# Software Development Methodologies

## How software engineering teams really work

William Chan
Lead Platform Engineer, 605.tv, Capital One, FreeWheel (Comcast)

# Objectives

- How has software development processes evolved over the course of the years?
- What are the different mental models associated to software development?
- What are the different software methodologies?
- When to use one methodology over another?

# Why Software Development Methodologies?

Software development methodologies are ways that you can use and follow to operationalize your team to *deliver software that provides value to your customers* 3

# General Development Process Review

1. Define the product
2. Gather requirements
3. Design the application
4. Implement the application
5. Test the application
6. Release the application
7. Gather user feedback

4

# The Old Days

5

# Waterfall

Thought process: The more time spent on upfront design, the less time and effort it'll require later on

- Requires a highly structured approach
- Requires thorough documentation
- Usually provides visible and linear milestones
- Suited for projects whose requirements and scope are fixed and that the technology is well understood

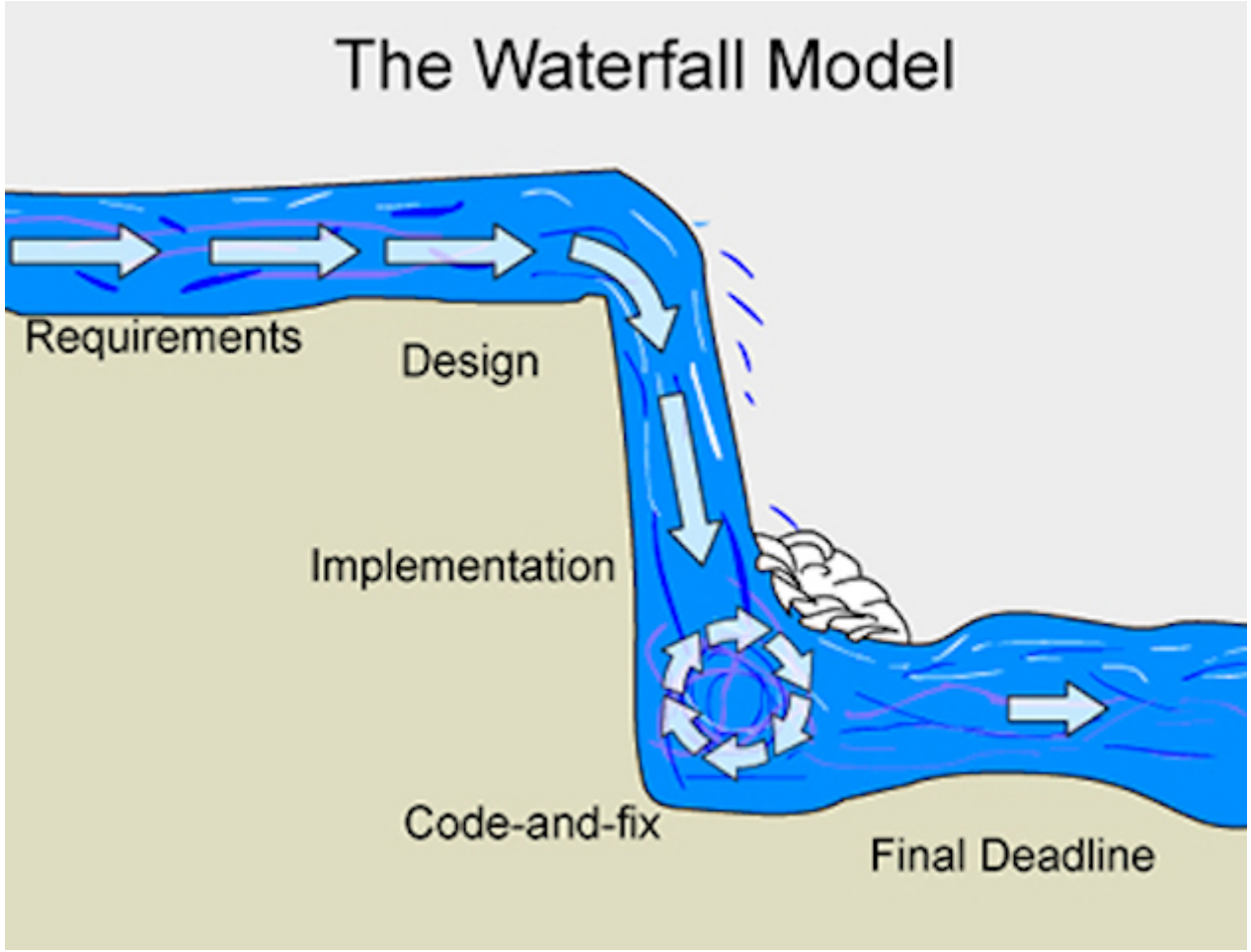
6

# Phases: Waterfall

1. Requirements
2. Design
3. Implementation
4. Testing
5. Release
6. Verification

7

# Waterfall




Source: http://jaymeholmes.com/jad.html

# Waterfall Model's Shortcomings

- Requirements often do change leading to increased costs when found at later stages of the development life cycle
- Unforeseen complexity in the technology or product logic

# A Group Exercise

10

# Problem Statement

How do I get the following done in the shortest amount of time?

