ENGR 301

Engineering Project Management

Lean and Kanban

Agile Software Development

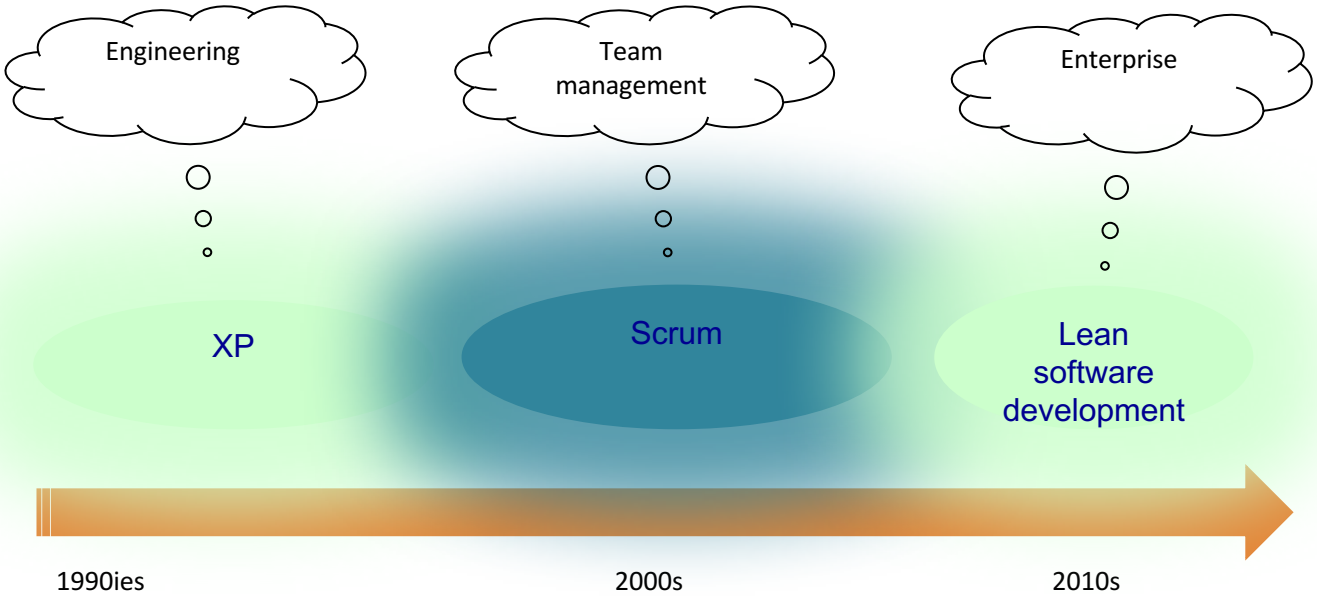
Craig Anslow, Aaron Chen,
James Quilty, James Noble

1897 VICTORIA UNIVERSITY OF WELLINGTON
TE HERENGA WAKA

Overview

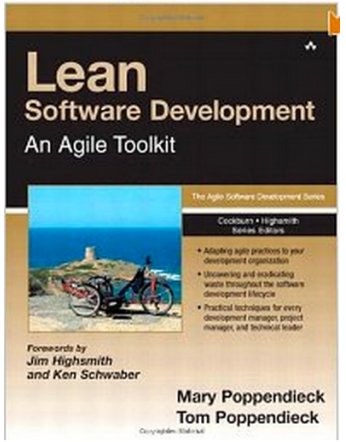
- **Lean History**
- Lean Principles
- Lean Software Development
  - Principles
  - Waste
  - People
  - Quality

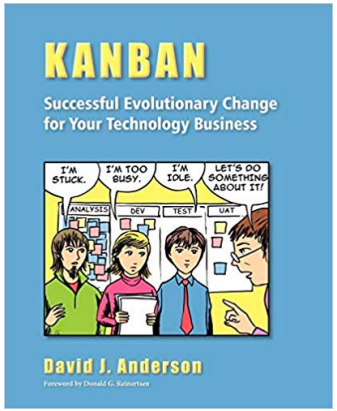
A Brief History of Agile Methods



References

- Mary & Tom Poppendieck: Lean Software Development: An Agile Toolkit, Addison-Wesley, 2003
- Mary & Tom Poppendieck: Implementing Lean Software Development, Addison-Wesley, 2007
- Kanban: Successful Evolutionary Change for Your Technology Business, Blue Hole Press, 2010





A Short History of “Lean”

- 18th century: Interchangeable parts
- Early 20th century: Interchangeable people
  - Ford Model T Assembly Line – 85% reduction of labor
- Mid 20th century:
  - Toyota Motor Corporation
  - Toyota Production System

Ohno, Taiichi (1988), *Toyota Production System: Beyond Large-Scale Production*, Productivity Press, ISBN 0-915299-14-3

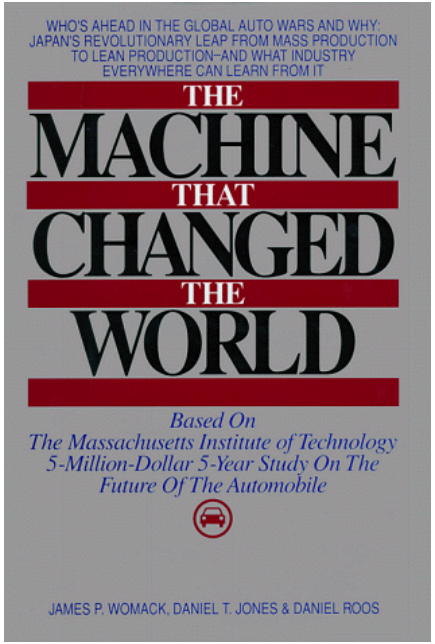
Henry Ford
Kiichiro Toyoda
Taiichi Ohno
Eli Whitney

Toyota Production System

- **Just-in-time flow** – managing complexity
  - Small batches
  - Requires quick changeover or equipment to produce different parts
- **Automation (Jidoka)** – stop the line
  - Detect abnormalities quickly
  - Stop work
  - Resolve problem – root cause analysis
  - Resume work
- **Zero inspection** – production must be designed to be mistake-proof

