28th Annual INCOSE international symposium

Washington, DC, USA
July 7 - 12, 2018

Synchronizing SE and Implementation in Lean-Agile Programs

Robert Maione
Techna Systems, Inc.
rmaione@technasystemsinc.com

Techna Systems Inc.

Mark A. Wilson
Strategy Bridge International, Inc.
mwilson@strategybridge.com

StrategyBridge®

Purpose and Assumptions

- Describe a technique to better manage value delivery in Lean-Agile development programs
- Make the case that the SE community must foster better collaboration between developers and systems engineers.
- Our assumptions for this presentation:
  - You already have a basic familiarity with Lean and Agile philosophies and methods.
  - We are addressing issues that pertain to large-scale, software-intensive systems where architecture matters.

Context: Large-Scale, Software-Intensive Systems

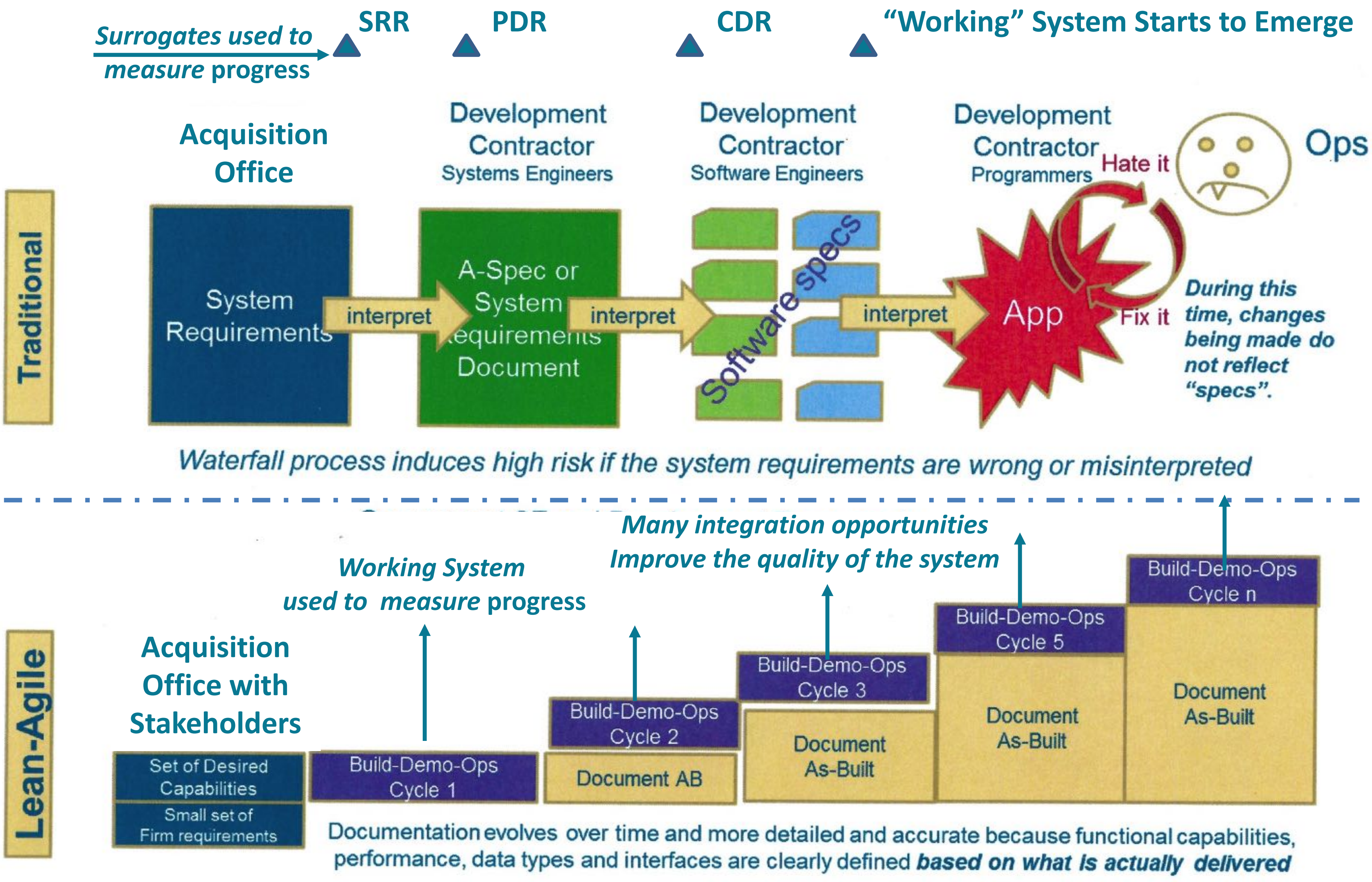
Problems We are Trying to Solve

- Over 72% of U.S. Government IT projects fail to meet cost, schedule, performance objectives (Source: TechFAR Handbook)
  - Of the 28% that succeed, over 50% of the functionality is either not used or is minimally used
- Agile developers are not aligned with SE
  - Developers can code faster than SEs can ERB
- Not all SEs are embracing the Agile philosophy
  - SE role is different in Agile development projects

Is our underlying theory of program/project management trapped in the past and no longer valid?

B

How Many Of Us Still Use The Traditional Acquisition Model?



Surrogates used to measure progress

SRR

PDR

CDR

"Working" System Starts to Emerge

Traditional

Acquisition Office

Development Contractor
Systems Engineers

Development Contractor
Software Engineers

Development Contractor
Programmers

Ops

System Requirements

interpret

A-Spec or System Requirements Document

interpret

Software specs

interpret

App

Hate it

Fix it

During this time, changes being made do not reflect "specs".

Waterfall process induces high risk if the system requirements are wrong or misinterpreted

