Virtual Learning Package 9: Lean & Agile Systems Engineering

Suzanne Miller, Richard Turner, Ph.D., Bart Hackemack
SEI Continuous Deployment of Capability Directorate

May 2020

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.

DM20-0408

Meeting Conventions for Today

Please stay on mute for the lecture portion of the course module

If you are “in” the Skype meeting via web or app, please ask questions via the Chat window.

- A facilitator will collect the questions and either pass them to the instructor if something immediate, or organize them for the Q&A portion of the course module

Those on dial in will enter questions via email to Dave Walbeck -- dtwalbeck@sei.cmu.edu

Instructor will call for participation and discussion at various points. Please remember to come off mute before talking.

When you are done talking, before going back on mute, please say “Over” so others know you are finished.

The lecture part of this session is being recorded. Recording will be turned off during discussions.

Topics the SEI will address in this module include:

- **Introduction (Why Agile SE)**
- **What Agile SE looks like**
- **Agility throughout the V (not just the bottom)**
- **The importance of batch size reduction**
- **Technical review**
- **Model-Based Engineering**
- **Lean SE Applied**

Bottom Line Up Front: Systems Engineering Benefits from Agile and Lean Principles

Systems Engineering

One of SAFe's reasons for popularity in DoD is its explicit inclusion of systems engineering concepts/activities

Agile is easier to implement when systems engineering looks at the whole system from an incremental delivery viewpoint

Integration & Test

Incremental, iterative testing provides both opportunities and challenges for developers and acquirers

Incorporate automated testing into program testing practices as soon as possible (or even sooner!)—it's a key success factor

CM, Security

Configuration management and cybersecurity are two stakeholders that shouldn't, but often are, left out of planning when Agile is being implemented. They are key contributors to successfully delivering incrementally

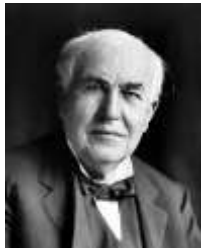
Where do all these Lean Engineering ideas come from?

Scientific method applied to management from F. W. Taylor

Creation of reusable knowledge comes from the Wright brothers

Lean principles come from Henry Ford and Taiichi Ohno (Toyota)

Development cadence comes from Thomas Edison



Learning cycles come from John Dewey

P-D-C-A comes from Walter Shewhart and W. Edwards Deming.

*These principles are the **beginning** rather than the end of improving product development productivity.*

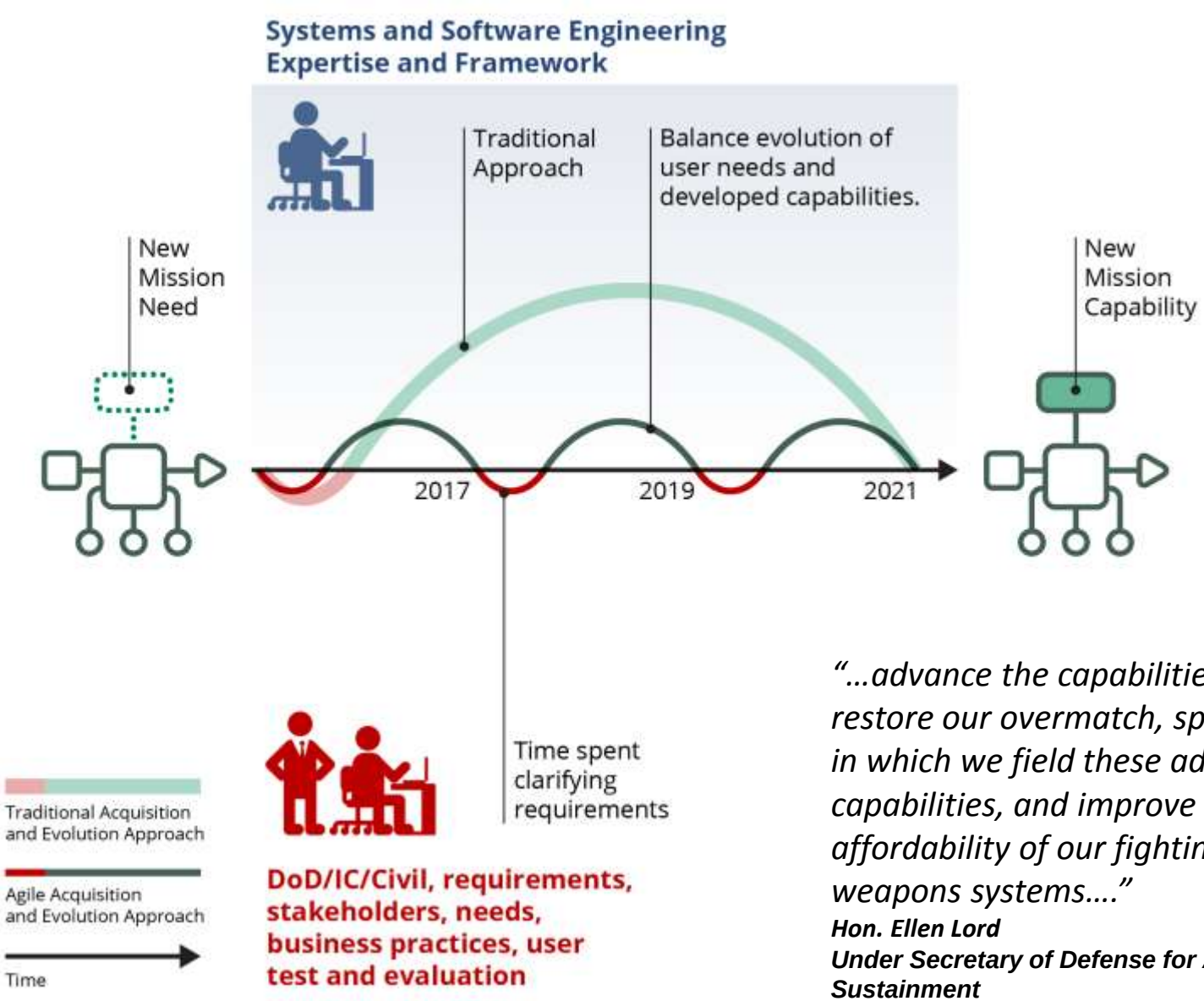
Source: adapted from D. Oosterwal, The Lean Machine, 2010.

Why do SEs Care about Agile?

Deliver performance at the speed of relevance

Streamline rapid, iterative approaches from development to fielding

National Defense Strategy Summary
Jan 2018



"…advance the capabilities required to restore our overmatch, speed the rate in which we field these advanced capabilities, and improve the overall affordability of our fighting forces weapons systems…."

Hon. Ellen Lord
Under Secretary of Defense for Acquisition and Sustainment

Let’s Get Some Perspective – Yours – Here’s a Poll (only participants in the Skype meeting will be able to answer, sorry!)

Which Statements Do You Agree With?

The following slides contain a list of statements quoted or adapted from The Lean Machine, a book about Harley Davidson's adoption of knowledge-based product development.

For each statement, answer as to whether or not you agree with the statement in your context.

Statements About Lean Engineering - 1

1. The purpose of engineering, or any other discipline associated with product development, is to create reusable knowledge that will be used in the development/evolution of products.

2. Product development inherently begins with incomplete knowledge.

3. Integration points and design cycles drive product development/evolution.

Source: adapted from D. Oosterwal, *The Lean Machine*, 2010.

Statements About Lean Engineering - 2

4. The greater the separation in time and space between an action and feedback, the more difficult learning becomes.

5. The speed and effectiveness of learning cycles determines a company's ability to innovate.

6. It is better to be approximately correct and try something than to be precise but wrong.

Source: adapted from D. Oosterwal, *The Lean Machine*, 2010.

Statements About Lean Engineering - 3

7. The amount of prototyping and testing on a project should be proportional to the knowledge gap that needed to be closed to resolve the design problem.

8. It's not the test information you collect that's important, but how it's digested and made visible to create reusable knowledge that's important.
9. The applicability of a solution to current need declines as the time to deliver capability increases

Source: adapted from D. Oosterwal, *The Lean Machine*, 2010.

What does Systems Engineering with Lean and Agile look like?

Systems Engineering Embraces the Agile Manifesto

We are uncovering better ways of developing software by doing it and helping others do it.

