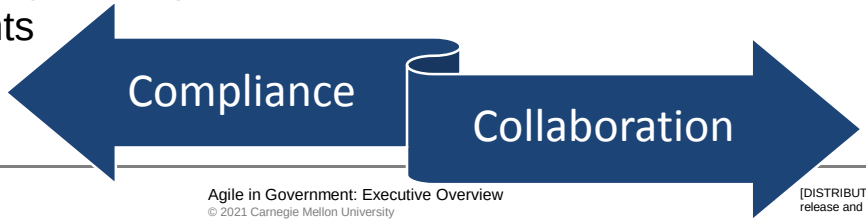
Compliance vs Collaboration

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



What About Scaling to Large Programs?

We operate on a massive scale – ***how does Agile work "in the large"?***

Some considerations when scaling above a few small teams:

- Managing interfaces among the many products/system components that multiple teams are working on…
- Synchronizing releases and events across multiple teams…
- Organizing inventory (backlog) of requirements productively to support the development pace of multiple small teams….
- Dealing with specialty disciplines (UX, security, etc.) that have significant inputs to the evolving product, but aren't needed as full time team members….
- Mindfully specifying architecture ("just enough") and other far-reaching concerns…
- Incorporating high assurance requirements (safety of flight, IA, nuclear surety…)

Lean principles help with scaling and with moving Agile out of the software arena

SAFe Lean-Agile principles

Highlights of a couple lean principles relevant to F-35

#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

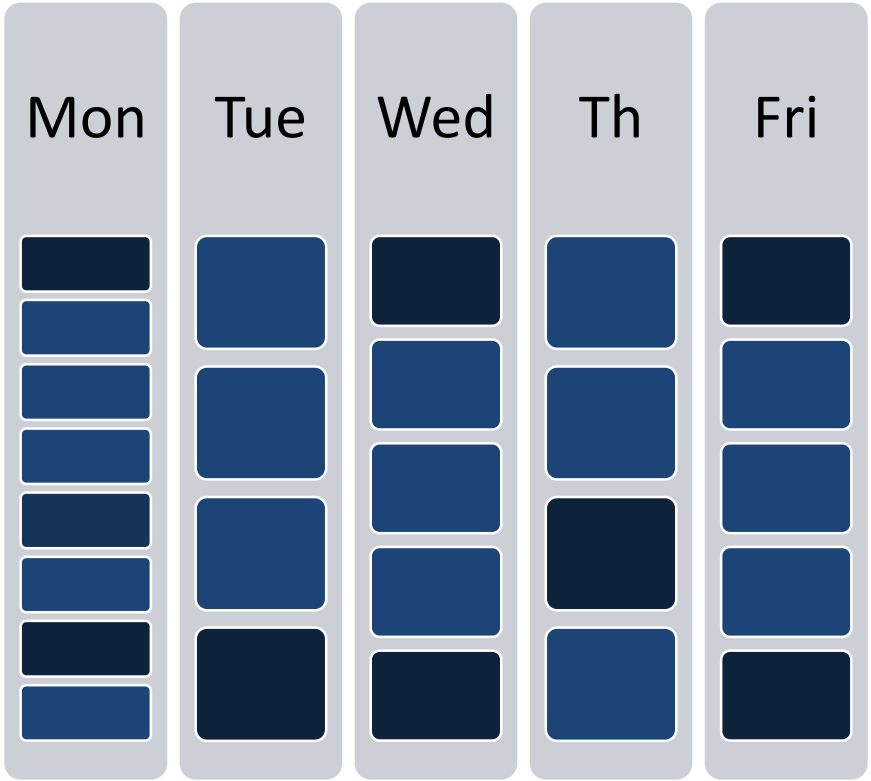
#6 - Visualize and limit WIP, reduce batch sizes, and manage queue lengths

#7 - Apply cadence, synchronize with cross-domain planning

#8 - Unlock the intrinsic motivation of knowledge workers

#9 - Decentralize decision-making

Utilization is the Wrong Goal



100% Utilization:

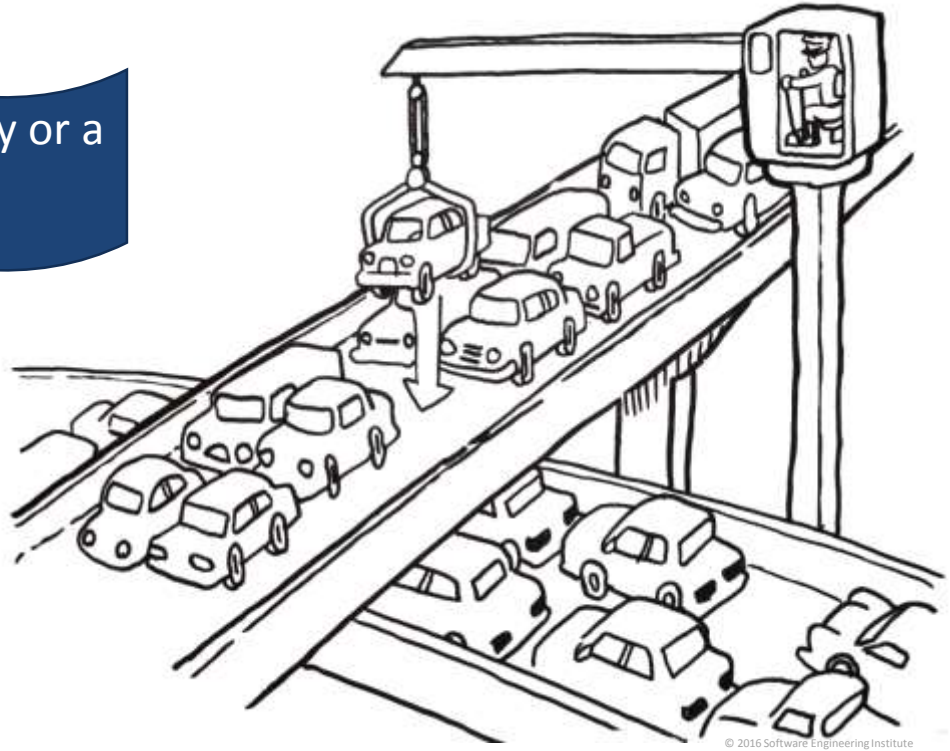
- Magnifies the impact of variation
- Maximizes task-switching overhead
- Assures slower overall progress

Change is inevitable, plan to learn

Multi-tasking is a myth we don't accurately comprehend

Maximum Utilization is Counterproductive

Do you want a highway or a parking lot?