Lean-Agile

Many integration opportunities
Improve the quality of the system

Working System used to measure progress

Acquisition Office with Stakeholders

Set of Desired Capabilities

Small set of Firm requirements

Build-Demo-Ops Cycle 1

Build-Demo-Ops Cycle 2

Document AB

Build-Demo-Ops Cycle 3

Document As-Built

Build-Demo-Ops Cycle 5

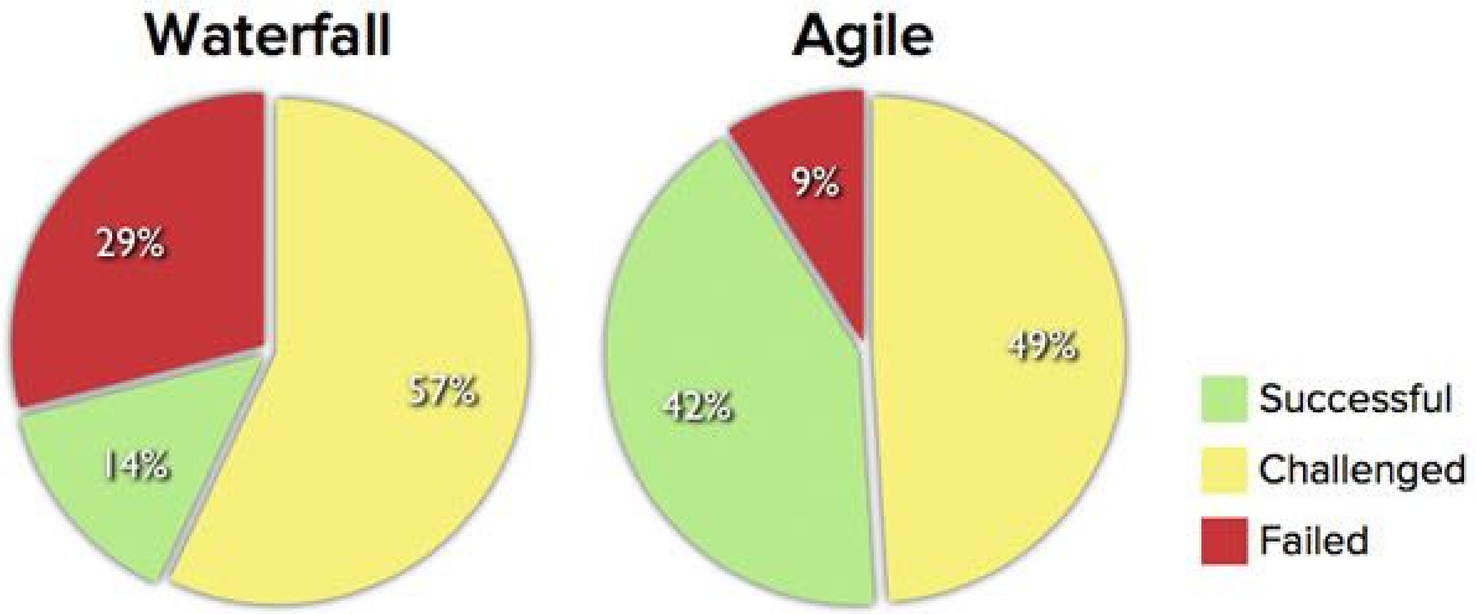
Document As-Built

Build-Demo-Ops Cycle n

Document As-Built

Documentation evolves over time and more detailed and accurate because functional capabilities, performance, data types and interfaces are clearly defined ***based on what is actually delivered***

Agile is better than Waterfall, but..



Source: The CHAOS Manifesto, The Standish Group, 2012.

…a 42% success rate isn’t exactly stellar

Where were the systems engineers on the the 58% challenged/failed projects?

M

What is Lean-Agile Development?

80%

How to **think**

Philosophy: a fundamental proposition that serves as the foundation for behavior and reasoning.

20%

How to **do** things

Process: steps taken to implement a set of values/principles.

4 AGILE Values

- Individuals and interactions over processes/tools
- Working software over comprehensive documentation
- Customer collaboration over contract negotiation
- Responding to change over following a plan

12 AGILE Principles

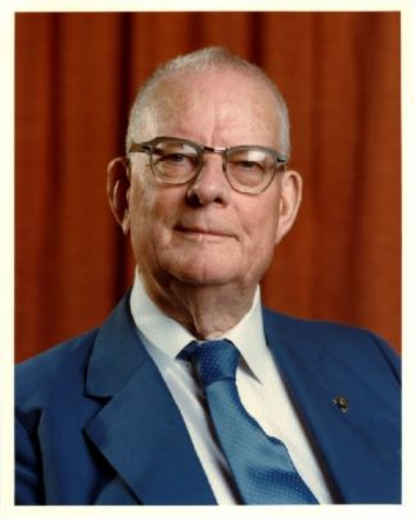
Specific Methods

- SCRUM
- SAFe
- XP
- FDD
- Crystal
- …

Best Practices

Unfortunately, we see more focus on the 20% than the 80%.

The SE Role

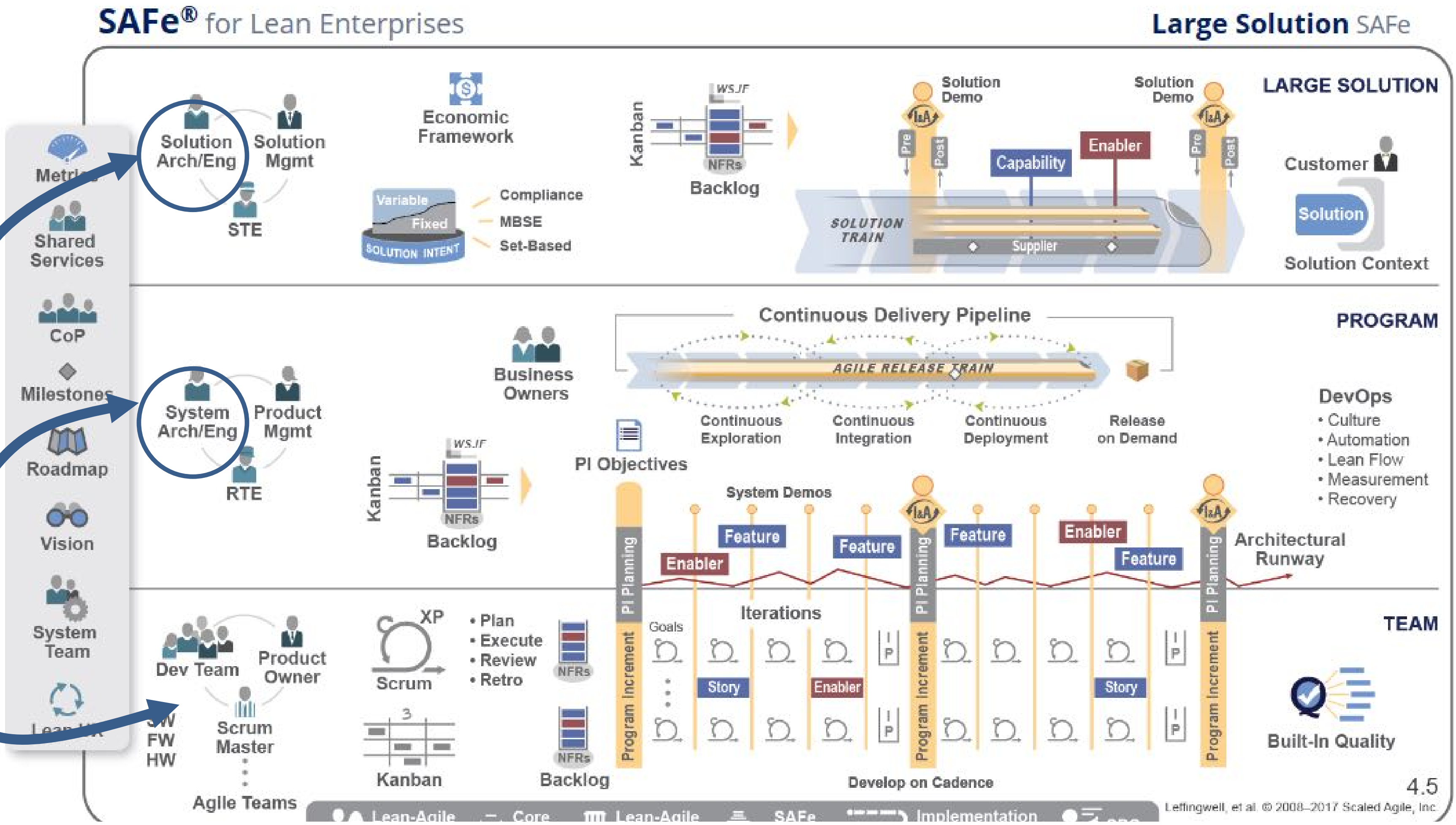


A system is a network of interdependent components that work together to try to accomplish the aim of the system. A system must have an aim. Without an aim, there is no system.