Lean – Moving to North America

- US car manufacturing loosing ground against Japanese competition
- Womack, Jones, Roos: The Machine That Changed The World
- "The truly lean plant … *transfers the maximum number of tasks and responsibilities to those workers actually adding value to the car on the line, and it has in place a system for detecting defects that quickly traces every problem, once discovered, to its ultimate source.*"



Lean Family History

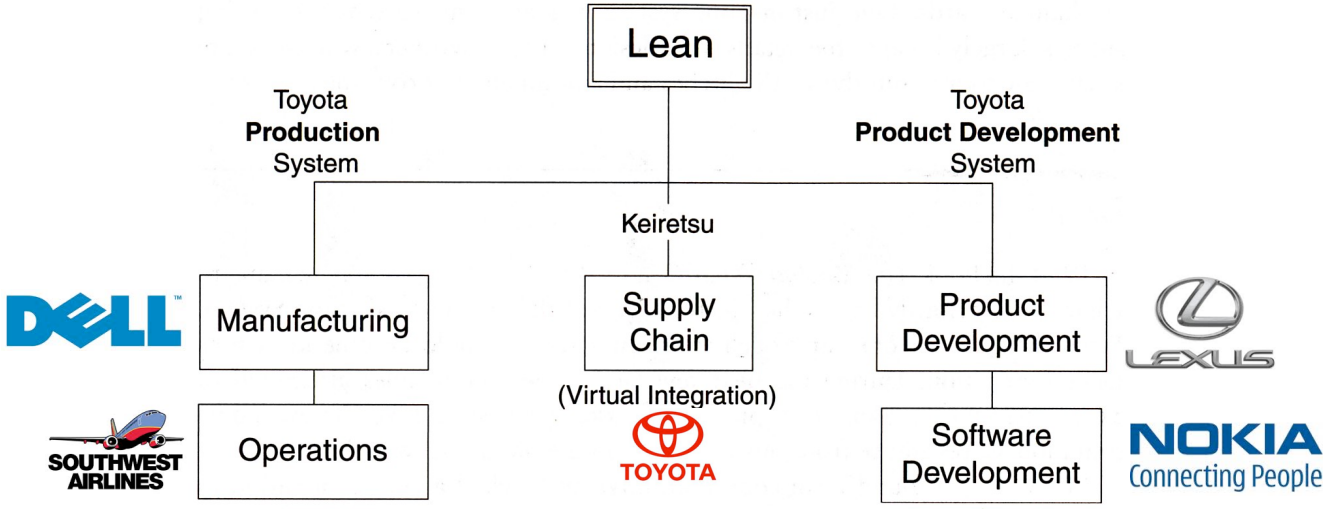
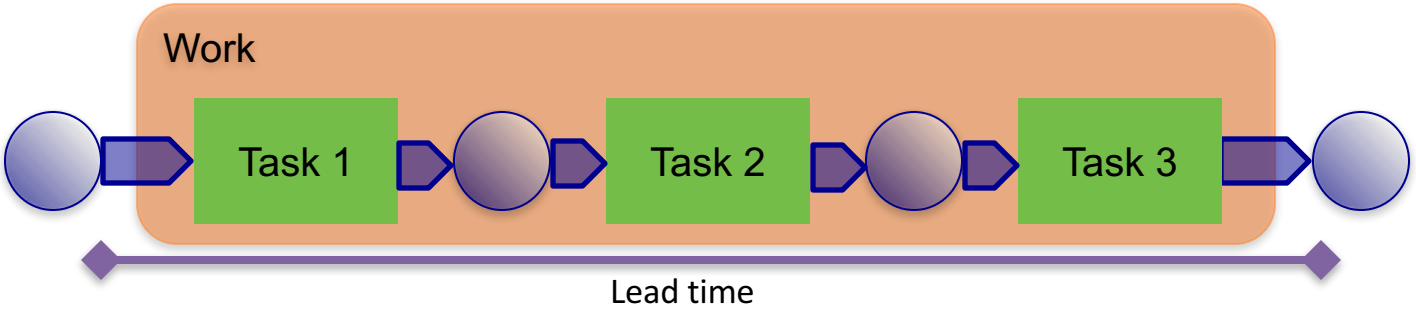


Figure 1.4 *The lean family tree*

Overview

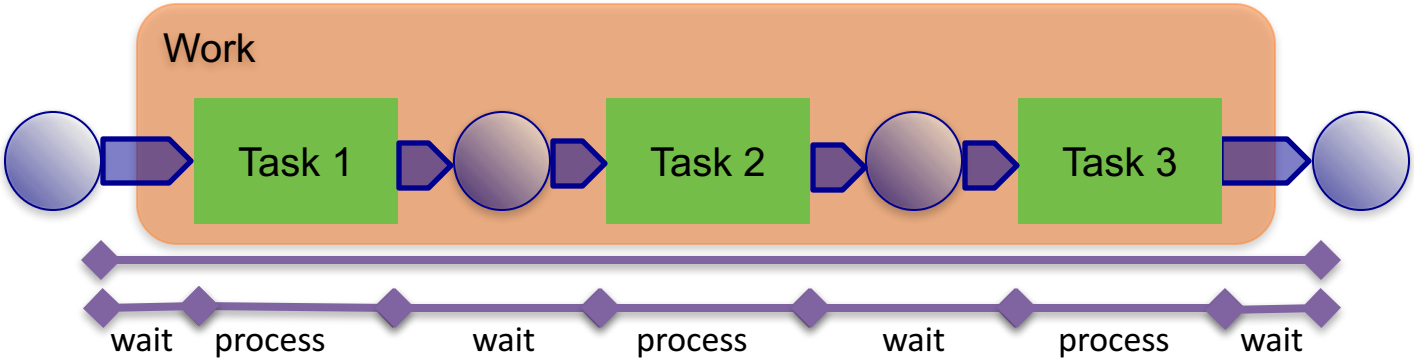
- Lean History
- **Lean Principles**
- Lean Software Development
  - Principles
  - Waste
  - People
  - Quality