Systems…

Through this work we have come to value:

Individuals and interactions over processes and tools

Working software over comprehensive documentation

Customer collaboration over contract negotiation

Responding to change over following a plan

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

agilemanifesto.org

SAFe Lean-Agile Principles Target Systems Engineering Concerns

#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

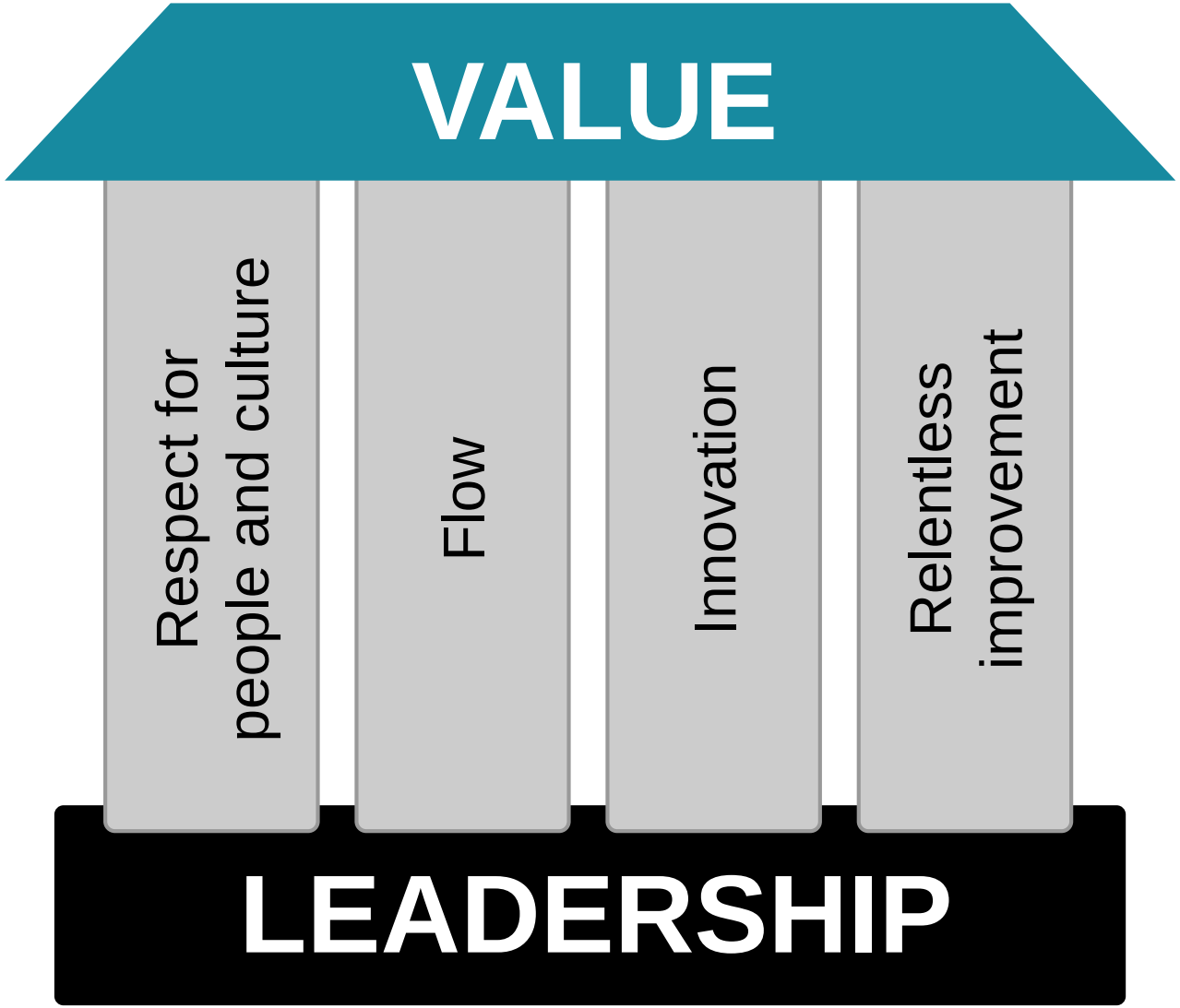
#9 Decentralize decision-making

#10 Organize around value

Systems Engineering Has a Key Role in the House of Lean

Systems Engineering is

- A key designer as well as implementer of FLOW,
- A key focus for RESPECT FOR PEOPLE AND CULTURE
- A key driver of INNOVATION
- A key advocate for RELENTLESS IMPROVEMENT
- Continually focused on delivering VALUE through LEADERSHIP in all these areas



Key Elements of Lean Engineering

Transparent visualization of work

Fast feedback/learning cycles

Small batches

Extreme modularization

Integration, Integration, Integration!

Limit Work in Progress – achieve flow

Multiple design parameter activities to generate knowledge

Postpone decisions to “last responsible moment”

Base decisions on working product, not projective documents

Use models to gain understanding of design parameters

“Big Room” Planning with all relevant stakeholders

Optional homework: map these elements back to the Lean principles…

Wikispeed Car Build…Lots of Examples of Applying Lean and Agile Techniques

Selected items from this video (also in your learning package)

https://youtu.be/lU6FeBw7Pws

Contract first engineering

Extreme modularity

Flexible manufacturing “mini-factories”

Pair work

Test-driven development

Reduce cost of change

Why Systems Engineering for Agile is Challenging

System Engineering Challenges

SE is a holistic discipline, but is

- Traditionally taught (and executed) as a single process with fixed steps
- Perceived as up front, unidirectional and static
- Not fully aligned with SW and other disciplines that:
  - Have become culturally and intellectually biased toward multi-processing, quick decisions and immediate feedback
  - Tend to prefer experimentation to analysis

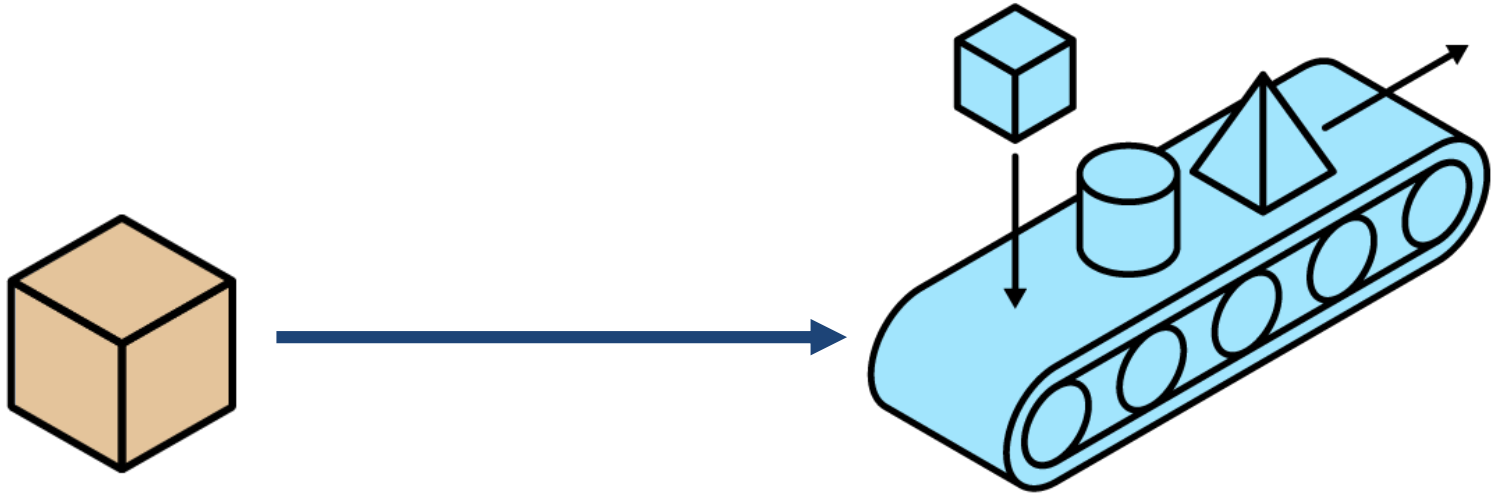
Firm requirements and a logically consistent flowdown of those requirements, with carefully established parent-child relationships, are at the core of present-day system engineering practice. Yet theory suggests that such practices produce results inferior to those which would be obtained using alternative methods.

Michael Griffin, 2010

Griffin, M. D., "How can we fix Systems Engineering?," IAC-10.D1.5.4, 61st International Astronautical Congress, Prague, Czech Republic, 2010.

Collopy, Paul, "Adverse Impact of Extensive Attribute Requirements on the Design of Complex Systems", AIAA 2007-7820, AIAA Aviation Technology, Integration, and Operations Conference, Belfast, Northern Ireland, 2007.

It Isn’t Just the Requirements That Change!



Building a
Box

Engineering a (most of the time) single product defined in one large batch and delivered one time

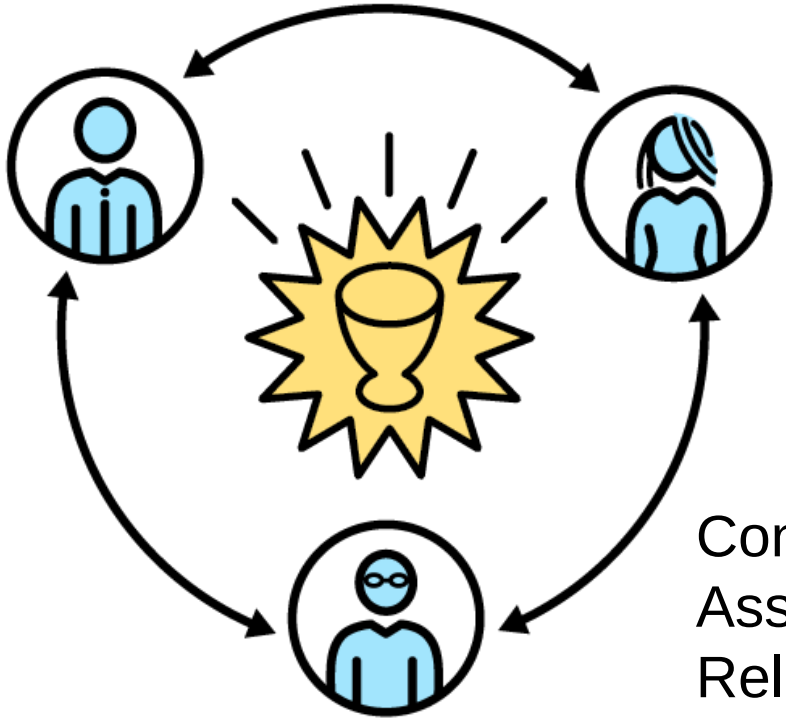
Building an ongoing
delivery stream of capabilities