- W. Edwards Deming (1900-1993)

B

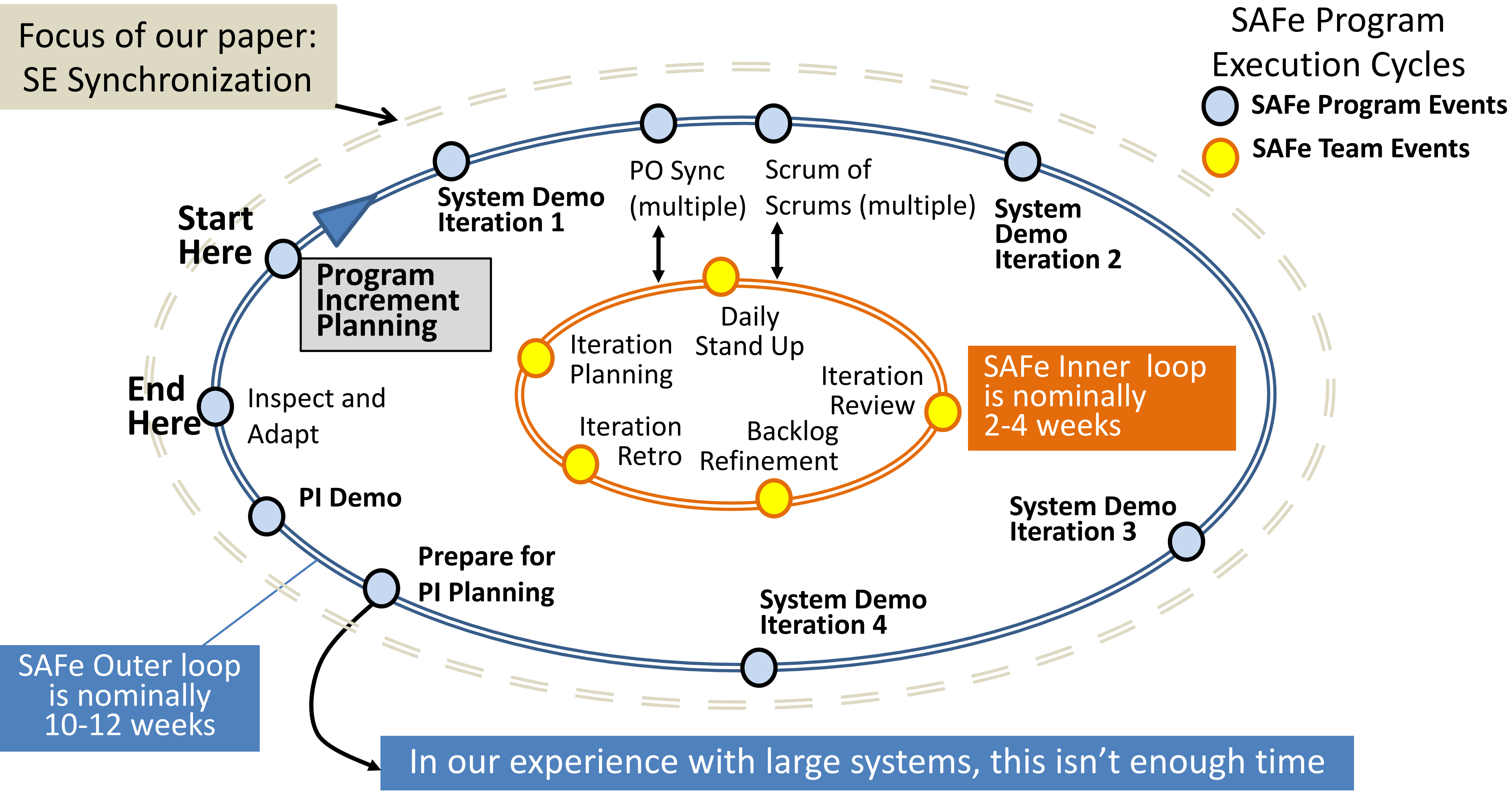
How and Where Do Systems Engineers Engage With Agile Release Trains



In large systems, SEs must interact with Agile release trains to ensure the system has an aim.

Where Do System Engineers “Plug In” to the Agile Development Cycle?

B



Focus of our paper: SE Synchronization

SAFe Program Execution Cycles

● SAFe Program Events

● SAFe Team Events

Start Here

Program Increment Planning

System Demo Iteration 1

PO Sync (multiple)

Scrum of Scrums (multiple)

System Demo Iteration 2

Daily Stand Up

Iteration Planning

Iteration Review

Iteration Retro

Backlog Refinement

SAFe Inner loop is nominally 2-4 weeks

End Here

Inspect and Adapt

PI Demo

Prepare for PI Planning

System Demo Iteration 3

System Demo Iteration 4

SAFe Outer loop is nominally 10-12 weeks

In our experience with large systems, this isn’t enough time

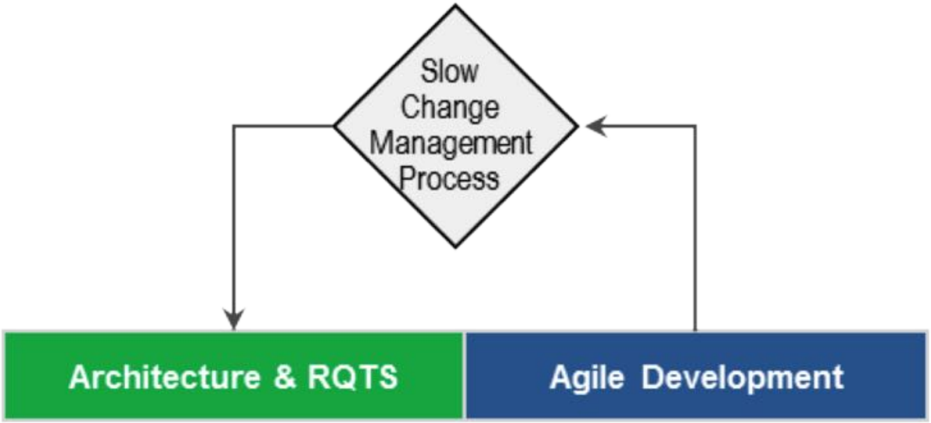
System Architecture/Design continuously evolves. Traditional SE organizations/processes are challenged by the fast pace in Agile development.

B

Typical SE Approaches to Agile….