**Situation:** You have 10 coins on the table with tails facing up
**Problem:** You want all 10 of the coins to have heads facing up
**Team:** 3 people will work with each other by flipping coins and passing it on to the next team mate

**Rules:**
1. Player 1 must pass on heads to player 2
2. Player 2 must pass on tails to player 3
3. Player 3 finishes by flipping the coins to heads

# First Solution: Batch

1. Player 1 flips **all the coins** from *tails to heads* and then **hands all the coins** all 10 coins to player 2
2. Player 2 flips **all the coins** from *heads to tails* and then **hands all the coins** all 10 coins to player 3
3. Player 3 flips **all the coins** from *tails to heads*

12

# Second Solution: Single Coin

1. Player 1 flips **one coin** from *tails to heads* and then **hands one coin** to player 2
2. Player 2 flips **one coin** from *heads to tails* and then **hands one coin** to player 3
3. Player 3 flips **one coin** from *tails to heads* at a time

13

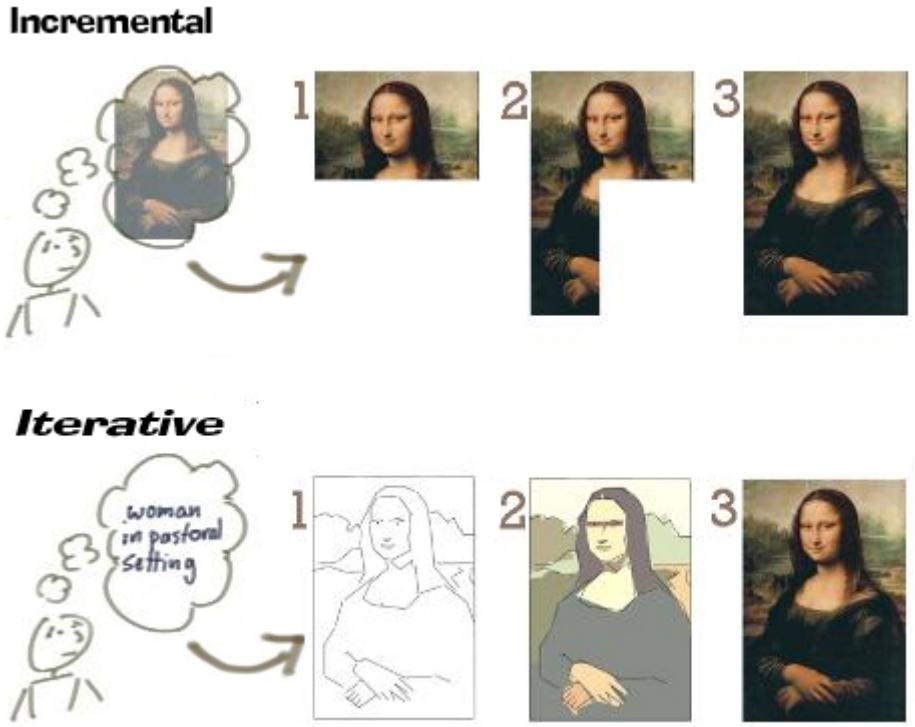
# Modern Software Development Mindsets

14

# Iterative vs Incremental

- Incremental - slowly build up to the envisioned product using cycles
- Iterative - build a simple version of the envisioned product to verify the market to see if you should continue investing to build it out further

# Iterative vs Incremental: A Demonstration




Source: https://www.infoq.com/news/2008/01/iterating-and-incrementing

16

# Lean

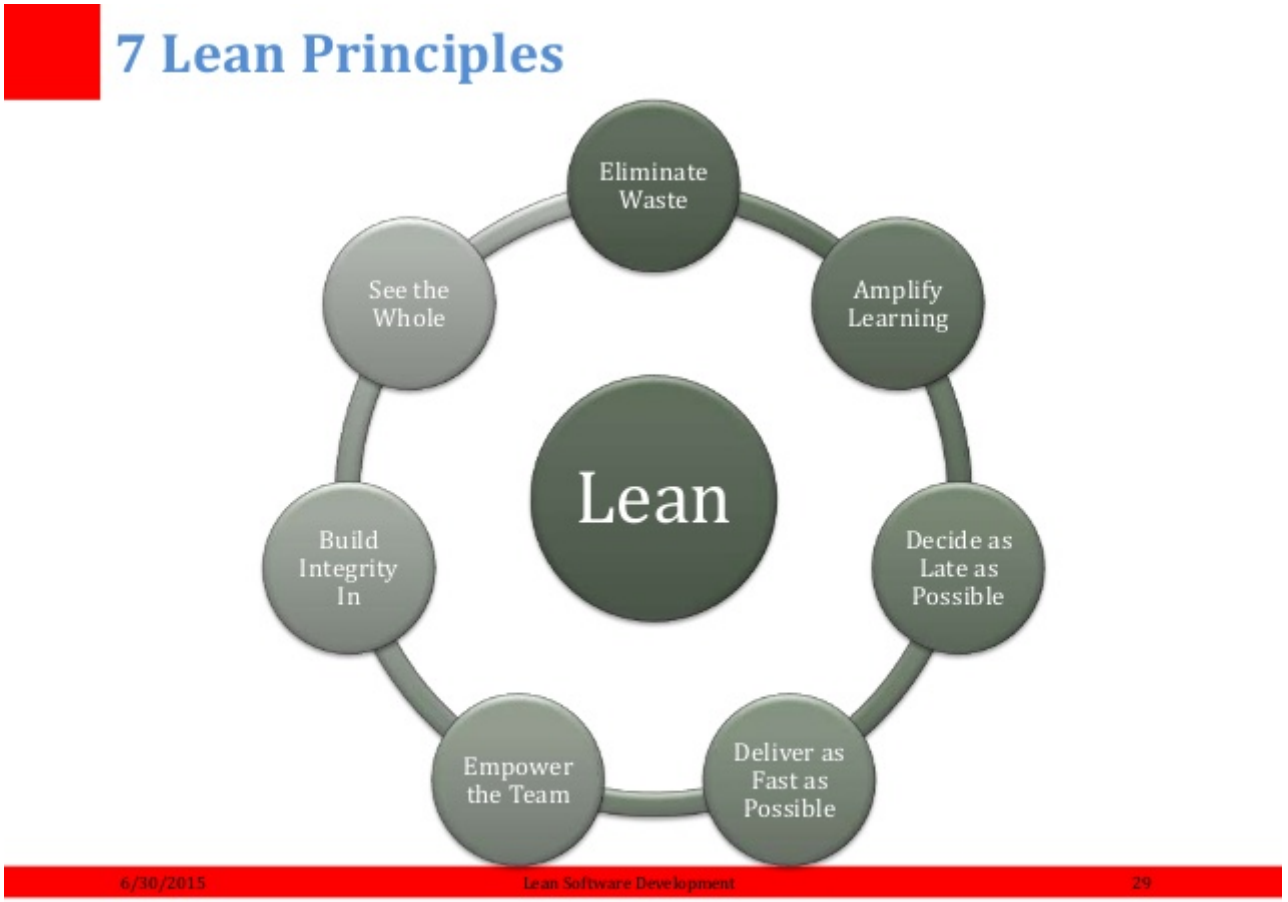
A is also not a process, methodology, or framework. It is the mindset that eliminates waste and maximizes value.

