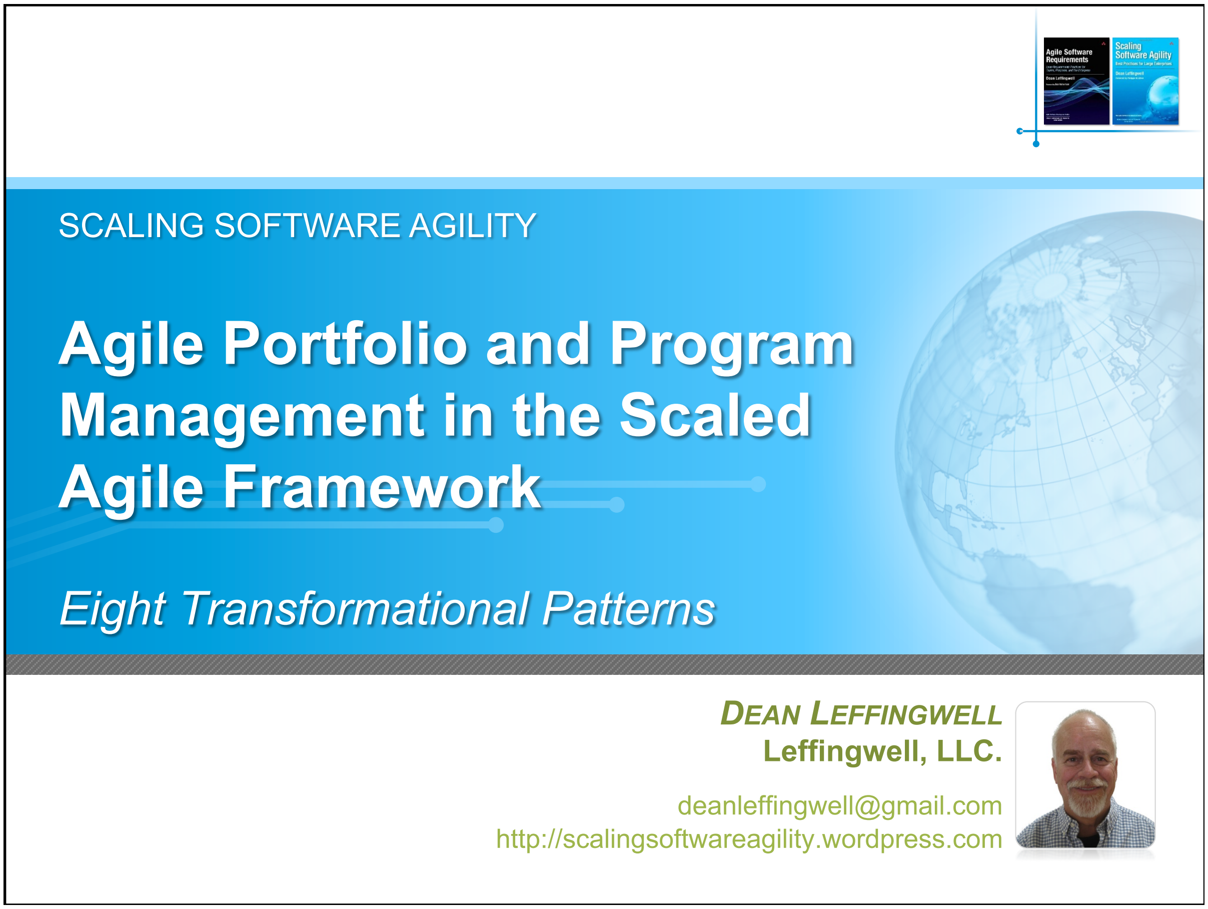


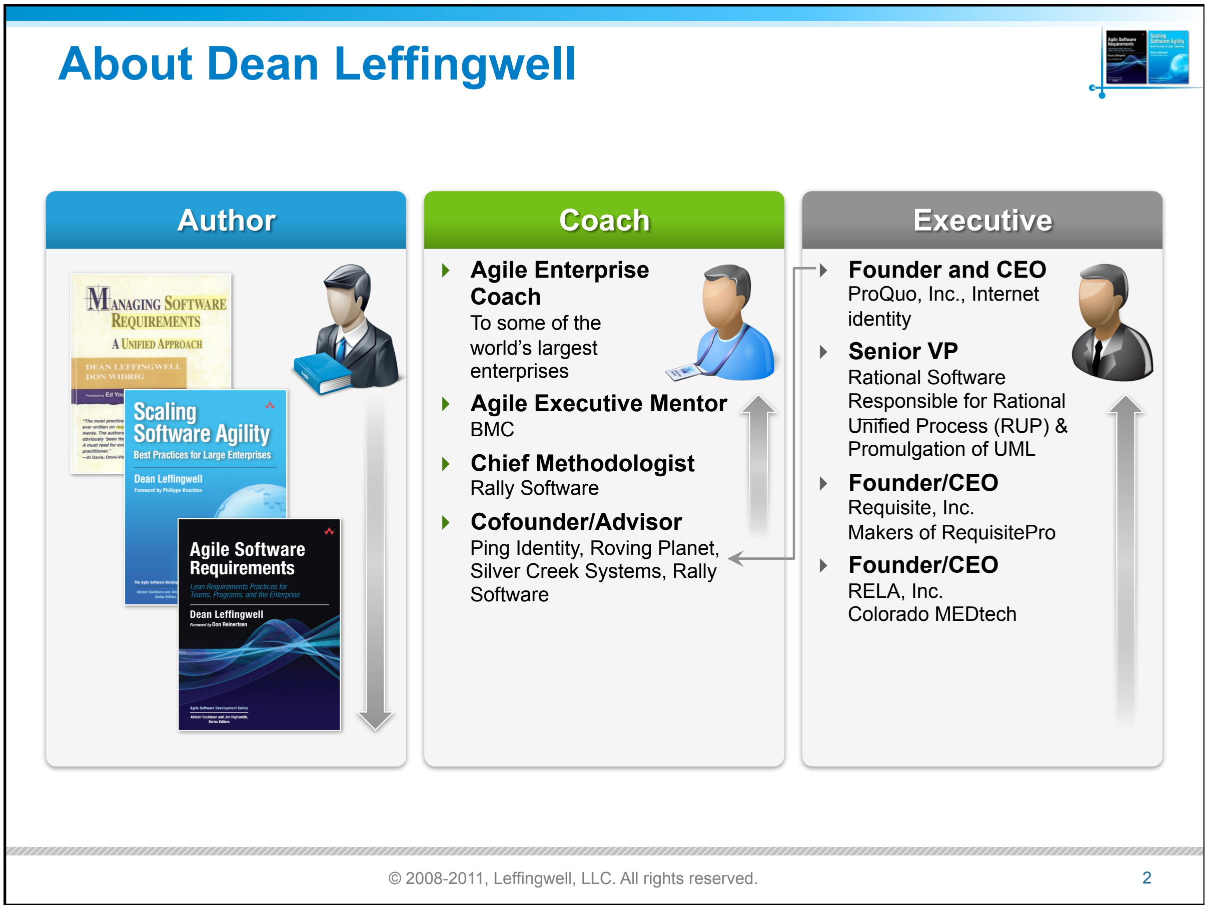
SCALING SOFTWARE AGILITY

Agile Portfolio and Program Management in the Scaled Agile Framework

Eight Transformational Patterns

DEAN LEFFINGWELL
Leffingwell, LLC.

