Lesson 3
Understanding SAFe Principles

Day 1

1. Introducing the Scaled Agile Framework
2. Embracing a Lean-Agile Mindset

   Break
3. Understanding SAFe Principles

   Lunch
4. Implementing an Agile Release Train

   Break
5. Experiencing PI Planning

Day 2

6. Executing and Releasing Value

   Break
7. Building an Agile Portfolio

   Lunch
8. Building Really Big Systems

   Break
9. Leading the Lean-Agile Enterprise

SAFe Lean-Agile principles

#1-Take an economic view

#2-Apply systems thinking

#3-Assume variability; preserve options

#4-Build incrementally with fast, integrated learning cycles

#5-Base milestones on objective evaluation of working systems

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

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

#8-Unlock the intrinsic motivation of knowledge workers

#9-Decentralize decision-making

Why the focus on principles?

A common disease that afflicts management the world over is the impression that "Our problems are different". They are different to be sure, but the principles that will help to improve quality of product and service are universal in nature. —*W. Edwards Deming*

100 books

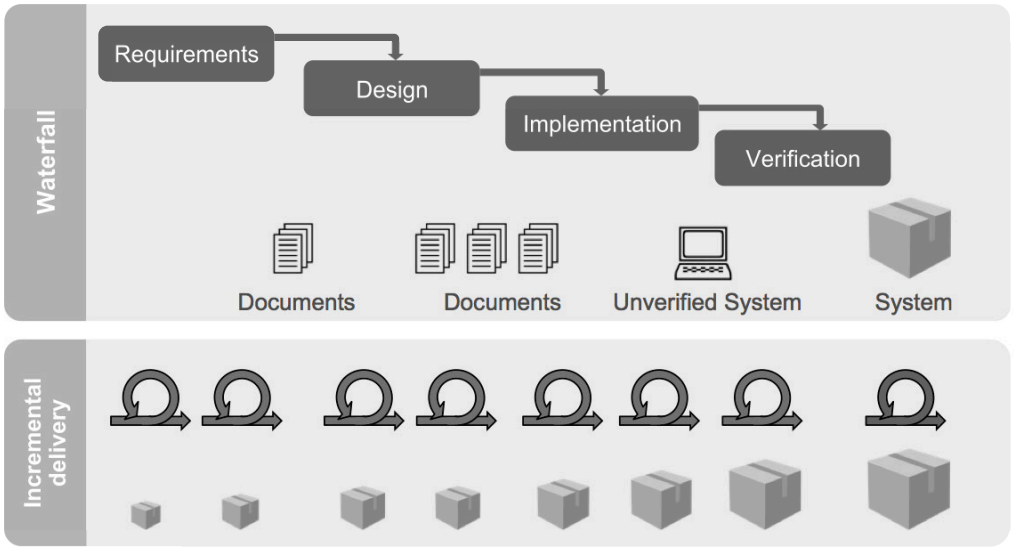
Principles over Practices

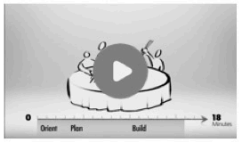
100 implementations

- A Lean-Agile transformation will deliver substantial benefits
- But it is a significant change and every implementation is different
- Leaders should understand why the practices work; it's part of "knowing what it is they must do"
- If a practice needs to change, understanding the principles will assure the change moves the enterprise in the right direction

#1 Take an economic view

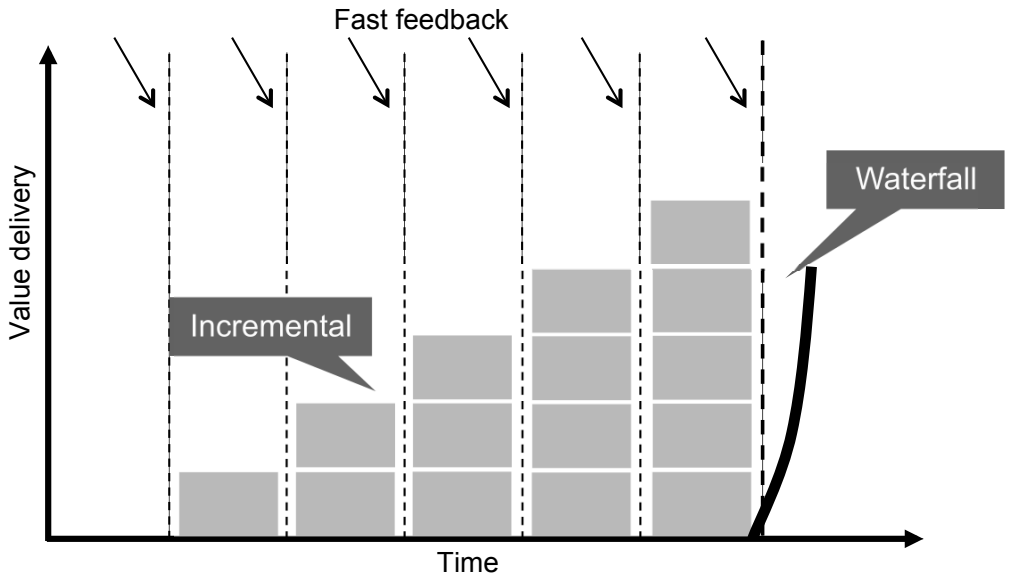
Agile economics: Deliver early and often