Thought process: Minimize waste and maximize value

**Principles**

- Eliminate waste
- Amplify learning
- Decide as late as possible
- Deliver as fast as possible
- Empower the team
- Build integrity in
- See the whole

# The Lean Cycle




Source: https://www.slideshare.net/SmartBizVN/introduction-to-agile-and-lean-software-development

# Agile Revolution

Problem: Waterfall didn't match real world expectations

Solution: The introduction of **agile** methodologies, essentially shorter waterfall cycles

19

# What is agile?

Agile is not a process, methodology, or framework. It is a mindset that welcomes uncertainty, embraces challenges, empowers individuals and views failure as a learning opportunity

Thought process: Change is constant and complexity can not be completely known

- Shorter development cycles to allow quick and early feedback
- Prioritizes learning and adaptation to respond quickly to changes
- Ship a "workable" product in each cycle

20

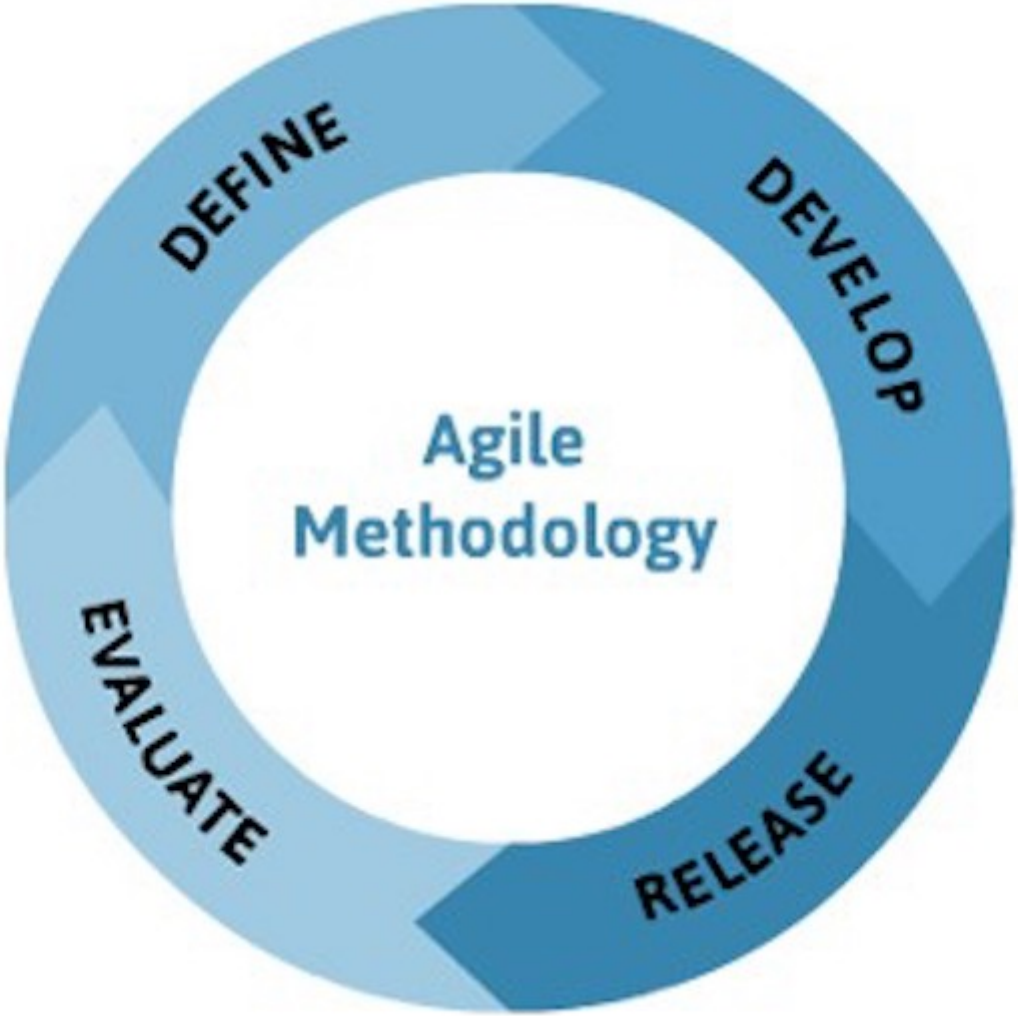
# Phases: Agile Methodologies

Very similar to waterfall but instead of months, quarters or years, think **weeks** to complete a small feature set based on the following.

1. Requirements
2. Design
3. Implementation
4. Testing
5. Release
6. Verification

21

# The Agile Cycle




Source: https://medium.com/@arun_87007/8-benefits-of-implementing-agile-methodology-in-your-project-b3552debeb0d

# Agile Manifesto

# Principles behind the Agile Manifesto

We follow these principles:
Our highest priority is to satisfy the customer through early and continuous delivery of valuable software.

Welcome changing requirements, even late in development. Agile processes harness change for the customer's competitive advantage.

Deliver working software frequently, from a couple of weeks to a couple of months, with a preference to the shorter timescale.

Business people and developers must work together daily throughout the project.