Contracting for a product that is not completely defined at the start, and explicitly evolves and is delivered multiple, potentially many times

Different SE Interactions are Needed to Support Evolving Products Delivered Multiple Times

Complete, Correct and Contracted-for Requirements and Approved IMS

Continuous Engineering Feedback

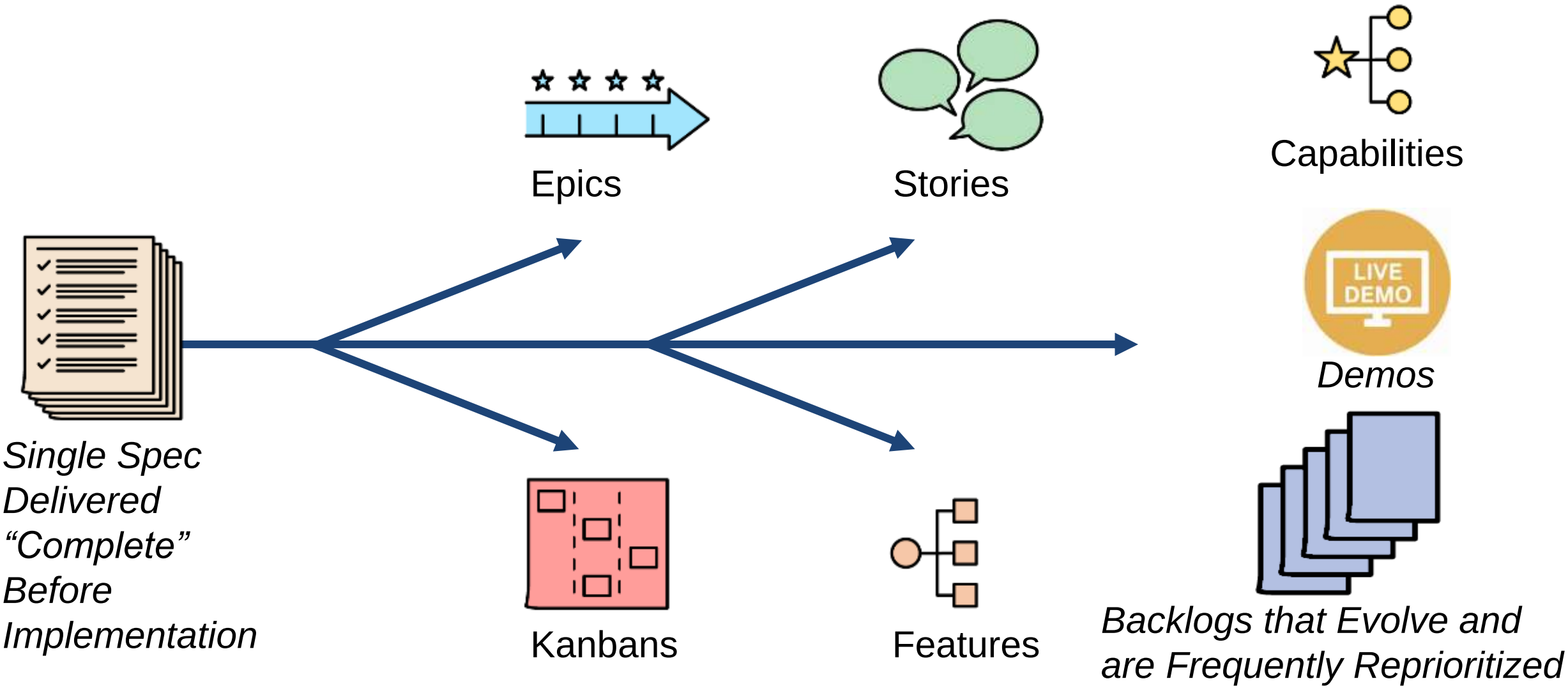


Constant Collaboration and Orchestration

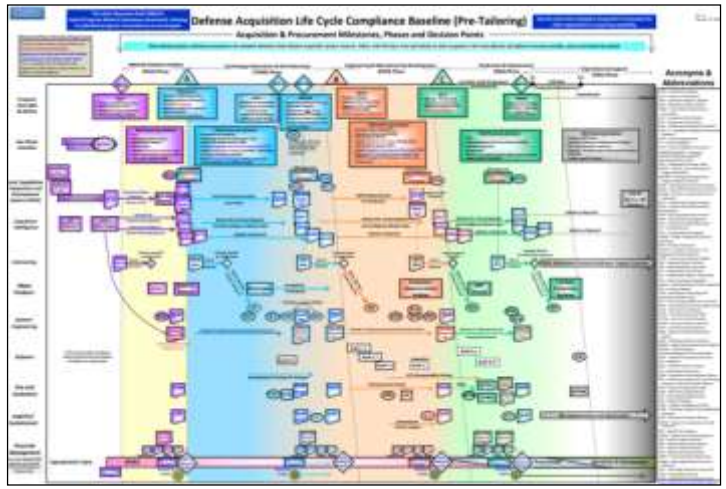
Continuous Assessment of Relevant Value

SE is continuously involved in all these interactions

Different Artifacts are Delivered and Validated



Different Delivery Modes/Cadence Mean Different SE Strategies



Date-based Milestones Emphasizing Completion of Activities

Increment-Based Demos
Definition of Done
Continuous Integration & Test

Flow-based Milestones Emphasizing Completion of Product Elements

Team Increment

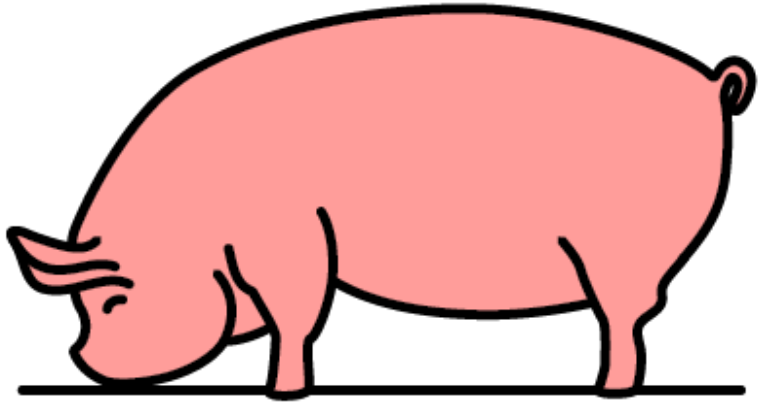
System Increment

Solution Increment

Release/ Deployment to Stakeholders

Moving to Government "Owning" the Technical Baseline

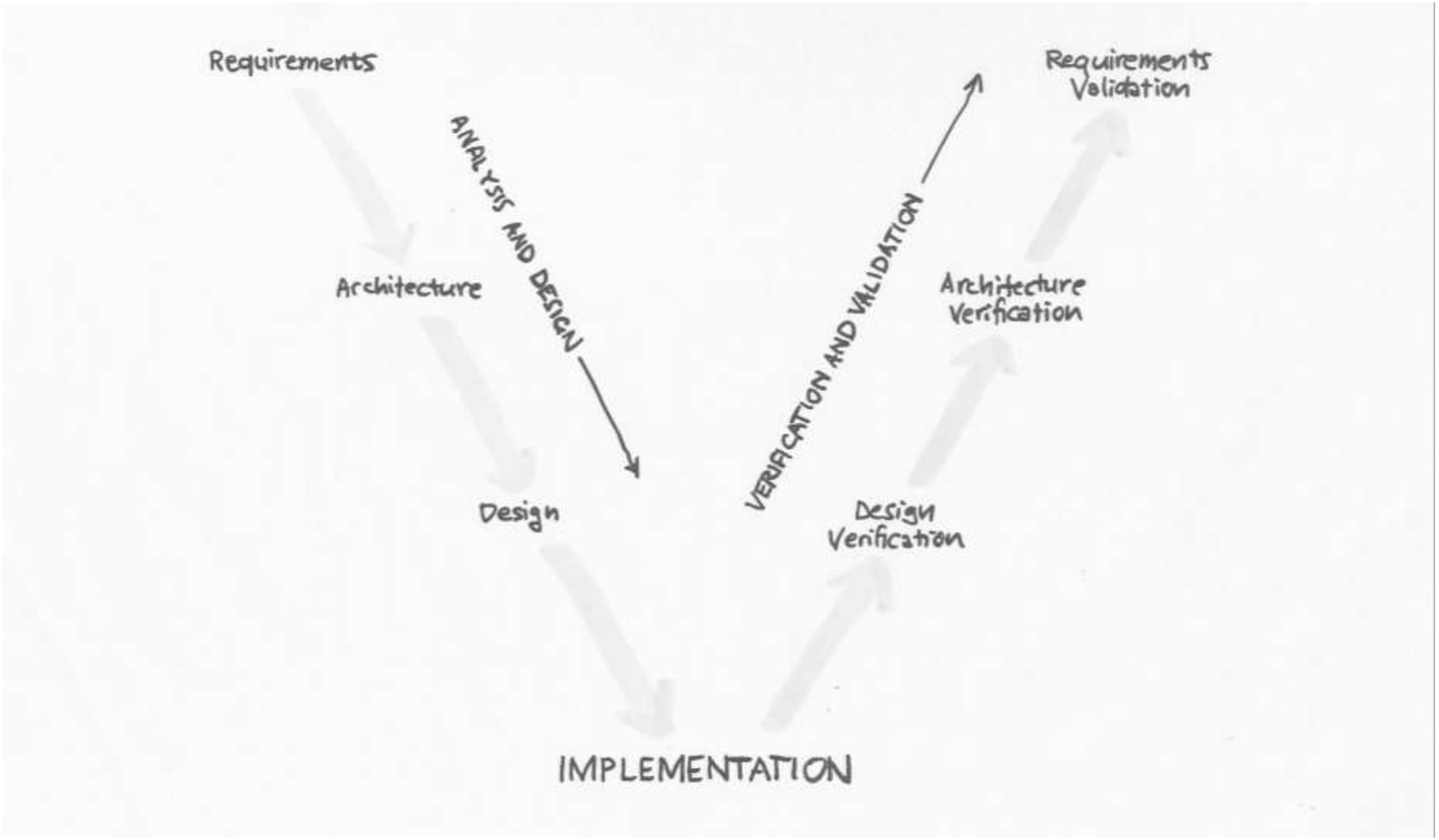
Gov't is "All In" All along the way—Systems Engineering is a Key Player in making this work!!!