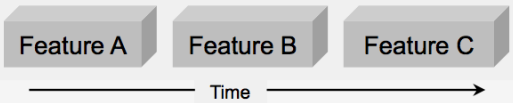
Iterative Learning: The "Marshmallow Challenge"
https://youtu.be/H0_yKBitO8M
7:22

Deliver value incrementally

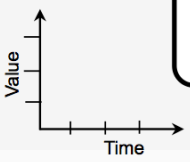


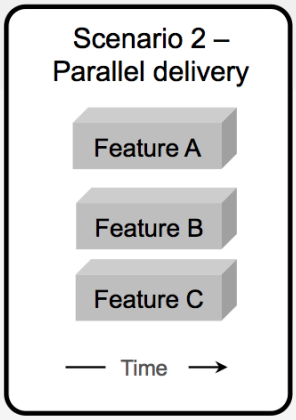
Exercise: Accelerating value delivery

Scenario 1 – Serial delivery



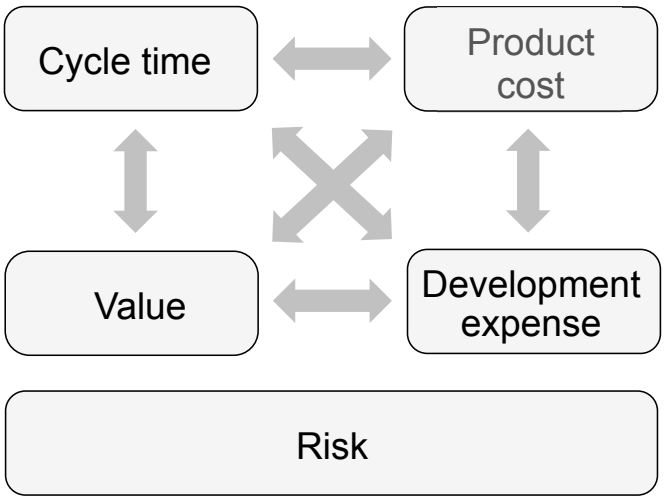
- Your backlog has three features. Each will take the entire team one month and delivers one unit of value.
- Plot the value delivery of serial and simultaneous/parallel implementation scenarios
  - Assume 20% task switching overhead for each team member in Scenario 2
- Hint: Plot the serial case first





Base decisions on economics

Understand tradeoff parameters



- Sequence jobs for maximum benefit
- Do not consider money already spent
- Make economic choices continuously
- Empower local decision making
- If you only quantify one thing, quantify the cost of delay

"Understanding economics requires understanding of the interaction amongst multiple variables."

—Don Reinertsen, *Principles of Product Development Flow*

#2 Apply systems thinking

Systems thinking

A system must be managed. It will not manage itself.

Left to themselves, components become selfish, independent profit centers and thus destroy the system…

The secret is cooperation between components toward the aim of the organization.

—W. Edwards Deming

Aspects of systems thinking

1. The solution itself is a system.

2. The enterprise building the system is a system too

3. Optimize the full value stream

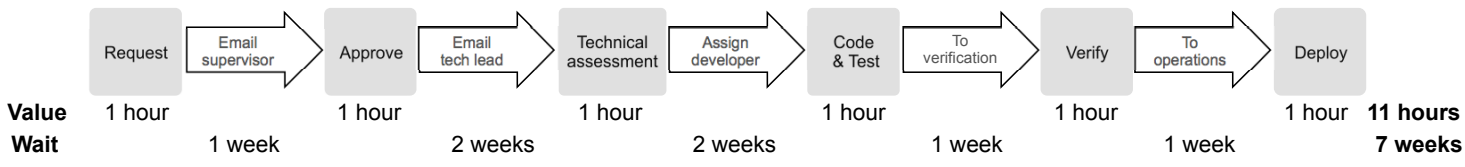
- Optimizing a component does not optimize the system
- For the system to behave well as a system, a higher-level understanding of behavior and architecture is required
- The value of a system passes through its interconnections
- A system can evolve no faster than its slowest integration point

Optimize the full value stream

All we are doing is looking at the timeline, from when the customer gives us an order to when we collect the cash. And we are reducing the timeline by reducing the non-value added wastes.
—Taiichi Ohno

- Most problems with your process will surface as *delays*
- Most of the time spent getting to market is a result of these delays
- Reducing delays is the fastest way to reduce time to market

Focus on the delays!