Besides Longer Cycle Time, Queues are Just Generally Bad

The Principle of Queueing Waste: Queues are the root cause of the majority of economic waste in product development.

Queues create:

- Longer Cycle Time
- Increased Risk
- More Variability
- More Overhead
- Lower Quality
- Less Motivation

Principles of Product Development Flow,
Don Reinertsen

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

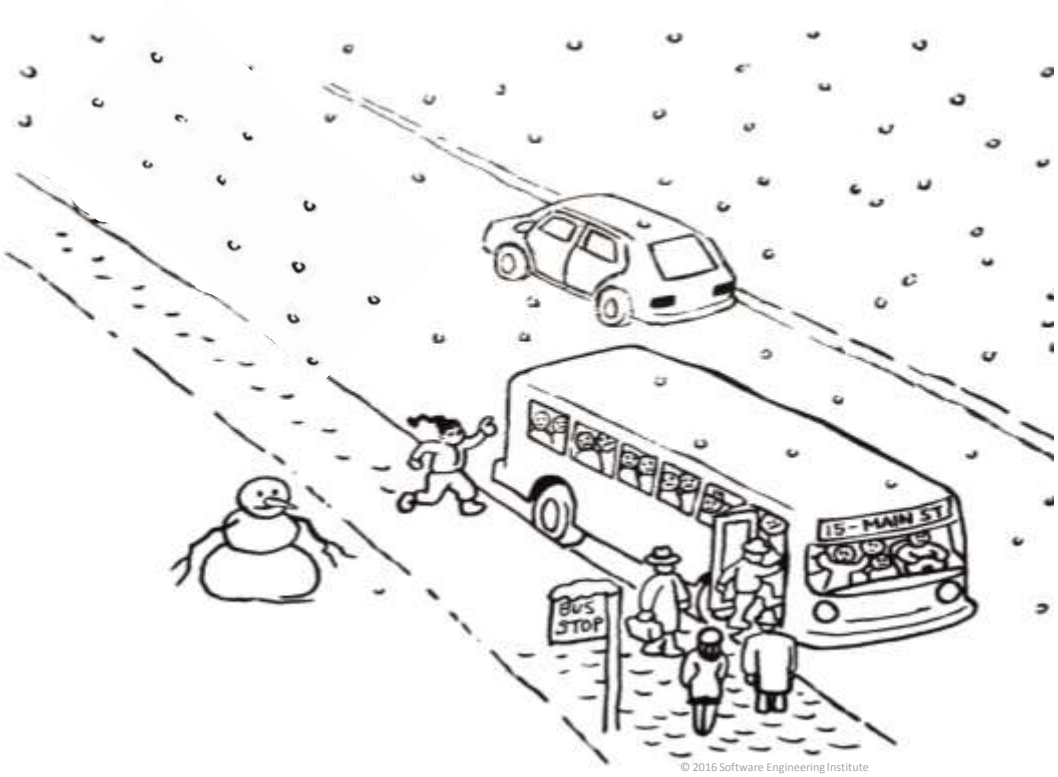
#6 - Visualize and limit WIP, reduce batch sizes, and manage queue lengths

#7 - Apply cadence, synchronize with cross-domain planning

#8 - Unlock the intrinsic motivation of knowledge workers

#9 - Decentralize decision-making

Cadence Enhances Predictability

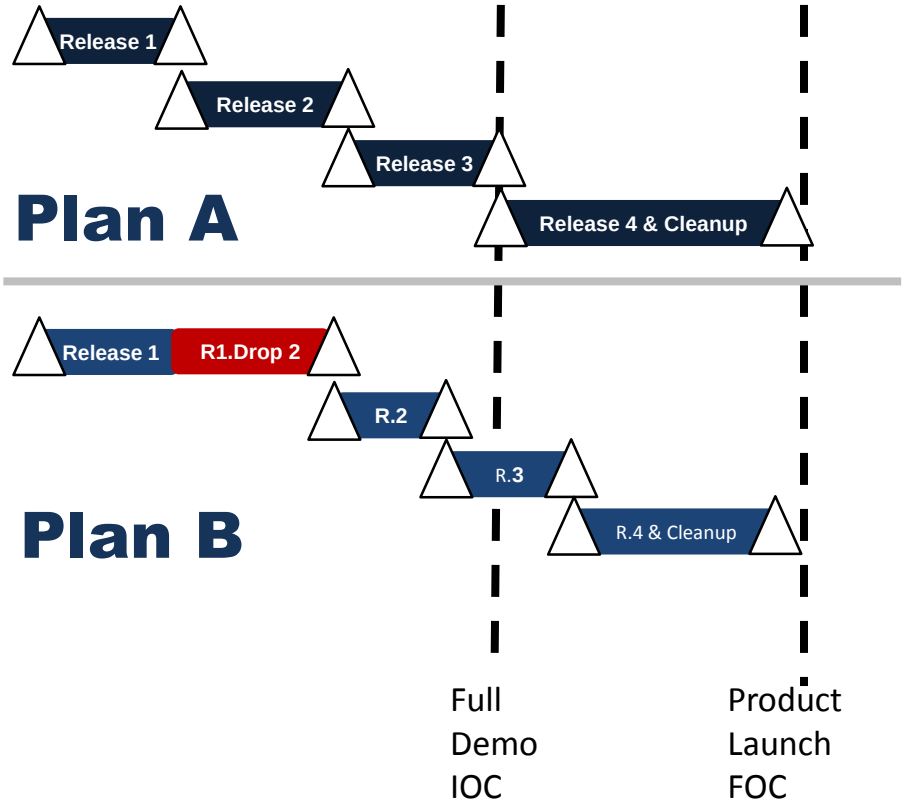


A Late Bus:

- Makes people scramble to get aboard
- They don't know when the next one will get here

Then the next bus comes along empty

Late Releases Become “Feature Magnets”

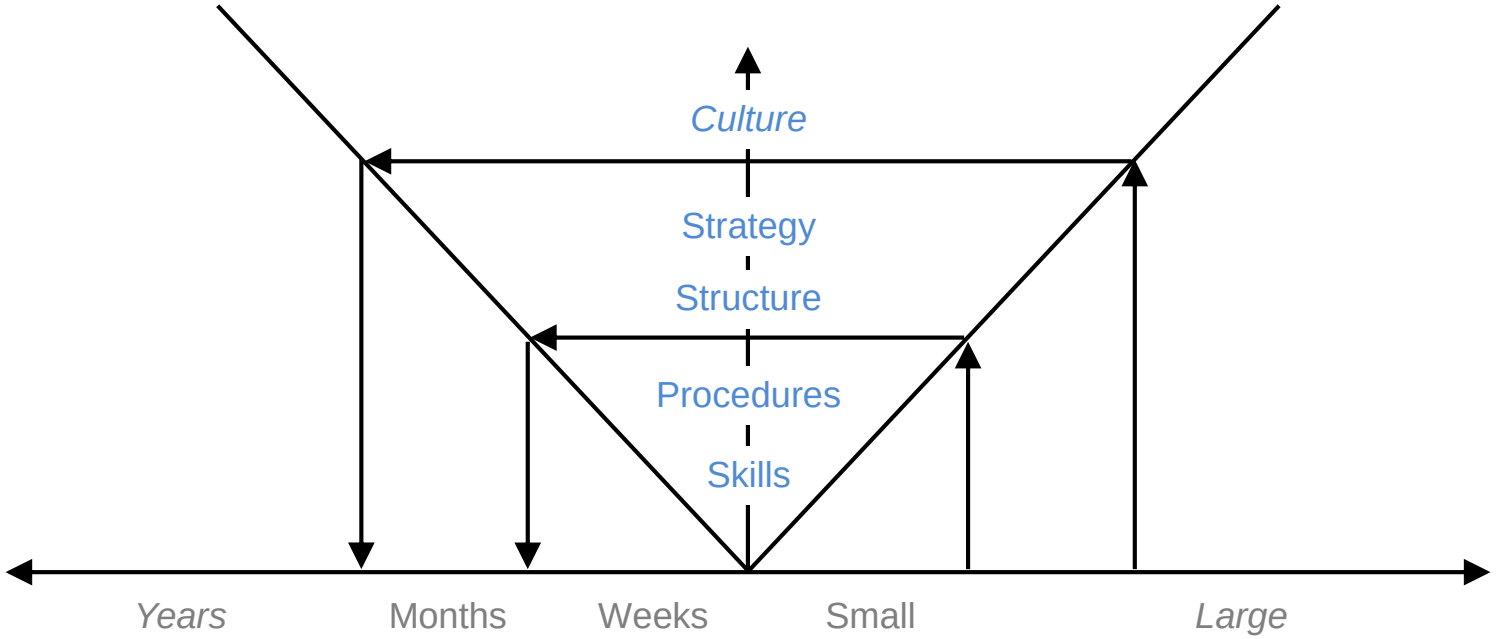


As things start to slip

- Influential people get ‘their priorities’ moved up, rather than deferred
- Pressure increases on early releases
- Functions slated for final release can’t be guaranteed…

If this is so great, why isn't everyone already doing it?