deanleffingwell@gmail.com
http://scalingsoftwareagility.wordpress.com



About Dean Leffingwell

Author

Managing Software Requirements
A Unified Approach
Dean Leffingwell
Don Widrig
Scaling Software Agility
Best Practices for Large Enterprises
Dean Leffingwell
Agile Software Requirements
Lean Requirements Practices for Teams, Programs, and the Enterprise
Dean Leffingwell

Coach

- **Agile Enterprise Coach**
  To some of the world's largest enterprises
- **Agile Executive Mentor**
  BMC
- **Chief Methodologist**
  Rally Software
- **Cofounder/Advisor**
  Ping Identity, Roving Planet, Silver Creek Systems, Rally Software

Executive

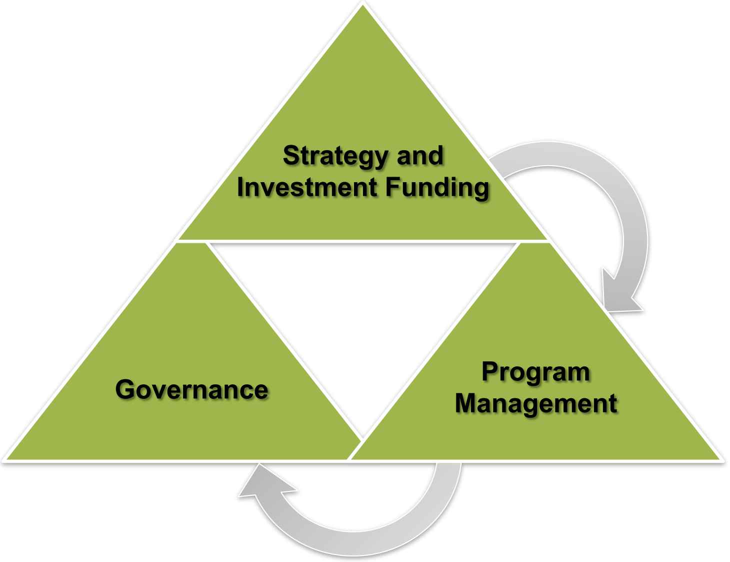
- **Founder and CEO**
  ProQuo, Inc., Internet identity
- **Senior VP**
  Rational Software
  Responsible for Rational Unified Process (RUP) & Promulgation of UML
- **Founder/CEO**
  Requisite, Inc.
  Makers of RequisitePro
- **Founder/CEO**
  RELA, Inc.
  Colorado MEDtech

Agenda

- Portfolio and Program Management Responsibilities
- Legacy Mindsets in Portfolio and Program Management
- The Scaled Agile Framework
- Eight Transformational Patterns
  - Rethinking Strategy and Investment Funding
  - Rethinking Program Management
  - Rethinking Governance
- Summary: The AGILE PPMO

PPMO Responsibilities

The PPMO has a central role in **strategy and investment funding**, **program management** and **governance**



Legacy Mindsets Handicap Agility

- Historically, PMO practices were based on the waterfall model of development

  (We shouldn’t blame these folks for governing us the way we told them to back when!)

- But now, we are transforming the entire business to a leaner and more agile enterprise, with ever-faster delivery of valuable, high quality software
- These legacy mindsets have to change……..(just like everything else did)

Legacy Mindsets

Strategy and Investment Funding

Mindset	Manifestation	Problems
“we can plan out a full year of projects”	Long term program commitments Teams must justify personnel a year out	Plans are obsolete, but not treated that way Projects impossible to kill once started Everyone lives a lie
“Maximize utilization”	Resources committed long range 100% allocation before “what if” Key resources assigned to multiple projects	No time to think or innovate Dedicate resources to task or lose resources Thrashing – productivity lost of most valuable resources No flex to changing priorities Exhaustion, burnout

Source: ***Establishing an Agile Portfolio to Align IT Investments with Business Needs*** -- Thomas and Baker, DTE Energy, by DTE Energy - Implementing and extending agile practices since 1998

Legacy Mindsets

Program Management

Mindset	Manifestation	Problems
"widget engineering"	Fixed schedule, functionality planning Big Up Front Design/ Analysis (BUFD) Detailed requirements specifications	Long range detailed commitment Resources committed year in advance Analysis paralysis Specs are wrong, hard to change
"Get it done"	Belief that best case plans must succeed	Deferred recognition of plan vs. actual Late discovery and re-negotiation Extended risk profile Loss of credibility, mistrust
"order taker mentality"	Do what you are told We are the boss of you	False agreements. No buy in. Misses IT innovation contribution Failure to meet expectations –mistrust No empowerment, lower motivation

Legacy Mindsets

Governance and Oversight

Mindset	Manifestation	Problems
"Control through data"	Fine grain reporting and overhead Detailed wbs, earned value metrics, loaded Gantt charts	Reporting overhead slows value delivery Metrics don't reflect actual progress
"Control through milestones"	Milestone reporting on intermediate artifacts	Milestones do not reflect actual progress Annoys the team "they just don't get it"

Eight Transformational Patterns

What we need is a transformation "roadmap", one that builds an Agile PPMO on Lean-Agile Principles

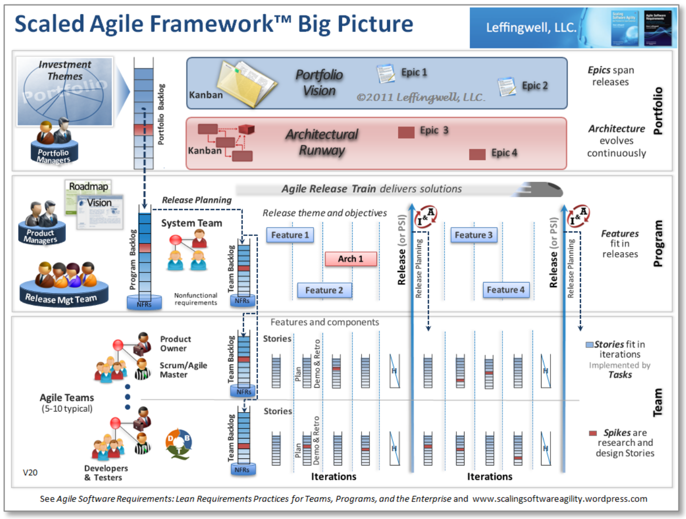
	Legacy Mindset	Lean-Agile Pattern
#1	Too Many Projects	Limiting Work in Process
#2	Detailed Project Plans	Lightweight Business Cases
#3	Annual Funding	Incremental Funding
#4	Centralized Annual Planning	Decentralized Rolling-Wave Planning
#5	Work Breakdown Structure	Agile Estimating and Planning
#6	Projects	Agile Release Trains
#7	PMBOK	Agile Project Management
#8	Waterfall Milestones	Fact-Based Governance