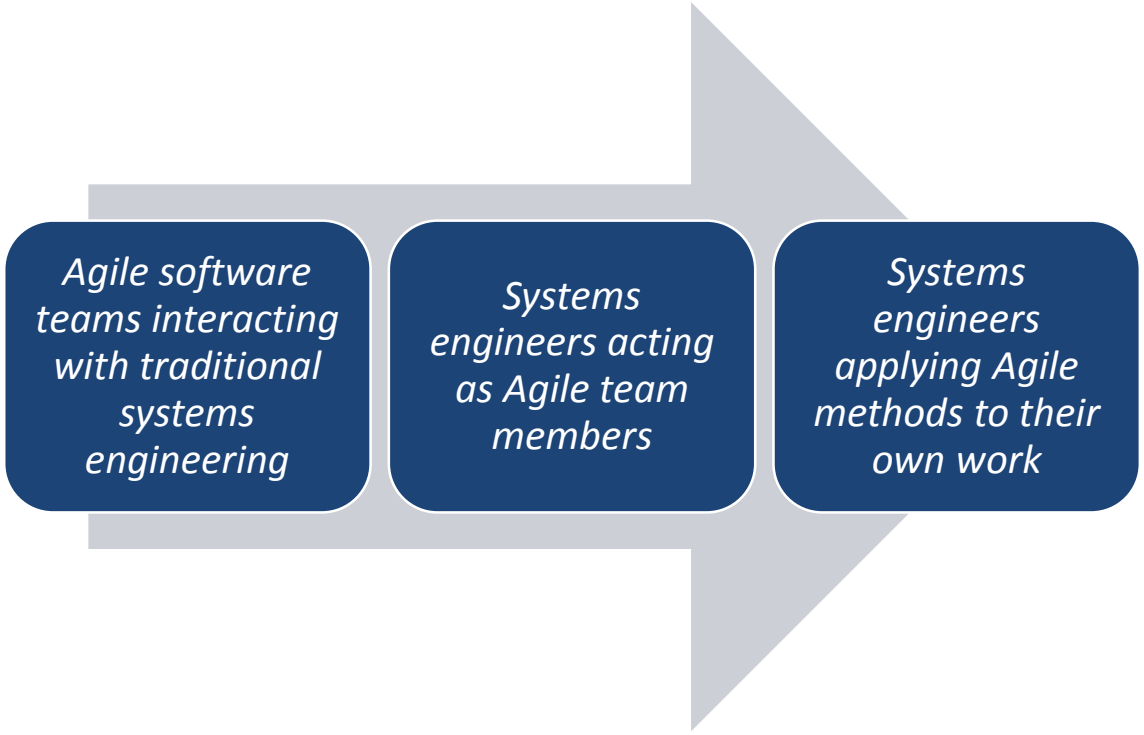
If this is where we are, let's make sure our systems engineering practices support the collaborative relationships needed for this to work!

Agile Throughout the V (not just the bottom!)

Traditional Systems Engineering V



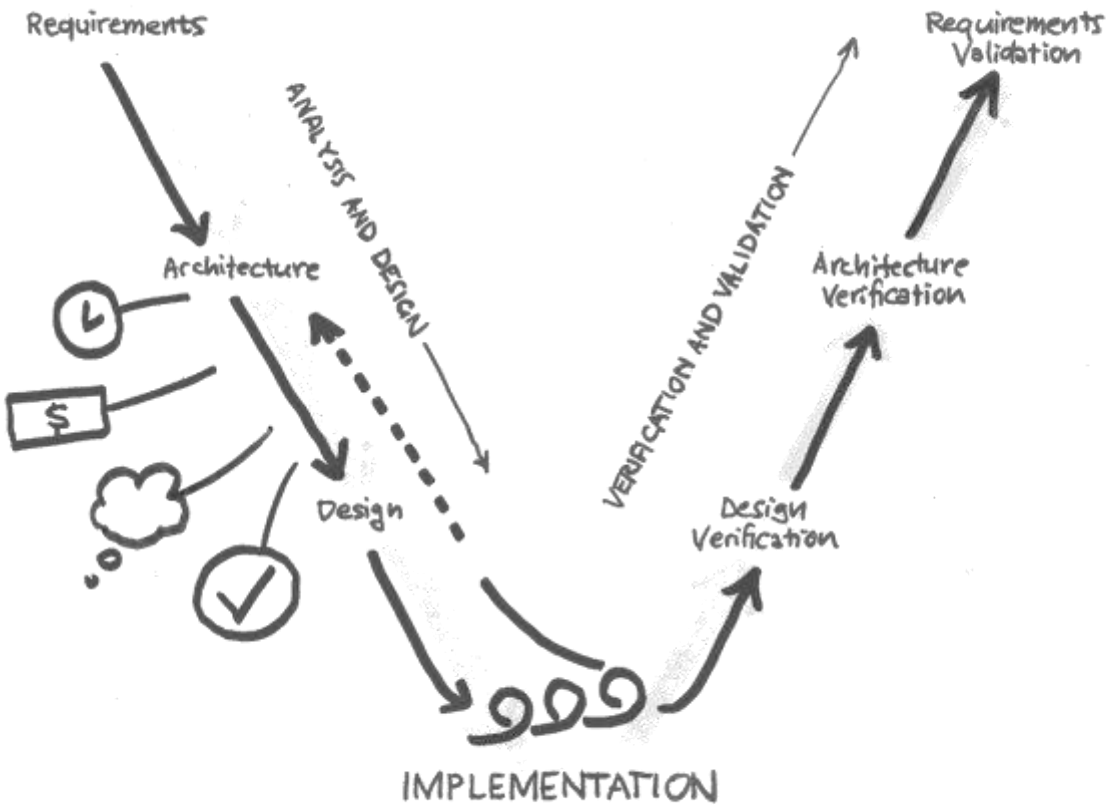
Systems Engineering with Agile Teams



In reality it is a continuum:

- As Agile teams demonstrate success, systems engineering teams and leaders need to get engaged
- The traditional approach leads to applying agile and lean at the "bottom" or apex of the V.

What Happens When "Large Batch" Systems Engineering Meets "Small Batch" at the Bottom of the V?



This approach:

- Completes most specification activities to fairly low level detail before development
- Misses opportunities for small batch learning cycles, limiting the ability to take advantage of these early learning opportunities
- Makes an assumption that upfront, large batch work for requirements, analysis and design is complete leading to "*False-positive feasibility*" and "*Fixed Intent*" too early
- Leads to cost and schedule impacts when adjusting these specification as a result of short learning cycles during system development

Large Batch to Small Batch

Evolving Systems Engineering to Small Batch in a Hardware-Dominant System?

Small Batch and Systems Engineering

Systems Engineering adopting a Small batch approach for system specification and verification/validation enables:

- An incremental and iterative approach
- Ongoing collaboration with stakeholders from other areas of the V (Test, Verification, Evaluation, Certification, Users, etc) throughout system specification
- Making Systems Engineering decisions at "last responsible moment", eliminating effects of False-positive feasibility and Fixing Intent too early
- Taking advantage of small batch learning cycles to create fixed intent for the current and near term increments
- Systems Engineering as a service enables integration of rapid learning cycles among various stakeholders

But How is This Possible with Complex Embedded Systems?

We have hardware to consider, after all!!!!

You can't build a modular car, or ship, or….

- Actually, modular cars and ships are being built
  - German ship builders have been modularizing ship construction for at least 15 years
  - Wikispeed is a proponent of "extreme modularization" for it's CCC crowdsourced vehicle

How do they do it?

