Types of Agile

Peter Capell

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.

DM22-0989

Why Agile

Lessons on Iterative Realization of Value

Abstractions of the product may not all be deliverable components

Iterations of process design may be paired with product increments

Not all iterative processes need to execute on the same cadence

Product demonstrations don't have to all be a 'run for score'

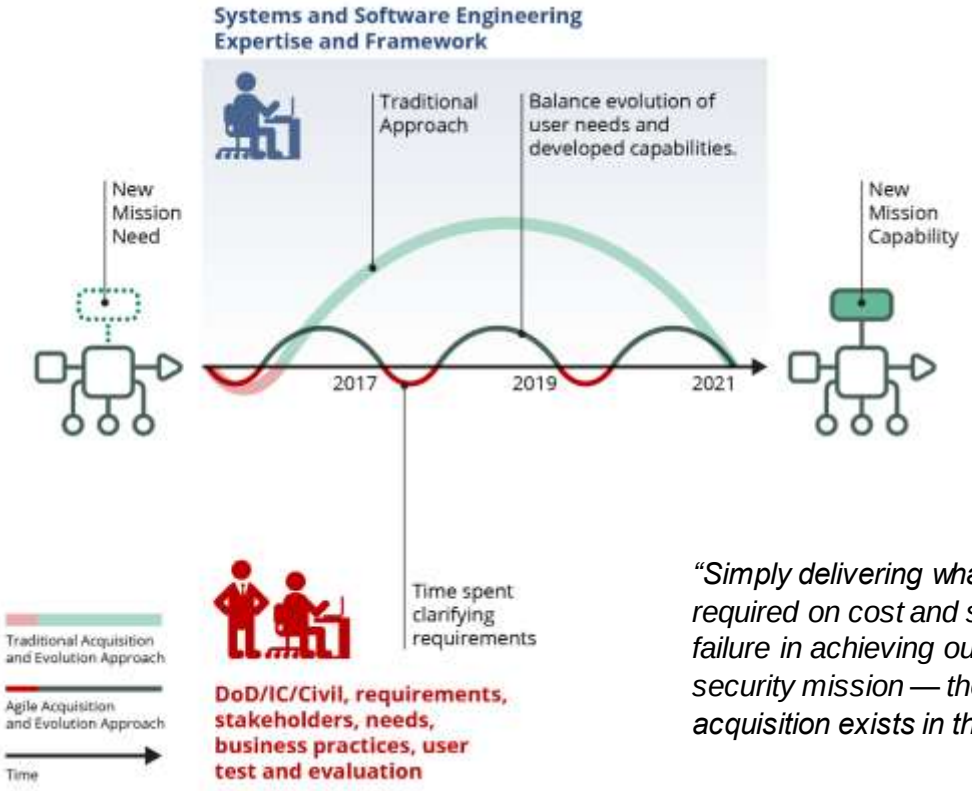
Mission and Product Vision drive process optimization – not the other way around

Why Does the DoD Care about Agile?

Deliver performance at the speed of relevance

Streamline rapid, iterative approaches from development to fielding

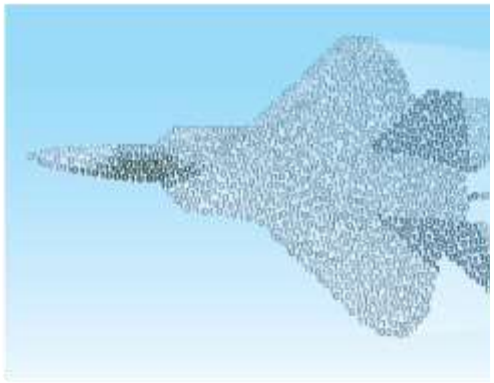
National Defense Strategy Summary
Jan 2018

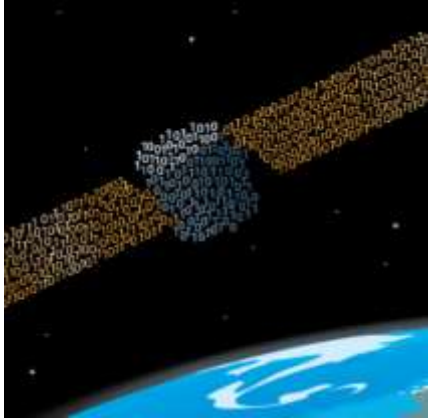


"Simply delivering what was initially required on cost and schedule can lead to failure in achieving our evolving national security mission — the reason defense acquisition exists in the first place."

Honorable Frank Kendall
Under Secretary of Defense (AT&L)
2015 Performance of The Defense Acquisition System

Why Does the DoD Care about Software?