Lean Principles: Basics



- Work converts inputs into outputs
- Work is a process
- A process consists of (sub)tasks and hand-offs
- **Lead time**: time between receiving a requirement and delivering on it to the customer

Lead time



- **Process lead time** = task times + wait times
- Process time comes from value adding activities
- **Value adding** = something the customer is willing to pay for

Value Adding Times

Goal:  Task Times = Lead Time

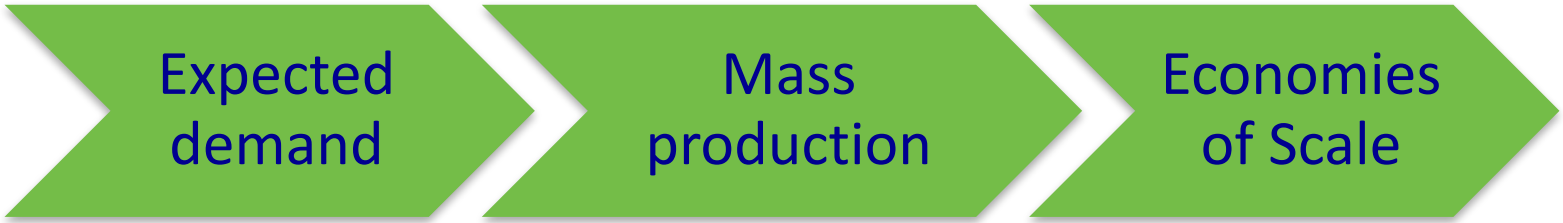
- However: every process has wait times

$$\%\, Value\ adding\ time\ = \frac{\sum Task\ Times}{Process\ Lead\ Time} * 100$$

- Goal reached when %VAT = 100 %

Push versus Pull Systems

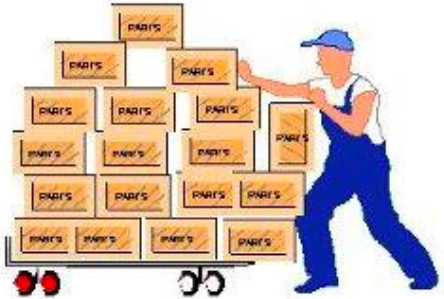
Push – up stream information



Pull – down stream information

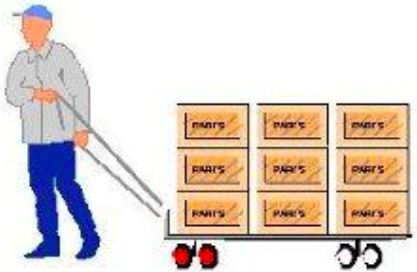
Push vs. Pull

Make all we can just in case.



- Production Approximation
- Anticipated Usage's
- Large Lots
- High Inventories
- Waste
- Management by Firefighting
- Poor Communication

Make what's needed when we need it



- Production Precision
- Actual Consumption
- Small Lots
- Low Inventories
- Waste Reduction
- Management by Sight
- Better Communication

https://www.industryweek.com/cloud-computing/article/22023873/push-vs-pull-manufacturing-is-a-kanban-pull-system-right-for-your-company

Number Multitasking Game - Setup

- Write the roman numerals **I** through **X** in a column from top to bottom. Use a **black** pen for this task.
- Write the letters **A** through **J** in another column from top to bottom. Use a **red** pen for this task.
- Write the numbers **1** through **10** in a final column from top to bottom. Use a **blue** pen for this task.

Number Multitasking Game – **Round 1**

- In the first iteration (**Round 1**) utilize the “resources” (the players) to the fullest and therefore want them to spend equal amounts of time on each project, because they’re all important. Write **row by row**:

Step	Roman (I-X)	Letters (A-J)	Numbers (1-10)
1	I	A	1
2	II	B	2
3	…	…	…

Number Multitasking Game – **Round 2**

- For (**Round 2**) focus on what's most important first and finish it before continuing on to the next task. In other words, work **column by column**:

Step	Roman	Letters	Numbers
1	I		
	II		
	…		

Step	Roman	Letters	Numbers
	I	A	
2	II	B	
	III	C	

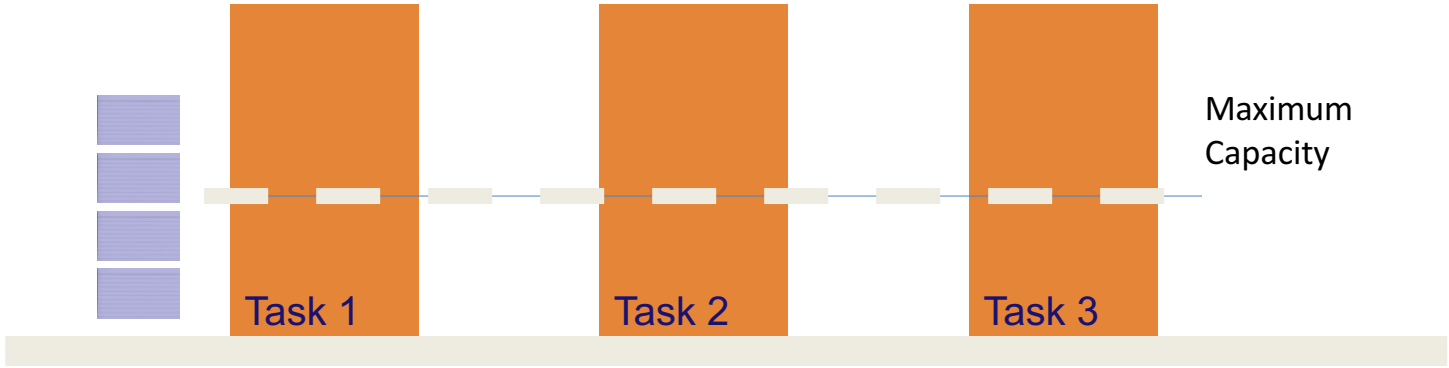
…

Number Multitasking Game – **Reflection**

- What happened to the total time? Why?
- What happened to the times for each individual project? Why?
- Was the first round harder? Why?
- Does the simulation resemble your work situation?
  - What are your individual projects?
  - What does “switching pens” represent in your context?
- How are projects prioritized at your company? Do you know what’s most important to work on right now?
- How many projects/different things are you working on right now?
- What happened to the quality of the produced result?
- How did the first approach feel? Did the second approach feel better?