# Principles behind the Agile Manifesto Cont'd

Build projects around motivated individuals.
Give them the environment and support they need,
and trust them to get the job done.

The most efficient and effective method of
conveying information to and within a development
team is face-to-face conversation.

Working software is the primary measure of progress.

Agile processes promote sustainable development.
The sponsors, developers, and users should be able
to maintain a constant pace indefinitely.

Continuous attention to technical excellence
and good design enhances agility.

# Principles behind the Agile Manifesto Cont'd

Simplicity--the art of maximizing the amount of work not done--is essential.

The best architectures, requirements, and designs emerge from self-organizing teams.

At regular intervals, the team reflects on how to become more effective, then tunes and adjusts its behavior accordingly.

# The Methodologies

27

# Agile Methodologies

## Methodologies

- Scrum - breaking down work into timeboxed iterations, called sprints
- Kanban - based on the flow of work and the elimination of wastes, focusing just on the value delivered
- Scrumban - combination of scrum and kanban

## For larger organizations

- LeSS - large scale scrum; like scrum but focused on larger product developments with coordination efforts
- SAFe - scaled agile framework; like scrum but for larger teams building a hierarchy to which multiple scrum teams report into called the agile release train

# Scrum

Thought process: Increasing the flexibility and the adaptability of your team to uncertainties

- Scrum teams are usually cross functional
- Scrum is timeboxed usually in the form of 1 or 2 week periods called **sprints**
- The most common metrics gathered are **velocity** and **capacity**
- Team members who finishes their task first often swarms to help other team members to complete the stories committed in the sprint

# Scrum: Terminologies

- Sprints - planned timeboxed cycles with a certain amount of stories allocated
- Story points - rough estimation of complexity by coming to a team consensus
- Capacity - the amount of story points that can completed by the team in a sprint cycle
- Velocity - the amount of story points completed per sprint, generally used to predict overall capacity
- Blockers - problems or issues that arise that is blocking a story from being complete
- Commitment level - team's comfort level with the story points allocated to a given sprint

30

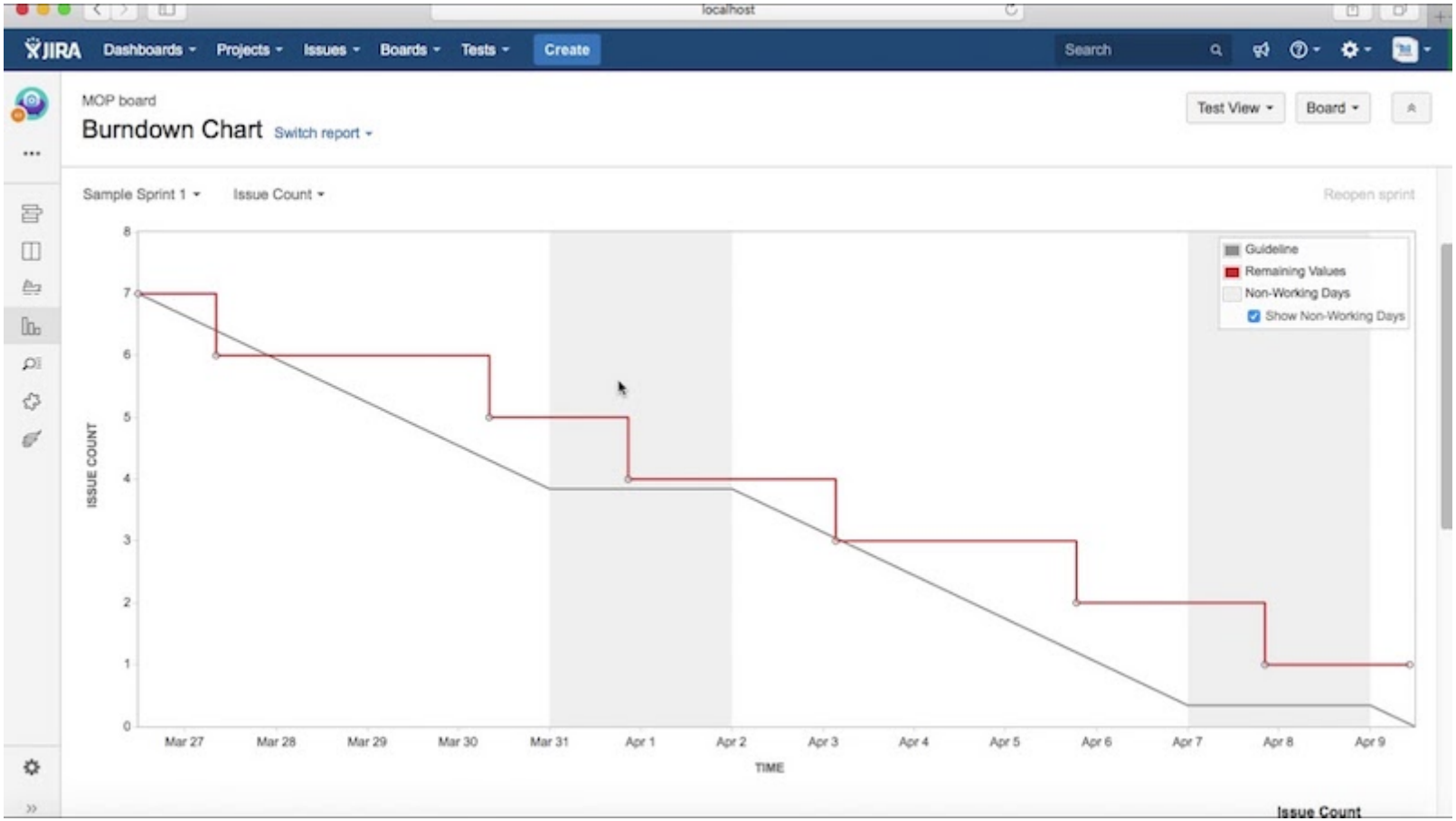
# What does a Sprint look like?