Often devolves into

Big Vee + Agile Development



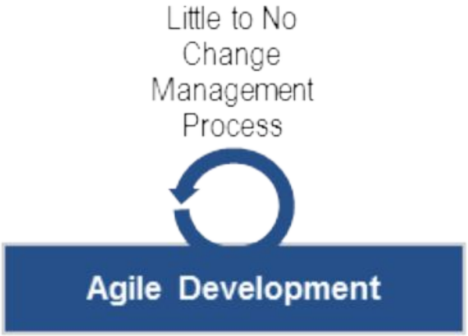
Slow Change Management Process

Architecture & RQTS

Agile Development

Waterfall Agile (WAGILE) Model

No or Disconnected SE



Little to No Change Management Process

Agile Development

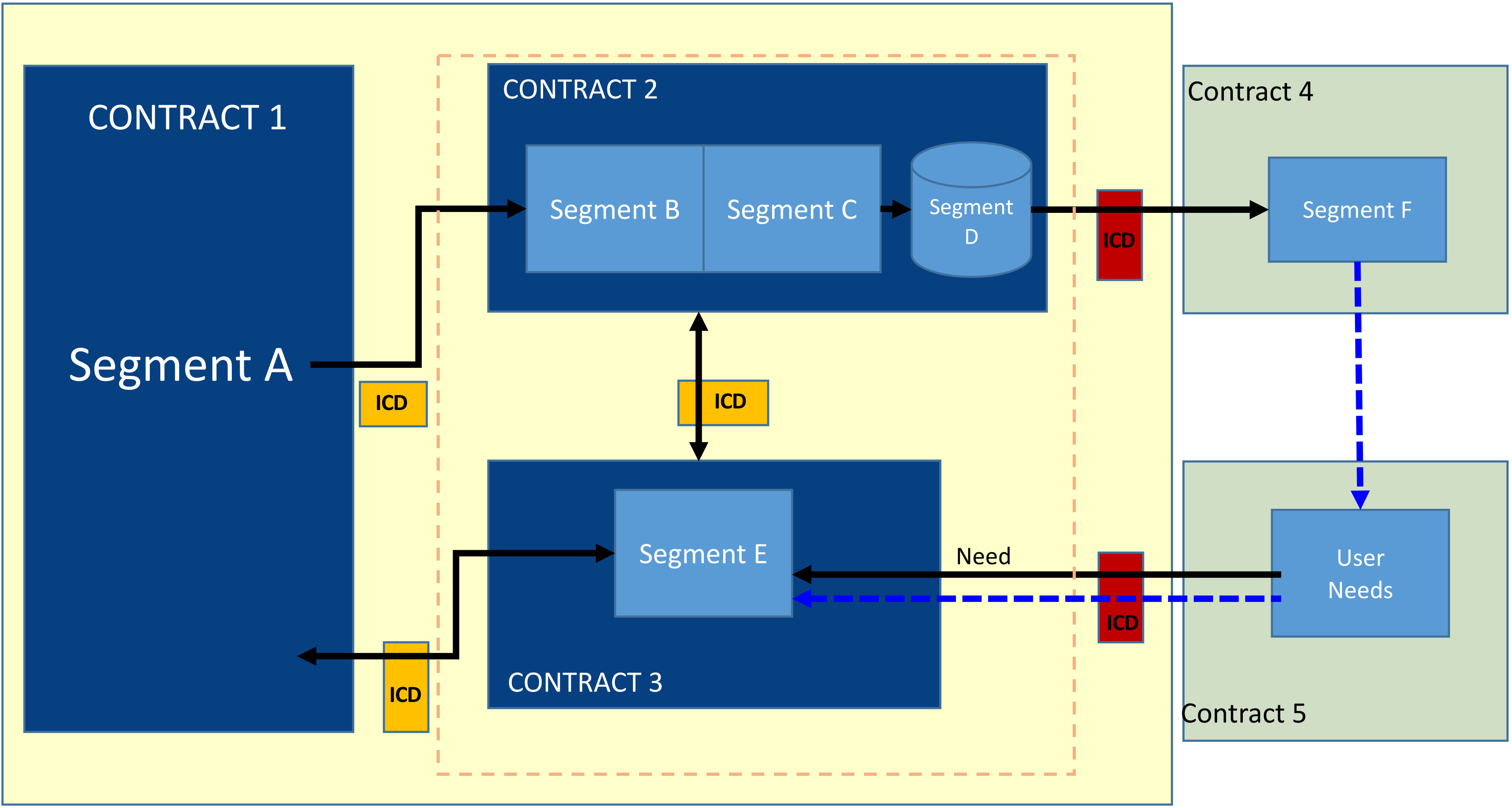
Unconstrained Agile Model

- System Engineers use the language of requirements to represent desired functionality.
- Agile developers think in terms of “backlog” and time.

The Solution?

Systems Engineering Professionals must drive the change!

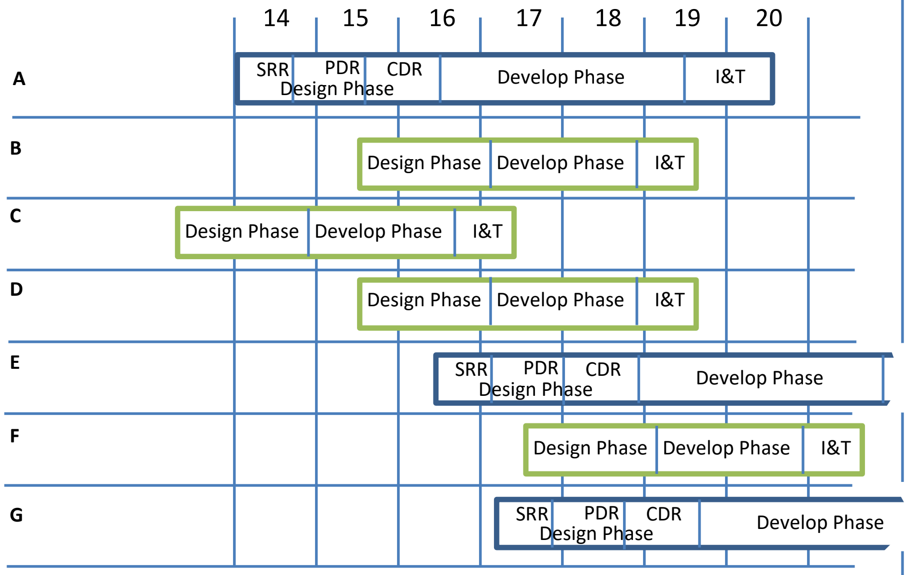
A Typical Complex System Development



- **Multiple Program Offices, multiple contracts, and many development contractors.**
  - SAFe® presumes the system is under the control of a single enterprise.

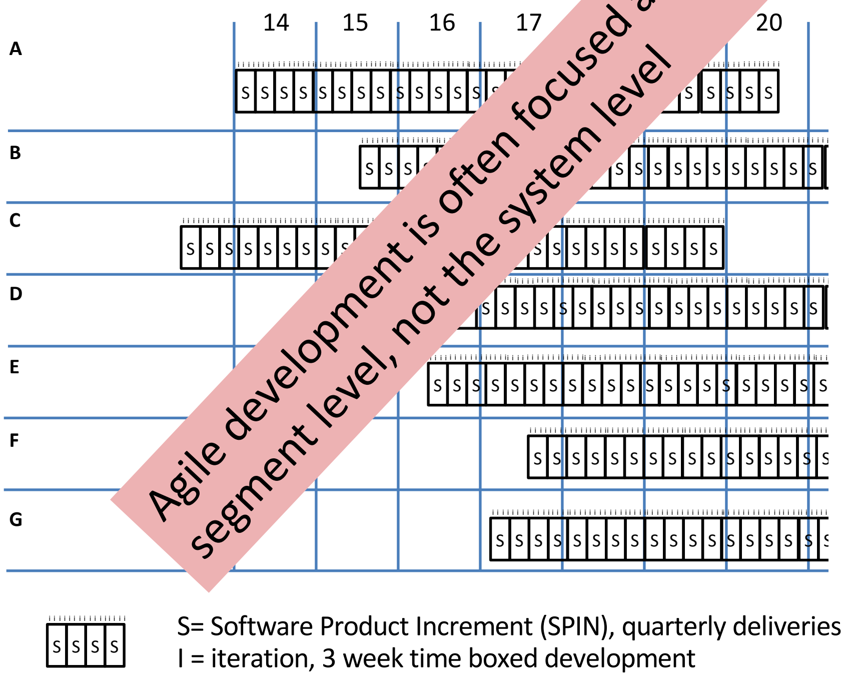
How do we develop, integrate and deploy key functions?