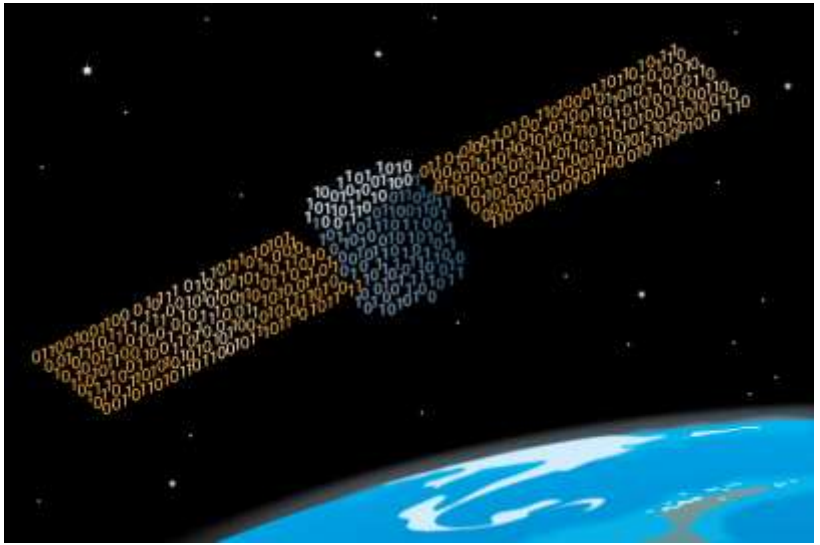




If the world isn't software-defined yet, it is certainly software-reliant!

Software is often part of a larger System Development Activity

- Major system engineering endeavors in aircraft vehicles, satellites, economic infrastructure, energy management…
  - Often these are contracted systems, where the software part of the system is being acquired by the owner of the overall system



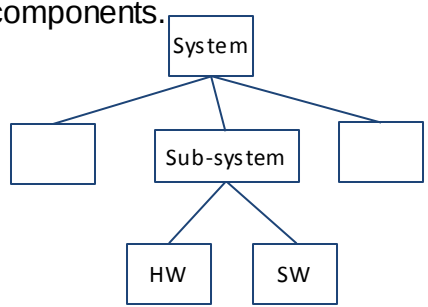
Many Project Management Practices Assume a Hardware-Centric System

Hardware Sys Development Assumptions (is a part of relationships)

Systems can be decomposed into discrete, independent, and hierarchically related components (or subsystems).

Component interaction can largely be derived from the original decomposition or BOM

Quality attributes can be allocated to specific components.

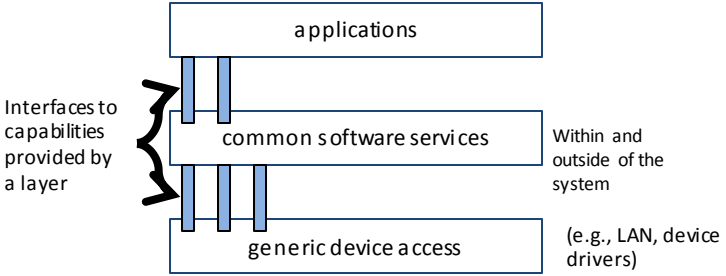


Software Realities (is used by, is controlled by) relationships

Complex interactions among components and across layers

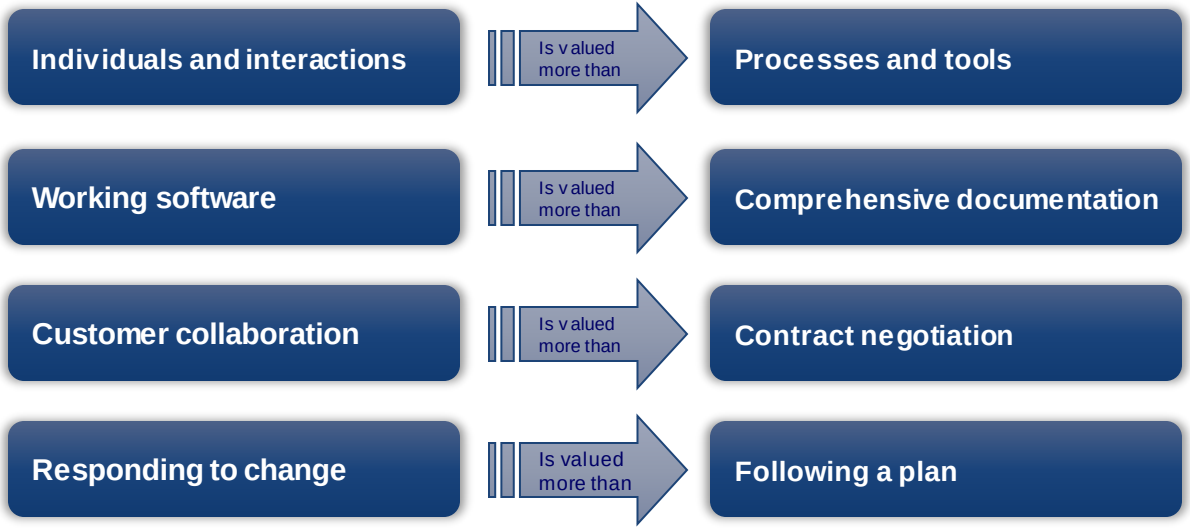
Interactions can take many different paths.

Quality attributes are achieved by component interactions (*not* by individual components).