1. Sprint planning - estimating complexity of story points and planning to start the sprint with stories matching capacity as close as possible
2. Daily standup - everyone stands up giving an update answering the questions of what they did yesterday, what they are going to do today and if there are any blockers
3. Backlog grooming - writing stories, figuring out requirements, estimating the points to for the story points
4. Retrospectives - a meeting to reflect on practices that led to positive outcomes and discontinuing bad practices that led to bad outcomes

# Scrum: Useful charts

- Burn down chart - a chart that keeps track of story points getting completed, measuring story points outstanding
- Velocity chart - measures the volatility, difference between story points completed vs outstanding

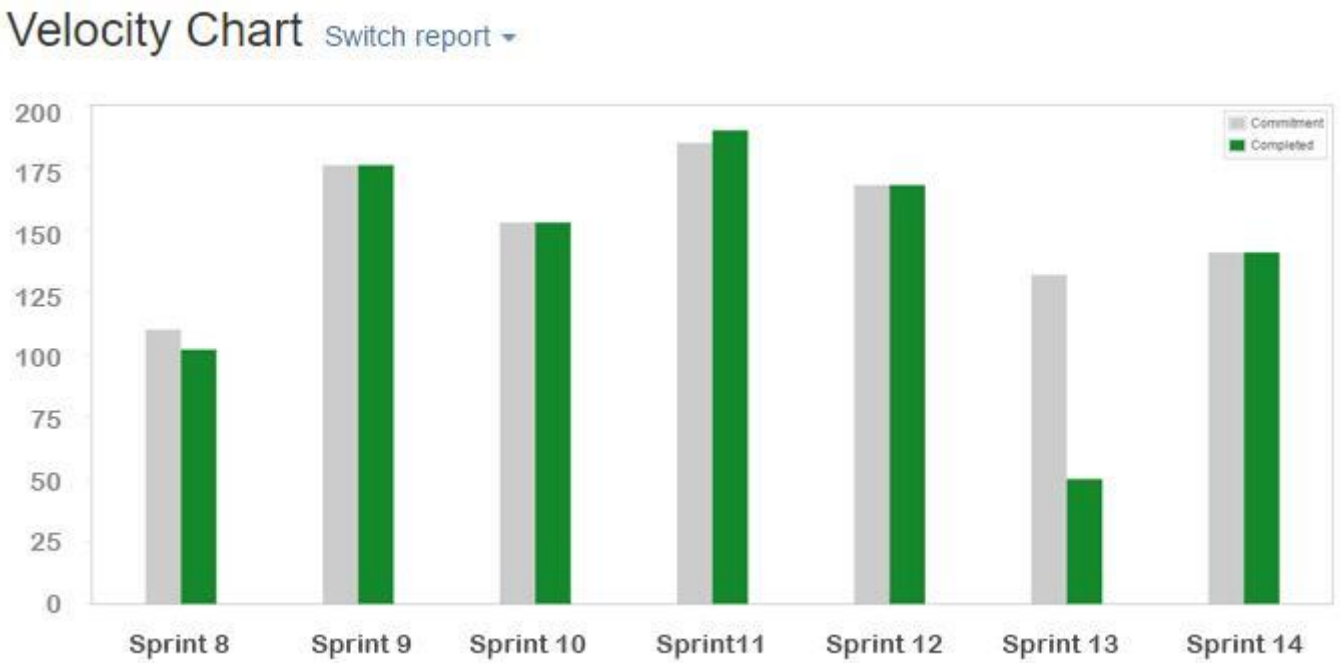
# Scrum: Burndown Chart




Source: https://www.youtube.com/watch?v=IyhaZ5gNZ_M

33

# Scrum: Velocity Chart




Source: https://www.researchgate.net/figure/Velocity-chart-using-JIRA-following-scrum-approach_fig2_316665790

# Kanban

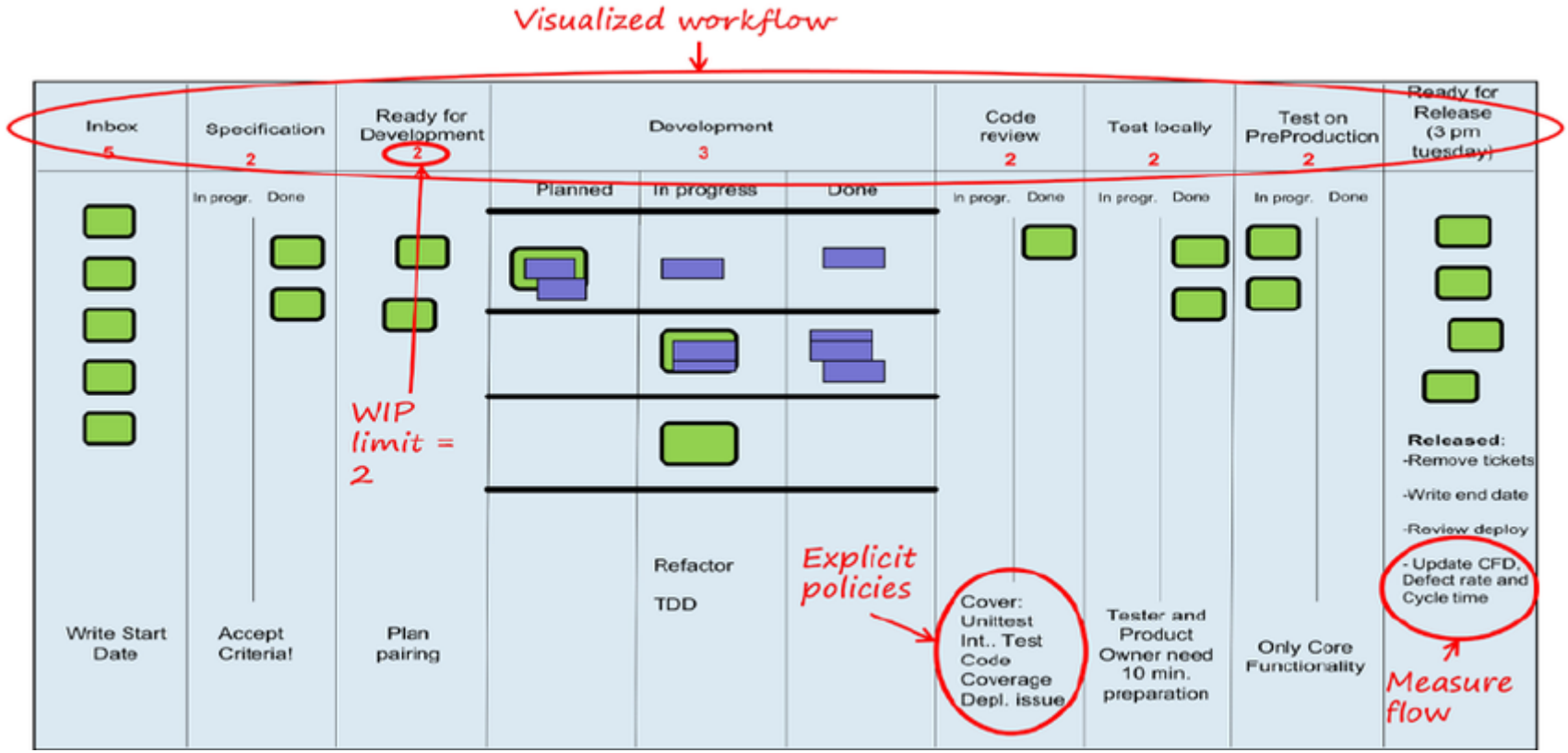
Thought process: Focus on the value delivered rather than extraneous processes, a lot like the lean methodology