Waterfall Development



- Alignment of design, development and integration activities is slow but controlled
- Integration issues compound non-linearly
- Schedule impacts compound
- SE processes slow but controlled

Un-aligned Agile Developm



Agile development is often focused at the segment level, not the system level

S= Software Product Increment (SPIN), quarterly deliveries
I = iteration, 3 week time boxed development

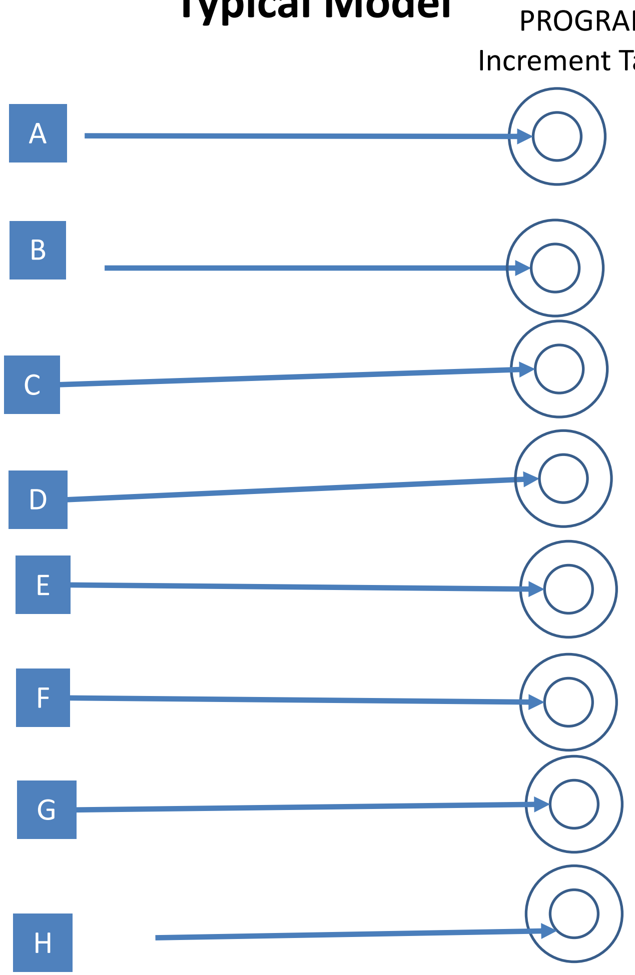
- Alignment of design, development and integration activities is not done or is disconnected
- Integration issues compound non-linearly
- More opportunities to adapt
- Limited SE rigor can lead to chaos

Both models lead to programmatic issues.

M

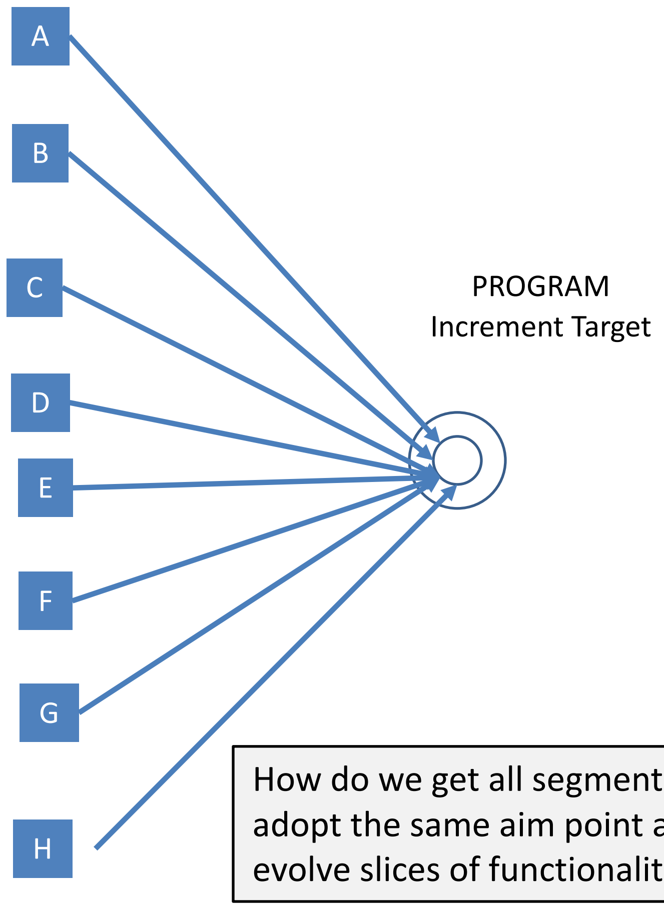
Alignment: another way to look at it

Typical Model



Still requires a waterfall integration; segments evolve independently of one another.

Where We Need to Be



How do we get all segments to adopt the same aim point and evolve slices of functionality?

Alignment requires a systems perspective across all elements.

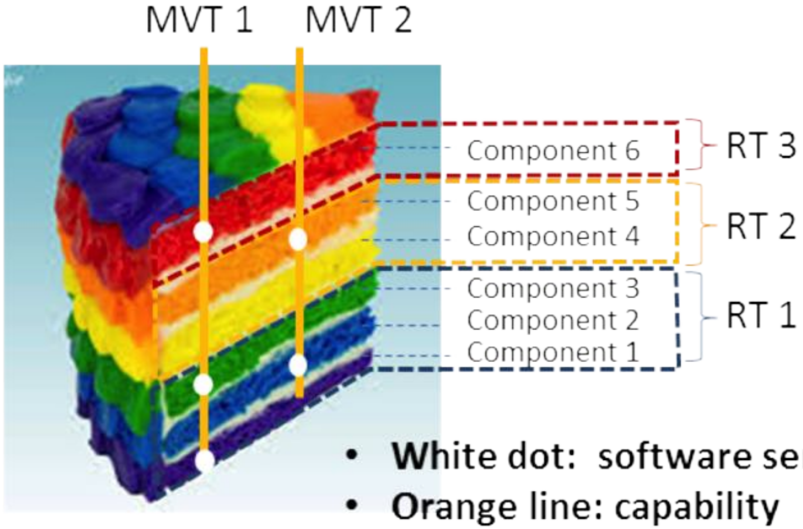
We Must Change The Way We Think About Realizing A System

B

The System is the entire cake

Each Layer of the cake is a is a System Component. The icing between layers are interfaces

Produce a fully functional "cake slice" <u>Every Program Increment</u>



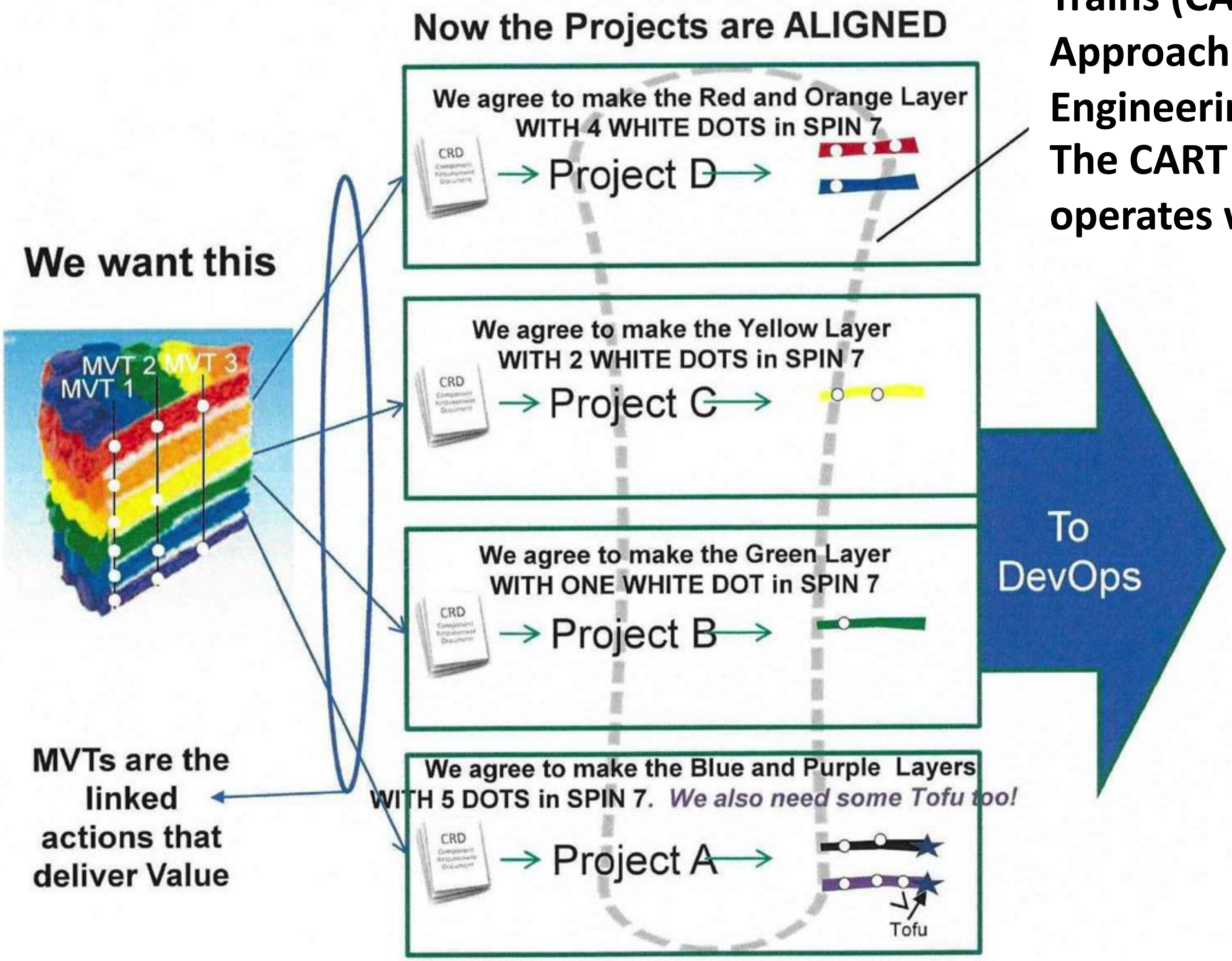
- White dot: software services
- Orange line: capability