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

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

➢ **Activity: Which principle(s) do you think are most applicable to the work you do?**

Testing is development, development is testing

Phased

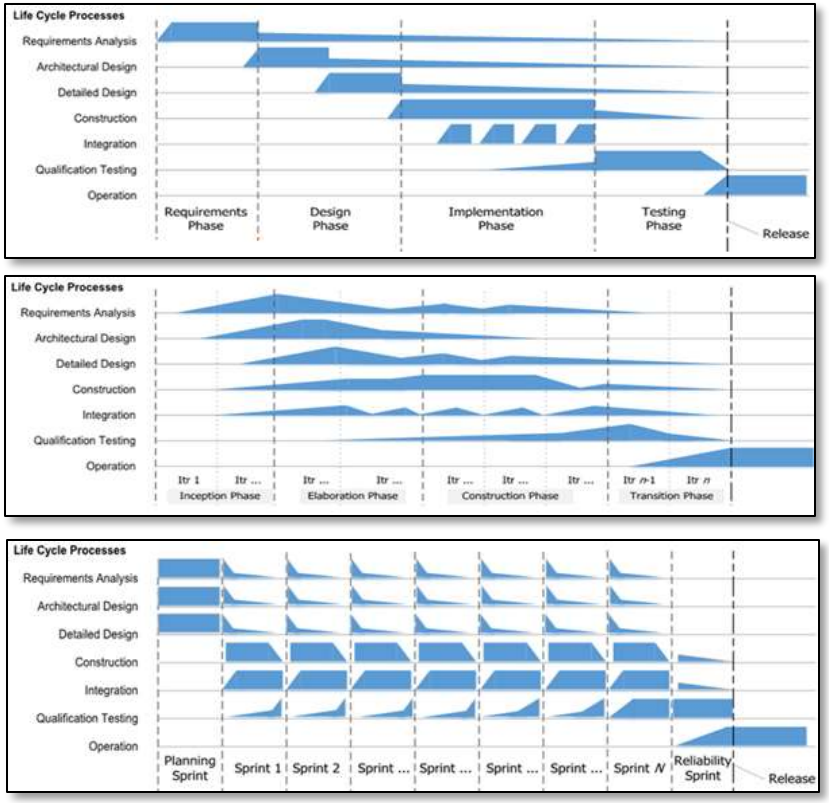
- Design, Code
- Test, Test, Test
- Release

Incremental

- Design a little, code a little, test a little…
- Test, Test, Test
- Release

Agile

- Test, fail, code, test, pass …
- Test, fail, code, test, pass …
- Test, fail, code, test, pass …
- Release

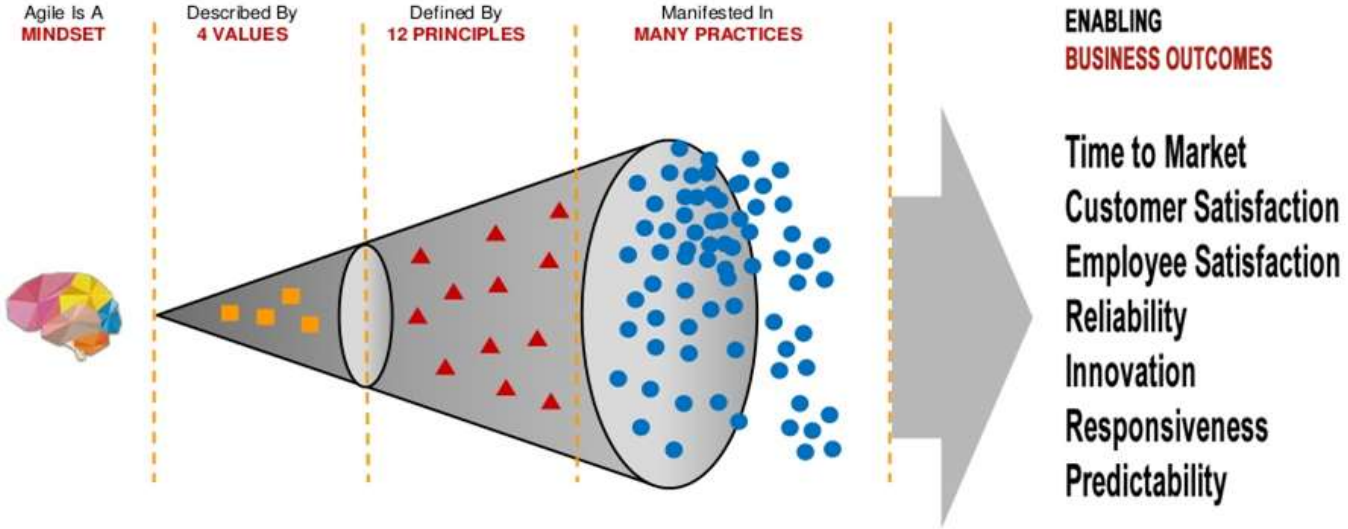


Source: IEEE Standard 1633P, *Recommended Practices for Software Reliability Engineering*

The Transition to Agile is Multi-Dimensional

	Traditional DoD	Agile DoD
Organization	Hierarchical, command and control based teams.	Self-organizing teams. Decentralized decision-making. Value stream networks in addition to functional hierarchy
Leadership	Leader as primary source of authority to act	Facilitative leadership
Staffing	Roles operate in silos participating only at distinct, defined points in the program life cycle.	Cross-functional teams including all roles operate throughout the lifespan of the program.
Practice Adoption	PMO oversight tool focused on demonstrating compliance	Agile advocate or coach works to assist the team in adopting Agile and Lean principles and practices.
Communications	Top-down communications structure dominate	Team-driven, informal communications dominate
Documentation	Focus on detailed documents as contracts, oversight mechanisms	Focus on “just enough” documentation, lightweight modeling and ongoing direct communications
Practice Improvement	Compliance to regulations, policies and procedures by team. Improvements made centrally and handed down.	Continuous improvement of practices driven by team.

