lean

software development

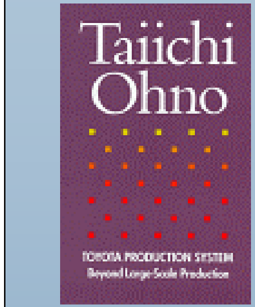
# World Class Software-Enabled Products

*Case Studies in Lean Thinking*

mary@poppendieck.com Mary Poppendieck www.poppendieck.com

# *The Toyota Production System*

## Taiichi Ohno

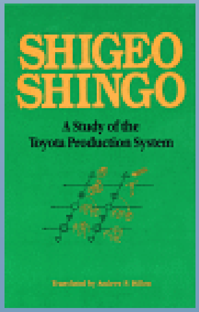


***The Toyota Production System,*** 1988 (1978)

- ✓ Eliminate Waste
  - × Just-in-Time Flow
- ✓ Expose Problems
  - × Stop-the-Line Culture

Taiichi Ohno
(1912-1990)

## Shigeo Shingo



***Study Of 'Toyota' Production System,*** 1981

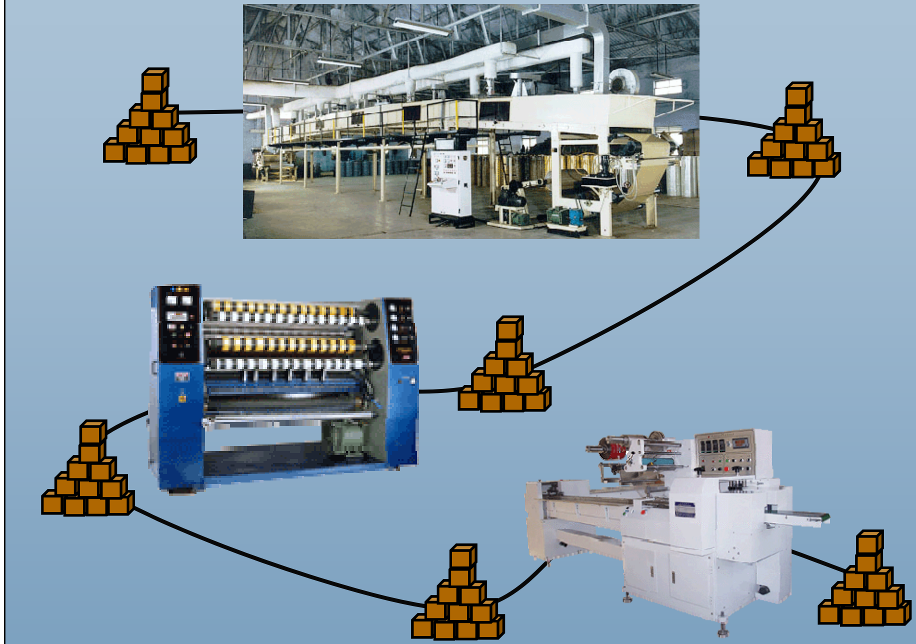
- ✓ Non-Stock Production
  - × Single Digit Setup
- ✓ Zero Inspection
  - × Mistake-Proof Every Step

Shigeo Shingo
(1909 – 1990)

l e a n

# *Pillars of Lean*

## Just-in-Time Flow



Stop trying to maximize local productivity – maximize ***FLOW***.

## Stop-the-Line Culture

Detect problems the moment the occur – ***STOP*** – find the root cause – fix it immediately.