https://livebook.manning.com/book/kanban-in-action/chapter-13/106

Kanban

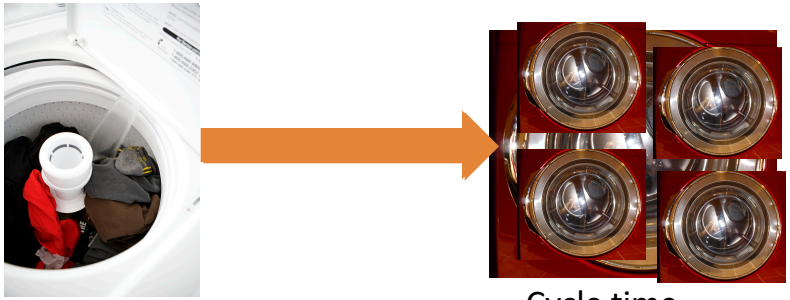


- **Kan** = visual and **Ban** = card
- Kanban = signal card → signal demand
- Capacity limits work in progress and encourages self-directed reallocation of resources
- Buffer slot empty → fill it

Cycle Time

- Cycle Time is the *average* length of time to complete a task or set of tasks.

Cycle time for "washing"?

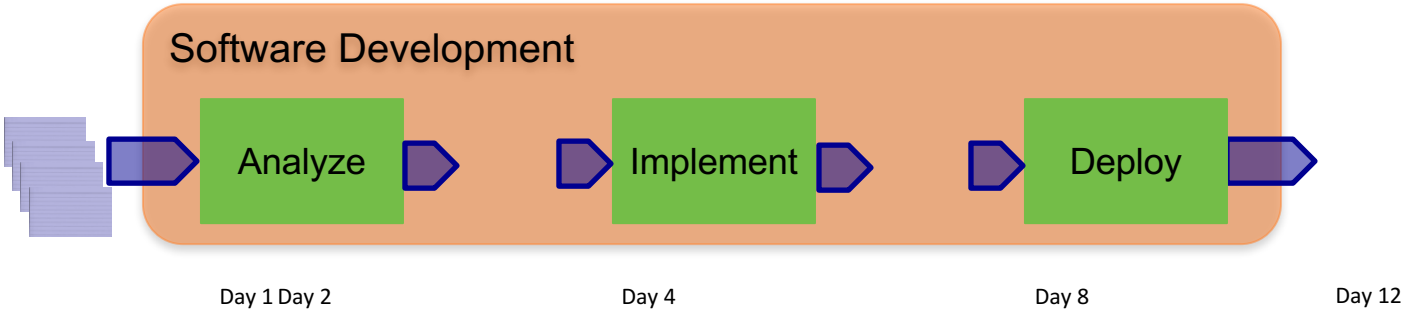


Cycle time
30 min

Cycle time
60 min

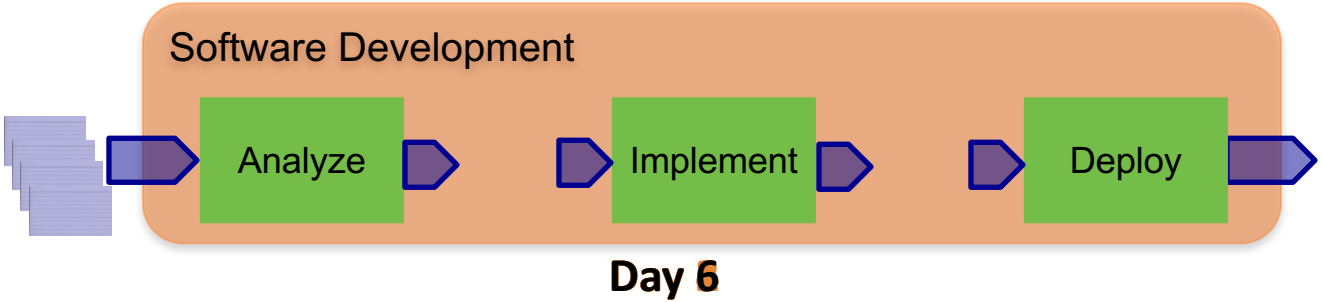
Overall cycle time for "doing laundry"?

Flow – The Impact of Batch Size on Cycle Time (1)



- 1 day per story, batch: 4 story cards
- How long for all stories to be delivered?
- Work: 4x1day + 4x1day + 4x1day = 12 days

Flow – The Impact of Batch Size on Cycle Time (2)



- 1 day per story, batch:1 story card
- How long for all stories to be delivered?
- Work: 4x1day + 4x1day + 4x1day = 12 days
- Delivery: 6 days

Numbers Multitasking Game

Learning Points:

- By taking a smaller set of requirements all the way to completion, you get something to the customer faster.
- Conversely, if all the requirements are processed at the same time, changes later in the cycle become more costly.
- **Single piece flow** (round 2) is often faster than **batch and queue** (round 1). Due to the fact that each cross-functional participant can take ownership of a module all the way to completion, reducing overall task-switching and hand-offs.