- Interfaces, interfaces, interfaces!!!!
- Modularizing the manufacturing process (think tool and die that's modular and evolvable too!) not just the components
- THINKING in small batches, not just decomposing a big thing into a smaller set of things
  - What's the smallest, cheapest thing we can do to learn if this is the right direction?
    - A powerful question to shift to small batch thinking

Incremental delivery of H/W by increasing fidelity. Emulators, Simulators, etc

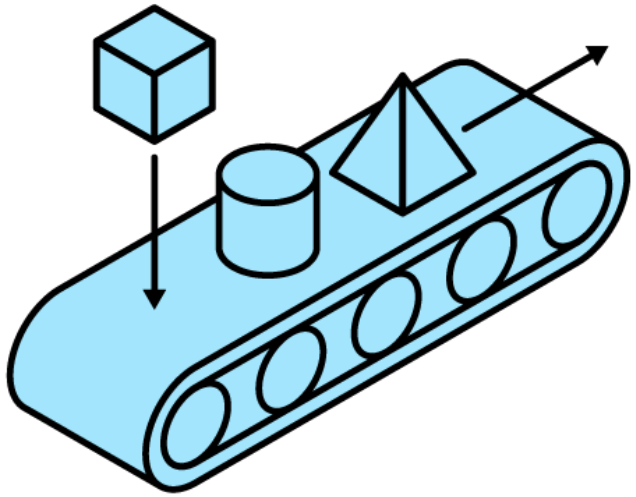
Back to the “Continuous Delivery Capability” Idea

How do I change my thinking to engineer for continuous delivery capability vs a single huge product?

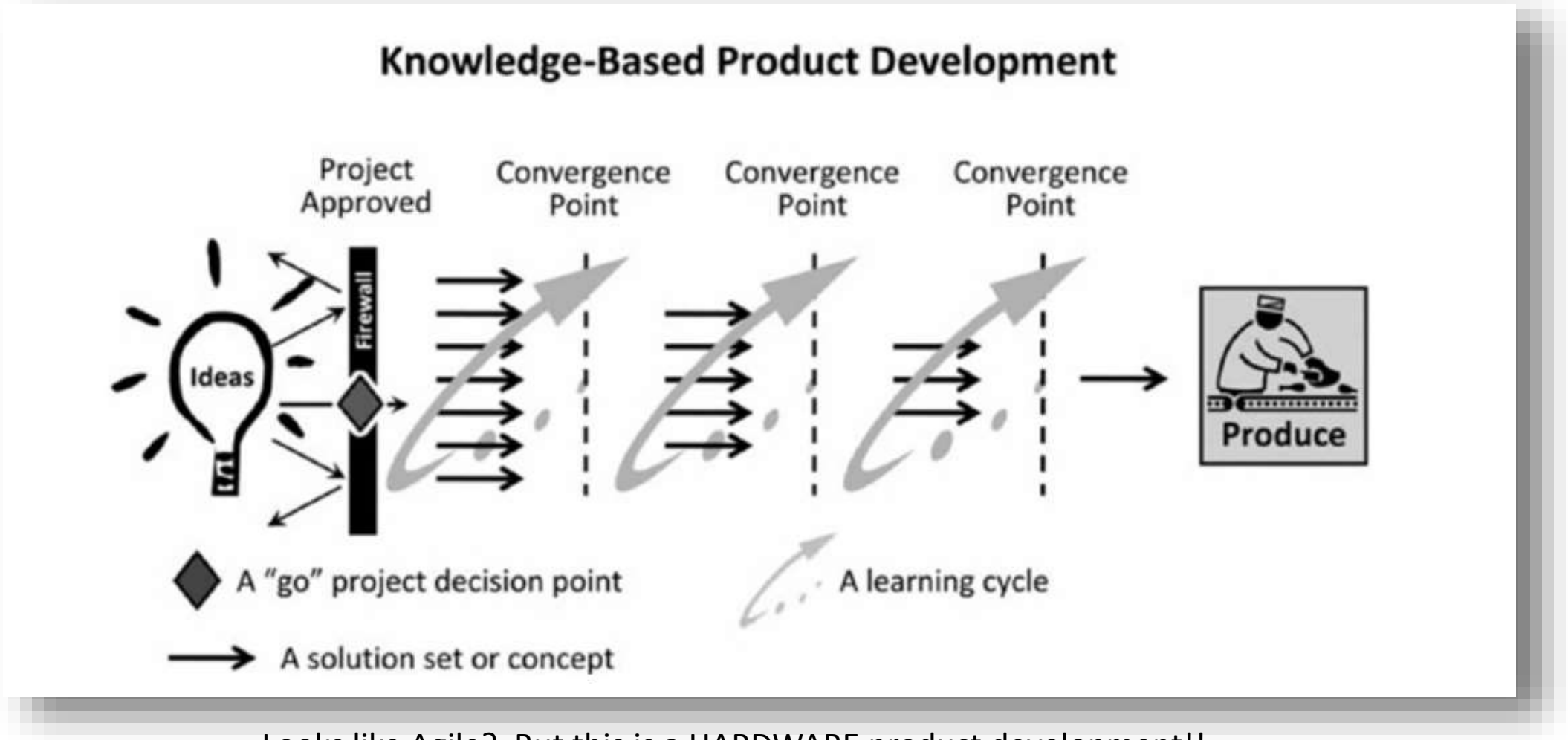
How do I change my early engineering activities to focus on “what do we need to learn before committing to a particular design or manufacturing approach”?

How do I change my engineering thinking to “write the test first”, the “test is the specification”?

How do I leverage models, mission threads, other software-y approaches to achieve extreme modularity, test driven development, etc?



Knowledge-Based Product Development a la Harley Davidson



Looks like Agile? But this is a HARDWARE product development!!

Source: D. Oosterwal, *The Lean Machine*, 2010.

Technical Review

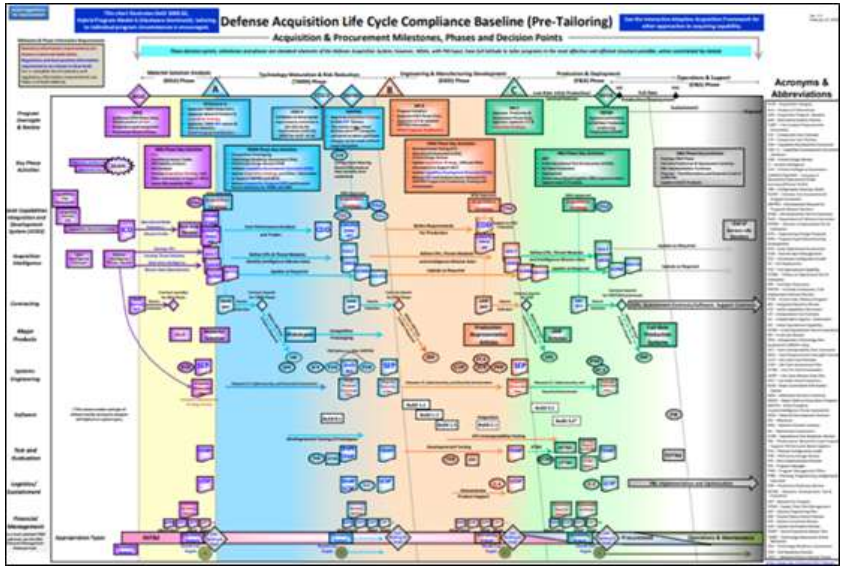
SE Responsibilities for Technical Review

Traditionally, SE is responsible for:

- Reviewing technical plans, interim documents, trades documentation, V&V plans
- Monitoring KPPs, TPMs, architectural compliance and interfaces
- Validating technical progress at major milestones
- Artifacts reviewed are generally large

Review cycle is generally from 30-120 days depending on the target

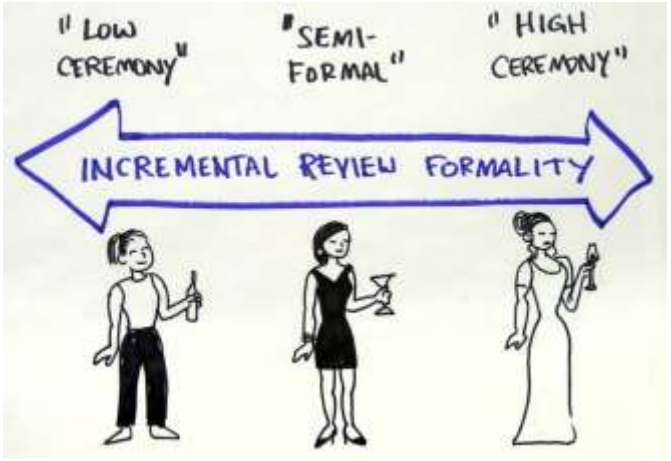
- Developer provides review target
- SE reviews and comments
- Developer makes changes and comments
- SE Reviews and comments
- …



SE Technical Review in Lean and Agile Organizations

In Agile and Lean organizations, SE is continuously supporting the development and use of evidence to measure progress

- Smaller batches and shorter intervals between review activities (a few weeks versus months/years)
- Focused on demonstrations (evidence) rather than documents
- Closer, continuing engagement with stakeholders
- Ensuring that the evidence presented is sufficient for the determination requested



Evidence- and Risk-based Major Technical Review (Boehm, 5000.02)

At major system reviews, technical review has the goal of supporting decisions as to acceptability, direction, opportunity and risk

In general, evidence should fully support that the system will

- Satisfy the requirements
- Support the operational concept;
- Be buildable within the budgets and schedules in the plan;
- Generate a viable return on investment;
- Generate satisfactory outcomes for all of the success-critical stakeholders;
- Resolve all major risks, treating evidence shortfalls as risks, by covering them by risk management plans
- Serve as basis for stakeholders' commitment to proceed.