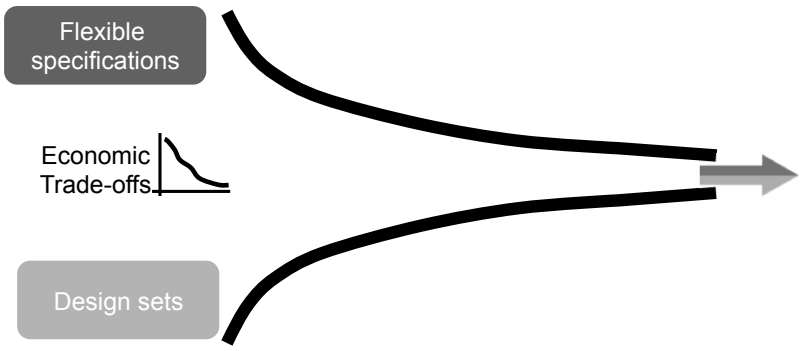


#3 Assume variability; preserve options

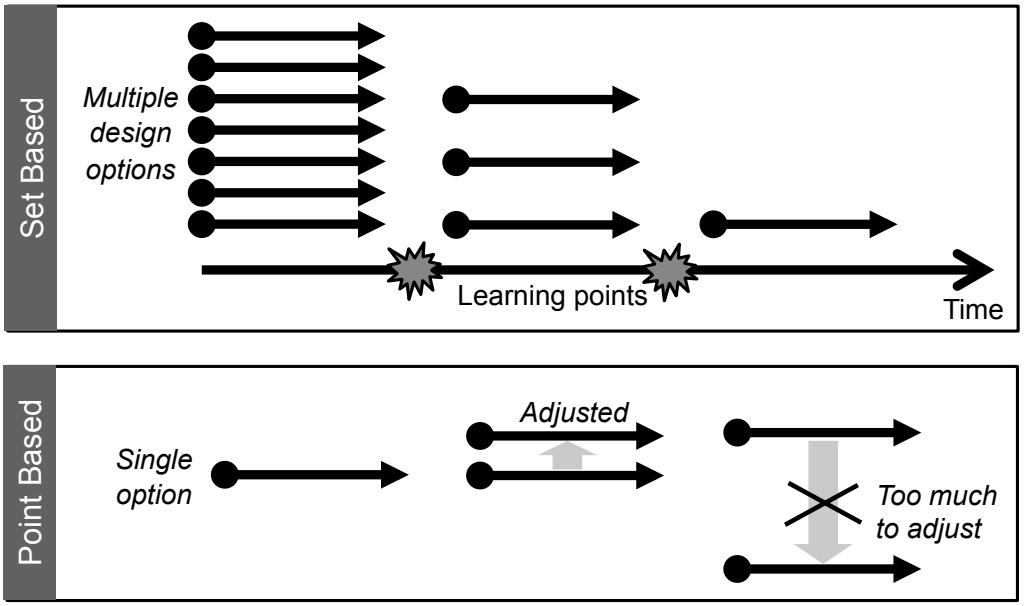
Development occurs in an uncertain world

Aggressively evaluate alternatives. Converge specifications and solution set.
—Allen Ward

- You cannot possibly know everything at the start
- Requirements must be flexible to make economic design choices
- Designs must be flexible to support changing requirements
- Preservation of options
- improves economic results



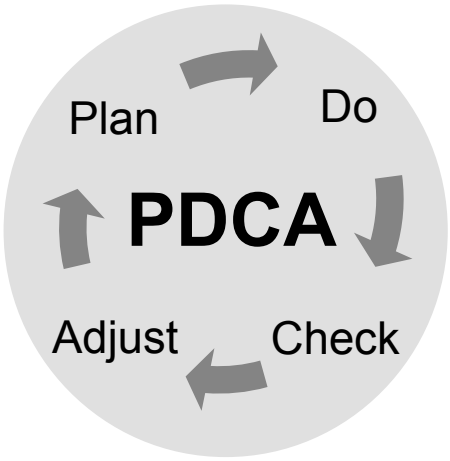
Apply a Set Based approach



#4 Build incrementally with fast, integrated learning cycles

Apply fast learning cycles

Fast feedback accelerates knowledge.



The iterative learning cycle

Principles of Product Development Flow, Don Reinertsen
Plan. Do. Check. Act. (Adjust), W. Edwards Deming, Walter Shewhart, et al.

- Improves learning efficiency by decreasing the time between action and effect
- Reduces the cost of risk-taking by truncating unsuccessful paths quickly
- Facilitated by small batch sizes
- Requires increased investment in development environment

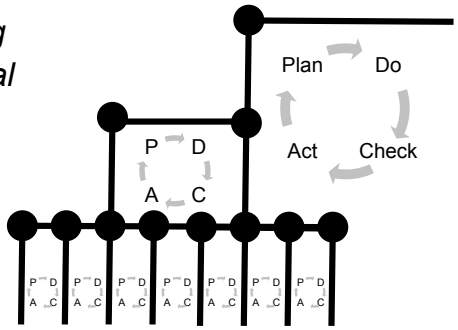
The shorter the cycles, the faster the learning