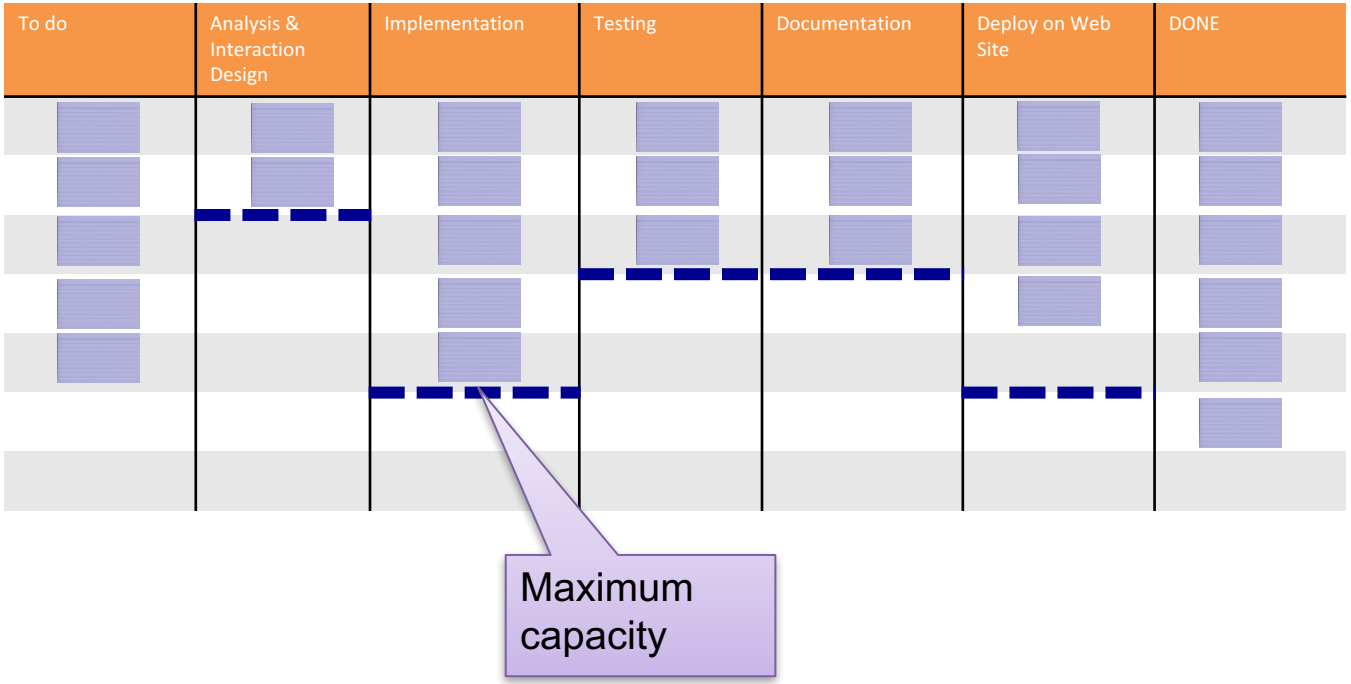
Other Games

- **Name Game:**
  http://www.leansimulations.org/2013/01/the-name-game-aka-hldittwan.html
- **Penny Game:**
  http://www.leansimulations.org/2010/11/penny-game.html
- **Paper Airplanes:**
  http://www.leansimulations.org/2012/09/more-lean-paper-airplanes-another-lean.html
- **Tasty Cupcakes (series of games):**
  http://tastycupcakes.org/

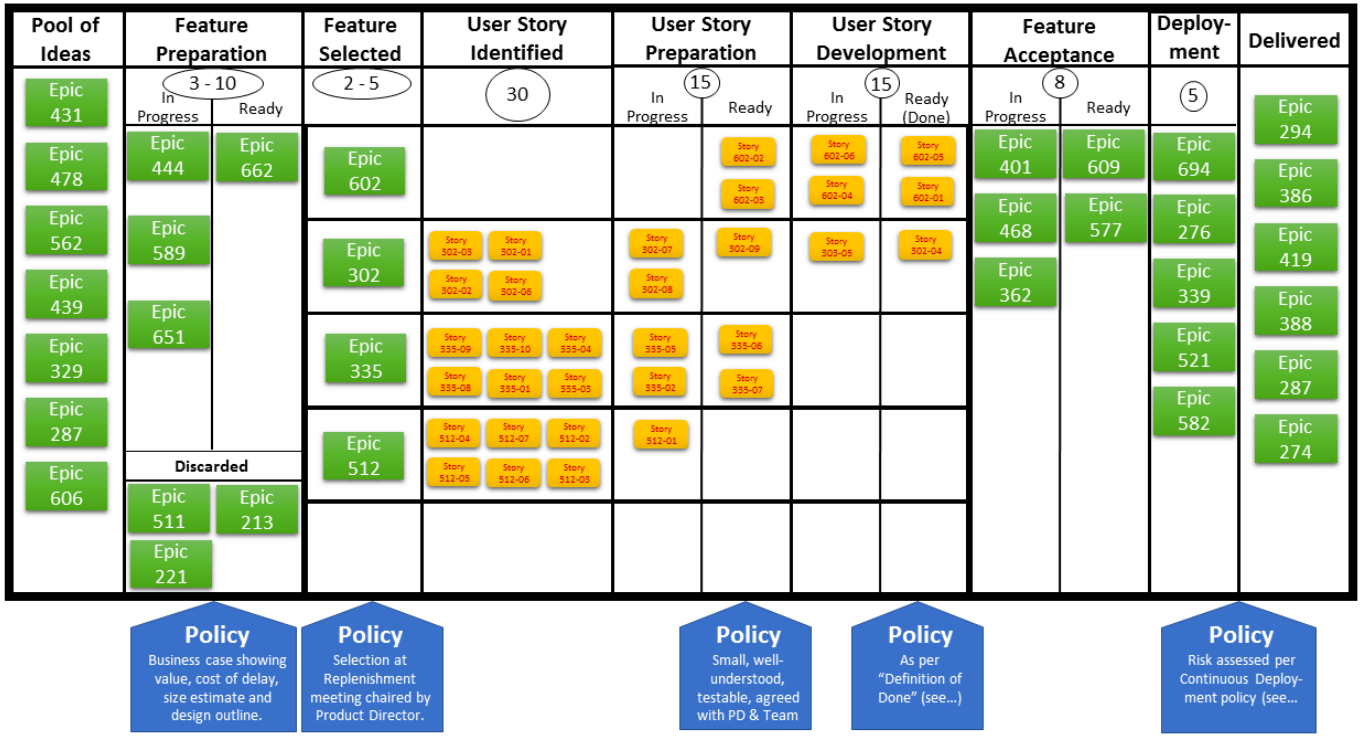
Lean Software Development - Flow

- The goal is a continuous and leveled flow from input to outputs
- Sustainable pace