Legacy PPMO **Agile PPMO**

The Scaled Agile Framework

The Scaled Agile Framework

The Scaled Agile Framework is a proven, framework for applying Lean and Agile practices at enterprise scale

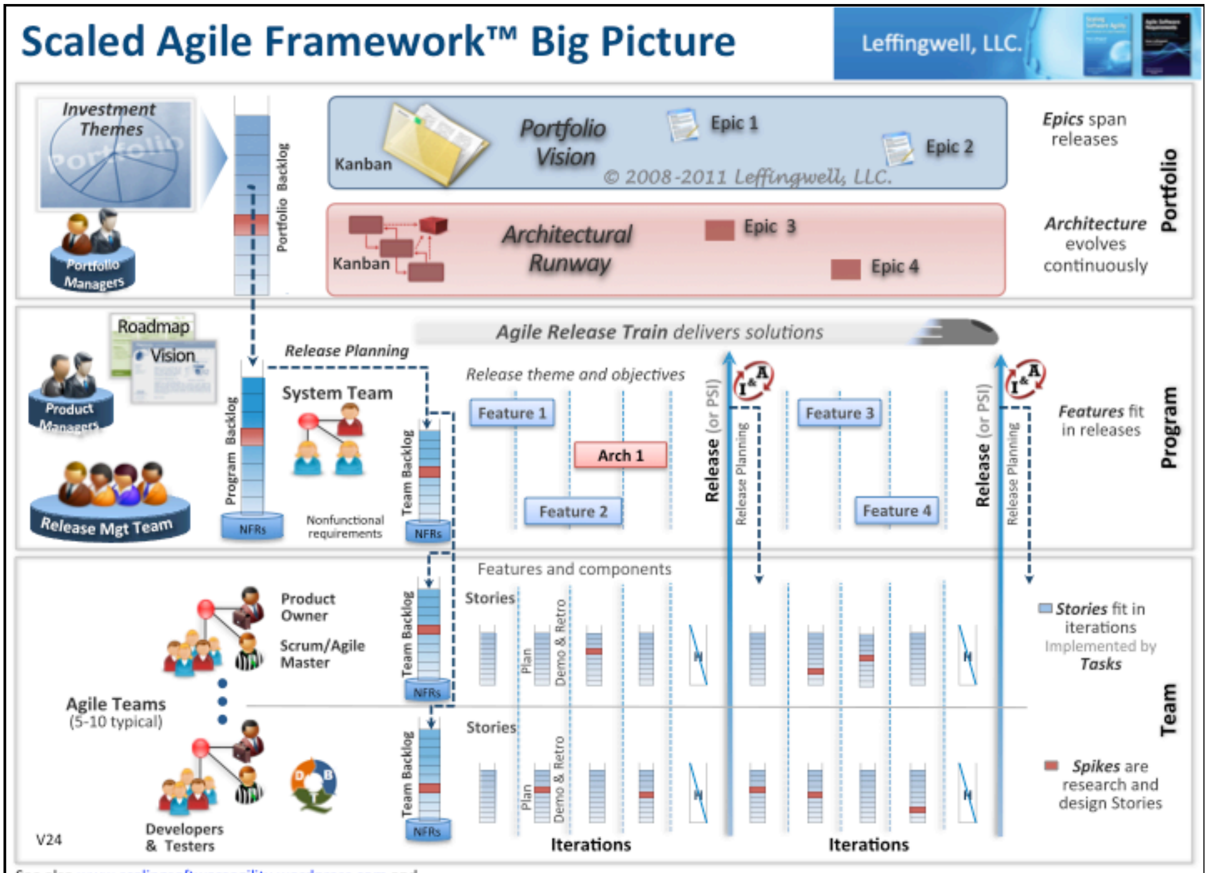


More on SAF:

- Synchronizes the vision, planning, interdependencies, and delivery of many teams
- Works well for teams of 50-100
- Has been scaled to over hundreds of teams and thousands of people
- Web version available to public in February 2012
- For more info, see ScaledAgileFramework.com

See Leffingwell, D. *Agile Software Requirements: Lean Requirements Practices for Teams, Programs, and the Enterprise* and www.scalingsoftwareagility.wordpress.com

Scaled Agile Framework™ Big Picture



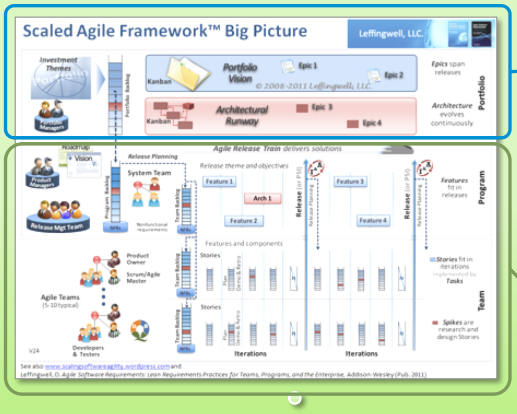
See also www.scalingsoftwareagility.wordpress.com and
Leffingwell, D. *Agile Software Requirements: Lean Requirements Practices for Teams, Programs, and the Enterprise*, Addison-Wesley (Pub. 2011)

PPMO in the Framework

SAF provides lean operating models for Strategy and Investment Funding, Program Execution, and Governance

Governance

- Clear content authority
- Drives incremental delivery
- Fact-based (code, test, quality, customer feedback) lifecycle governance



Strategy & Investment Funding

- Limit Work in Process
- Economic Prioritization

Program Management

- Continuous value delivery
- Self-managing programs

Rethinking Strategy and Investment Funding

1. Too Many Projects
2. Detailed Project Plans
3. Annual Funding
4. Centralized Annual Planning

#1. From Too Many Projects

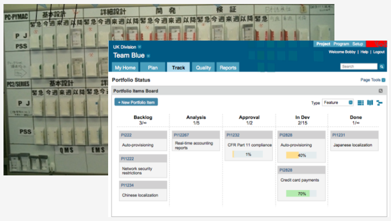
To Limiting Work in Process

Too Many Projects

- It is far easier to start projects than to complete them
- All dev teams have accepted (or been given) **more work than they can deliver**
- Result: excessive work in process, team members thrashing
- Lowers productivity (20% per switch) … causes delays in outcome…: Result: Even **more work is loaded** into the system
- Like a freeway, congestion results and throughput comes to a **halt**