Boehm, Lane, Koolmanojwong and Turner, *The Incremental Commitment Spiral Model,* Addison Wesley, 2014

SEI Found 3 Patterns in Agile Settings for PDR, CDR Design/Execution

Pattern A

- PMO uses traditional PDR and CDR in each block as traditional milestone events

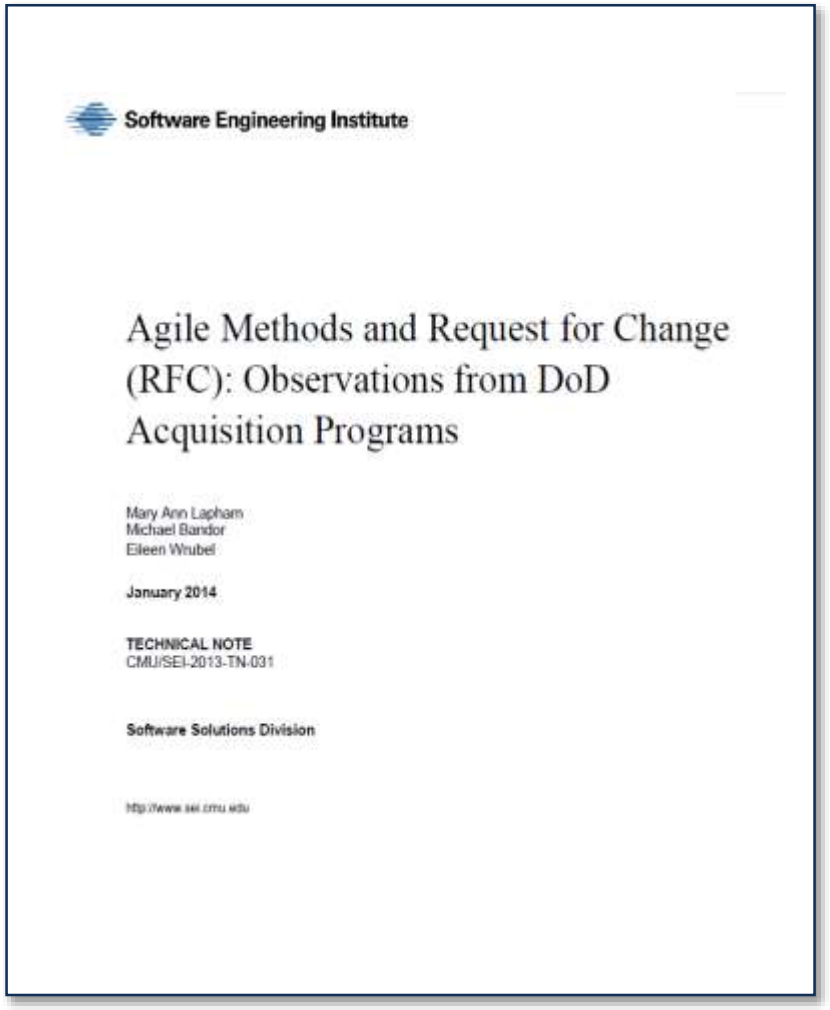
Pattern B

- PMO team participates in each of multiple Preliminary and Critical Design Working meetings (PPDW/PCDW)* – one per iteration
- PDR and CDR are still held at some level of technical discussion and also include management elements

Pattern C

- PMO technical staff (engineers) participate in each PPDW/PCDW (per iteration)
- PDR and CDR become management level reviews
- No technical detail is discussed in PDR and CDR other than a summary for management

*PPDW=Partial Preliminary Design Walkthrough;
PCDW=Partial Critical Design Walkthrough


Software Engineering Institute

Agile Methods and Request for Change (RFC): Observations from DoD Acquisition Programs

Mary Ann Lapham
Michael Bandor
Eileen Wrubel

January 2014

TECHNICAL NOTE
CMU/SEI-2013-TN-031

Software Solutions Division

http://www.sei.cmu.edu

Continuous Technical Review

In the agile and lean technical reviews, the emphasis is more on the completion of the agreed upon task and the actual value achieved versus expected

- Definition of done should include the evidence required
- Continuous integration and automated testing create a good deal of evidence
- Occurs more frequently with a much smaller scope
- Inherently builds up the evidence for milestone reviews (if required)

The way that we achieve evidence-based risk reduction is to focus on completion of value-added work

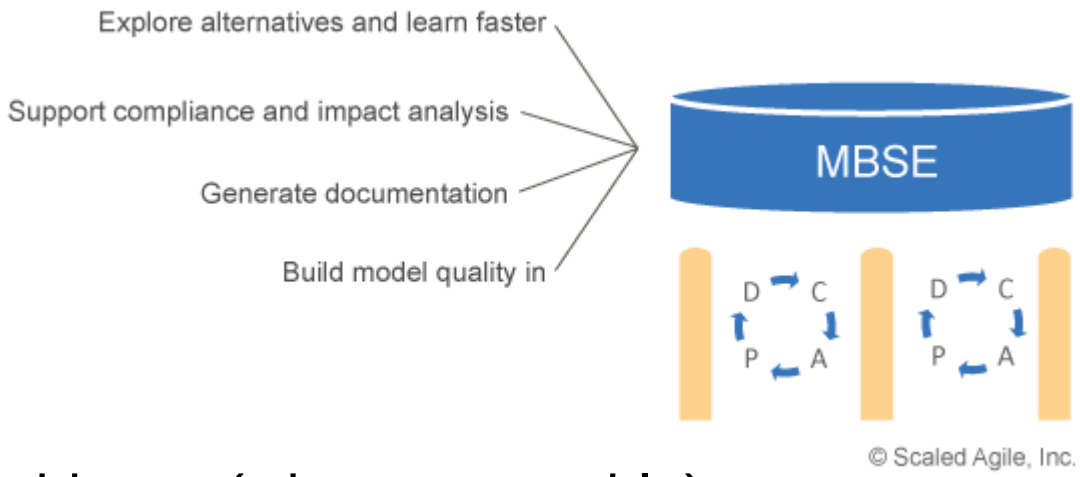
Model-based Engineering

Definition of MBSE

Model-Based Systems Engineering (MBSE) is the practice of developing a set of related system models and simulations that help define, design, analyze, and document the system under development.

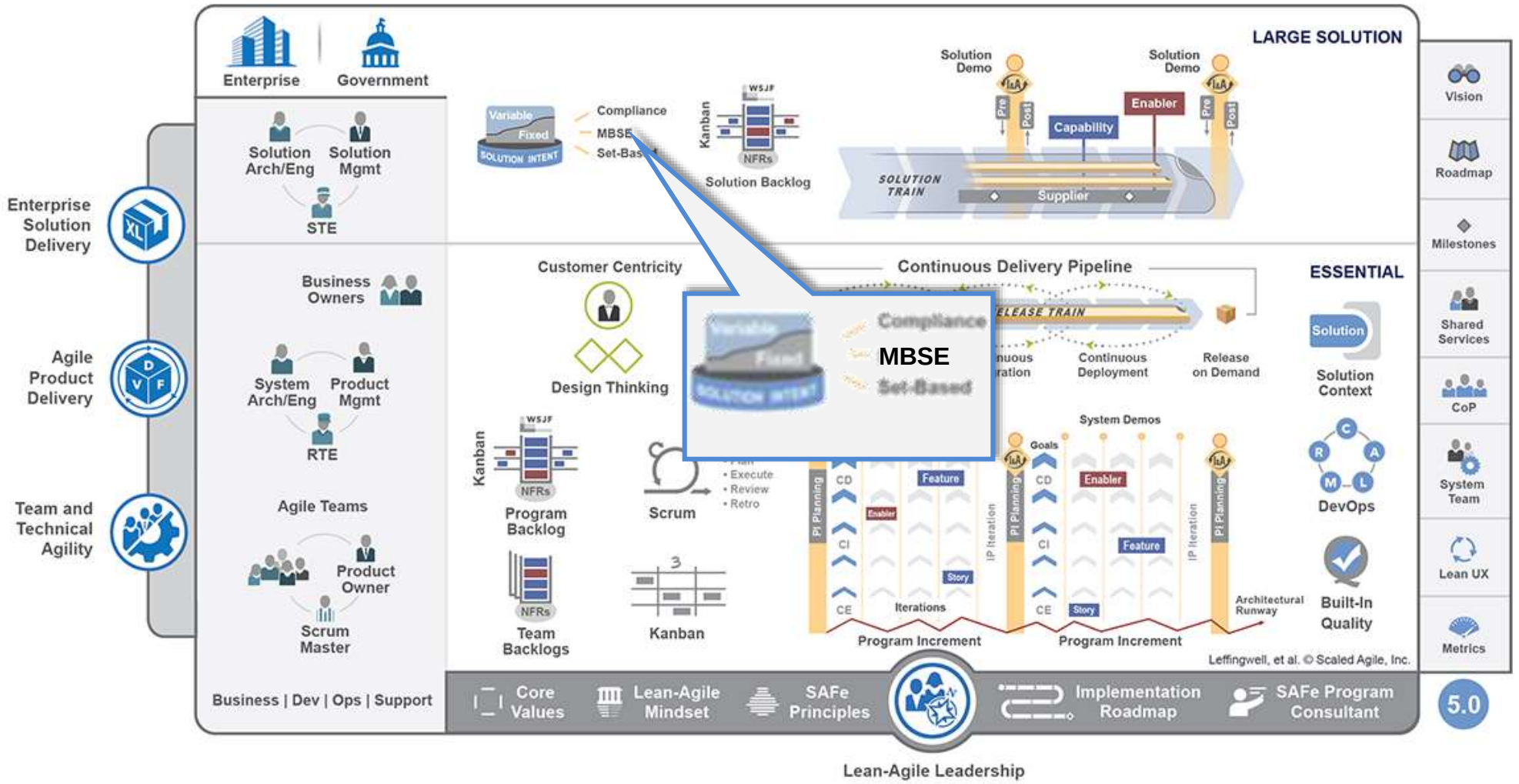
Models allow acquirers and developers (and SEs) to virtually prototype, explore, and communicate system aspects, while significantly reducing or eliminating dependence on traditional documents