Mission Value Threads define the end to end capability delivered in the slice

Drive as many integration opportunities and demos into the development plan as possible.

Use the CART to Drive Out Slices of Cake

Collaborate Across Release Trains (CART) is a Team of Teams Approach to Systems Engineering and Development. The CART is an Agile team and operates with an Agile Cadence



Now the Projects are ALIGNED

We want this

We agree to make the Red and Orange Layer WITH 4 WHITE DOTS in SPIN 7

We agree to make the Yellow Layer WITH 2 WHITE DOTS in SPIN 7

We agree to make the Green Layer WITH ONE WHITE DOT in SPIN 7

We agree to make the Blue and Purple Layers WITH 5 DOTS in SPIN 7. *We also need some Tofu too!*

MVTs are the linked actions that deliver Value

To DevOps

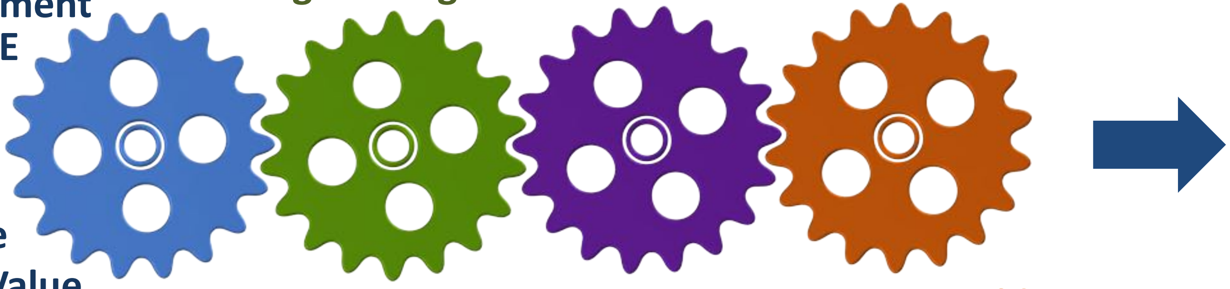
©2018 Techna Systems/Strategy Bridge International

B

Lean SE Best Practice
Concurrent or Simultaneous Systems Engineering

- A work methodology based on the parallelization of tasks (Small SE Batches tied to Small Development Batches)

- Integrate SE and Development engineering activities to reduce the elapsed time required to deliver value in an increment



Customer/ Government PM & SE

Elaborate Desired Value

Contractor System Engineering

Elaborate Desired Value With MBSE

Contractor Software Architects

Elaborate Arch/Design To Produce Value

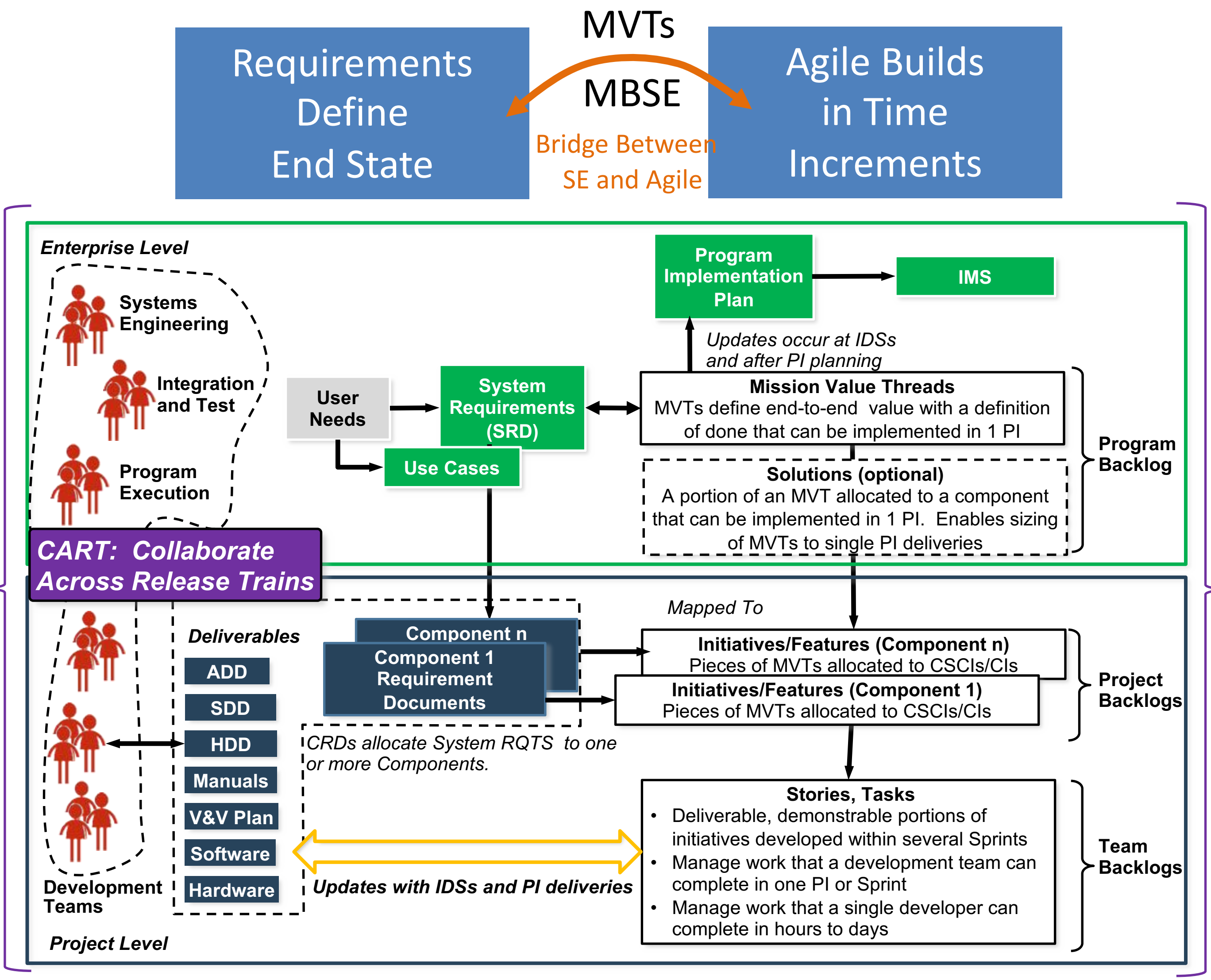
Contractor Development Teams

Design, Build Integrate, Test Value

Produce
End-to-End ***Value***
Every Program
Increment
(nominally 12 weeks)

Systems Engineering must be synchronized and on the same cadence as agile development teams!