Controlling WIP

- Development wip is invisible and has no natural predators
- Provide visual indicators of work in process
- Establish WIP limits for # projects in flight
- Build a Portfolio Management Kanban System



Portfolio Status

Portfolio Items Board

Backlog 3/∞	Analysis 1/5	Approval 1/2	In Dev 2/15	Done 1/∞
PI222 Auto-provisioning	PI12267 Real-time accounting reports	PI1232 CFR Part 11 compliance 1%	PI2828 Auto-provisioning 40%	PI1231 Japanese localization
PI1222 Network security restrictions			PI2828 Credit card payments 70%	
PI1234 Chinese localization				

Portfolio Kanban System

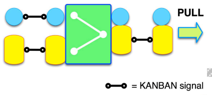
Kanban systems manage the
flow of value via visibility and Work in Process (WIP) limits

What Kanban Systems do

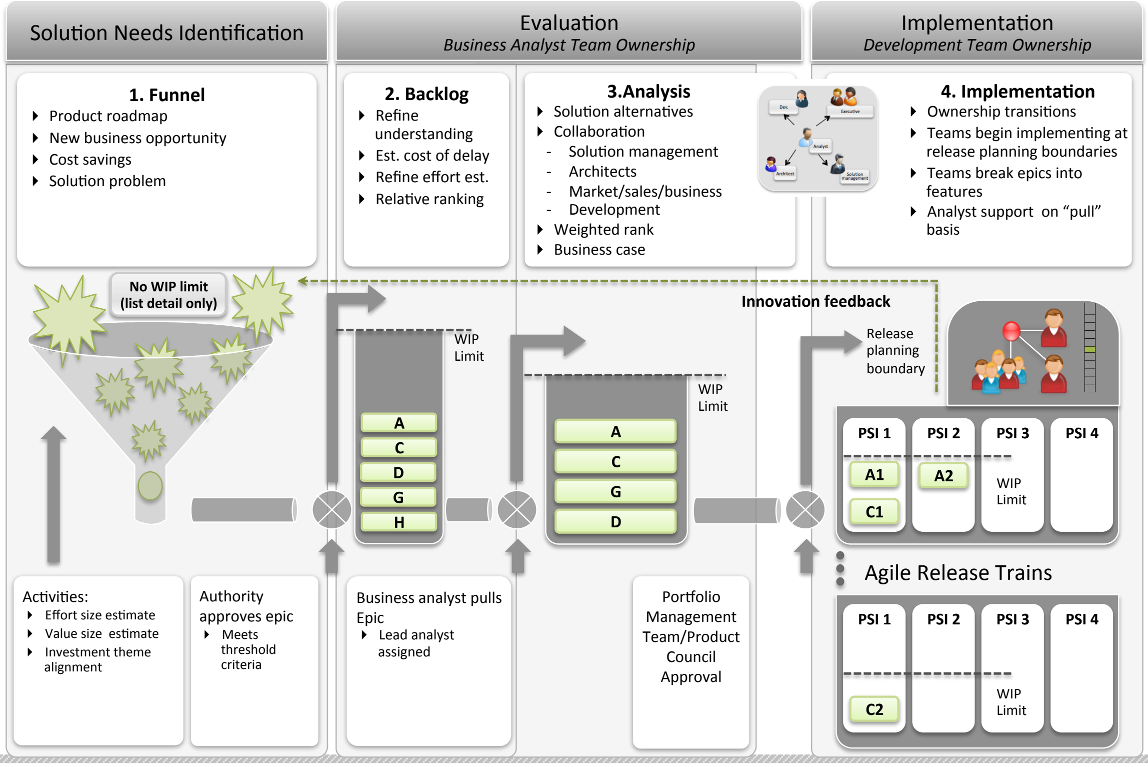
- Visually manages units of value through the whole system, from when they enter, until they leave
- Limiting WIP, creates an optimum, sustainable pipeline of value flow
- Limiting WIP provides visibility as to when there is capacity for new work
- WIP Limits are adjusted and their effects measured as the system is continuously improved

Why we need one for the Portfolio

- Adopt and drive agile thinking, language and behavior
- Drive incremental-ism in implementation of business epics
- Establish priorities based on economics
- Make work in process (WIP) visible
- Establish WIP limits to control queue sizes, limit global WIP and help assure continuous flow
- Drive effective collaboration with development teams



Portfolio Kanban System



Solution Needs Identification

Evaluation
Business Analyst Team Ownership

Implementation
Development Team Ownership

1. Funnel
- Product roadmap
- New business opportunity
- Cost savings
- Solution problem

2. Backlog
- Refine understanding
- Est. cost of delay
- Refine effort est.
- Relative ranking

3.Analysis
- Solution alternatives
- Collaboration
  - Solution management
  - Architects
  - Market/sales/business
  - Development
- Weighted rank
- Business case

4. Implementation
- Ownership transitions
- Teams begin implementing at release planning boundaries
- Teams break epics into features
- Analyst support on "pull" basis

No WIP limit (list detail only)

Innovation feedback

Release planning boundary

WIP Limit

Agile Release Trains

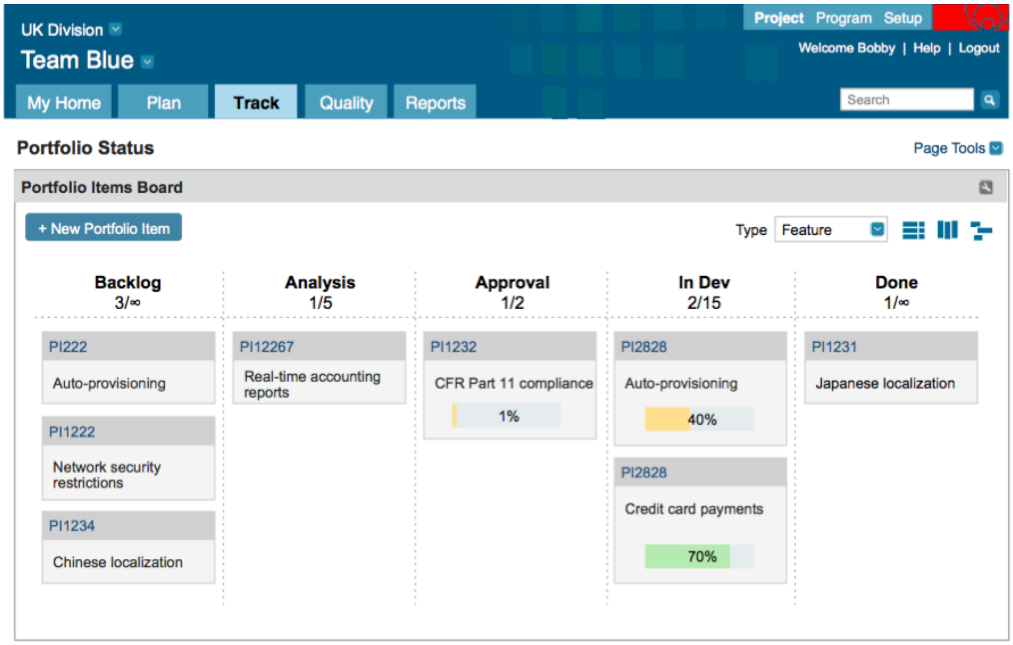
Activities:
- Effort size estimate
- Value size estimate
- Investment theme alignment