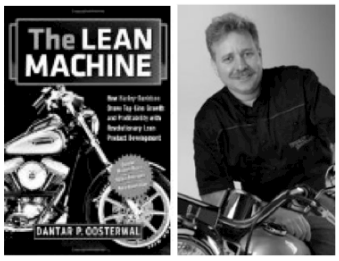
Apply fast learning cycles

Product development is the process of converting uncertainty to knowledge —Dantar P. Oosterwal

Integration points control product development

- Integration points accelerate learning
- Development can proceed no faster than the slowest learning loop
- Improvement comes through synchronization of design loops and faster learning cycles

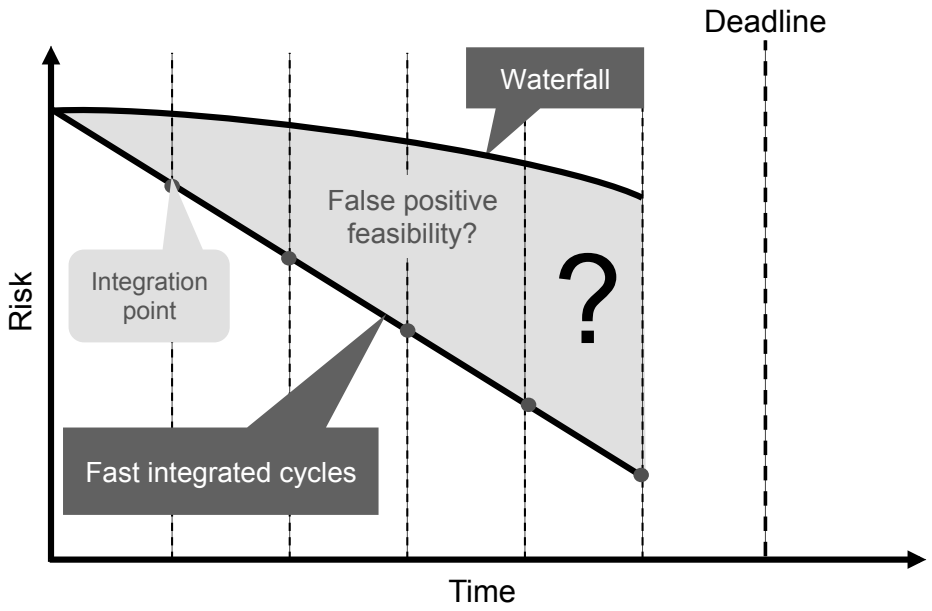




***The Lean Machine**: How Harley Davidson Drove Top-Line Growth and Profitability with Revolutionary Lean Product Development*

—Dantar P. Oosterwal

Integration points reduce risk

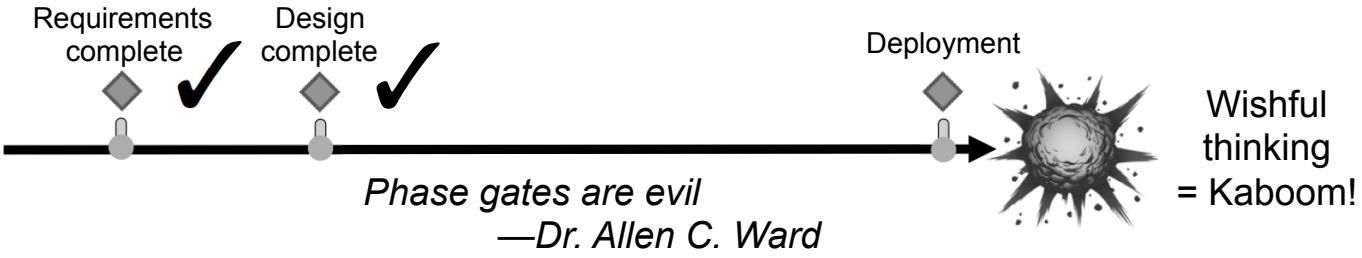


#5 Base milestones on objective evaluation of working systems

The problem of phase gate milestones

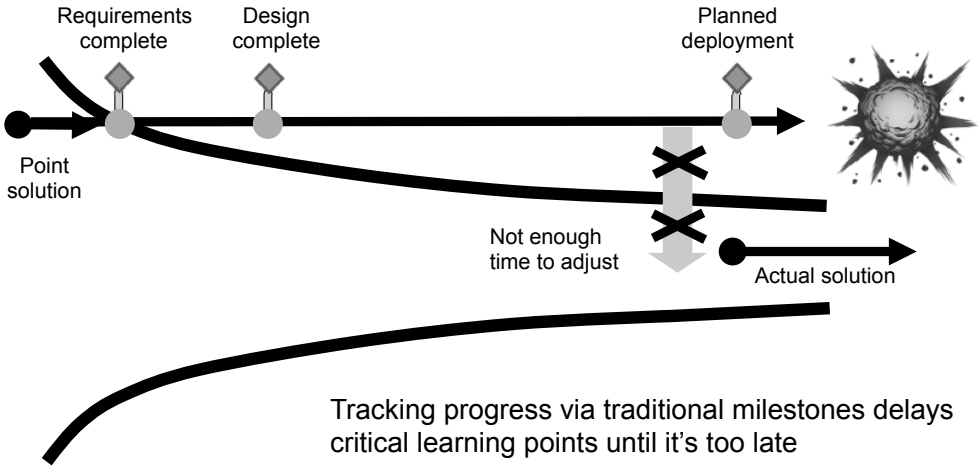
There was in fact no correlation between exiting phase gates on time and project success… the data suggested the inverse might be true. —*Lean Machine*