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

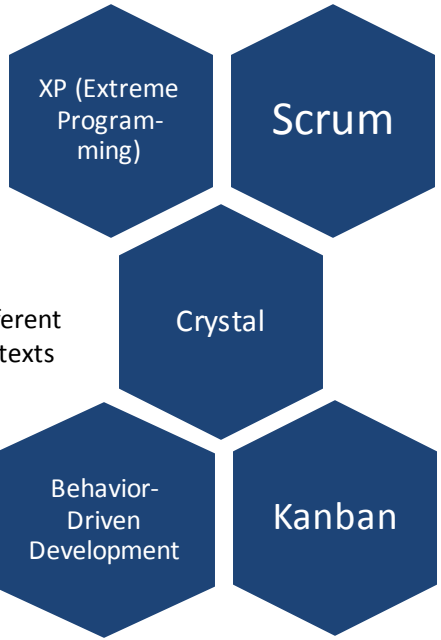
Is NOT Agile!

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.

Types of Agile

Team Methods Generally Termed “Agile”

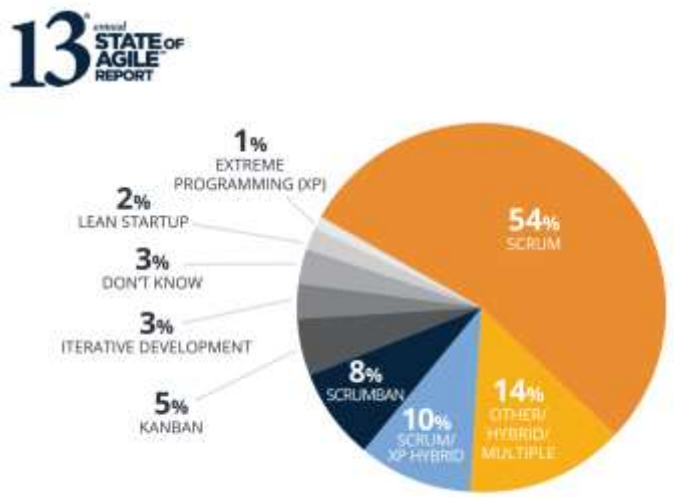
focused on team technical practices; gaining popularity in USAF due to Kessel Run



focused on team management practices

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

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

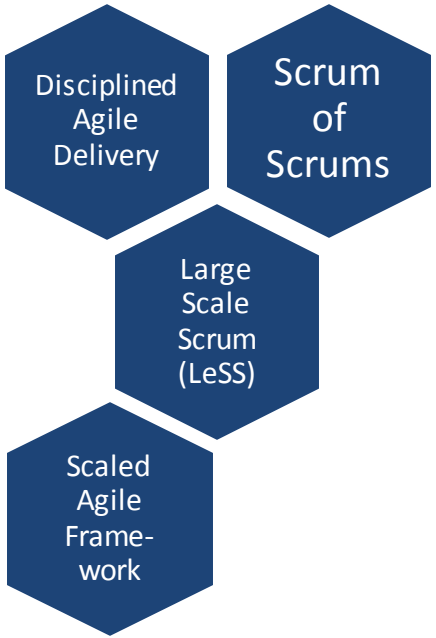


Some Agile Techniques are Transcending Methodologies

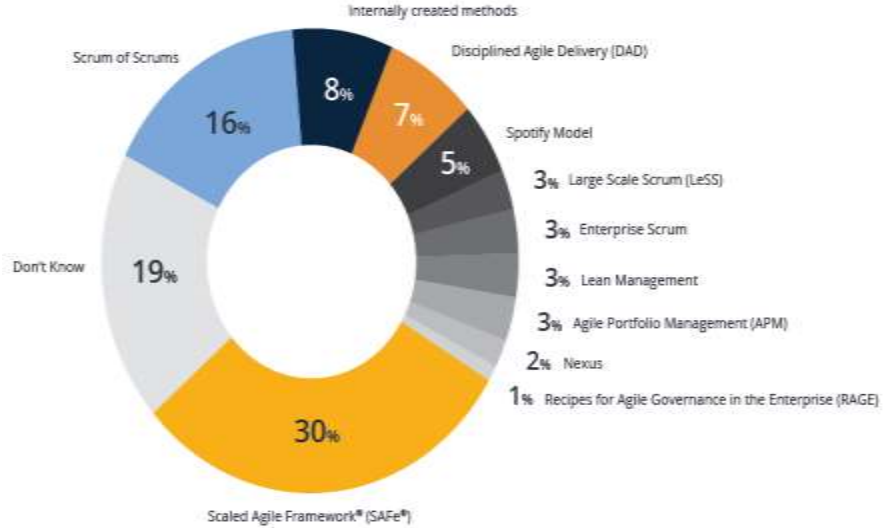
Source: 13th Annual Survey on State of Agile, Version One, April 2019