Authority approves epic
- Meets threshold criteria

Business analyst pulls Epic
- Lead analyst assigned

Portfolio Management Team/Product Council Approval

Agile Portfolio Management Tools



Portfolio Status

Portfolio Items Board

Type: Feature

Backlog 3/∞	Analysis 1/5	Approval 1/2	In Dev 2/15	Done 1/∞
PI222 Auto-provisioning	PI12267 Real-time accounting reports	PI1232 CFR Part 11 compliance 1%	PI2828 Auto-provisioning 40%	PI1231 Japanese localization
PI1222 Network security restrictions			PI2828 Credit card payments 70%	
PI1234 Chinese localization				

#2. From Detailed Project Plans

To Lightweight Business Cases

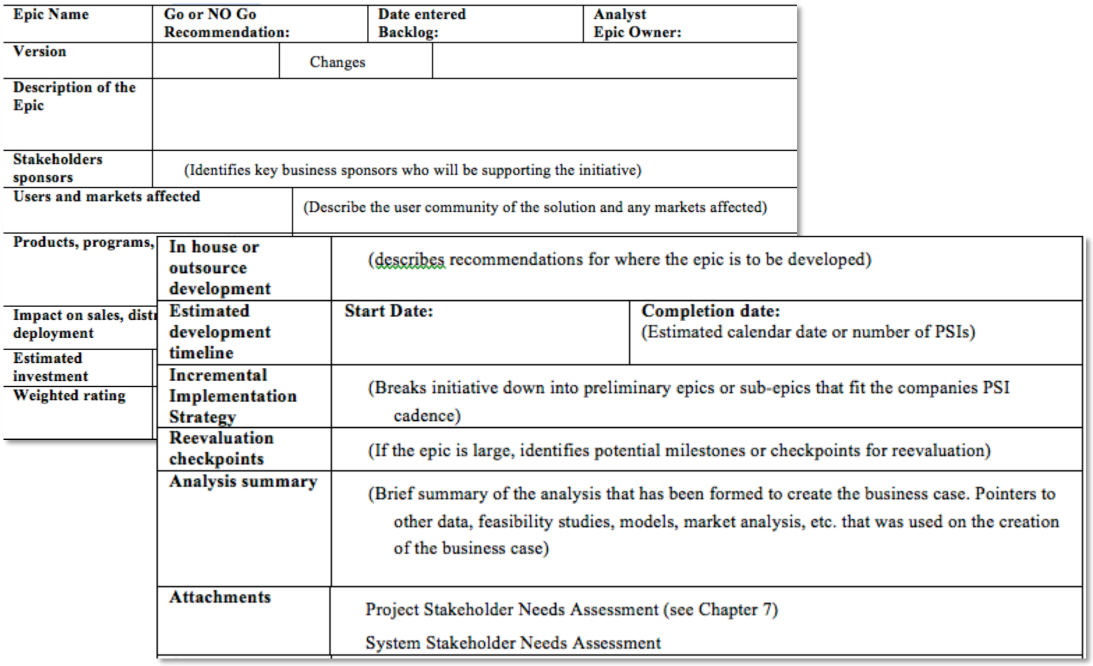
Detailed Project Plans

- Detailed business case justifications become project plans
- False precision – detailed requirements over-constrain solution implementation
- May contain redundant schedule, budget, ROI information
- Investment in the business case causes resistant to changing the case and plan
- Too much overhead for quarterly update

Lightweight Business Cases

- 1-2 page business case form
- Not much detail
- Business cases that make the cut get exploratory iterations funding
- Easily cancelled if progress not acceptable
- Fast ROI if it is
- Weight Shortest Job First lean economic prioritization
- Updated quarterly – changes only

Lightweight Business Case



Epic Name	Go or NO Go Recommendation:	Date entered Backlog:	Analyst Epic Owner:
Version		Changes	
Description of the Epic			
Stakeholders sponsors	(Identifies key business sponsors who will be supporting the initiative)		
Users and markets affected	(Describe the user community of the solution and any markets affected)		
Products, programs,			
Impact on sales, distr deployment			
Estimated investment			
Weighted rating			

In house or outsource development	(describes recommendations for where the epic is to be developed)	
Estimated development timeline	Start Date:	Completion date: (Estimated calendar date or number of PSIs)
Incremental Implementation Strategy	(Breaks initiative down into preliminary epics or sub-epics that fit the companies PSI cadence)	
Reevaluation checkpoints	(If the epic is large, identifies potential milestones or checkpoints for reevaluation)	
Analysis summary	(Brief summary of the analysis that has been formed to create the business case. Pointers to other data, feasibility studies, models, market analysis, etc. that was used on the creation of the business case)	
Attachments	Project Stakeholder Needs Assessment (see Chapter 7) System Stakeholder Needs Assessment	

See Leffingwell, Dean. Agile Software Requirements: Lean Requirements Practices for Teams, Programs and the Enterprise.

#3. From Annual Funding

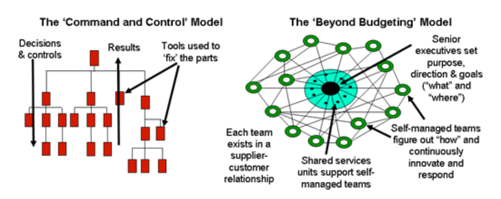
To Incremental Funding

Annual All or None Funding

- Legacy model drives “all or none” funding.
- Once funds committed, they are virtually impossible to de-allocate.
- Commitments are included in the annual budget planning process
- Departments must justify the use of resources **through** the fiscal year.
- So they create more detailed plans that justify the annual budget.
- Resulting projects are almost impossible to kill
- This vicious cycle drives the opposite of the behavior we are trying to achieve.