- Force too early design decisions; encourages false positive feasibility
- Assume a "point" solution exists and can be built right the first time
- Create huge batches and long queues; centralizes requirements and design in program management



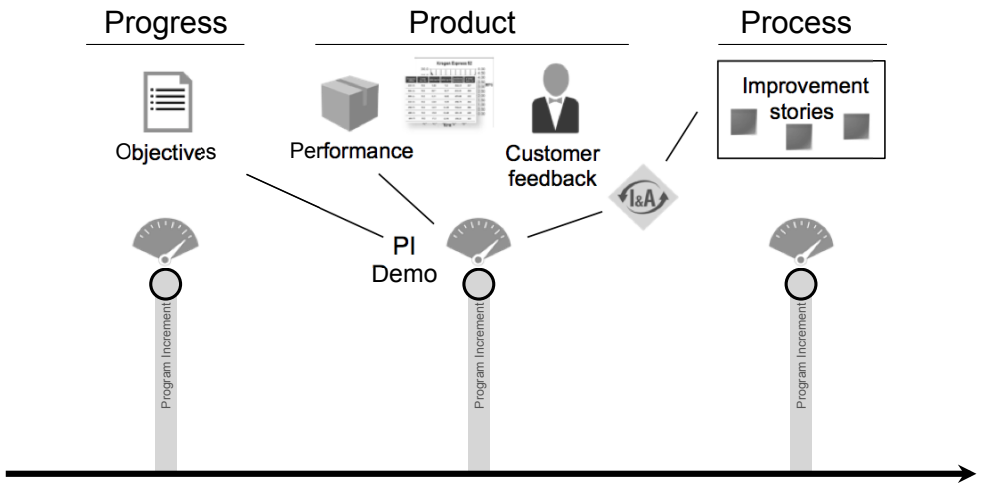
The problem of phase gate milestones

Phase gates fix requirements and designs too early, making adjustments costly and late as new facts emerge.



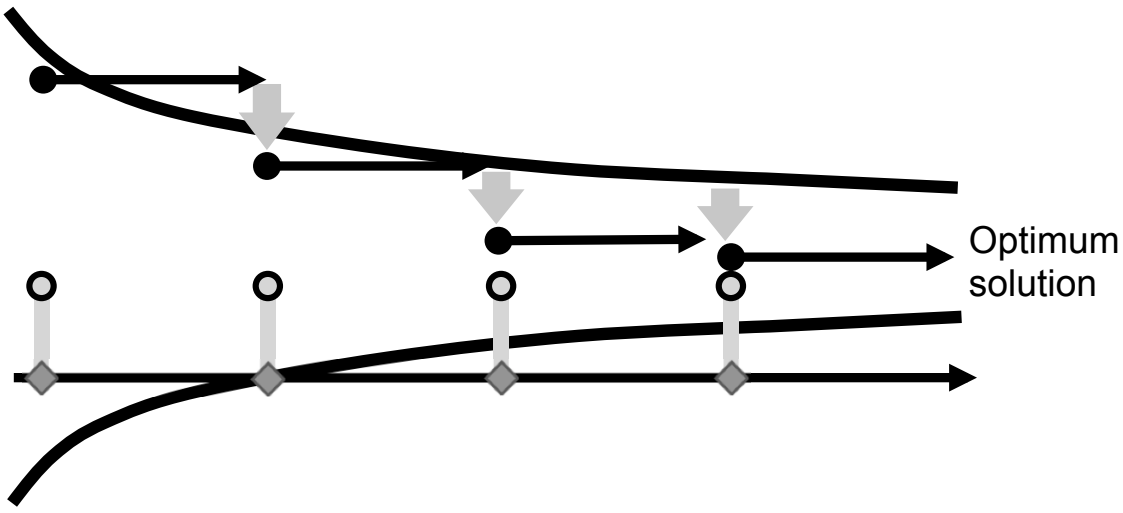
Apply objective Milestones

PI Demos are orchestrated to deliver objective progress, product and process metrics.