l e a n

Copyright©2007 Poppendieck.LLC

# *Think Products, not Projects*

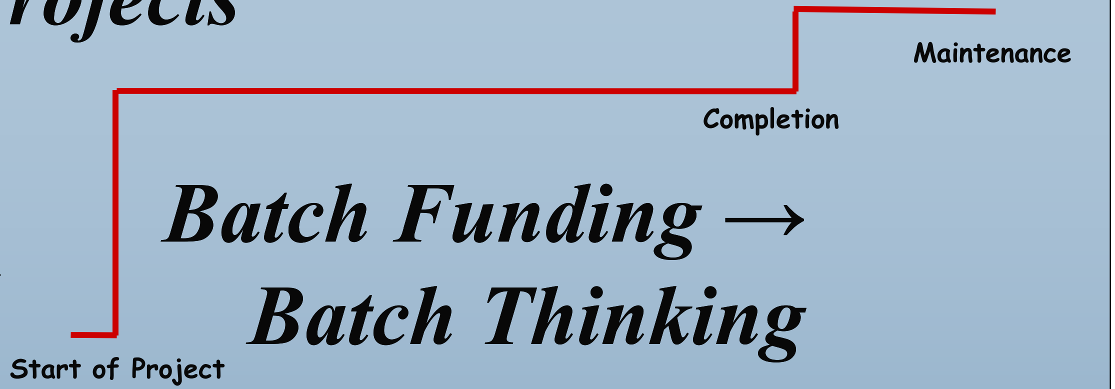
## ***Projects***

Up-front funding

Scope fixed at onset

Success = cost/schedule/scope

Team often disbands at completion



***Batch Funding →
Batch Thinking***

## ***Products***

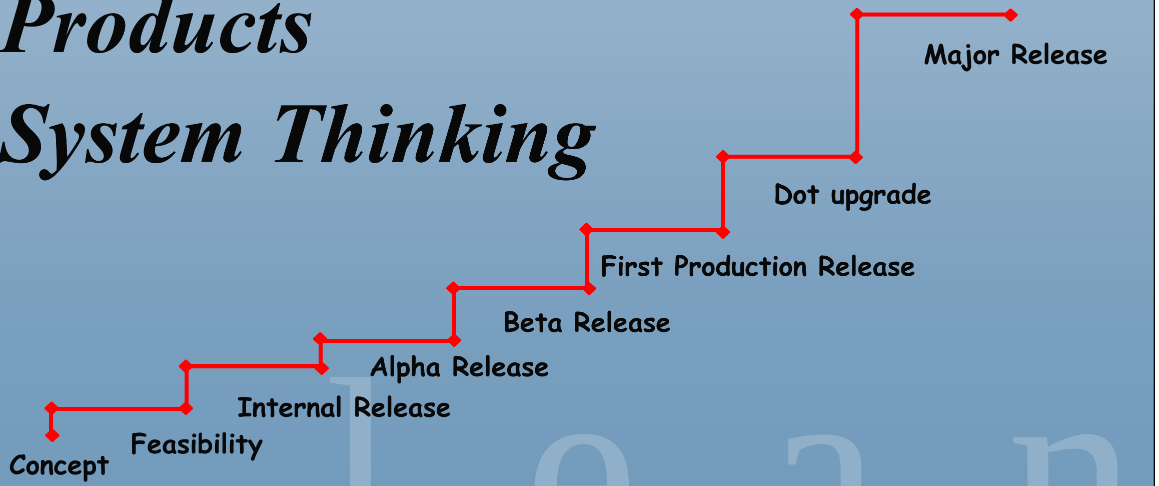
## ***System Thinking***

Incremental funding

Scope expected to evolve

Success = profit/market share

Team often stays with product



Copyright©2007 Poppendieck.LLC

l e a n

# Think Whole Product
# Not Just Software

Software is rather useless

– all by itself

Software is embedded

In hardware

In a process

In an activity

software
inside

*The product [or process or activity] should be designed and developed as a* ***system*** *– by a* ***complete*** *team.*

The Overall Product/Process

- ✓ Is usually developed at the same time as its software
- ✓ Evolves as the development process generates learning
- ✓ Generates changing demands for embedded software
- ✓ In this context:

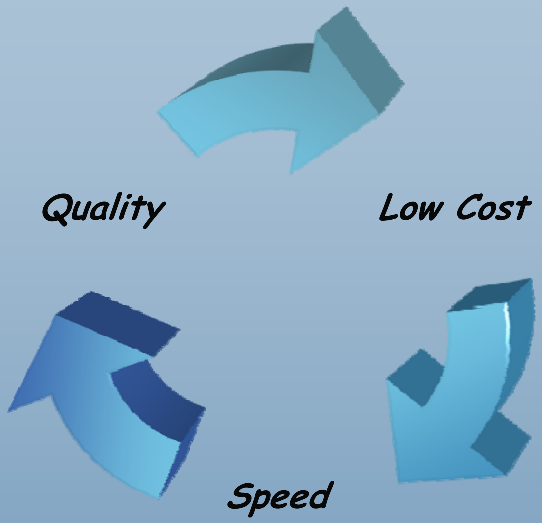
***What are Requirements?***

***Product design decisions that the software team doesn't participate in.***

l e a n

# Principles of Lean Software Development

1. Eliminate Waste
   - ✓ *Focus on the Flow of Value*
2. Focus on Learning
   - ✓ *Pursue Relentless Improvement*
3. Build Quality In
   - ✓ *Mistake-Proof Every Step*
4. Defer Commitment
   - ✓ *Maintain Options*
5. Deliver Fast
   - ✓ *Don’t Batch & Queue*
6. Respect People
   - ✓ *Decide as Low as Possible*
7. Optimize the Whole
   - ✓ *Measure UP*