Originated from Toyota as a manufacturing process to ensure demand and capacity across the value chain.

- Making work visible to find where the bottlenecks are
- Limit the work in progress and based on completion to avoid expensive context switching and makes problems visible
- Focus on flow

35

# The Kanban Cycle




Source: https://www.researchgate.net/figure/Kanban-board-and-principles-in-action-adapted-from-Boeg-11_fig1_267514980

# What the process looked like?

1. Kanban card determines the production of new parts
2. If there are lack of cards, production is stopped to ensure that the number of parts is balanced with demand, to limit overproduction
3. Use similar size cards
4. Avoid impediments and only start development on stories that are ready

**Advantages**

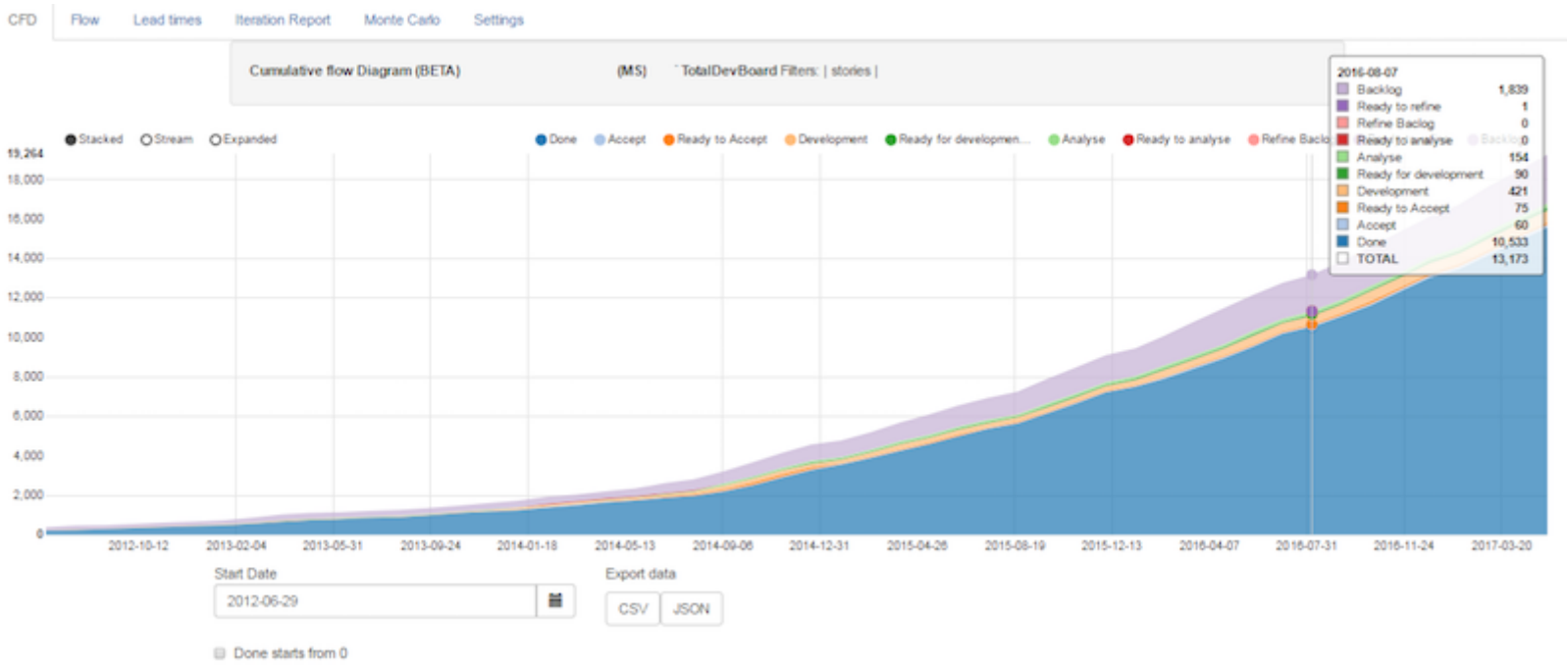
- Preventing overproduction and limits work in progress to not tie up resources
- Gives transparency into work where bottlenecks are easily seen
- Can be measure capacity with respect to demand