Iterate to the optimum solution

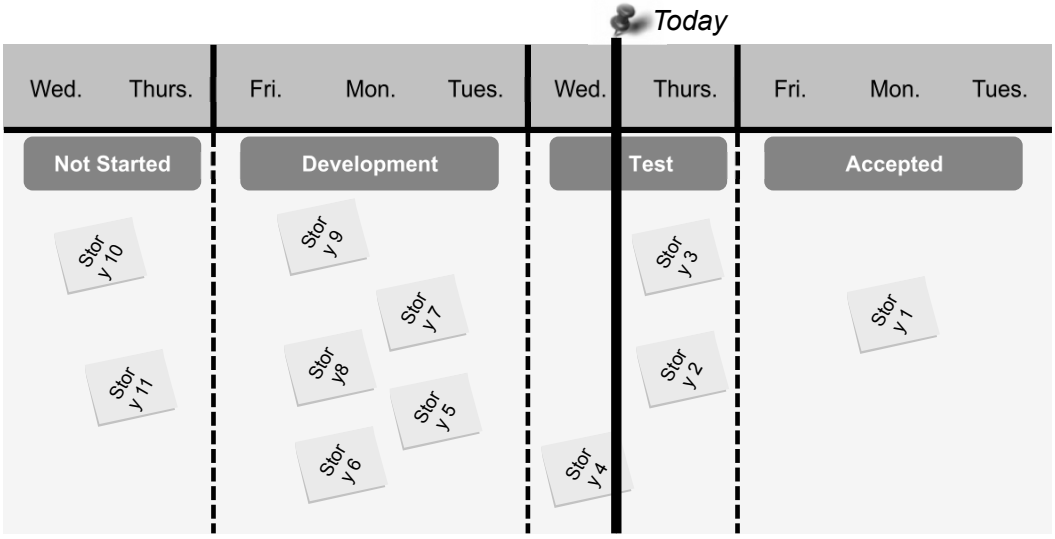
Objective milestones facilitate learning and allow for continuous, cost-effective adjustments towards an optimum solution.



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

Visualize and limit work in progress

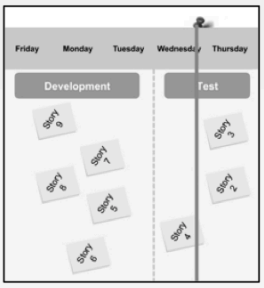
One team's Big Visible Information Radiator (BVIR)



How is this team doing? How do you know that?

Exercise: Work in progress constraints

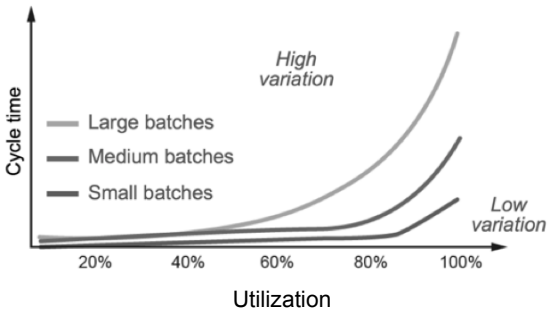
- Consider the BVIR on the previous page, then discuss:
- What would the effect be of a three-story WIP constraint on Development and Test?
- Scenario: You're a developer. You just finished story 6. What would you do if:
  a. There is no WIP constraint?
  b. The three-story WIP constraint is in place?
- Which scenario has the highest throughput?



Reduce batch size

Small batches go through the system faster, with lower variability.

- Large batch sizes increase variability
- High utilization increases variability
- Severe project slippage is the most likely result
- Most important batch is the transport (handoff) batch
- Proximity (co-location) enables small batch size
- Good infrastructure enables small batches



Implementing Lean Software Development, Mary Poppendieck

Principles of Product Development Flow, Don Reinertsen

Exercise: Large batch push

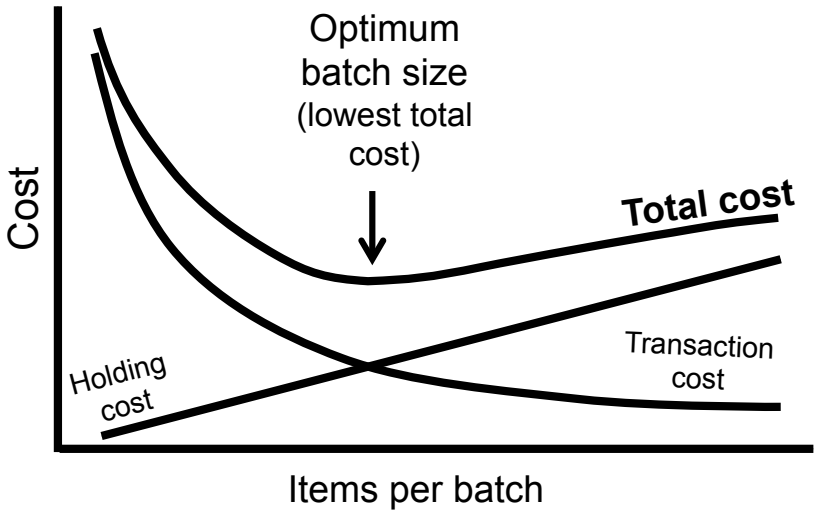
- Create groups of five people, with 10 coins per group. One person is the timekeeper. The remaining four people process the coins.
- Person by person, flip all coins one at a time, recording your own results (heads or tails)
- Pass all coins at the same time to the next person
- Time keeper records time from the start of the first flip to the completion of the last flip for the group

Exercise: Small batch pull

- Similar four person process
- Each person flips each coin one at a time and records the result
- But, passes each coin as flipped
- The time keeper records the time from the start of the first flip to the completion of the last flip

Finding optimum batch size

Optimum batch size is an example of a U-curve optimization.