Kanban Boards



Kanban Boards



https://en.wikipedia.org/wiki/Kanban_(development)

Kanban versus Scrum

- Kanban – steady flow
- Scrum – “hard” iterations
- Introduction to Kanban in Under 5 Mins: https://www.youtube.com/watch?v=R8dYLbJiTUE
- Kanban Applied to Scrum: https://www.youtube.com/watch?v=0EIMxyFw9T8
- Scrum vs Kanban https://www.youtube.com/watch?v=rIaz-l1Kf8w

Overview

- Lean History
- Lean Principles
- **Lean Software Development**
  - Principles
  - Waste
  - People
  - Quality

Lean Software Development

- Software is expected to change
  (no change → put it into hardware)
- Deterministic school:
  - Create complete product definition
  - Build it
- Empirical school:
  - Establish product vision
  - Use feedback loops to develop product that realizes the vision

Lean Product Development

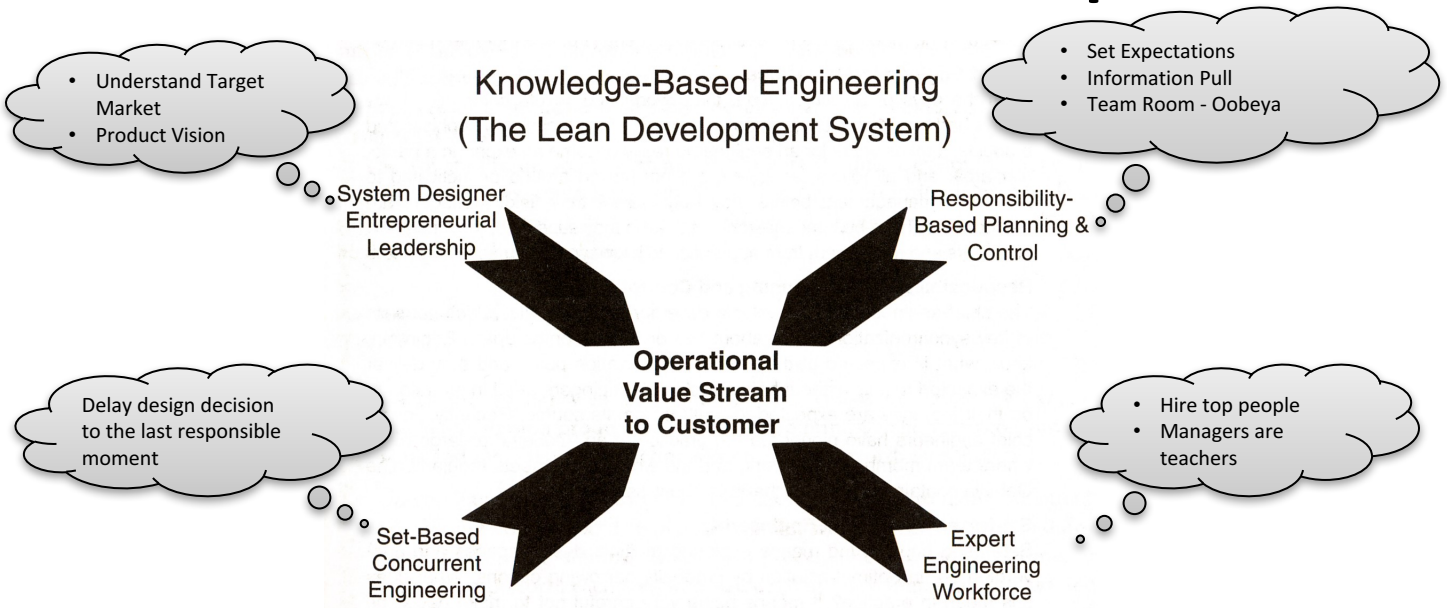
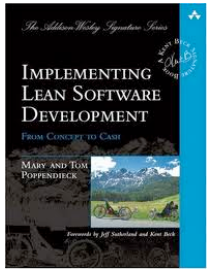


Figure 1.5 *Cornerstone elements of the Toyota Product Development System*[18]

Lean Software Development - Principles

1. Eliminate Waste (myth: early specification reduces waste)
2. Build Quality In (myth: the job of testing is to find defects)
3. Create Knowledge (myth: predictions create predictability)
4. Defer Commitment (myth: planning is commitment)
5. Deliver Fast (myth haste makes waste)
6. Respect People (myth: there is one best way)
7. Optimize The Whole (myth: optimize by decomposition)



Overview

- Lean History
- Lean Principles
- Lean Software Development
  - Principles
  - **Waste**
  - People
  - Quality

The Seven Wastes

Manufacturing	Software Development
In-Process Inventory	Partially done work (e.g. tasks not finished at the end of an iteration)
Over production	Extra features (e.g. gold plating a system)
Extra processing	Relearning (e.g. frequent project switches)
Transportation	Handoffs (e.g. strong role specialization)
Motion	Task switching (e.g. inability to complete tasks after work on it has started)
Waiting	Delays
Defects	Defects

Finding Waste – Value Stream Mapping

- "All we are doing is looking at the timelines from the moment a customer gives us an order to the point when we collect the cash. And we are reducing that timeline by removing the non value-add wastes." Taiichi Ohno
- Map the current process of an "average" project
- Choose when to start and stop the timeline = what is in/what is out