Agile – Incremental Funding

- Bad paths can be truncated more quickly
- Resources can be moved to the best opportunities throughout the fiscal year
- Base incremental funding on objective demonstrations of working software (instead of milestones based on proxy documentation)
- Continuous opportunities to assess and adjust
- Teams have real motivation to deliver immediate value


The ‘Command and Control’ Model
Decisions & controls
Results
Tools used to ‘fix’ the parts
The ‘Beyond Budgeting’ Model
Senior executives set purpose, direction & goals (“what” and “where”)
Each team exists in a supplier-customer relationship
Shared services units support self-managed teams
Self-managed teams figure out “how” and continuously innovate and respond

See “Beyond Budgeting”. www. www.bbrt.org

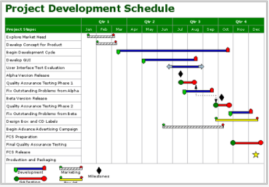
#4. From Centralized, Annual Planning

To Decentralized, Rolling-Wave Planning

From: Gantt

Issues:

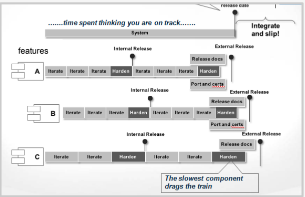
- Irrelevant to agile teams
- Hard to maintain
- Always obsolete
- If a team isn’t on the plan, is it a “bad” team or “bad” plan?
- Measure *paper, not software*



To: Independent, asynchronous, Sprints

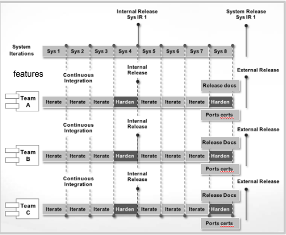
Issues:

- Most agile companies start here
- Little coordination amongst teams
- Non-harmonized schedules
- No visibility beyond the next sprint
- Little or no system level visibility



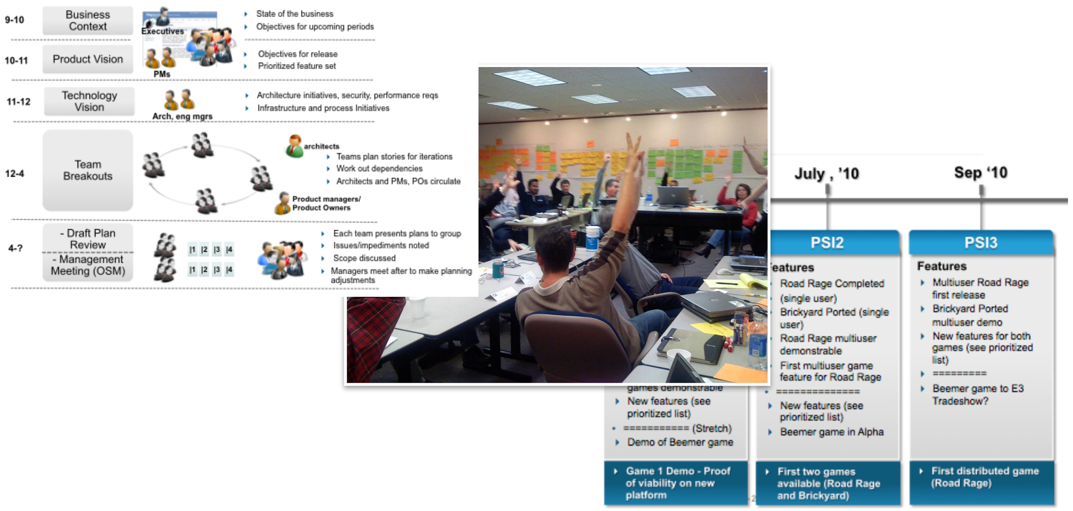
To: Agile Release Train

- Coordinates sprints
- Multi- sprint visibility and commitment
- Teams work out interdependencies on the fly
- Full system visibility
- Requires set-based development



Decentralized, Rolling-wave Planning

- Teams plan Face-to-Face on a fixed cadence
- Result: Agreed-to objectives
- Commitment from the ream
- Updated roadmap: high confidence “Release Next”



9-10 Business Context

- State of the business
- Objectives for upcoming periods

10-11 Product Vision

- Objectives for release
- Prioritized feature set

11-12 Technology Vision

- Architecture initiatives, security, performance reqs
- Infrastructure and process Initiatives

12-4 Team Breakouts

- Teams plan stories for iterations
- Work out dependencies
- Architects and PMs, POs circulate

4-? - Draft Plan Review
- Management Meeting (OSM)

- Each team presents plans to group
- Issues/impediments noted
- Scope discussed
- Managers meet after to make planning adjustments

games demonstrable
- New features (see prioritized list)
- =========== (Stretch)
- Demo of Beemer game
- Game 1 Demo - Proof of viability on new platform

July , ’10

PSI2

eatures
- Road Rage Completed (single user)
- Brickyard Ported (single user)
- Road Rage multiuser demonstrable
- First multiuser game feature for Road Rage
- ==============
- New features (see prioritized list)
- Beemer game in Alpha
- First two games available (Road Rage and Brickyard)

Sep ‘10

PSI3

Features
- Multiuser Road Rage first release
- Brickyard Ported multiuser demo
- New features for both games (see prioritized list)
- =========
- Beemer game to E3 Tradeshow?
- First distributed game (Road Rage)

Rethinking Program Management

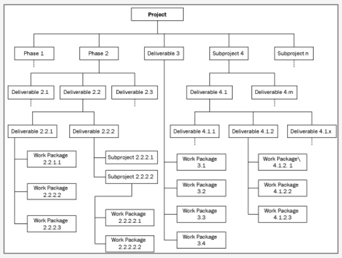
5. WBS
6. Projects
7. PMBOK

#5. From WBS

To Agile Estimating and Planning

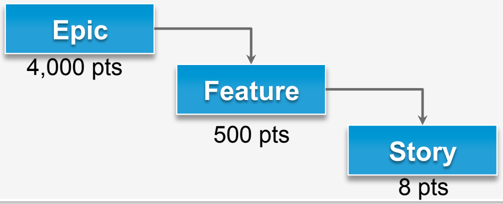
WBS Estimating and Planning

- Traditional project estimates tasks at the lowest leaf
- Requiring all leafs be identified before estimate is given
- Forces Big Up Front Analysis and estimates based on false precision
- Force fits later activities into a flawed WBS