Next: Change The Way We Define and Use Requirements


Requirements
Define
End State
MVTs
MBSE
Bridge Between
SE and Agile
Agile Builds
in Time
Increments
Enterprise Level
Systems
Engineering
Integration
and Test
Program
Execution
Program
Implementation
Plan
IMS
Updates occur at IDSs
and after PI planning
User
Needs
System
Requirements
(SRD)
Use Cases
Mission Value Threads
MVTs define end-to-end value with a definition
of done that can be implemented in 1 PI
Solutions (optional)
A portion of an MVT allocated to a component
that can be implemented in 1 PI. Enables sizing
of MVTs to single PI deliveries
Program
Backlog
CART: Collaborate
Across Release Trains
Mapped To
Deliverables
ADD
SDD
HDD
Manuals
V&V Plan
Software
Hardware
Component n
Component 1
Requirement
Documents
Initiatives/Features (Component n)
Pieces of MVTs allocated to CSCIs/CIs
Initiatives/Features (Component 1)
Pieces of MVTs allocated to CSCIs/CIs
Project
Backlogs
CRDs allocate System RQTS to one
or more Components.
Stories, Tasks
• Deliverable, demonstrable portions of
initiatives developed within several Sprints
• Manage work that a development team can
complete in one PI or Sprint
• Manage work that a single developer can
complete in hours to days
Team
Backlogs
Updates with IDSs and PI deliveries
Development
Teams
Project Level

CART is a continuous process that occurs concurrently with systems development

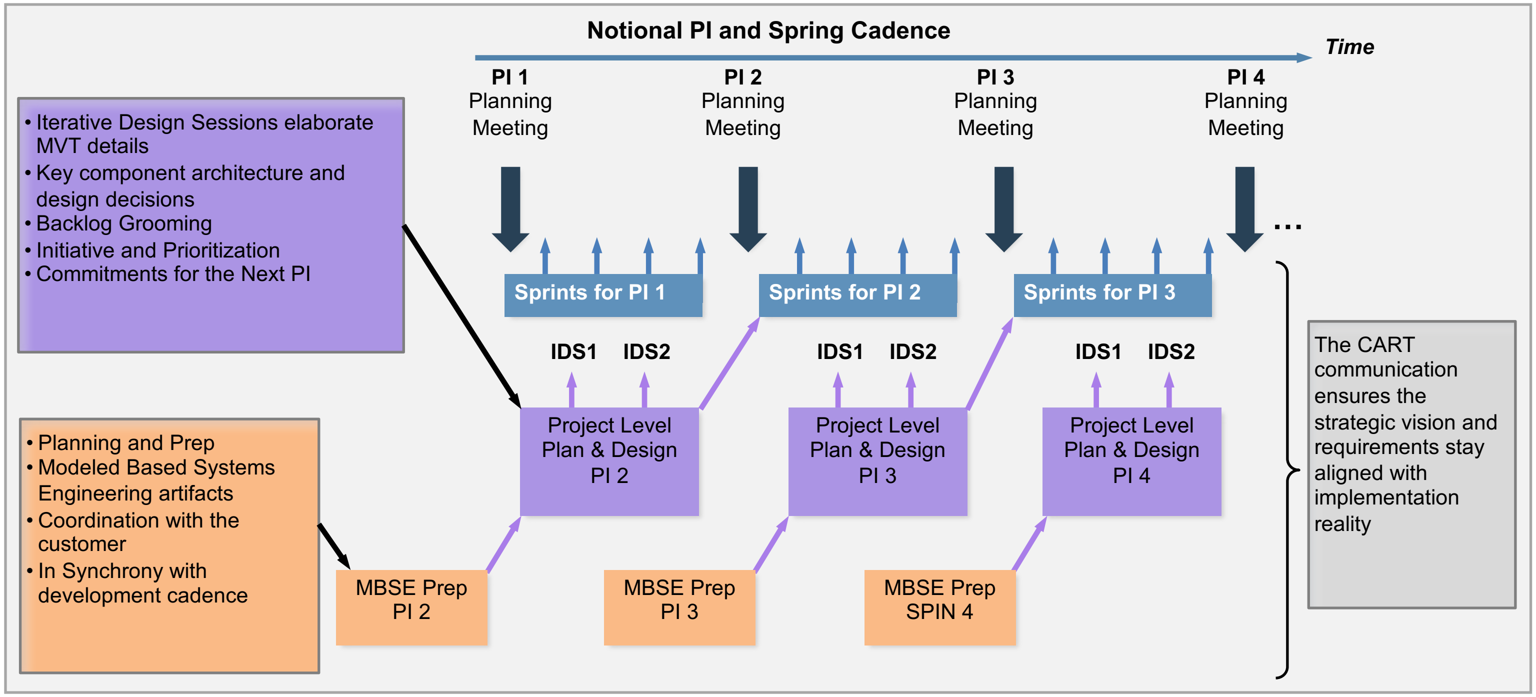
Agile Developers work off backlogs.

The CART ensures that useful SE artifacts elaborate solution intent in time for PI planning

SEs have to be on the Agile train!

A Failure to Communicate:

change the cadence and synchronization of SE processes



- Iterative Design Sessions elaborate MVT details
- Key component architecture and design decisions
- Backlog Grooming
- Initiative and Prioritization
- Commitments for the Next PI

- Planning and Prep
- Modeled Based Systems Engineering artifacts
- Coordination with the customer
- In Synchrony with development cadence

The CART communication ensures the strategic vision and requirements stay aligned with implementation reality

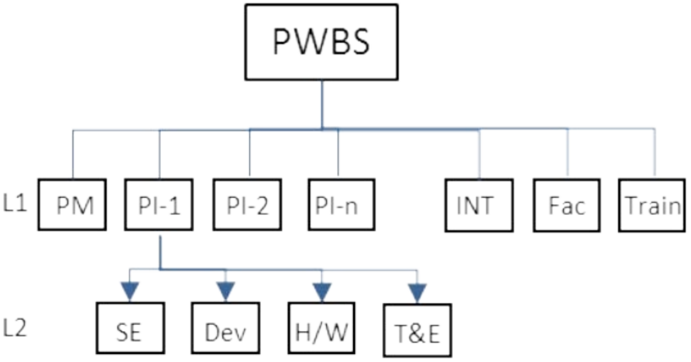
To keep SE connected to development, systems engineering must elaborate SE products in time to support Program Increments.