Level of Learning Required

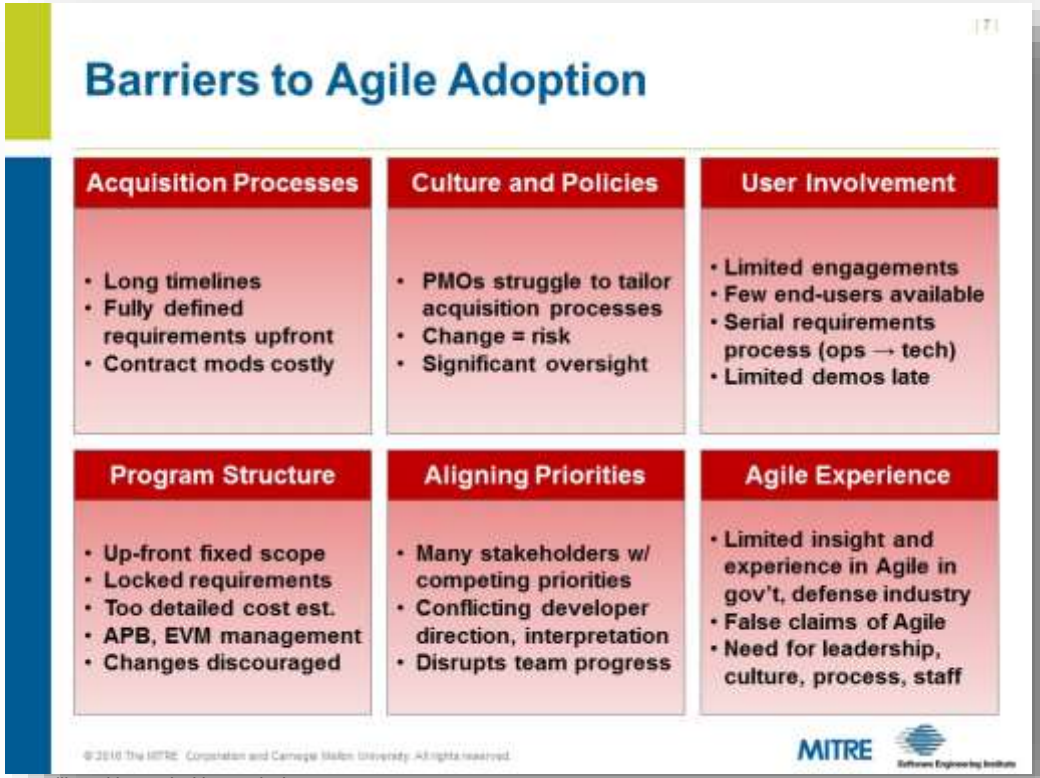


Time to adjust

Magnitude of Technological Change Sought

SEI Observations on Agile Adoption Barriers

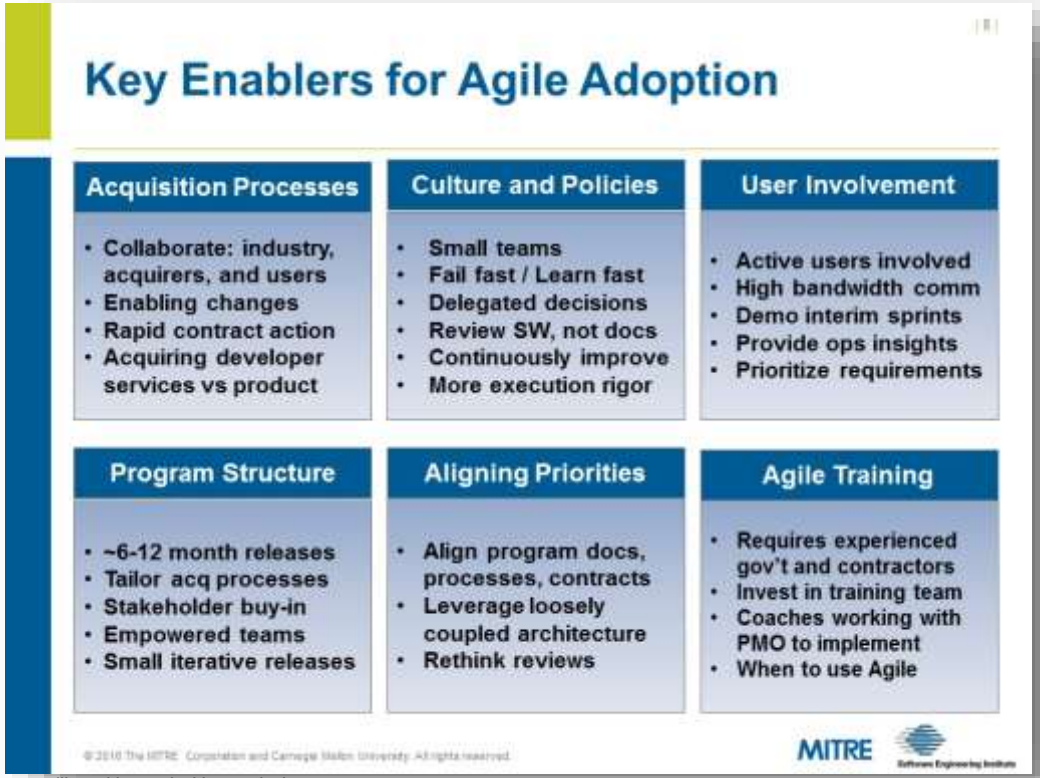
Which of these do your programs face?



Source: 2016 SEI/Mitre meeting with General E. Pawlikowski. Used with permission

SEI Observations on Key Enablers to Agile Adoption

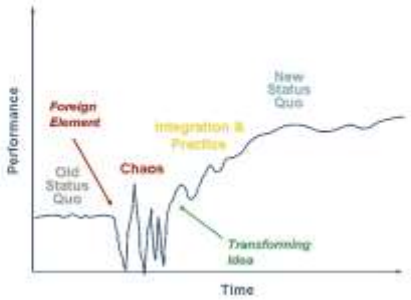
Which of these do your programs exhibit?



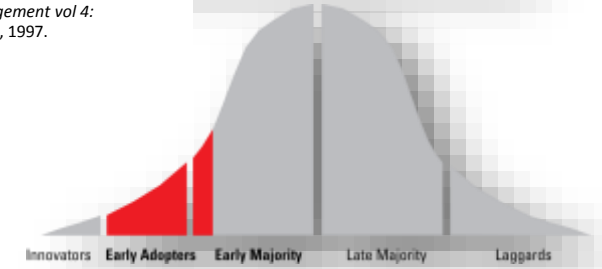
Source: 2016 SEI/Mitre meeting with General E. Pawlikowski. Used with permission

“Traditional” Adoption Tools and Methods Work Well with Agile Adoption

Understand the Change Cycle and Your Adoption Population

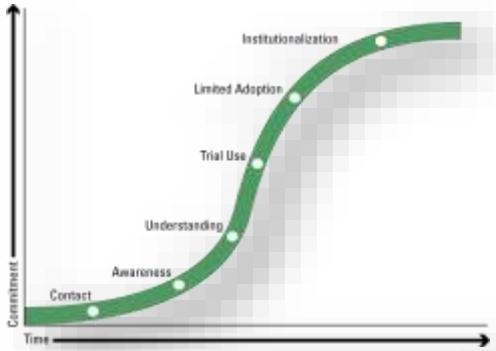


Adapted from *Quality Software Management vol 4: Anticipating Change,* Gerald Weinberg, 1997.