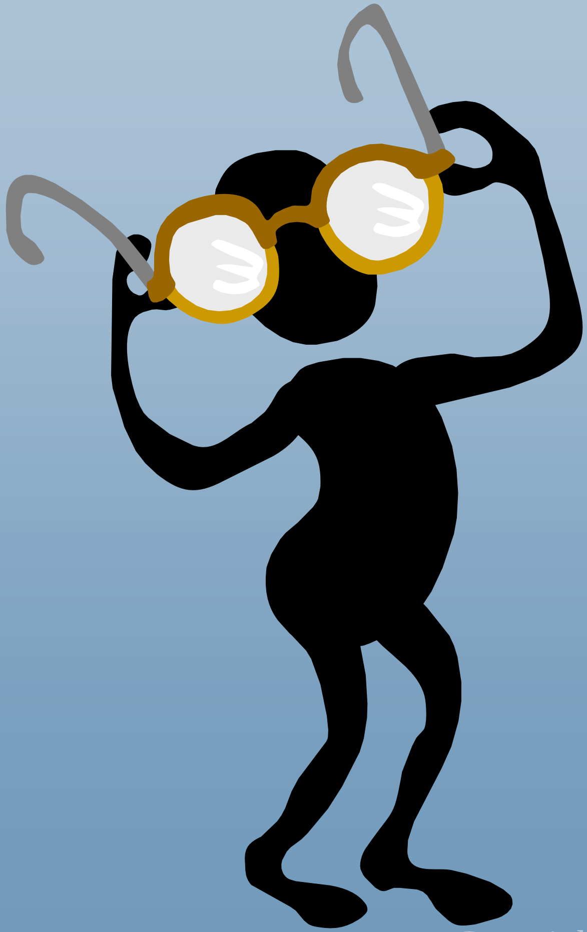


Copyright©2006 Poppendieck.LLC

l e a n

# *Principle 1: Eliminate Waste*

Put on Customer Glasses



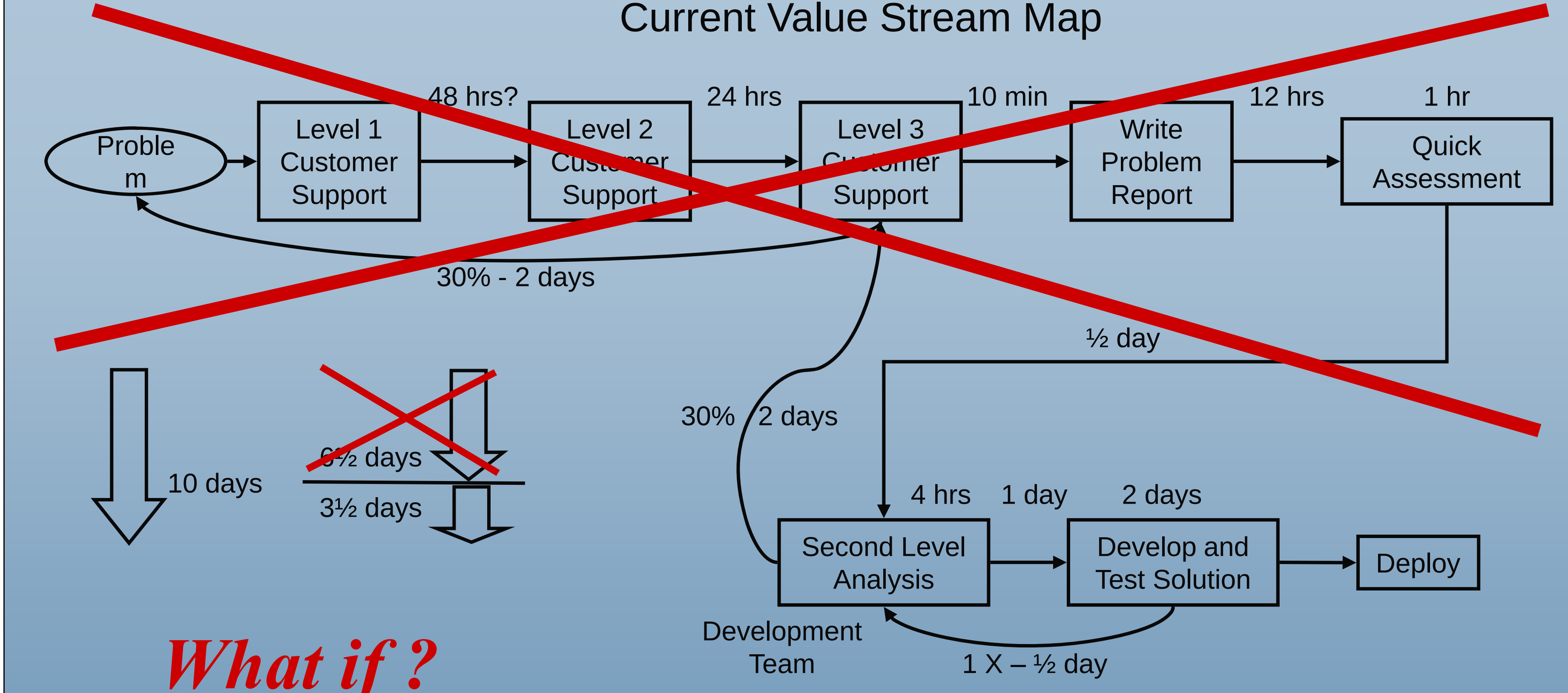
**MUDA**

**anything that does not add**

**VALUE**

# *Case Study: Critical Defects*

## Current Value Stream Map



Problem

Level 1 Customer Support

48 hrs?

Level 2 Customer Support

24 hrs

Level 3 Customer Support

10 min

Write Problem Report

12 hrs

1 hr

Quick Assessment

30% - 2 days

½ day

30% - 2 days

10 days

~~6½ days~~

3½ days

4 hrs

1 day

2 days

Second Level Analysis

Develop and Test Solution

Deploy

Development Team

1 X – ½ day

***What if ?***

l e a n

# *Case Study: Critical Defects*

Future Value Stream Map

Questions:

Who will staff the phones?

- ✓ Developers – in rotation

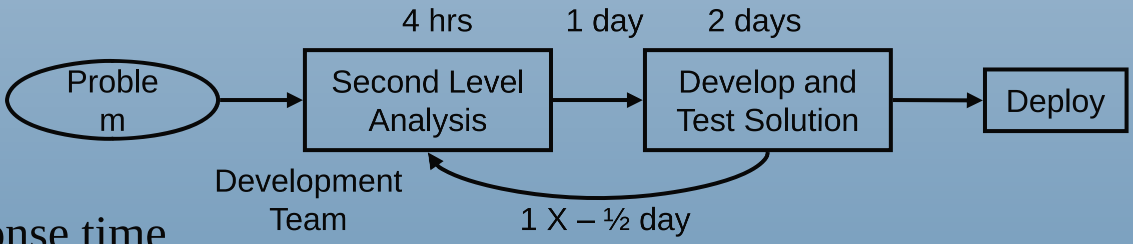
How many will we need?

- ✓ Experiment – find out

Two Rules:

1. Immediately after a release, responsible team takes calls.
2. Learning from each call MUST be recorded in knowledge base which is available to customers.