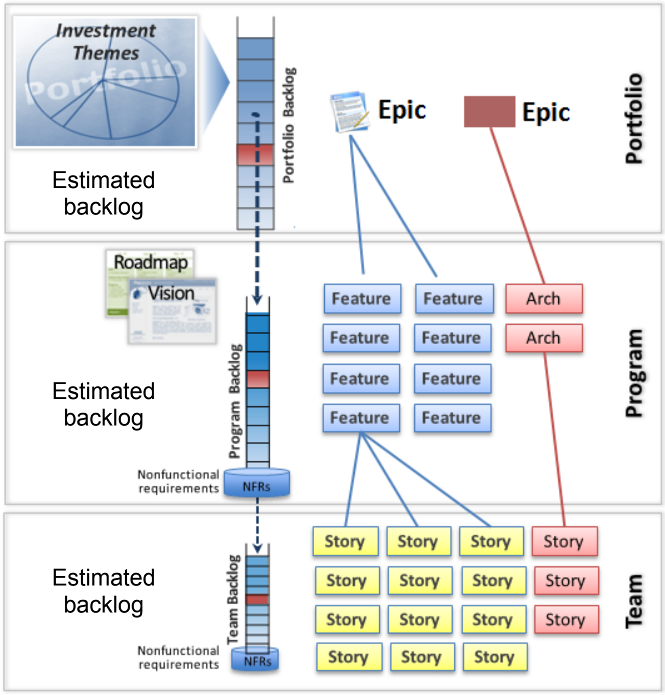


Agile Estimating and Planning

- Agile teams develop and monitor “velocity” based on story points at iteration level
- Story points can be normalized across teams
- Standardized estimating by analogous model can also be applied at the level of Epics and Features
- Epic estimating can be used for gross, portfolio-level planning
- Feature estimates can be used for release (product) level planning
- Story points used for iteration planning



Enterprise Backlog Estimation



- Portfolio estimates
- Program estimates
- Team estimates

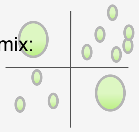
(all in story points)

#6. From Projects

To Agile Release Trains

Everything is a Project

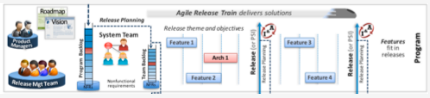
- Getting anything done (new feature or epic) requires creation of a new project
- Projects have significant overhead, planning, resourcing, execution, closure
- Once started, projects take on a life of their own. They develop antibodies to change and closure.
- Many small projects cause people to multiplex
- *Each project switch takes 20% overhead*



Project, portfolio mix:
Size, risk, reward

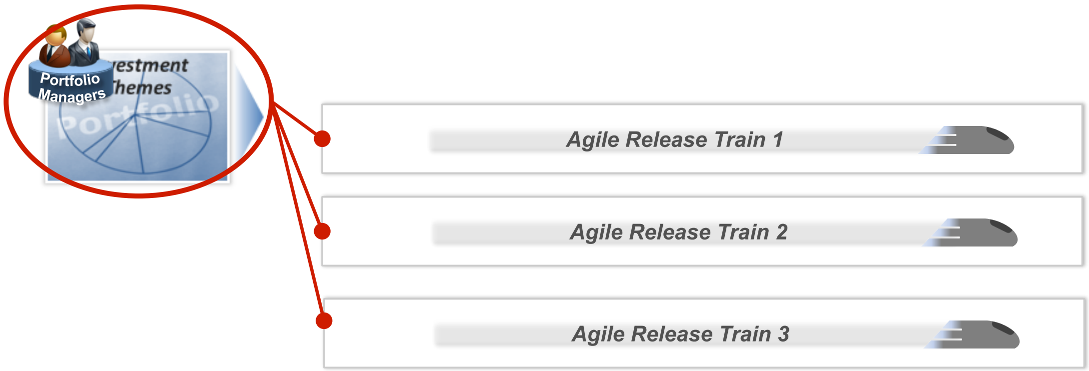
Continuous Content Delivery

- Dedicated teams stop multiplexing. No one works on more than one team. No team on more than one program.
- Project scheduling replaced by standard cadence
- New initiatives appear as new content : prioritized at each iteration/ release boundary
- Work in a PSI is fixed
- Team resources are adjusted only at cadence boundaries



Agile Release Trains

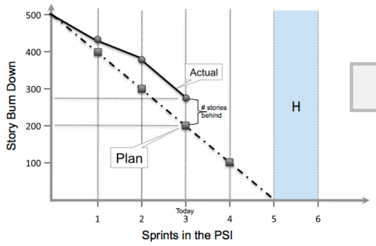
Investment Themes are realized by one or more “Agile Release Trains”



- Release trains are long lived value streams that align teams to a common mission
- *Each provides* continuous product development flow (individual project chartering not required)

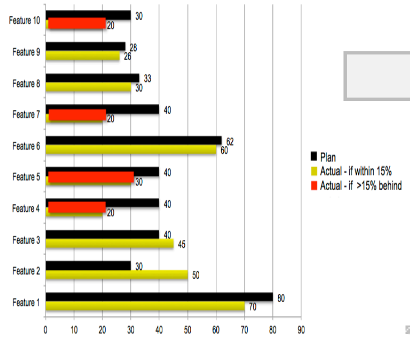
Release Train Tracking

Release Burndown



- Release burndown shows overall progress of release against release plan
- A standard agile report
- Automatically compiled from “time remaining” story updates in agile project management tooling
- Most valuable to program, release and executive managements

Feature Completeness Report

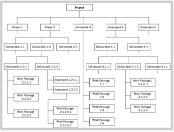


- Feature completeness report shows status of each release feature over time
- Automatically compiled from stories completed/stories remaining in agile project management tooling
- Most valuable to program and product managers

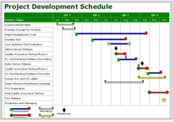
#7. From PMBOK

To Agile Project Management

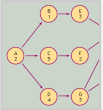
Traditional, performed by the *Project Manager*