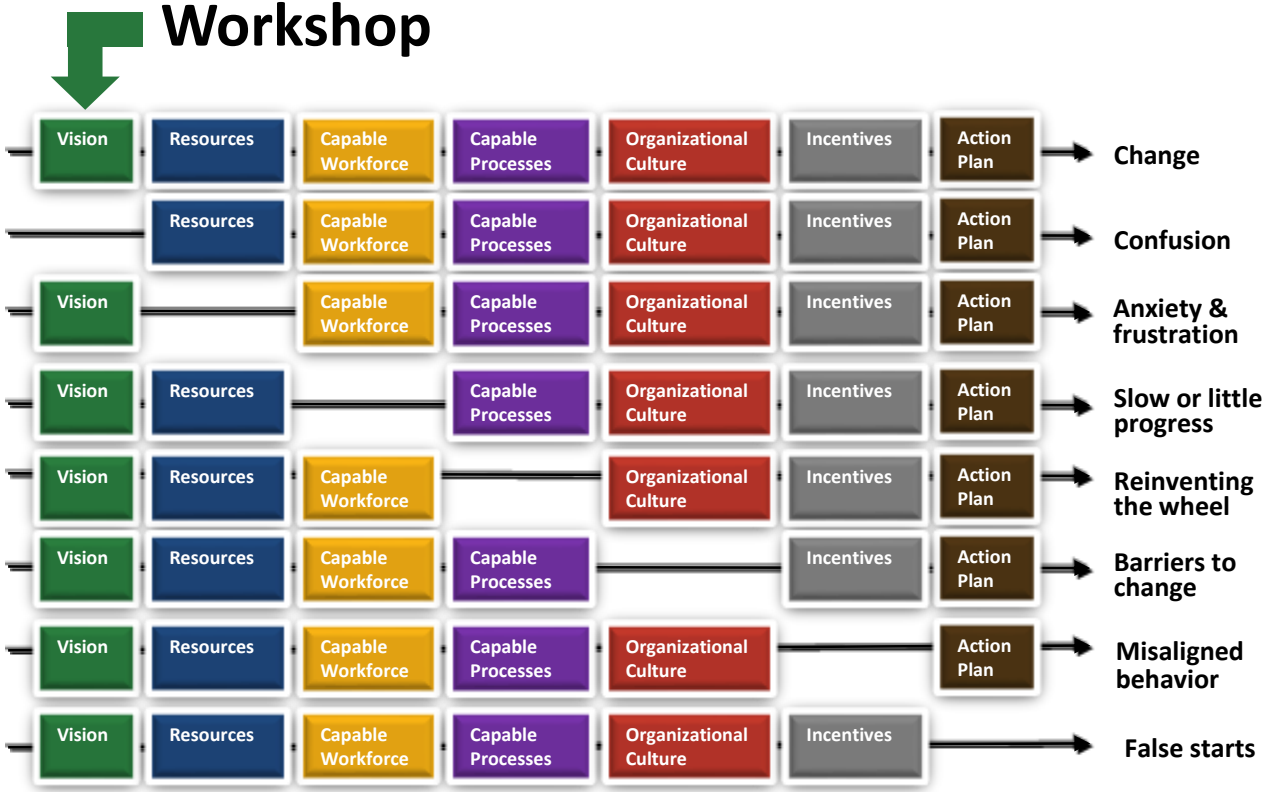
Adapted from *Crossing the Chasm*, Geoffrey Moore, 1991

Prepare for Both Communication and Implementation Support Mechanisms that are Needed



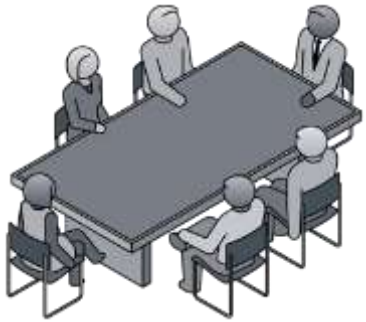
*Adapted from Daryl R. Conner and Robert W. Patterson, “Building Commitment to Organizational Change,” *Training and Development Journal* (April 1983): 18-30.

Where Leadership, Vision, and Goals Fit into Organizational Improvement



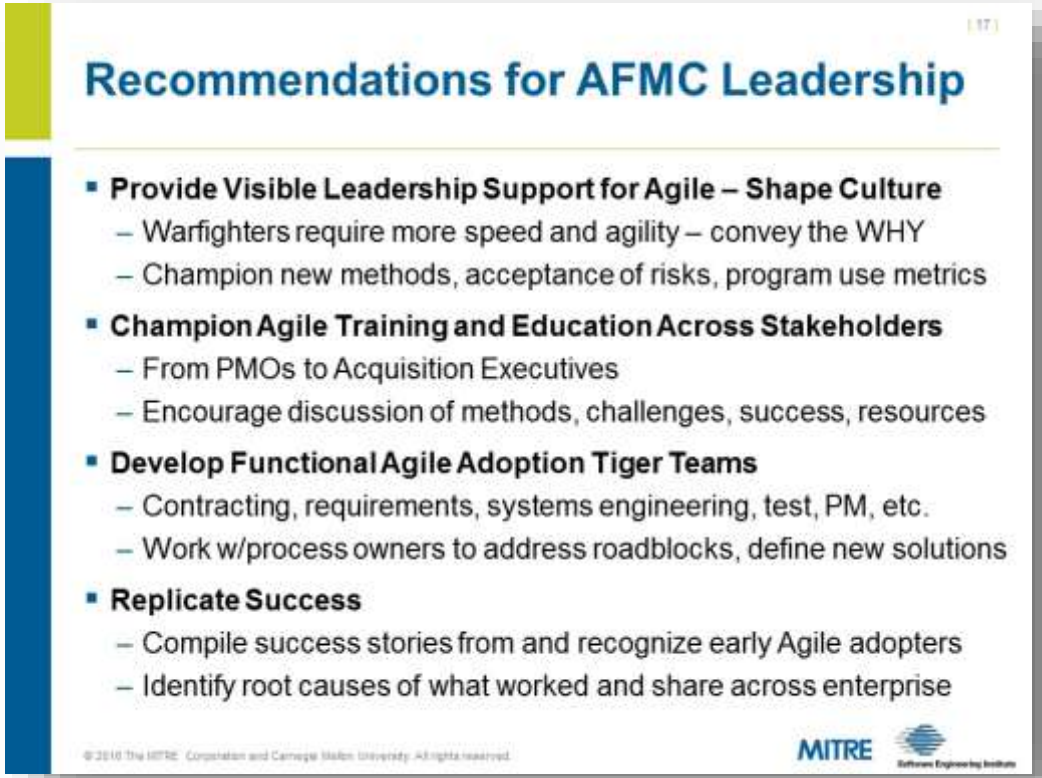
Adapted by Buttles (2010) from: Delorise Ambrose, 1987

Reminder: Attributes of Agile Success in Government Organizations



- **Top Cover**
- **Permission to "fail fast" (learn fast)**
- **Training in Agile**
- Use of Agile coach
- Enough up-front system and software architecture
- Willing to work collaboratively across government/ contractor boundary
- Willing and open to adopt new modes of operation
- Dedicated staff

What do leaders have to do to change the environment?