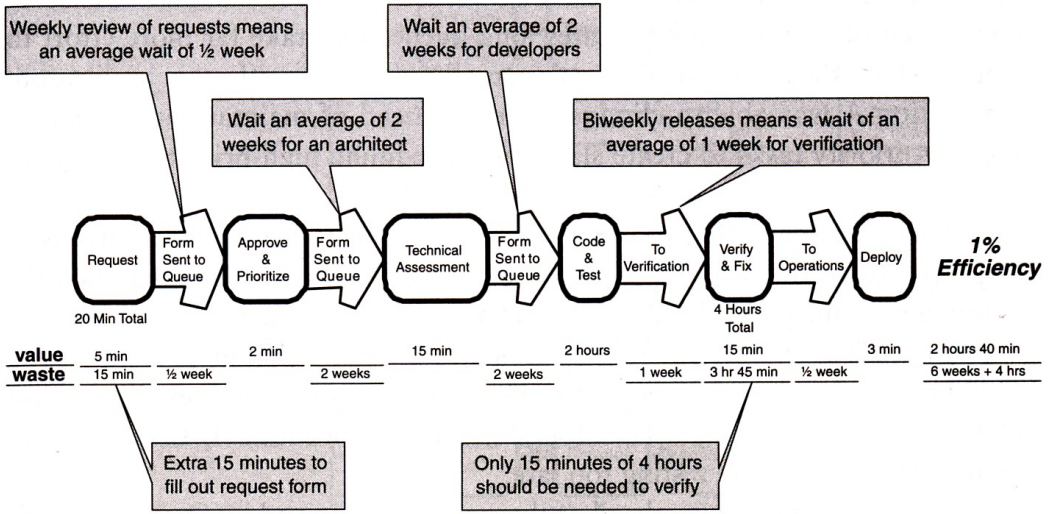


Figure 4.4 *Value stream map of a small, high-priority feature change request—Organization B*

Task cycle time

Improving a Lean Process

- Remove tasks that do not add value
- Reduce lead time
  - Reduce wait times
  - Reduce task times
- Reduce cycle time (improve process flow)

Reduce Cycle Time – Queuing Theory

- Even out the arrival of work – backlog
- Minimize #things in process – deliver fast
- Minimize batch size – small iterations
- Establish regular iterations
- Limit work to capacity – say no if limits are exceeded
- Use pull scheduling

How to Address Slow Steps?

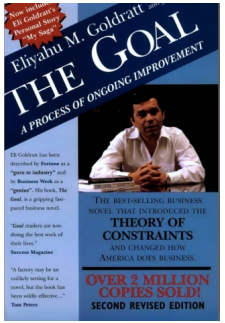
- Reduce activity time for steps on critical path
- Move resources to help out → relax constraint
- Requires skill flexibility

Balance Cycle Times of Process Steps

- 5 days implement + 1 day per story for other steps,
  batch: 1 story card
- How long for all stories to be delivered?
- Design: 1, 2, 3, 4
- Implement: 6, 11, 16, 21
- Test: 7, 12, 17, 22
- Deploy: 8, 13, 18, 23

Theory of Constraints

- “The title Theory of Constraints (TOC) adopts the common idiom "A chain is no stronger than its weakest link" as a new management paradigm. This means that processes, organizations, etc., are vulnerable because the weakest person or part can always damage or break them or at least reduce the outcome.” Wikipedia
- The underlying premise of Theory of Constraints is that organizations can be measured and controlled by variations on three measures:
  - **Throughput** - is the rate at which the system generates money through sales.
  - **Inventory** - is all the money that the system has invested in purchasing things which it intends to sell.
  - **Operational Expense -** is all the money the system spends in order to turn inventory into throughput”



Five Focusing Steps

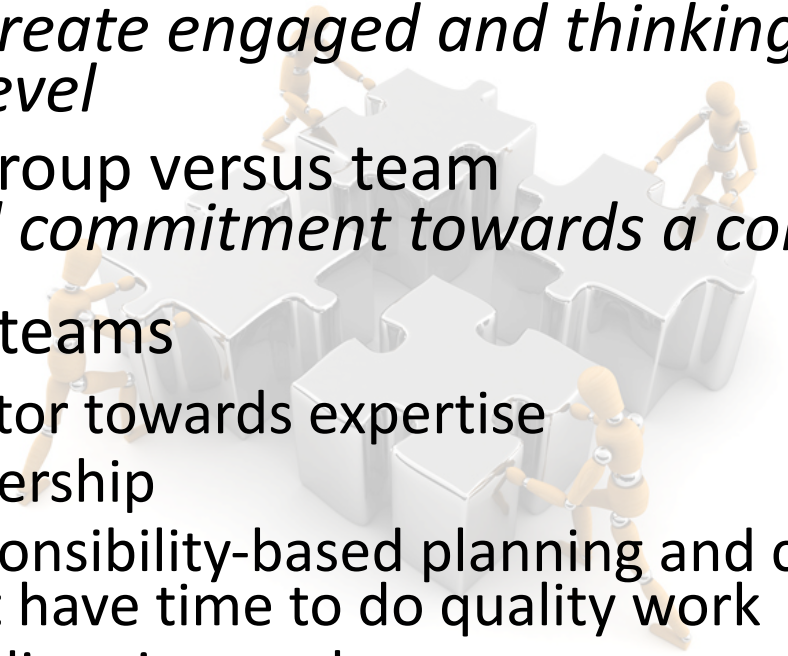
- After articulating the goal of the organization:
  1. Identify the constraint
  2. Decide how to exploit the constraint
  3. Subordinate all other processes to above decision
  4. Elevate the constraint
  5. If, as a result of these steps, the constraint has moved, return to Step 1. Don't let inertia become the constraint.