# Kanban's Shortcomings

- Focused on repetitive productions plans and less on bugs or unforeseen issue
- Does not work well when there is high volatility and wait times between tasks being done
- Measurements such as velocity stops becoming valuable

# Kanban: Cumulative Flow Chart

Cumulative flow diagram - keeping track of stories added, completed and being worked in progress




Source: https://medium.com/@siverbrant/powerfull-reporting-with-jira-flow-companion-707e11c09ad3

# Scrum vs Kanban

These are the most popular two methodologies. Below is a chart for comparison

| | Scrum | Kanban |
|---|---|---|
| Cadence | Regular fixed length sprints (ie, 2 weeks) | Continuous flow |
| Release methodology | At the end of each sprint if approved by the product owner | Continuous delivery or at the team's discretion |
| Roles | Product owner, scrum master, development team | No existing roles. Some teams enlist the help of an agile coach. |
| Key metrics | Velocity | Cycle time |
| Change philosophy | Teams should strive to not make changes to the sprint forecast during the sprint. Doing so compromises learnings around estimation. | Change can happen at any time |

# Scrumban

- Scrum + Kanban
- Use Kanban to increase flow, visibility, and pull tasks as they're finished
- Use Scrum ceremonies to plan ahead

# Scrum for Larger Organizations

42

# LeSS

Abbreviation: Large Scale Scrum

**Team**

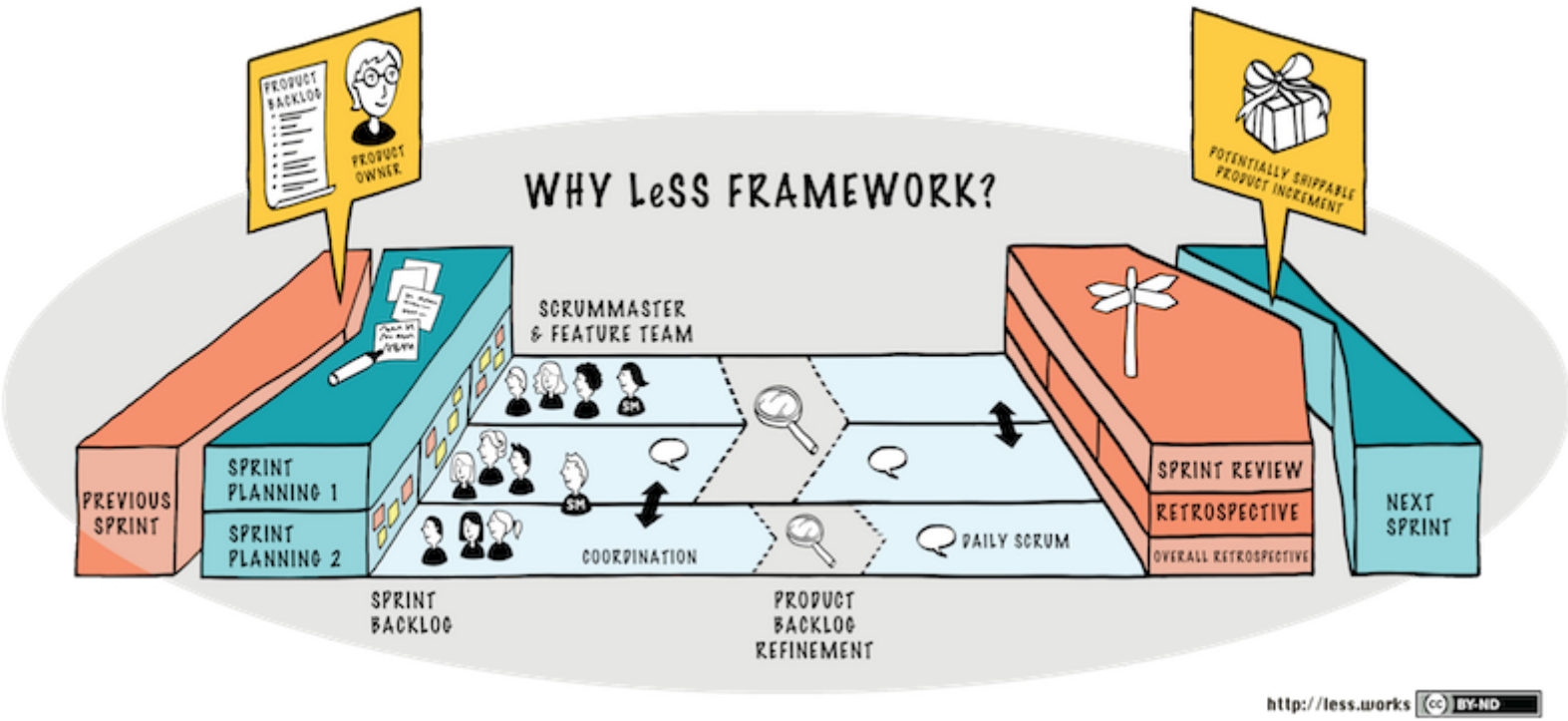
- Up to eight teams
- Up to a few thousand people per product
- One product owner

**Differences from Scrum**

- Single product backlog (as if it were a single team)
- Same deliverable of a shippable product at the end of each sprint (as if it were a single team)
- Cross functional teams

43

# The LeSS Cycle




Source: https://less.works/less/framework/index.html

# What does the LeSS process look like?

All the ceremonies remain the same but with the coordination of multiple teams

1. Sprint planning - instead of including a single team, includes all teams for the product
2. Daily standup - where a team member from from a team observes another team's daily standup
3. Backlog grooming - multiple teams sit together to improve learning and coordination
4. Retrospective - explore improving the system of teams