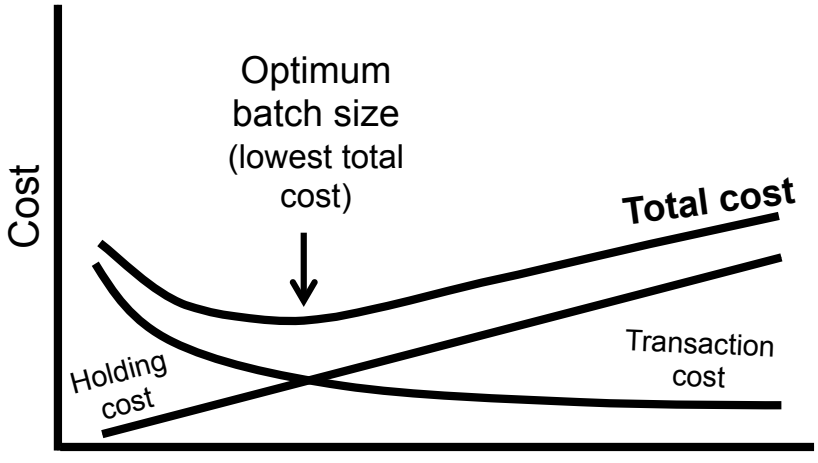


Principles of Product Development Flow, Don Reinertsen

- *Total* costs are the sum of holding costs and transaction costs
- Higher transaction costs shift optimum batch size higher
- Higher holding costs shift batch size lower

Reducing optimum batch size

Reducing transaction costs reduces total costs, and shifts optimum batch size lower.



Principles of Product Development Flow, Don Reinertsen

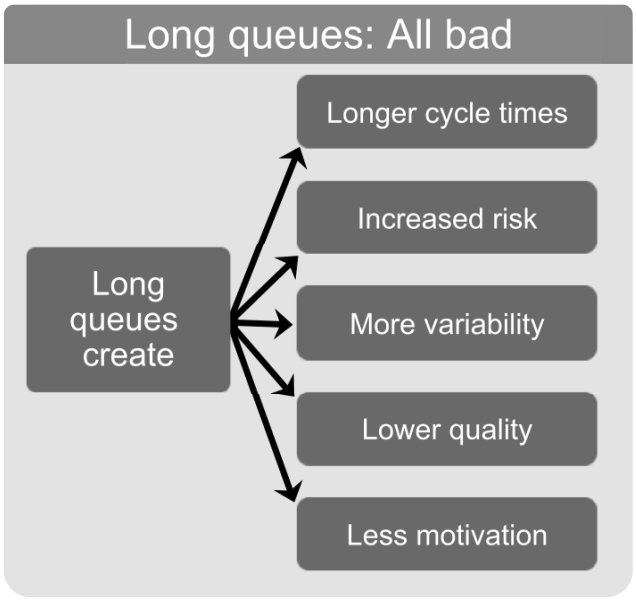
- Reducing batch size:
  - Increases predictability
  - Accelerates feedback
  - Reduces rework
  - Lowers cost
- Batch size reduction probably saves twice what you think

Reducing transaction costs example
https://youtu.be/RRy_73ivcms
2:09

Manage queue lengths

Email from a client service organization:

Thank you for contacting us.
We are experiencing increased volumes and apologize in advance for the delay. Our goal is to contact you within . . .



Principles of Product Development Flow, Don Reinertsen

Reduce queue lengths

- Understand Little's Law
- Faster processing time decreases wait
- Control wait times by controlling queue lengths

$$W_q = \frac{L_q}{\lambda}$$

Average wait time = average queue length / average processing rate

Example: given average processing speed of 10 features per quarter and a committed set of 30 features, a new feature will experience approximate wait time of:

$$\frac{30 \text{ items}}{10 \text{ items/Q}} = 3 \text{Q}$$

#7 Apply cadence, synchronize with cross-domain planning

Cadence and synchronization

Cadence

- Converts unpredictable events into predictable ones. Lowers cost.
- Makes waiting times for new work predictable
- Supports regular planning and cross-functional coordination
- Limits batch sizes to a single interval
- Controls injection of new work
- Provides scheduled integration points