Lean and agile, with the emphasis on short learning cycles, continuous value delivery and stakeholder involvement are well-served by the use of MBSE to:



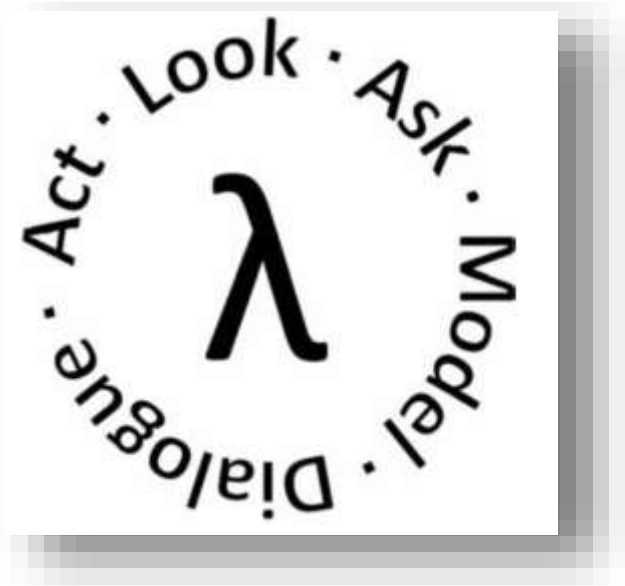
- Accelerate learning
- Bridge the physical and virtual worlds
- Support decision making by providing additional evidence (when executable)
- Support continuous integration and system integration labs.

SAFe 5.0 includes MBSE as a large system enabler

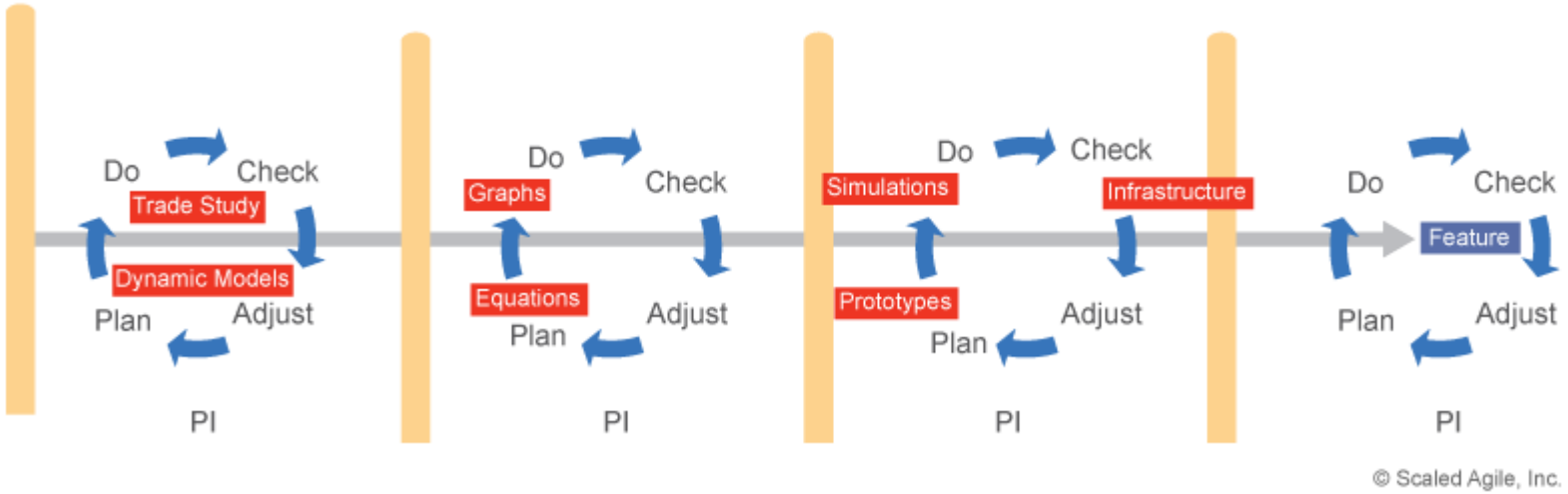


Modeling Supports the Learning Cycle, Expands PDCA

LAMBDA



SAFe - PDCA



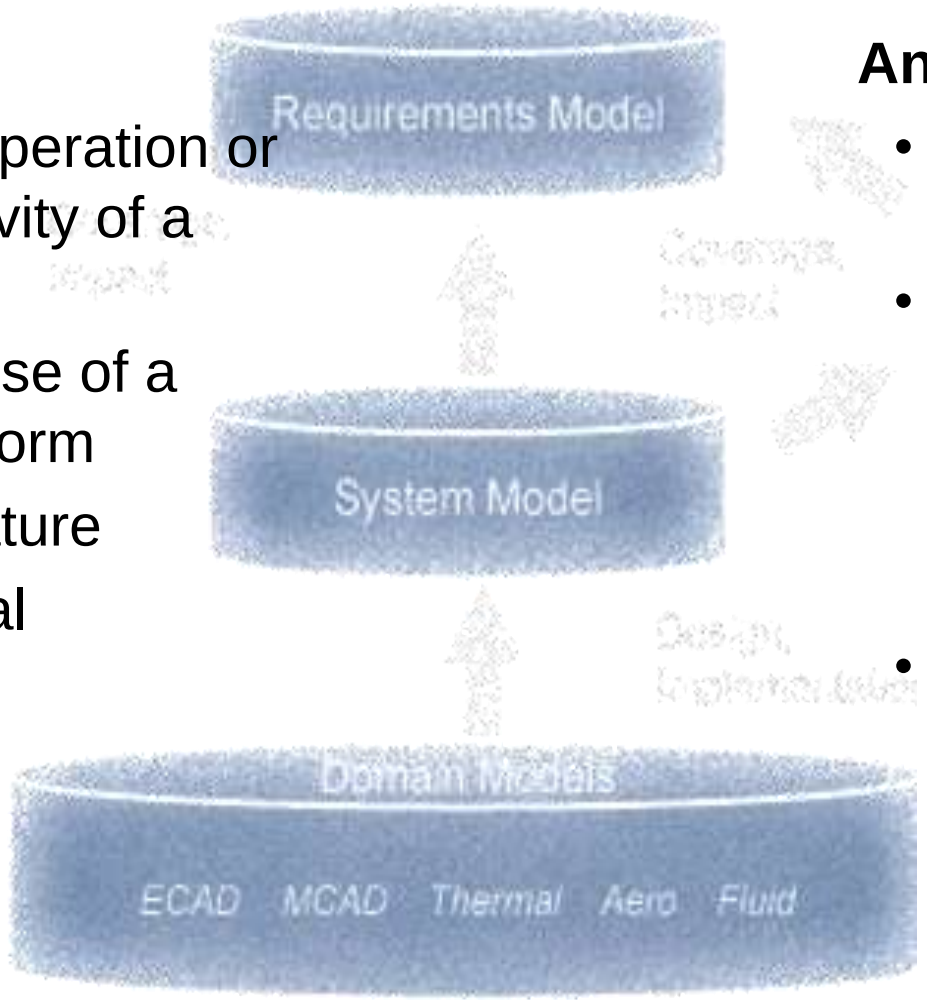
OSD "Digital Engineering" strategy is encouraging greater use of engineering modeling throughout the system life cycle

Source: D. Oosterwal, *The Lean Machine*, 2010.