***Why not?***

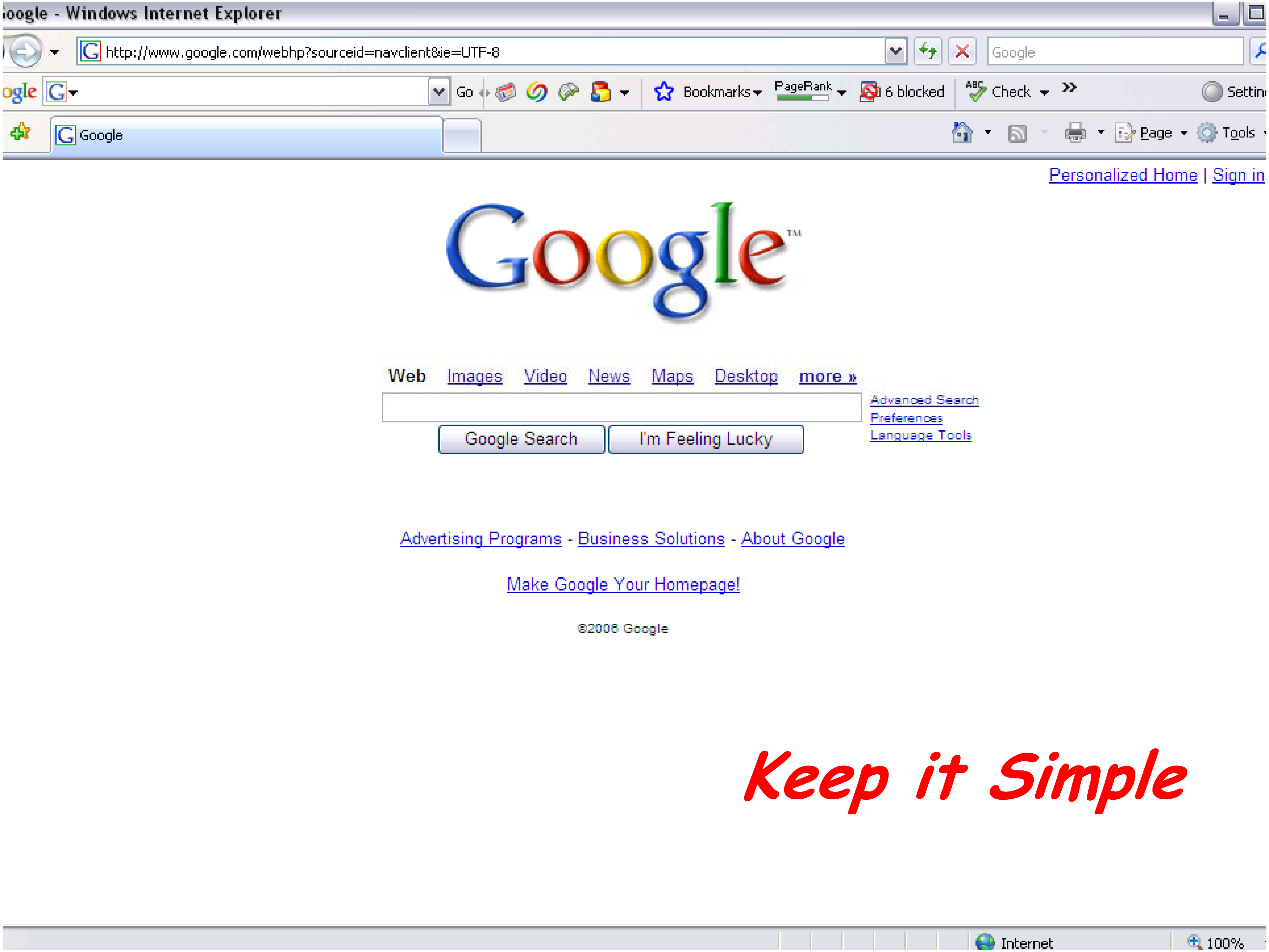


Results:

- ✓ 65% increase in response time
- ✓ 40% increase in available development time (for 800 developers)!
  - **Before**: 60% of development time spent on critical defects
  - **After**: 20% of development time spent on critical defects

l e a n

Copyright©2006 Poppendieck.LLC

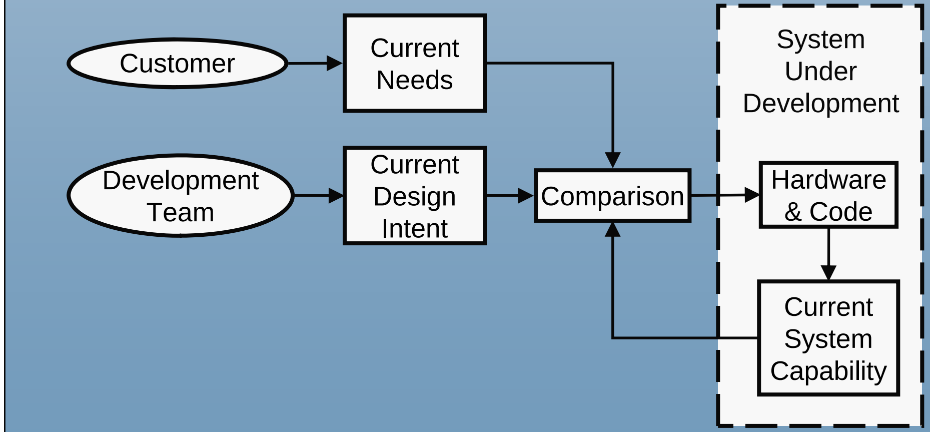


**Keep it Simple**

# Principle 2: Focus on Learning

## Cycles of Discovery

- ✓ Products emerge through iterations of learning.
- ✓ The best way to improve a product development process is to add more feedback.



## Relentless Improvement

- ✓ All products and all work activities are designed and constantly improved by the people doing the work.
- ✓ Managers act as teachers, helping workers use the scientific method to shape and improve both products and processes.

Copyright©2007 Poppendieck.LLC

Google™

## Our Philosophy

## *Never settle for the best.*

**Ten things Google has found to be true**

1. ***Focus on the user and all else will follow.***
2. It's best to do one thing really, really well.
3. ***Fast is better than slow.***
4. Democracy on the web works.
5. You don't need to be at your desk to need an answer.
6. You can make money without doing evil.
7. ***There's always more information out there.***
8. The need for information crosses all borders.
9. You can be serious without a suit.
10. ***Great just isn't good enough.***

http://www.google.com/corporate/tenthings.html
January 5, 2007

l e a n

Copyright©2006 Poppendieck.LLC

# *Principle 3:*
# *Build Quality In*

## ***There are Two Kinds of Inspection***

1. Inspection to Find Defects – WASTE
2. Inspection to Prevent Defects – Essential

## ***The Role of QA***

The job of QA is not to swat misquotes,

The job of QA is to put up screens.

A quality process builds quality into the code

✓ If you routinely find defects during verification

– your process is defective.

*Make it Flawless*

lean

Copyright©2006 Poppendieck.LLC

# Mistake-Proof Every Step

## *Case Study: Mobile Spectrometer to Analyze Grain*

Techniques:

- ✓ Trouble log with different behaviors depending on development or field platform and severity of error.
- ✓ Dual-targeting: Bracket HW-dependent code and run only with target HW, mock-out otherwise.
- ✓ Isolate HW driver code, use scripts to test it with HW
  - × Became the HW acceptance tests
- ✓ Isolate and test domain-level code (eg communications)
- ✓ Special tests for unique domains (eg math algorithms)

Result:

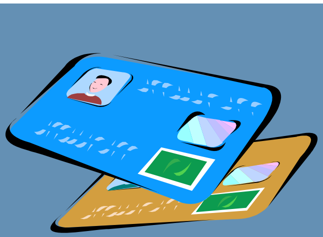
- ✓ In 3 years, only 51 defects (18 critical, 23 moderate, 10 cosmetic), with a maximum of 2 open at once!
- ✓ Productivity 3X similar embedded software teams.
- ✓ HW engineers trusted SW and used it to debug HW.