Note: Delivering on cadence requires scope or capacity margin

Synchronization

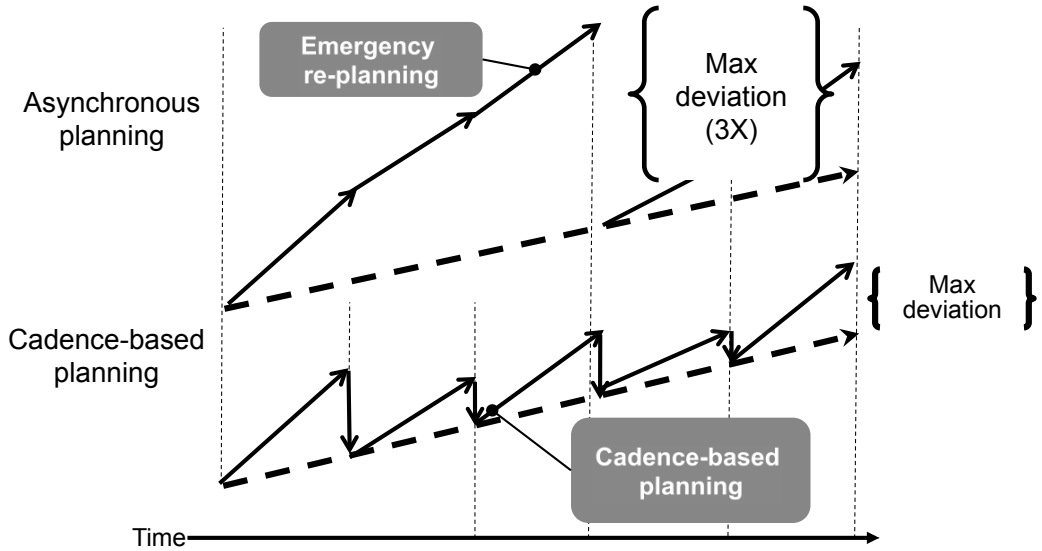
- Causes multiple events to happen at the same time
- Facilitates cross-functional tradeoffs
- Provides routine dependency management
- Supports full system and integration and assessment
- Provides multiple feedback perspectives

Note: To work effectively, design cycles must be synchronized

Principles of Product Development Flow, Don Reinertsen
The Lean Machine, Dantar Ootserwall

Control variability with planning cadence

Cadence-based planning limits variability to a single interval.



Synchronize with cross-domain planning

Future product development tasks can't be pre-determined. Distribute planning and control to those who can understand and react to the end results.

— Michael Kennedy, Product Development for the Lean Enterprise

- All stakeholders face-to-face, (but typically multiple locations)
- Management sets the mission, with minimum possible constraints
- Requirements and design happen
- Important stakeholder decisions are accelerated
- Teams create—and take responsibilty—for plans

#8 Unlock the intrinsic motivation of knowledge workers

Drive: The puzzling puzzles of Harry Harlow

The 1949 experiment

- Eight rhesus monkeys for a two week experiment on motivation and learning
- Puzzles were placed in their cages

Results

- Unprompted, the monkeys solved the puzzles on their own
- An interesting and not understood phenomenon
- As a motivator, raisins were added as rewards
- Result: the monkeys made more errors and solved the problems less frequently

It appears that the performance of the task provides its own intrinsic reward … this drive … may be as basic as the others —The Surprising Truth About What Motivates Us, Daniel H. Pink

#9 Decentralize decision-making

Decentralize decision-making

Define the economic logic behind a decision; empower others to actually make them.

Centralize	**De-centralize** everything else
Infrequent - Not made very often and usually not urgent *(ex: internationalization strategy)*	**Frequent and common** - Routine, every day decisions *(ex: team and program backlog)*
Long lasting - Once made, highly unlikely to change *(ex: common technology platform)*	**Time critical** - High cost of delay *(ex: point release to customer)*
Significant economies of scale - Provides large and broad economic benefit *(ex: compensation strategy)*	**Require local information** - Specific, and local technology or customer context is required *(ex: Feature criteria)*

Decentralizing decision making in nuclear submarine command
https://youtu.be/OqmdLcyES_Q
9:47

Homework exercise: Decentralize decision-making

- Consider three significant decisions you are currently facing
- Rate each item using the table below
- Would you centralize or decentralize the decision?

Decision	Frequent? Y=2 N=0	Time-critical? Y=2 N=0	Economies of scale? Y=0 N=2	Total

Scale: 0-2 (Low to high) Then add the total: 0-3: Centralize; 4-6: Decentralize

Principles are great, but…

Clarity on how to think, without clarity on how to act, leaves people unmoved.

—Dan Pink

. . . it's time to put this thinking to work.
Let's move to doing.

Lesson summary

In this lesson, you have started a journey to know how to apply SAFe Lean-Agile principles, including:

#1-Take an economic view

#2-Apply systems thinking

#3-Assume variability; preserve options

#4-Build incrementally with fast, integrated learning cycles

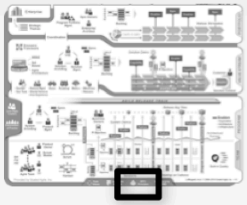
#5-Base milestones on objective evaluation of working systems

#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



Suggested Scaled Agile Framework reading: "SAFe Principles" article