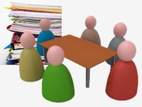
Work Breakdown Structure estimating



Gantt Charts scheduling

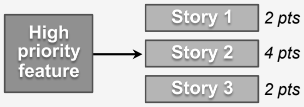


Critical Path analysis



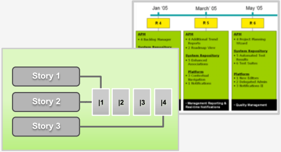
Reporting

Performed by *the team*



Actual velocity based estimating

Iteration planning



Release planning and roadmap

Release/iteration review/ retrospective

Rethinking Governance

8. Waterfall Milestones

#8. From Waterfall Milestone

To Fact-Based Governance

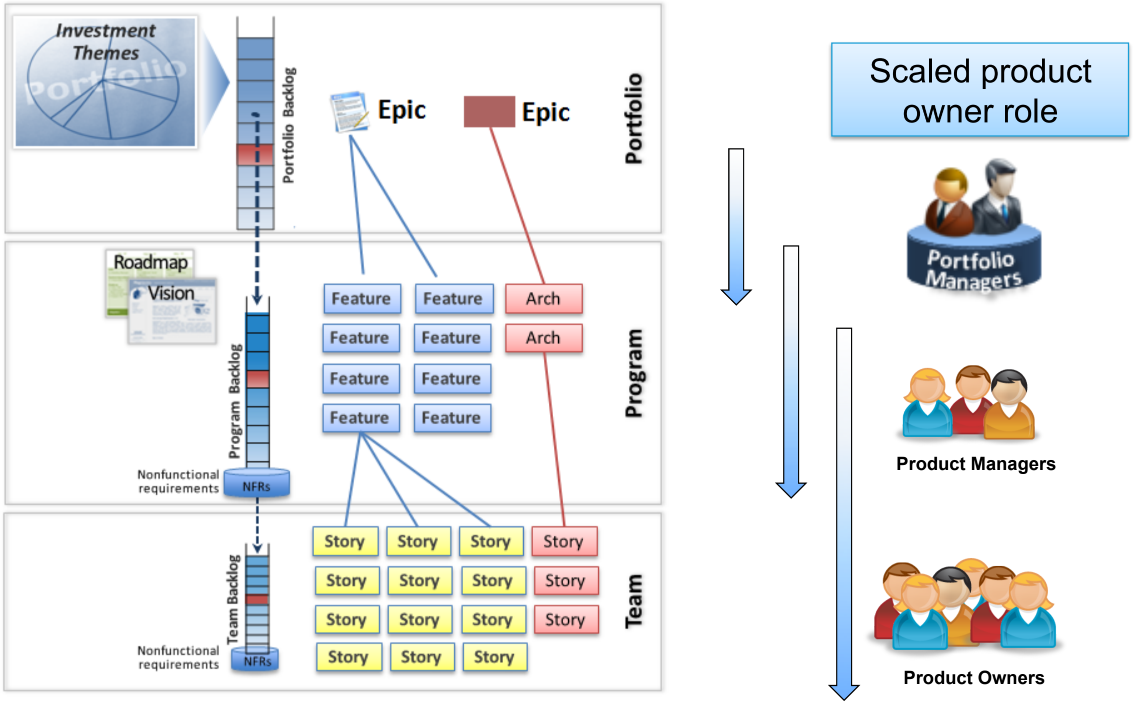
Waterfall Milestones

- Teams report milestones with document -based reviews
- Subjective, milestone reports do not correlate to actual project status
- Teams “report to” project office (leader as conductor/boss)
- Teams *cannot proceed* until and unless they pass milestones (start-wait-start-wait waste cycle)
- Scheduling delays and overhead
- Process changes dictated by “those who know best”

Fact-based

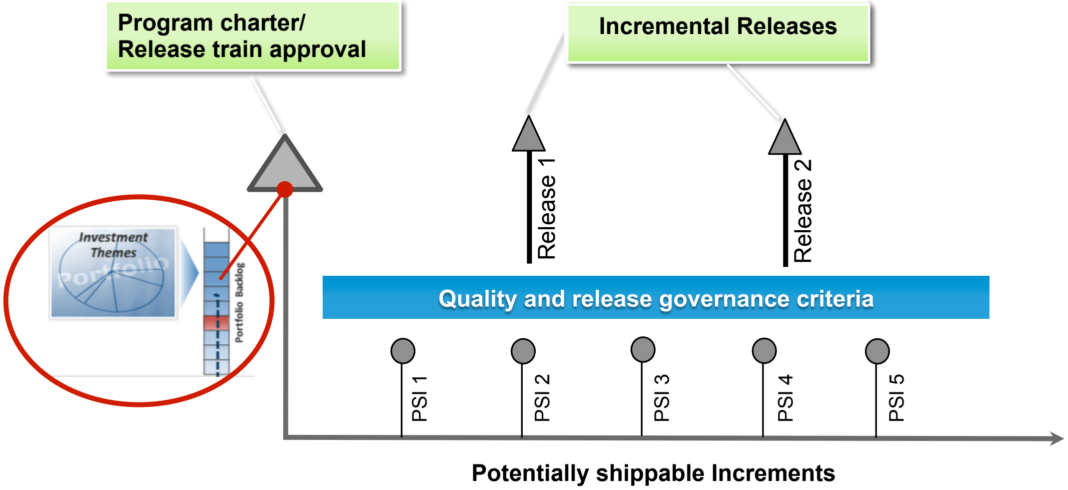
- Milestones are iterations ***and incremental releases of working code***
- Status and quality are quantitative, not subjective
- Project office “comes to teams” (enabling leadership model)
- Teams *default model is to proceed* unless stopped by business case (no process driven delays/waste)
- No scheduling delays and overhead
- Process changes applied and harvested from “those who do”

Governance – Content Authority



Governance – Agile Milestones

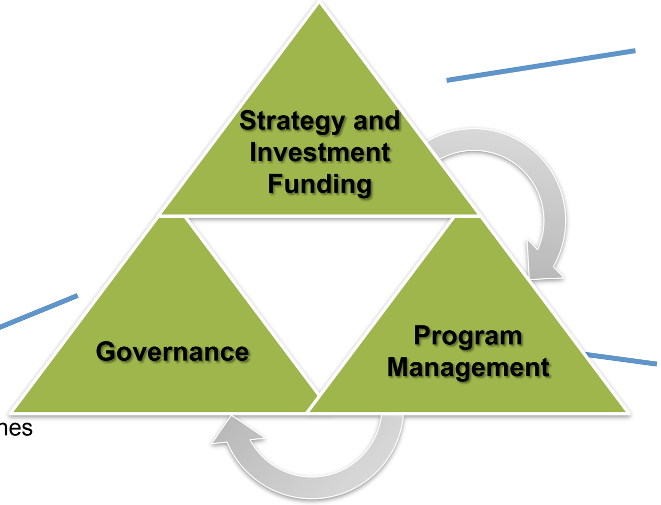
- All development occurs in PSI increments
- Agile milestones drive and support incremental delivery
- Quality and release governance standards apply to all releases



The Agile PPMO

The Agile PPMO

Mission: Enable, foster, and empower enterprise agility for business results



- Limiting Work in Process
- Lightweight business cases
- Incremental funding
- Decentralized rolling wave planning

- Agile estimating and planning
- Agile Release Trains
- Agile Project Management

- Fact-based assessment
- Agile milestones

The Agile PPMO enables, fosters, and promotes lean and agile practices across the enterprise

Sources and More About

- Leffingwell, Dean. 2011. *Agile Software Requirements: Lean Requirements Practices for Teams, Programs, and the Enterprise.*
- Sliger and Broderick. 2008.*The Software Project Manager's Bridge to Agility*
- Krebs, Jochen. 2008. *Agile Portfolio Management*
- Agile Project Management Leadership Network http://apln.org
- Thomas and Baker. 2008. *Establishing an Agile Portfolio to Align IT Investments with Business Need.*
  DTE Energy, Proceedings of Agile 2008