Scaling Methods Generally Termed “Agile”

Originally derived from Rational Unified Process, designed to scale



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



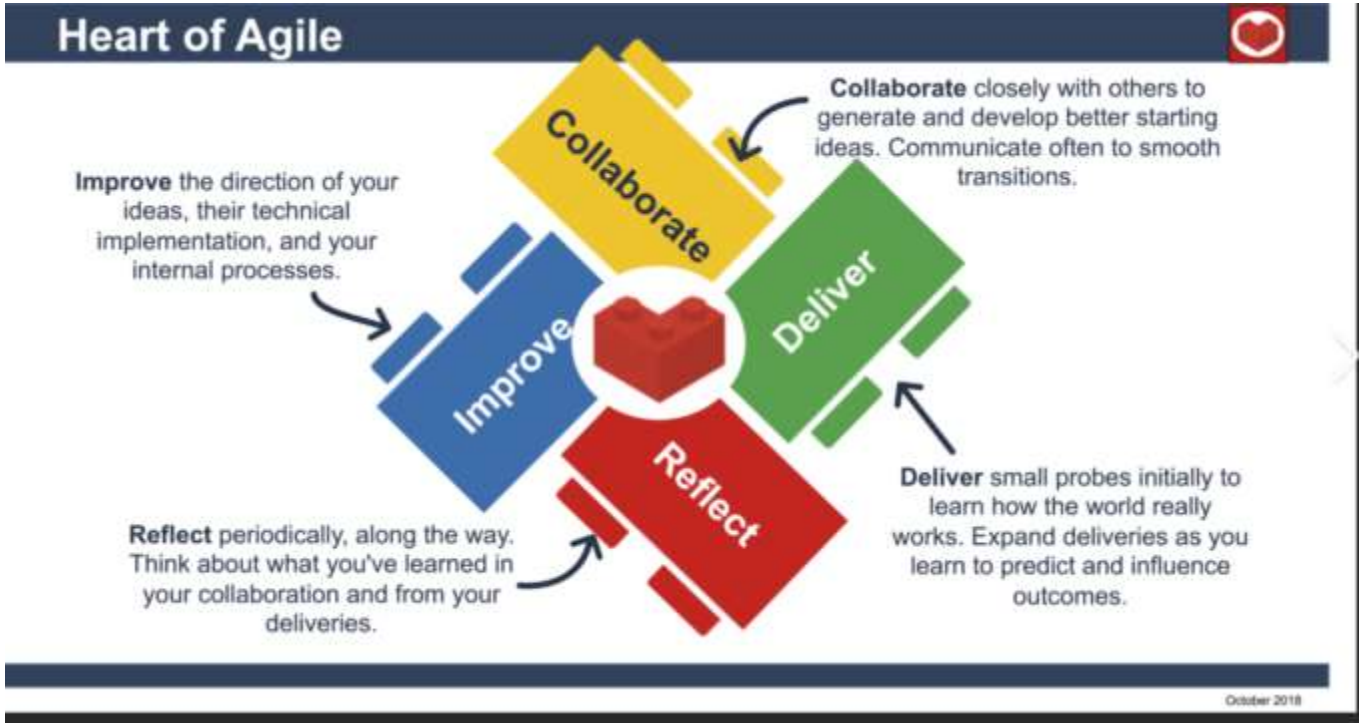
Or these

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

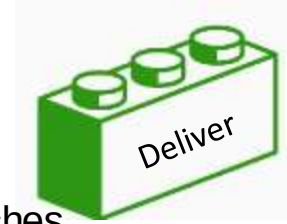


Source: heartofagile.com

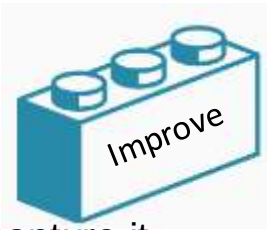
Agile Team Method Commonalities: Scrum, XP, Kanban Are The Most Used

- 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).

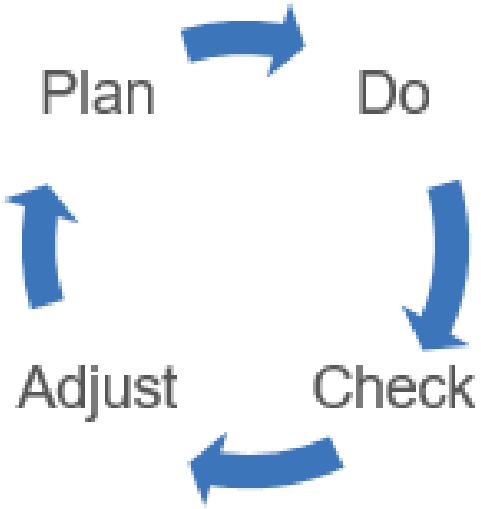
- Implement the change you decided on in the plan phase.

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.