Understand Two Different Focuses for Modeling

Descriptive

- Used to illustrate the operation or structure and connectivity of a system
- May capture the purpose of a system in algorithmic form
- Often conceptual in nature
- Examples: Architectural drawings, Visio, PPT engineering…

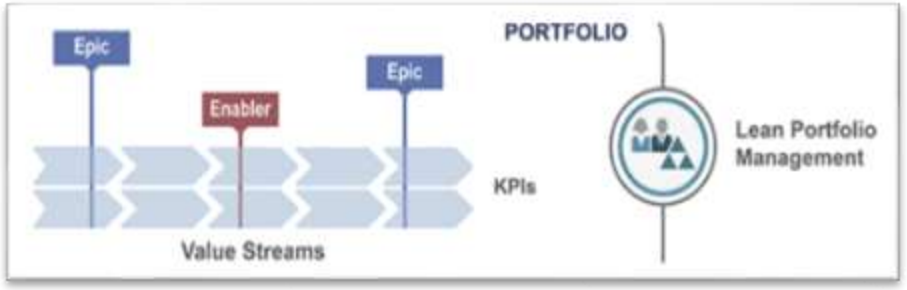


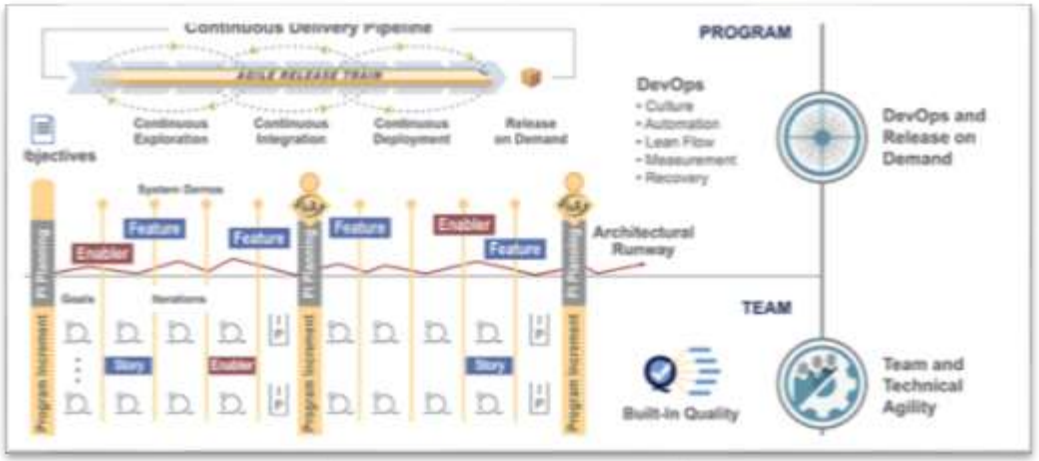
Analytical

- Used to evaluate some behavior or performance
- Often a simulation of a conceptual model with either measured or expected values or equations assigned to model components
- Examples: Physics-based component models, AADL, FMECA

Lean SE Applied

Where Systems Engineering Fits in the SAFe View of Roles?





Portfolio: (MAJCOM is primary actor). Enterprise level staff who are involved in defining and prioritizing strategic themes of the command (typically interacting at PEO level and higher). *Systems engineers likely to be involved in constructing the Solution Context that drives which Epics move forward.*

Large Solution: (SPO/MAJCOM are primary actors). For government, *Solution Architect* and *Solution Managers* are primary roles to fill/translate.). *Systems engineers are the Solution Architects team building the Solution Context, Solution Intent and defining the models that lead to set-based design, as well as architectural runways that will be needed by the ARTs*

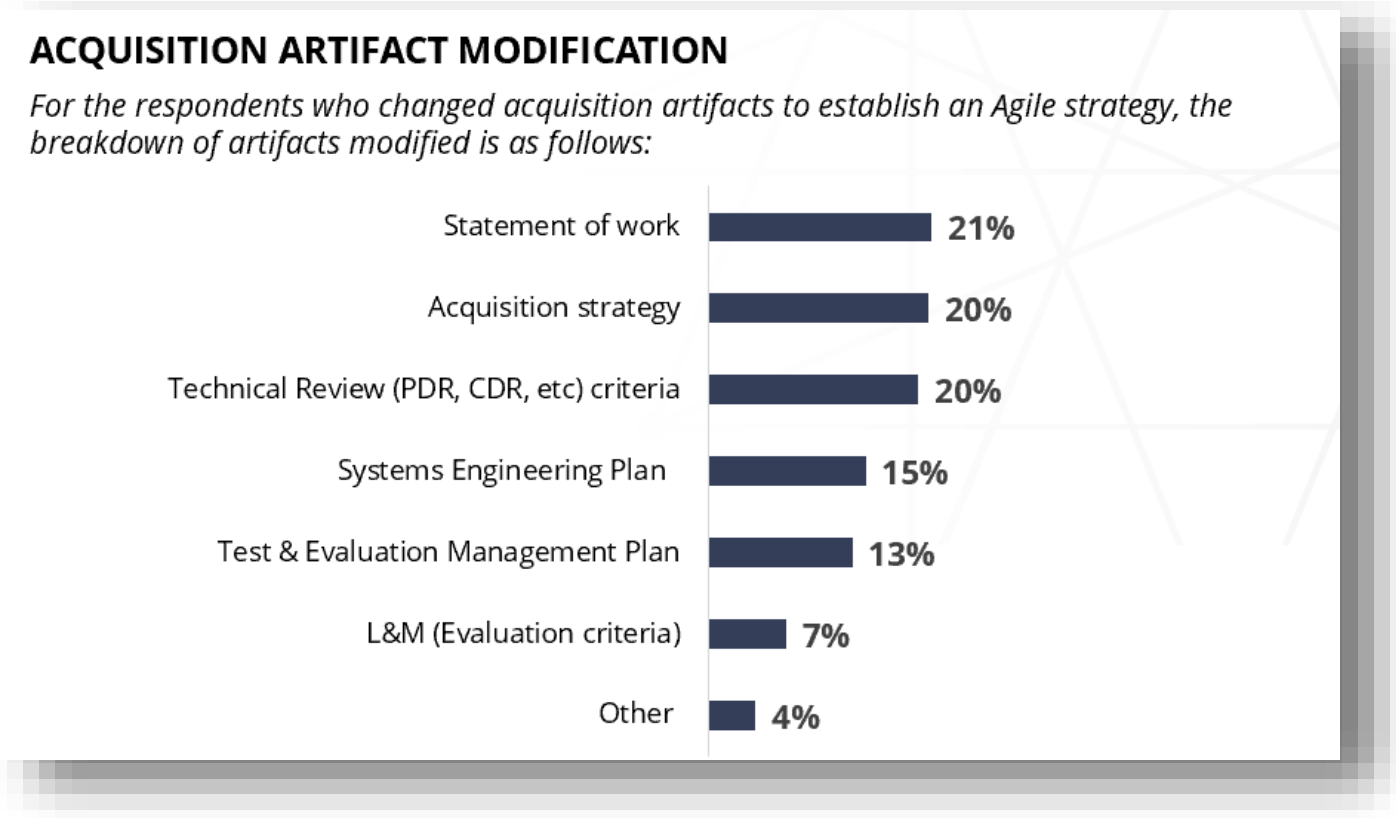
Program/Features: (SPO, Contractor, and their Suppliers are primary actors at Release Train level) For government, *Product Manager* and *System Architect/Engineer* are primary roles. *Systems engineers and architects define the architecture runways, build in the learning points needed to understand how integration needs to occur, and work with the System Team to ensure that hardware and software-based integration is occurring productively.*

Team: (Contractor and their Suppliers are primary actors at Team level. Contractor provides execution team members, *Scrum Master*, and *Product Owner* roles. *Systems Engineers tend to be SMEs in this layer, and typically review technical progress via the Iteration and System Demos.*

Data from SMC

In their first State of Agile report, they asked programs (who are using Agile) which acquisition artifacts they needed to change to achieve their Agile goals

- ***Note two typical systems engineering documents – Systems Engineering Plan and Technical Review criteria— are called out as needing to be reviewed and potentially changed to support Agile and Lean mindsets and practices***



Summary

Agile+Lean is Key to the System "Sprinting", Not Just the Software

Reduce work in progress (completion of value-added functionality trumps a large batch of ongoing work that has no defined end point)

Small batch sizes (allow for change incorporation with least rework and better estimation)

Incorporate cost of delay and value of information in planning (provides greatest overall value soonest)

Defer decisions until the latest reasonable time (Decisions limit adaptability to change; some early decisions, such as architecture, are key enablers)

Manage flow rather than tasks (Maintain a cadence of delivery rather than forced scheduling of individual tasks)

A Few Context Considerations to Get Started Thinking About Lean SE

Are we engineering for:

- Major hw/sw system, complex system of systems (implying dependencies with other DoD programs)?
- COTS/GOTS solutions that need to be configured/glued together?
- Existing legacy system enhancements and transition to modernized system?
- Transitioning the system and program from traditional development to Agile development, including leveraging existing acquisition artifacts (CDD, etc.)?
- Govt as integrator with Govt owning the technical baseline?
- Services to provide enhancements of legacy system and development of replacement modernized system, rather than contracting for an end product?
- Contractor to provide Agile development and DevOps as part of an end product delivery?

What's in it for SEs

Better engineering by reducing the amount of information you have to process at one time

>> Incremental Development

Higher likelihood of timely delivery of useful and usable capability to the field

>> Eliminating false-positive feasibility

Continuous flow of value to the stakeholder (including the end user)

>> Low WIP, Small Batch size

System Engineering commitment throughout value delivery

>>Lean SE throughout the V

Help Us Improve the Virtual Schoolhouse Experience

Please take a moment to complete the Virtual Schoolhouse Student evaluation:

https://sei.az1.qualtrics.com/jfe/form/SV_6ulmAjyMFwyk8Jv

Thanks!!!

Contact Information

Stephen Beck
SEI LRSO Technical Lead
SSD/TSAPP
Email: srbeck@sei.cmu.edu

David Walbeck
SEI LRSO Team
SSD/TSAPP
Email: dtwalbeck@sei.cmu.edu

Suzanne Miller
Principal Researcher
SSD/CDC
Email: smg@sei.cmu.edu

U.S. Mail
Software Engineering Institute
Customer Relations
4500 Fifth Avenue
Pittsburgh, PA 15213-2612
USA

Customer Relations
Email: info@sei.cmu.edu
Telephone: +1 412-268-5800
SEI Phone: +1 412-268-5800
SEI Fax: +1 412-268-6257