Source: https://less.works/less/framework/index.html

# SAFe

Abbreviation: Scaled Agile Framework

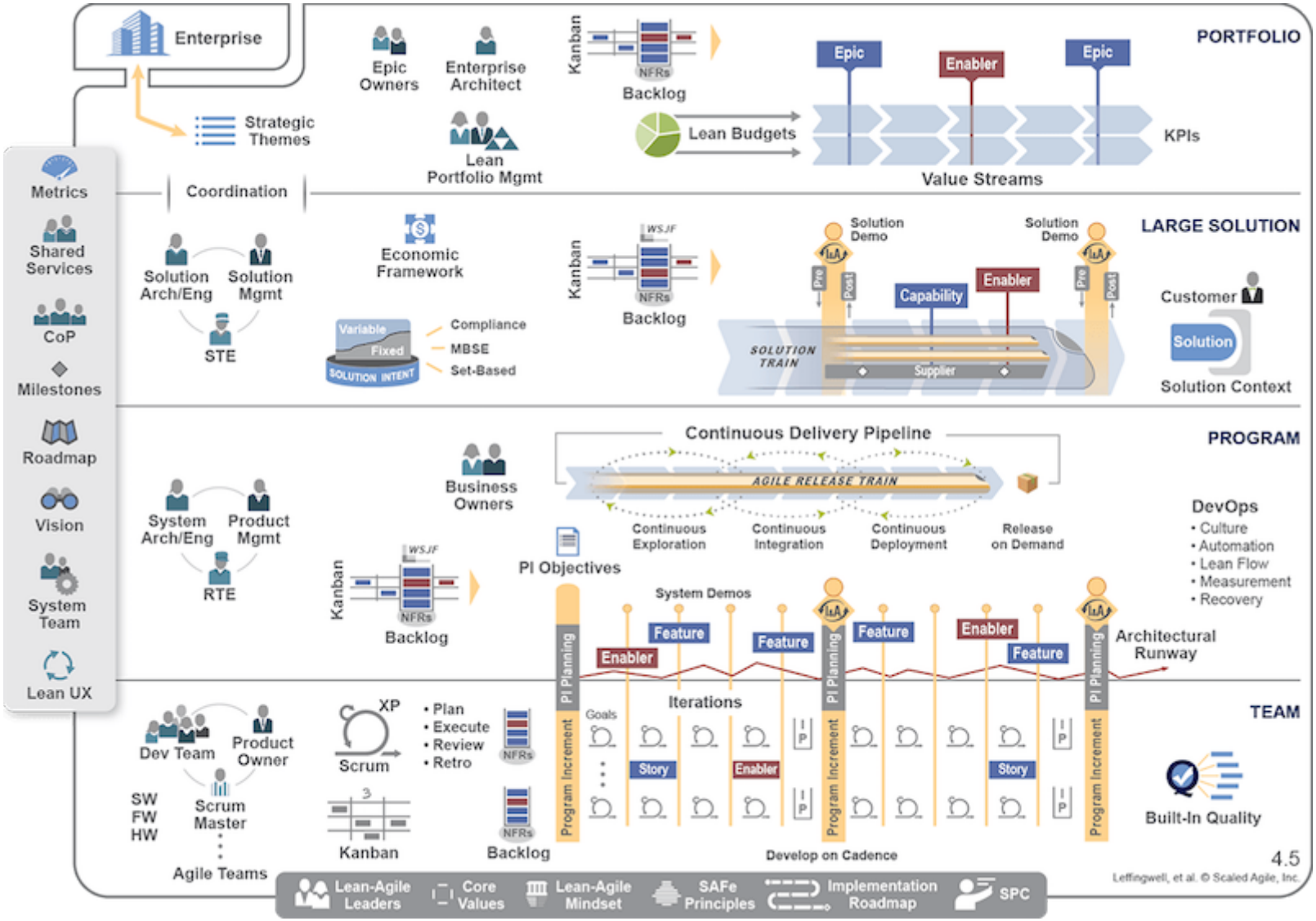
**Team**

- There is a program manager that manages the different product teams and facilitates efforts for delivery
- Multiple scrum teams, each with its own product owner
- Product increment planning meetings are used to facilitate between different teams

**Differences from Scrum**

- A larger hierarchical scrum team that manages local scrum teams to deliver a product
- All the individual underlying teams are part of an "agile release train" to release a product
- Every couple of sprints, teams from the agile release train gets together for a product increment planning meeting
- Product increment planning meets are to align teams on release dates to have a product ready for release of a "product increment"

# The SAFe Cycle




Source: https://www.scaledagileframework.com/

# What does the SAFe process look like?

1. Sprint planning - instead of including a single team, includes all teams for the product
2. Daily standup - where a team member from from a team observes another team's daily standup
3. Backlog grooming - multiple teams sit together to improve learning and coordination
4. Team retrospective - explore improving the system of teams
5. Product increment planning - a meeting that facilitates the releases of different teams to release a product increment
6. Product increment retrospective - exploring the continuation of good practices that led to positive outcomes and stopping bad practices that led to negative outcomes

# SAFe vs LeSS

- SAFe has multiple product owners vs LeSS only has one
- Coordination efforts in LeSS is increased due to teams having to have to gather their own requirements
- SAFe requires less modifications to existing processes and structure than LeSS

# Summary

- Software development processes are very similar
- Favor iterative rather than increment to be able to deliver a usable product and a better product each cycle
- Agile is the mindset of shortening feedback cycles to uncover uncertainty and increase adaptability and flexibility
- Lean is the mindset of eliminating waste and maximizing value
- Scrum relies on timeboxed cycles that enable effective measurements to uncover uncertainty
- Kanban relies on flow to eliminate waste
- LeSS focuses on organizing teams together into a large scrum to release a product
- SAFe focuses on additional process and ceremonies to help facilitate efforts to release a product

# Thank you

William Chan
Lead Platform Engineer, 605.tv, Capital One, FreeWheel (Comcast)
http://linkedin.com/in/wchan2 (http://linkedin.com/in/wchan2)