SAFe Overview

Don’t Forget Lean Principles

SEVEN WASTES

#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 Lean-Agile Principles

Lean principles focus on more of an enterprise level than Agile’s
Lean principles are compatible with Agile and DevSecOps

Success at Scale

Source: 13th Annual Survey on State of Agile, Version One, April 2019

Difference between SAFe Configurations

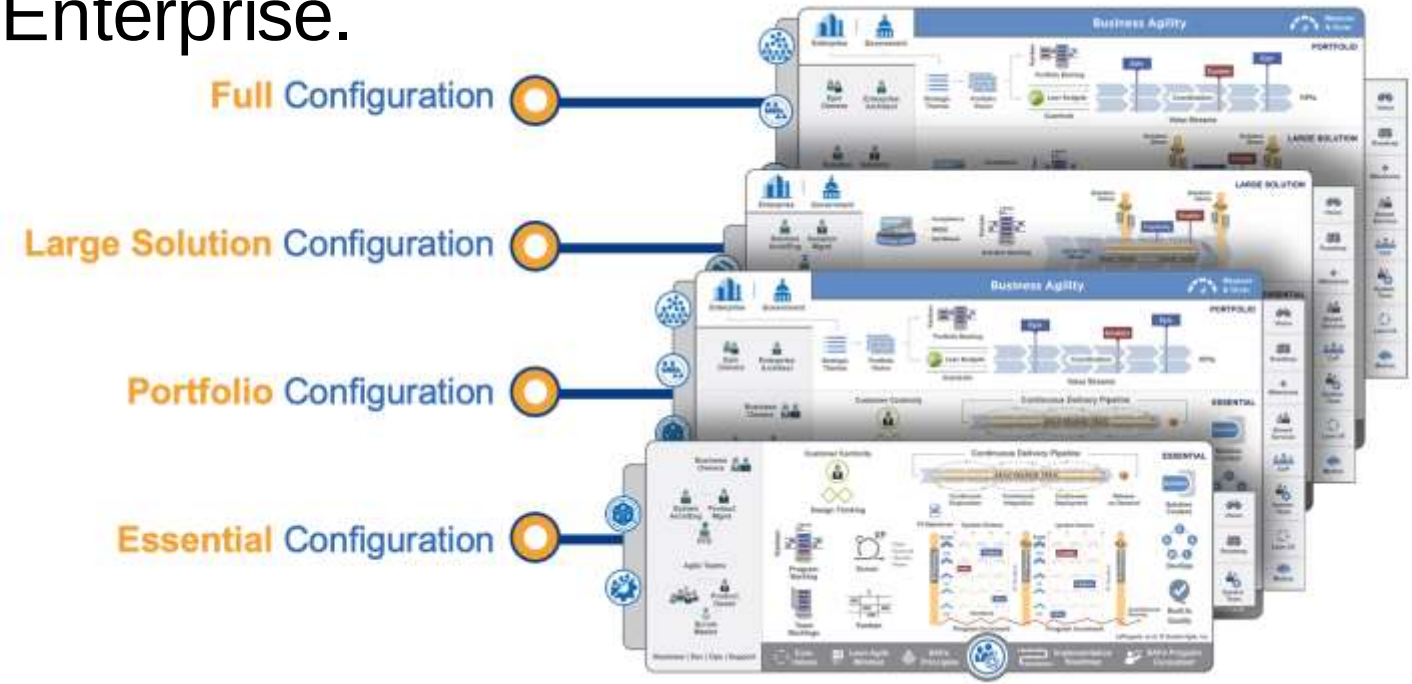
All support the developers

Layers for coordination not command/control

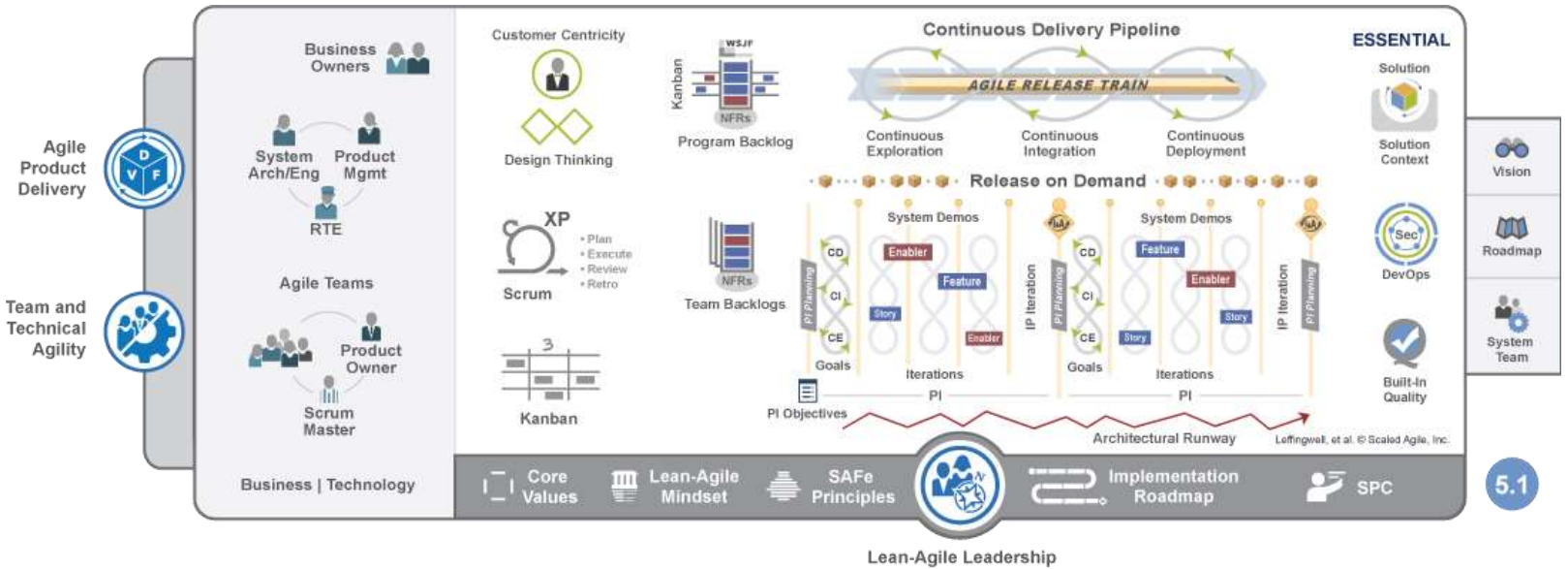
- Essential (the basis for success)
- Portfolio (adds lean portfolio governance)
- Large Solution (coordinates ARTs with a solution train)
- Full (some enterprise require full SAFe) Systems of systems