Source: 2016 SEI/Mitre meeting with General E. Pawlikowski. Used with permission

Leadership oversight measures: Ask new questions

Category	Description
Flow	Flow measures come out of the lean engineering and management environment. They focus on understanding the "idea to realization" cycle time. Flow measures for senior oversight focus on the development organization's ability to consistently meet timelines for deployment of IT functions according to a roadmap. These are cycles measured in weeks and months, rather than quarterly or annual cycles seen traditionally.
Engagement	Engagement measures help oversight organizations understand the level of collaboration that has been achieved. Timely involvement of stakeholders from the workflow supported by the IT system results in a deeper understanding of intended usage. Evolution of the workflow to better utilize technology results from engagement with the correct decision makers.
Quality	Quality measures at senior oversight levels have less to do with software defect rates than they do with the quality of the services supported by the IT systems. For example, improvements in wait times for key services, or percentage of "made it through in one pass" attempts to use a service are potential quality measures. These measures, in turn, drive the priorities for quality measures among software teams.
Risk	Risk measures for senior oversight can focus on the development organizations' performance in managing threats to their success, more than those threats themselves. When using Agile methods, confidently asserting the expected success of a program is no longer based on the comprehensive- ness of up-front specification documents. Therefore, an oversight approach for Agile cannot rely on review and approval of such projective documents as the primary mode of risk identification. The short and steady cadence of Agile promotes rapid learning.

Source: SEI Congressional testimony July 14, 2016 to House Ways and Means Committee.

Important Points to Remember: Agile Basics

Agile is an iterative, incremental, highly collaborative approach that prioritizes responsible responsiveness to changing conditions and as-built product over projections

- There are many valid ways to implement the principles
- A wide variety of popular engineering methodologies fall under the umbrella of “Agile”

Agile approaches require collaboration across the enterprise to be successful

- Contracts, finance, test, end users…

Agile approaches support fast learning cycles and adaptation to changing conditions/volatility

- Changes in technology, threats, priorities and diverse stakeholders, unknown solutions/experimentation
- Traditional highly sequential (“waterfall”) approaches are well-suited to homogeneous, stable environments with slowly changing requirements

Welcome to your 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!

Contact Information

Suzanne Miller
Principal Researcher
Software Engineering Institute
Email: smg@sei.cmu.edu
Phone: +1 412 512-6300

SEI Customer Relations
Software Engineering Institute
4500 Fifth Avenue
Pittsburgh, PA 15213-2612
Email: info@sei.cmu.edu
Telephone: +1 412-268-5800
Fax: +1 412-268-6257

Mark A. Ramsey
TSS PMO Acquisitions Lead
Program Manager, NH-04, USAF
Training Systems and Simulation PMO
F-35 Lightning II Joint Program Office
Email: Mark.Ramsey@jsf.mil
D: 703-604-6081
C: 571-451-5054
DSN: 329-6081

A Few Relevant Resources

USAF Chief Software Officer:

https://software.af.mil/

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

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:

SEI Congressional testimony:

SEI Researchers Provide Congressional Testimony on Social Security (cmu.edu)

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

SMC Agile/Lean Self-Service Learning Paths Catalog

- 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

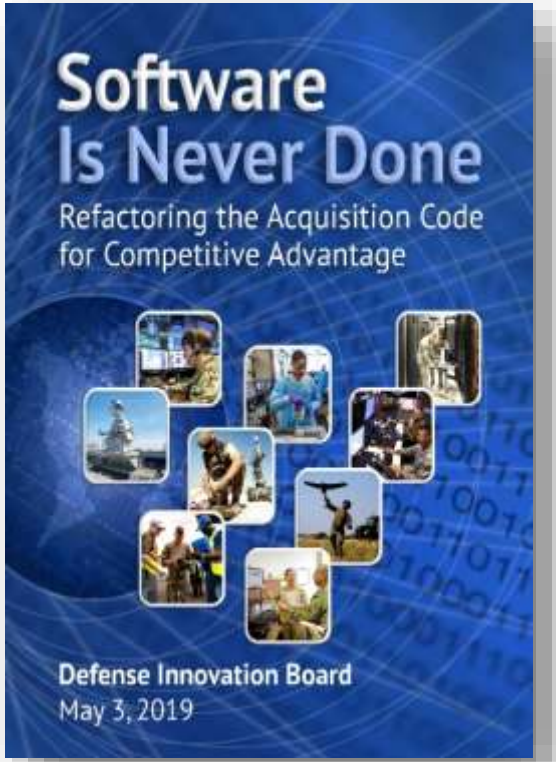
Software Engineering Institute Agile Resources: podcasts, blog posts, Technical Notes on many Agile in Government topics:

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

BACKUP MATERIALS

Software Engineering Institute
Carnegie Mellon University
Pittsburgh, PA 15213

2019 DIB SWAP Study



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/

Traditional vs. Agile Approaches

Traditional approach

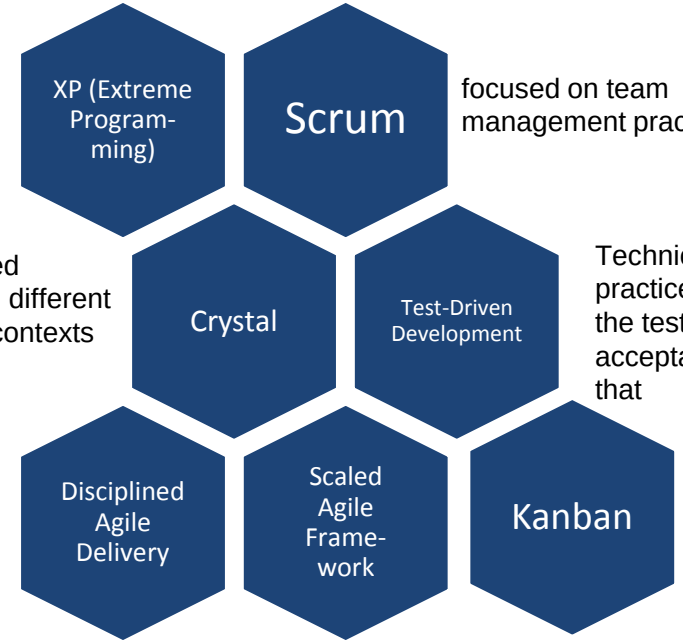
- Is consistent with the acquisition lifecycle provided in typical acquisition guidance
- Works well for
  - programs with stable requirements and environment, with known solutions to the requirements
  - programs with a homogeneous set of stakeholders who communicate well via documents
  - programs for which the technology base is evolving slowly (technology is not expected to be refreshed/replaced within the timeframe of the initial development)