Overview

- Lean History
- Lean Principles Lean Software Development
  - Principles
  - Waste
  - **People**
  - Quality

Respect People

- Goal: *Create engaged and thinking people at every level*
- Work group versus team
  Mutual commitment towards a common goal
- Create teams
  - Mentor towards expertise
  - Leadership
  - Responsibility-based planning and control: people must have time to do quality work
  - Self-directing work



"Year after year, Toyota has been able to get more out of its people than its competitors have been able to get out of theirs"
Gary Hamel

Incentives

- Commitment is a two way street: companies and employees

- Performance evaluations considered harmful
  - Individual rewards create competition not cooperation
  - Promotions must be transparent and perceived to be fair
  - Reward based on span of influence

Overview

- Lean History
- Lean Principles Lean Software Development
  - Principles
  - Waste
  - People
  - **Quality**

Quality

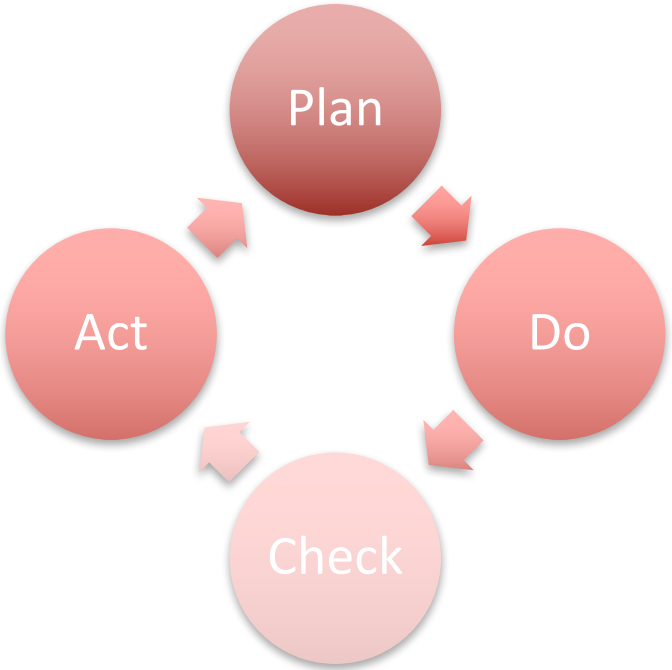
- The five S's – organize the workspace
  - Sort (Seiri)
  - Systematize (Seiton)
  - Shine (Seiso)
  - Standardize (Seiketsu)
  - Sustain (Shitsuke)
- Mistake-Proofing – design system so that things *cannot* go wrong

HDMI IDE

"You can't go fast without building quality into the product, and that takes a lot of discipline."
Poppendiecks

Kaizen – Continuous Process Improvement

- Scientific method
  1. Define the problem
  2. Analyze the situation
  3. Create a hypothesis
  4. Perform the experiments
  5. Verify results
  6. Follow-up/standardize



Five Whys – Root Cause Analysis

- My car will not start. (the problem)
  1. *Why?* - The battery is dead. (first why)
  2. *Why?* - The alternator is not functioning. (second why)
  3. *Why?* - The alternator belt has broken. (third why)
  4. *Why?* - The alternator belt was well beyond its useful service life and has never been replaced. (fourth why)
  5. *Why?* - I have not been maintaining my car according to the recommended service schedule. (fifth why, a root cause)
- I will start maintaining my car according to the recommended service schedule. (solution)

Continuous Improvement – Deming

1. Create constancy of purpose toward improvement of product and service, with the aim to become competitive and stay in business, and to provide jobs.
2. Adopt the new philosophy. We are in a new economic age. Western management must awaken to the challenge, must learn their responsibilities, and take on leadership for change.
3. Cease dependence on inspection to achieve quality. Eliminate the need for massive inspection by building quality into the product in the first place.
4. End the practice of awarding business on the basis of price tag. Instead, minimize total cost. Move towards a single supplier for any one item, on a long-term relationship of loyalty and trust.
5. Improve constantly and forever the system of production and service, to improve quality and productivity, and thus constantly decrease costs.
6. Institute training on the job.
7. Institute leadership (see Point 12 and Ch. 8 of "Out of the Crisis"). The aim of supervision should be to help people and machines and gadgets to do a better job. Supervision of management is in need of overhaul, as well as supervision of production workers.
8. Drive out fear, so that everyone may work effectively for the company. (See Ch. 3 of "Out of the Crisis")
9. Break down barriers between departments. People in research, design, sales, and production must work as a team, to foresee problems of production and in use that may be encountered with the product or service.
10. Eliminate slogans, exhortations, and targets for the work force asking for zero defects and new levels of productivity. Such exhortations only create adversarial relationships, as the bulk of the causes of low quality and low productivity belong to the system and thus lie beyond the power of the work force.
11. a. Eliminate work standards (quotas) on the factory floor. Substitute leadership.
    b. Eliminate management by objective. Eliminate management by numbers, numerical goals. Substitute leadership.
12. a. Remove barriers that rob the hourly worker of his right to pride of workmanship. The responsibility of supervisors must be changed from sheer numbers to quality.
    b. Remove barriers that rob people in management and in engineering of their right to pride of workmanship. This means, *inter alia," abolishment of the annual or merit rating and of management by objective (See Ch. 3 of "Out of the Crisis").*
13. Institute a vigorous program of education and self-improvement.
14. Put everybody in the company to work to accomplish the transformation. The transformation is everybody's job.

Seven Deadly Diseases + Obstacles

Diseases

1. Lack of constancy of purpose
2. Emphasis on short-term profits
3. Evaluation by performance, merit rating, or annual review of performance
4. Mobility of management
5. Running a company on visible figures alone
6. Excessive medical costs
7. Excessive costs of warranty, fueled by lawyers who work for contingency fees

Obstacles

1. Neglecting long-range planning
2. Relying on technology to solve problems
3. Seeking examples to follow rather than developing solutions
4. Excuses, such as "our problems are different"
5. Obsolescence in school that management skill can be taught in classes[25]
6. Reliance on quality control departments rather than management, supervisors, managers of purchasing, and production workers
7. Placing blame on workforces who are only responsible for 15% of mistakes where the system desired by management is responsible for 85% of the unintended consequences
8. Relying on quality inspection rather than improving product quality

Lean and Agile Differences

- The Difference Between Lean and Agile https://www.youtube.com/watch?v=aUd3xTdtXqI

Summary

- **Lean software development** is based on ideas from lean manufacturing
- Cycle time, flow, batch size
- Kanban
- 7 wastes
- Respect people
- Plan-do-check-act
- Kaizen