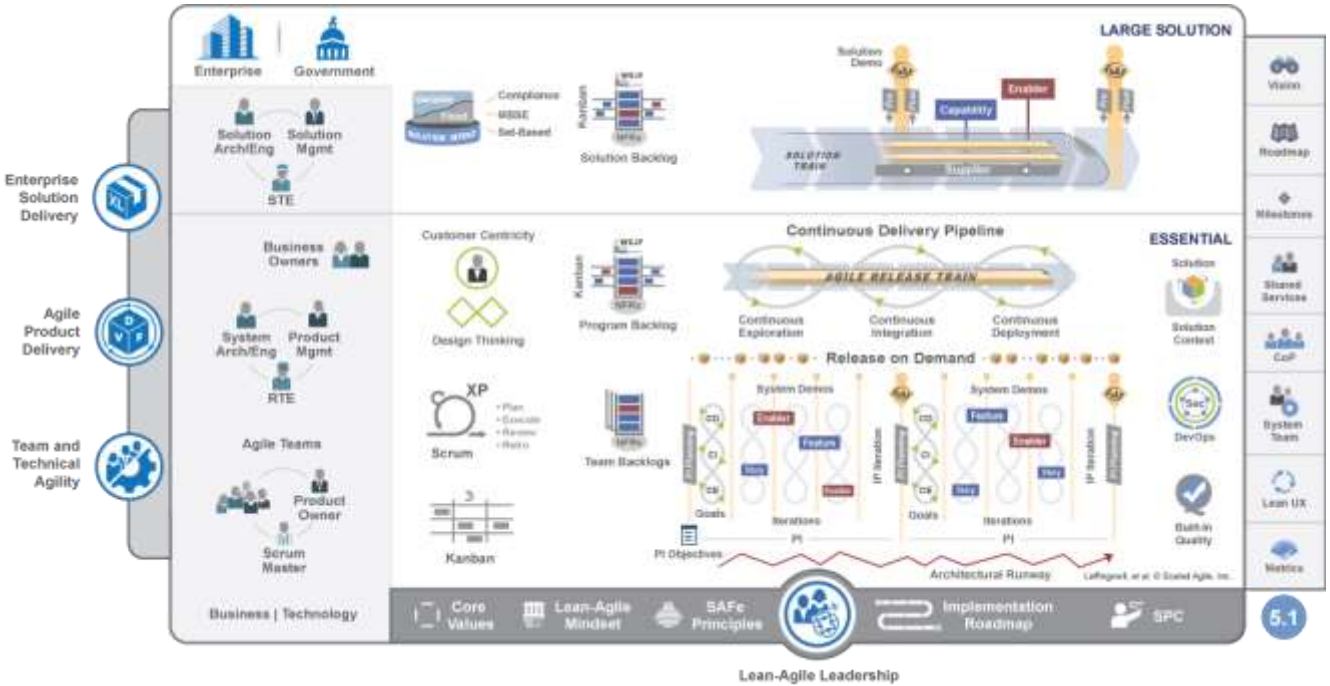
SAFe configurations

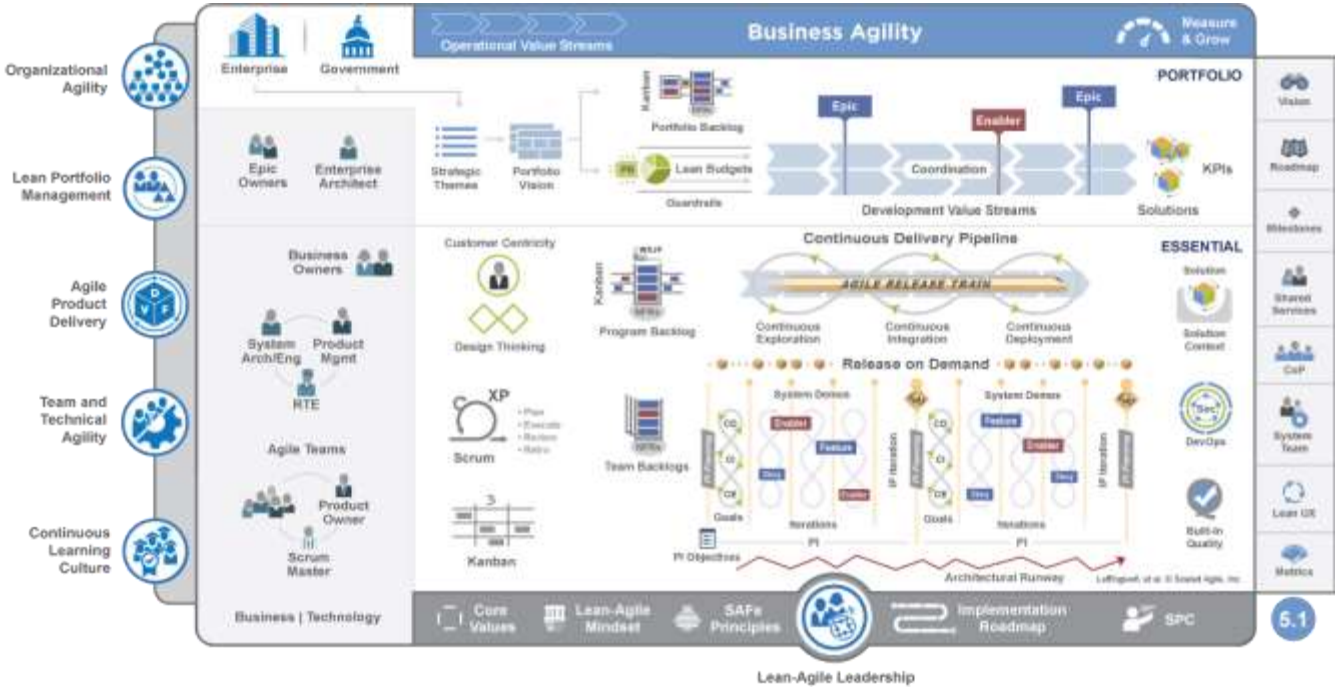
Four configurations provide the right solution for each Enterprise.

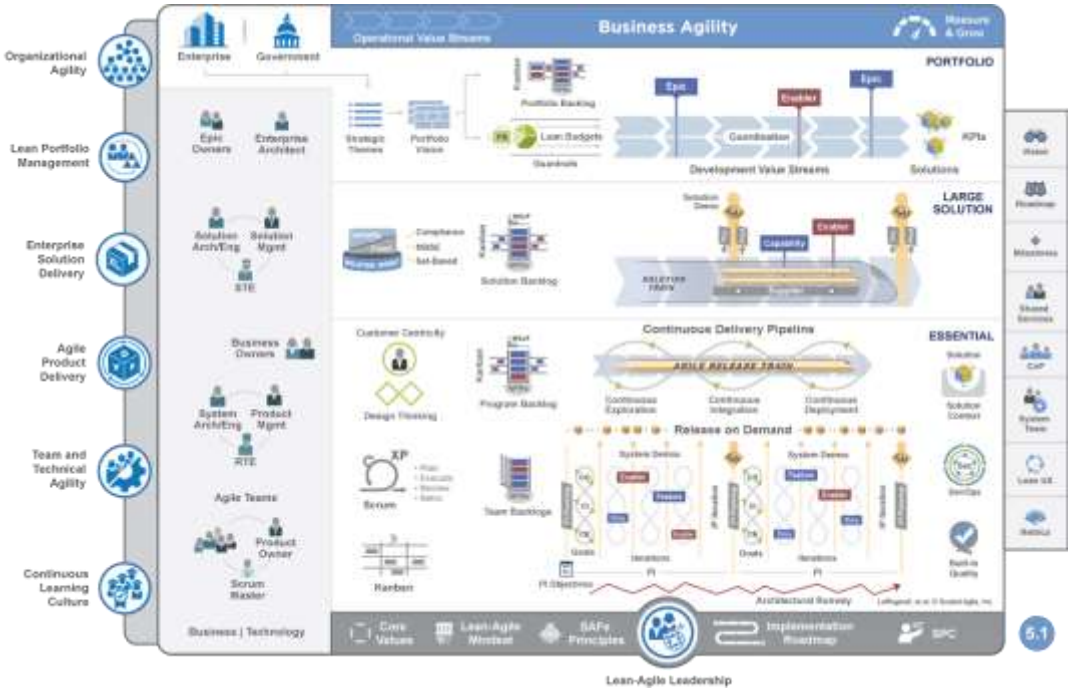


Backup









Agile Principles—Focused on Small Teams-1

1. Highest priority is satisfy the customer through early and continuous delivery of software.
2. Welcome changing requirements, even late in development…
3. Deliver working software frequently, from a couple of weeks to a couple of months...
4. Business people and developers must work together daily throughout the project.
5. Build projects around motivated individuals. Provide environment and support they need…
6. The most efficient and effective method of conveying information to and within a development team is face-to-face conversation.

Agile Principles—Focused on Small Teams–2

7. Working software is the primary measure of progress.
8. Agile processes promote sustainable development…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.

Adapted from http://agilemanifesto.org/principles.html

Useful Interpretation of Agile Principles for Government Settings For Your Reference
(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 For Your Reference
(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 For Your Reference
(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.