Agile approach works well for

- programs with volatile requirements and environment
- programs where solutions are sufficiently unknown that significant experimentation is likely to be needed
- programs for which the technology base is evolving rapidly
- *programs with stakeholders who can engage with developers in ongoing, close collaboration*

Nidiffer, K. Miller, S. & Carney, D. *Potential Use of Agile Methods in Selected DoD Acquisitions: Requirements Development and Management* (CMU/SEI-2013-TN-0006), September 2013.

Many Methods Generally Termed “Agile”



focused on team technical practices

focused on team management practices

Encourages risk-based selection of practices; different patterns for different contexts

Technical and management practices focused on writing the test that proves acceptance, then coding to that

Originally derived from Rational Unified Process, designed to scale

Pull-based approach particularly favored for services like security, systems engineering

Merger of lean, Kanban, and other Agile methods to support large scale projects

A Word about Sample RFP Language

No “iconic” RFP language for encouraging Agile development practices exists

- Lots of factors go into what language would be appropriate
- DCMA is considering changes to their policies related to audit points, etc, which could point to some new language—not expected for another year
- NDIA System Engineering Agile working group developed a Special Report on this topic:

  RFP Patterns and Techniques for Successful Agile Contracting

  http://resources.sei.cmu.edu/library/asset-view.cfm?assetid=484056

Useful Interpretation of Agile Principles for Government Settings (1/3)

Agile Principle	Useful Interpretations in Government Settings
The highest priority is to satisfy the customer through early and continuous delivery of valuable software.	In government, the "customer" is not always the end user. The customer includes people who pay for; people who use; people who maintain; as well as others. These stakeholders often have conflicting needs that must be reconciled
Welcome changing requirements, even late in development. Agile processes harness change for the customer's competitive advantage.	Rather than saying "competitive" advantage, we usually say "operational" advantage. This principle causes culture clash with the "all requirements up front" perspective of many large, traditional approaches.
Deliver working software frequently, from a couple of weeks to a couple of months, with a preference for the shorter timescale.	What it means to "deliver" an increment of software may well depend on context. With large embedded systems, we are sometimes looking at a release into a testing lab. Also, for some systems, the operational users are not able to accept all: "deliveries" on the development cadence – because there are accompanying changes in the workflow supported by the software that require updates.
Business people and developers must work together daily throughout the project.	In government settings, we interpret "business" people to be end users and operators, as well as the other types of stakeholders mentioned in Principle 1, since in many government settings, the business people are interpreted as the contracts and finance group.

Source: SEI Congressional testimony July 14, 2016 to House Ways and Means Committee.

Useful Interpretation of Agile Principles for Government Settings (2/3)

Agile Principle	Useful Interpretations in Government Settings
Build projects around motivated individuals. Give them environment and support they need, and trust them to get the job done.	A frequent challenge in government is to provide a suitable technical and management environment to foster the trust that is inherent in Agile settings. Allowing teams to stay intact and focused on a single work stream is another challenge.
The most efficient and effective method of conveying information to and within a development team is face-to-face conversation.	In today's world, even in commercial settings, this is often interpreted as "high bandwidth" rather than only face-to-face. Telepresence via video or screen-sharing allows more distributed work groups than in the past.
Working software is the primary measure of progress.	Our typical government system development approaches use *surrogates* for software – documents that project the needed requirements and design – *rather than the software itself*, as measures of progress. Going to small batches in short increments allows this principle to be enacted, even in government setting, although delivery may well to be a test environment or some internal group other than users themselves.
Agile processes promote sustainable development. The sponsors, developers, and users should be able to maintain a constant pace indefinitely.	This principle is a caution against seeing agility just as "do it faster." Note that this principle includes stakeholders outside of the development team as part of the pacing.

Source: SEI Congressional testimony July 14, 2016 to House Ways and Means Committee.

Useful Interpretation of Agile Principles for Government Settings (3/3)

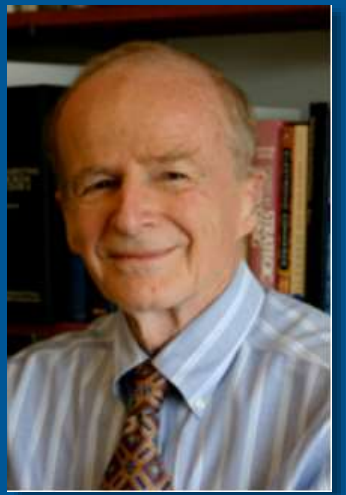
Agile Principle	Useful Interpretations in Government Settings
Continuous attention to technical excellence and good design enhances agility	This is a principle that often is cited as already being compatible with traditional government development.
Simplicity– the art of maximizing the amount of work not done– is essential.	One issue with this principle in government setting is that our contracts are often written to penalize the development organization if they don't produce a product that reflects 100% of the requirements. This principle recognizes that not all requirements we think are needed at the onset of a project will necessarily turn out to be things that should be included in the product.
The best architectures, requirements, and designs emerge from self-organizing teams.	Note that the principle does not suggest that the development team is necessarily the correct team for requirements and architecture. It is however, encouraging teams focused in these areas to be allows some autonomy to organize their work. Another complication in many government settings is that we are often re-architecting and re-designing existing systems.
At regular intervals, the team reflects on how to become more effective, then tunes and adjusts its behavior accordingly.	This principle is an attempt to ensure that “lessons learned” are actually learned and applied rather than just being “lessons written”

Source: SEI Congressional testimony July 14, 2016 to House Ways and Means Committee.