B

Change Contract Docs to Enable Lean-Agile Execution

Some Notable Examples

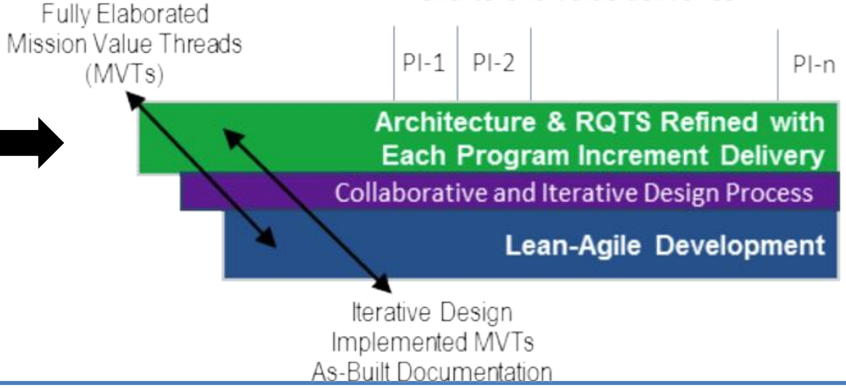
Cost Accounting and EVM



Cross Functional, Linked Actions
To Produce Value in Each Program Increment

Statement of Work: Define Tasks Using LSE and Tailored Lean-Agile Development Approaches

Measure program progress and cost with end-to-end Value deliveries



Contract Data Requirements List (CDRLs)

- **Tailor for Lean-Agile Development**
- **Reduce the Number of Docs**

IDSs vs Formal Design Reviews

Test Driven Development with executable Docs

Value Focused Meetings

Infrastructure As Code

Systems Engineering Management Plan (SEMP)

CART (rapid) vs Traditional (slow) Change Management

Use Agile Ceremonies vs Formal Boards

Incremental OT&E vs Big Bang

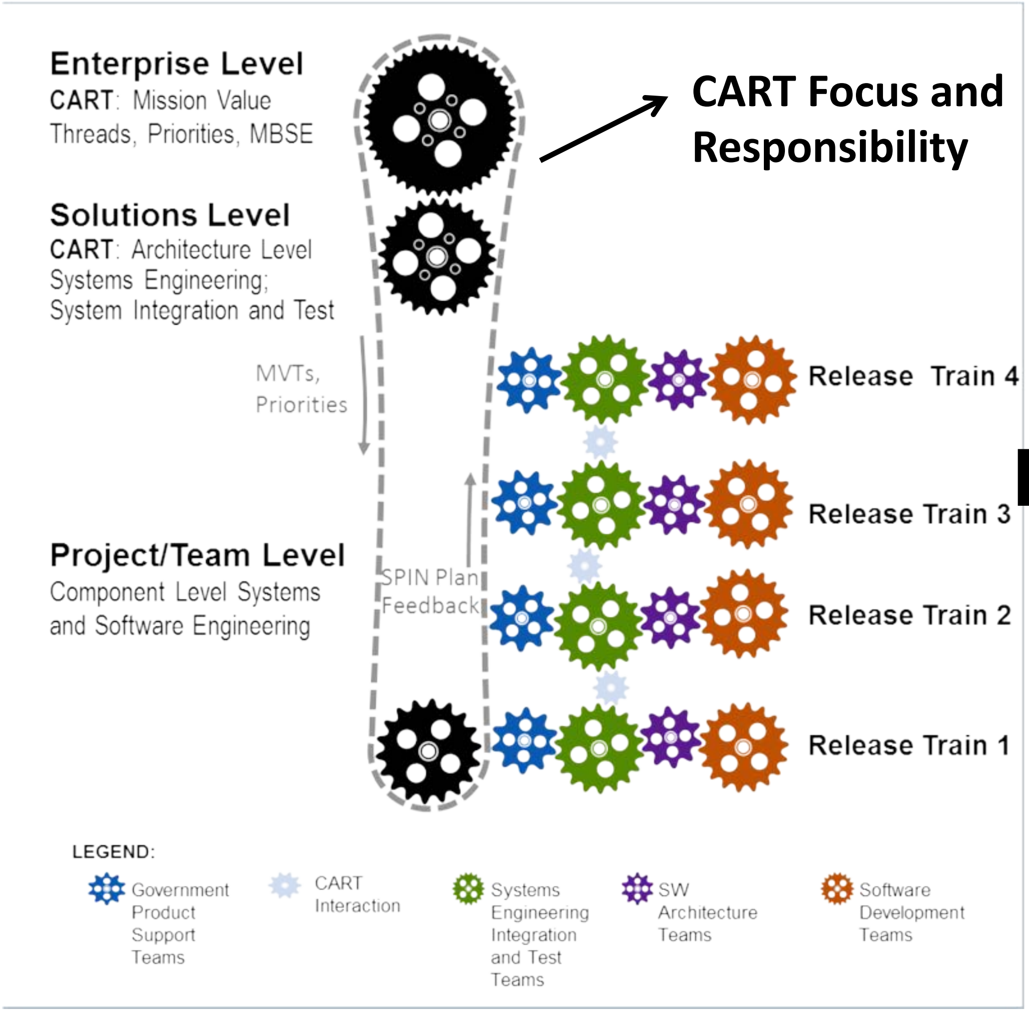
Incremental Requirements Verification and Validation vs Big Bang

Leverage Software Tool Output (e.g. JIRA, Confluence, MBSE)

Contract documentation is reflective of organizational culture: BIG culture changes needed for successful Lean-Agile execution.

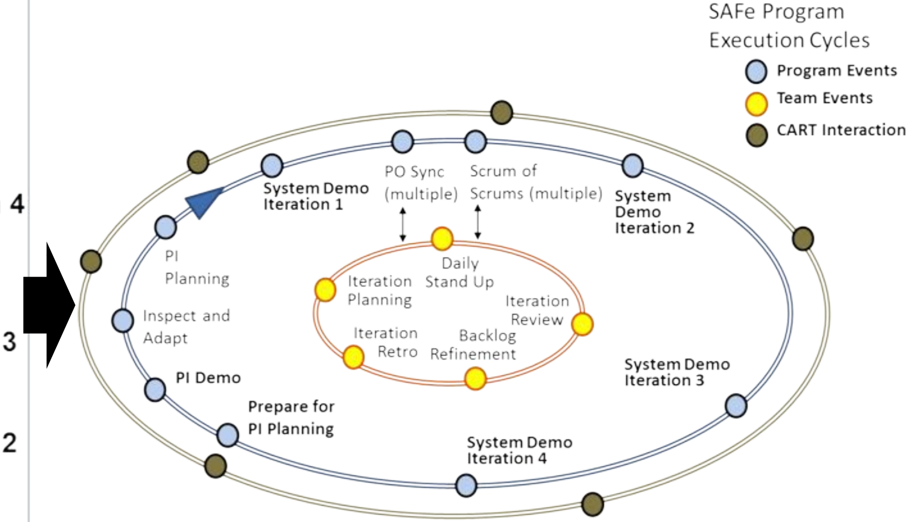
What Lean-Agile Development With Integrated SE (CART) Looks Like

B


Enterprise Level
CART: Mission Value Threads, Priorities, MBSE
Solutions Level
CART: Architecture Level Systems Engineering; System Integration and Test
MVTs, Priorities
SPIN Plan Feedback
Project/Team Level
Component Level Systems and Software Engineering
Release Train 4
Release Train 3
Release Train 2
Release Train 1
LEGEND:
Government Product Support Teams
CART Interaction
Systems Engineering Integration and Test Teams
SW Architecture Teams
Software Development Teams

CART Focus and Responsibility

The CART Has to Cycle Fast Enough to Support PI Planning



You know you are there when SE successfully communicates PI solution intent IN TIME for PI Planning and Development COMMITS TO IT!

Final Thoughts

- Lean-Agile is a mindset has to permeate across the entire enterprise to fully realize the promise of Agile methodologies.
- Expecting "the best architectures, requirements, and designs" to emerge from self-organizing teams remains a challenge in large scale systems. *(Agile Manifesto Principle 11).*
- A successful Lean-Agile development strategy can work for large-scale systems IF systems engineering stays connected to – **and keeps pace with** – Agile development teams and activities.
- CART with Mission Value Threads (MVTs) has succeeded in synchronizing Lean-Agile activities across very large programs.
- More work is needed in formal training classes as well as in software tools to better enable CART.

28th Annual INCOSE international symposium

Washington, DC, USA
July 7 - 12, 2018

www.incose.org/symp2018

Techna Systems Inc.

StrategyBridge®