Little's Law

$$L = \lambda W$$

…the long-term average number L of customers in a stationary system is equal to the long-term average effective arrival rate λ multiplied by the average time W that a customer spends in the system…



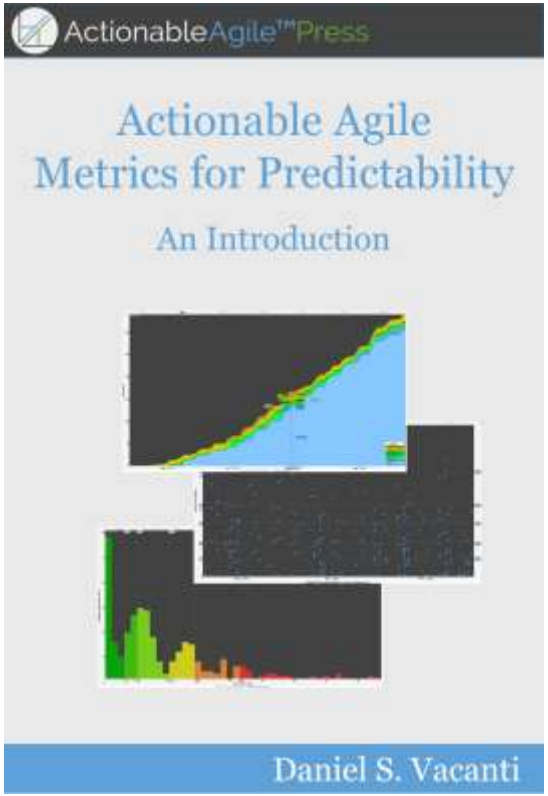
http://mitsloan.mit.edu/faculty-and-research/faculty-directory/detail/?id=41432

Little's Law in Agile Metrics

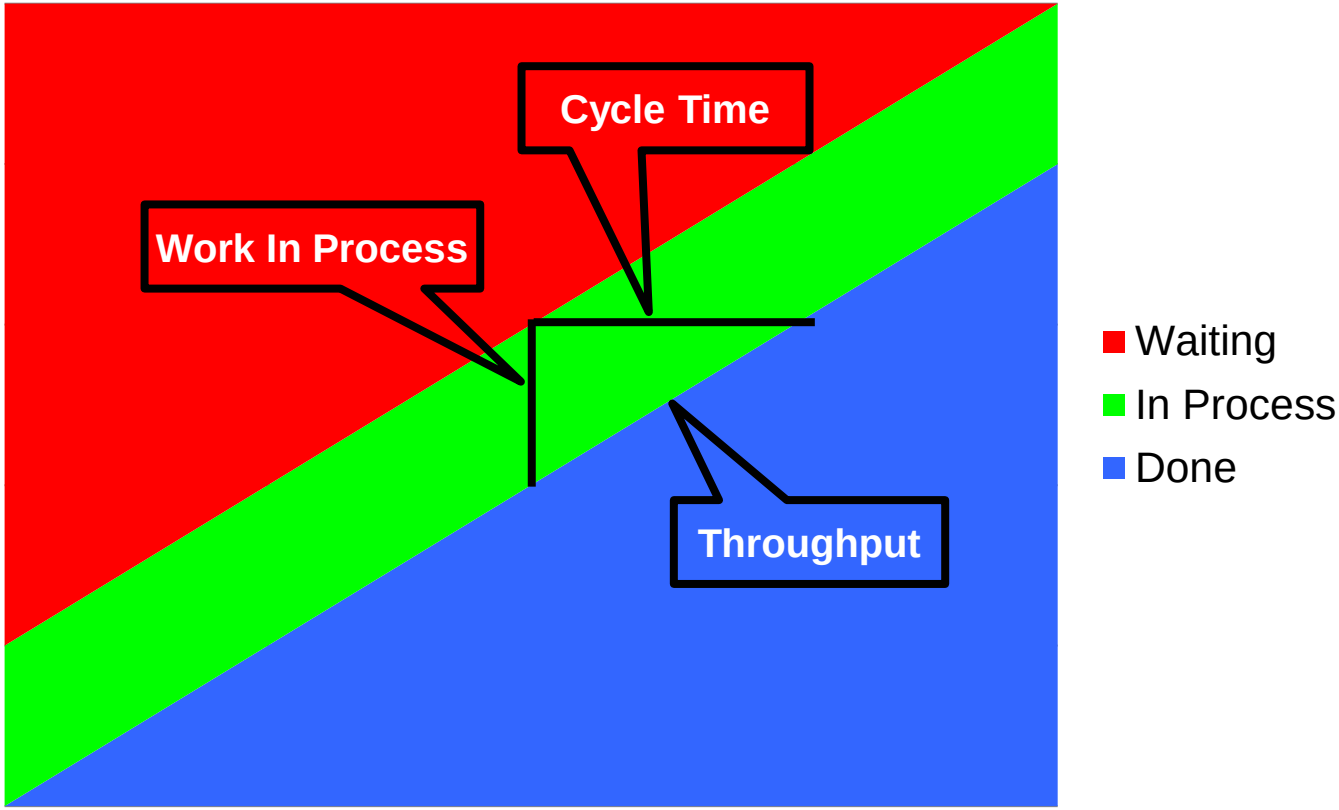
Three Metrics Emphasized*:

1. **Work In Progress** (the number of items that we are working on at any given time),
2. **Cycle Time** (how long it takes each of those items to get through our process), and
3. **Throughput** (how many of those items complete per unit of time).

** Excerpted from page 13 of the book depicted on the right.*



Utility of Little's Law



Program Level vs. Team Level Measures

Geared to External Stakeholders

Intended to Serve Needs of the Team Typically Not Shared Out-side the Team