Taken from: **Taming the Embedded Tiger – Agile Test Techniques for Embedded Software,** Nancy Van Schooenderwoert & Ron Morsicato, ADC 2004 & **Embedded Agile Project by the Numbers with Nubies,** Nancy Van Schooenderwoert, Agile 2006

Copyright©2006 Poppendieck.LLC

lean

# Technical Debt



*Anything that makes code difficult to change*
*(The usual excuse for batches & queues)*

✓Complexity

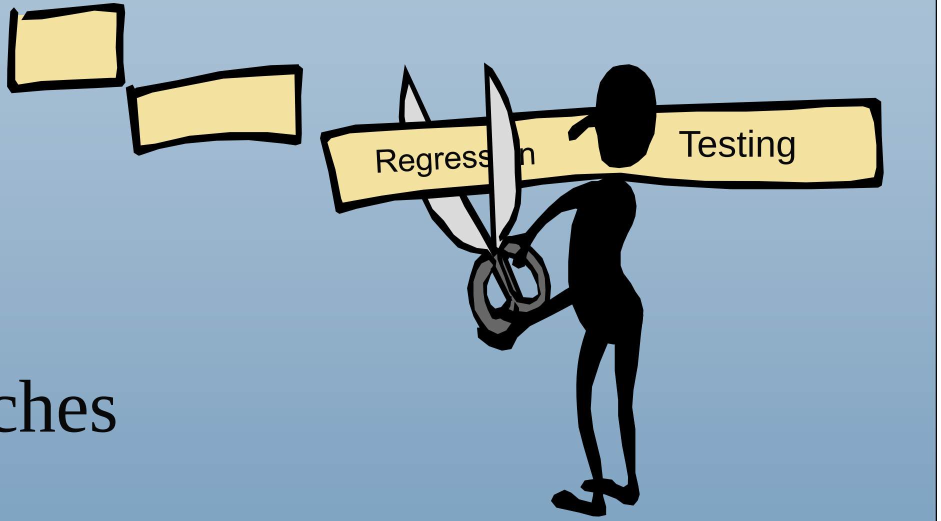
*The cost of complexity is exponential.*

✓Regression Deficit

*Every time you add new features*
*the regression test grows longer!*

✓Unsynchronized Code Branches

*The longer two code branches remain apart, the more difficult merging will be.*



Perfection is ***One-Piece-Flow:***

*Any useful feature set – at any time – in any order*

***Let it Flow***

lean

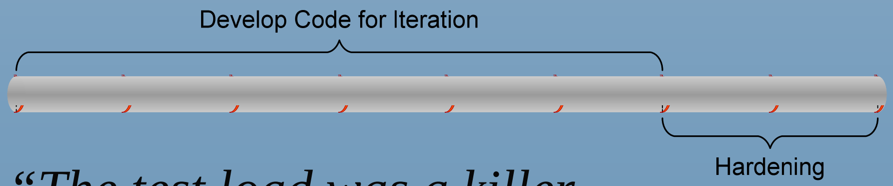
Copyright©2006 Poppendieck.LLC

# Case Study: Rally Software Development

*"We found ourselves doing waterfall in time-boxed increments. During the first year we had a lot of technical debt."*

Testing:

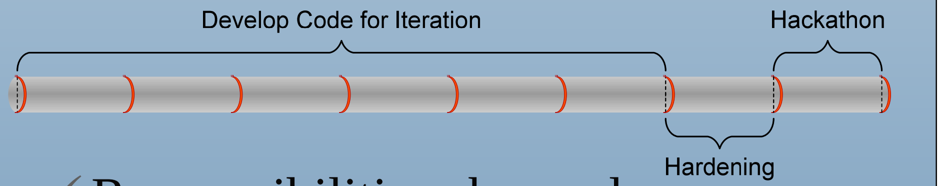
- ✓ JUnit for unit tests
- ✓ HTTPUnit for testing the GUI
  - × Not capable of testing page flows
  - × Most GUI testing manual
  - × All acceptance testing manual
- ✓ 6 weeks to develop, 2 weeks to test, and not all testing was done.



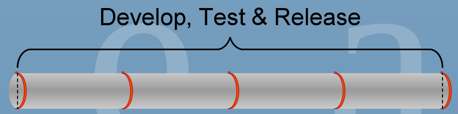
*"The test load was a killer, and it just kept going up."*

Ryan Martens, CTO

- ✓ Gradually moved page flow platform to Spring and AJAX
  - × Tested Spring with FIT & Fitnesse
  - × Tested AJAX by using JIFFIE to bind Java to IE. Wrote tests in Java to test AJAX through the browser.
- ✓ Hardening was reduced to 1 week.

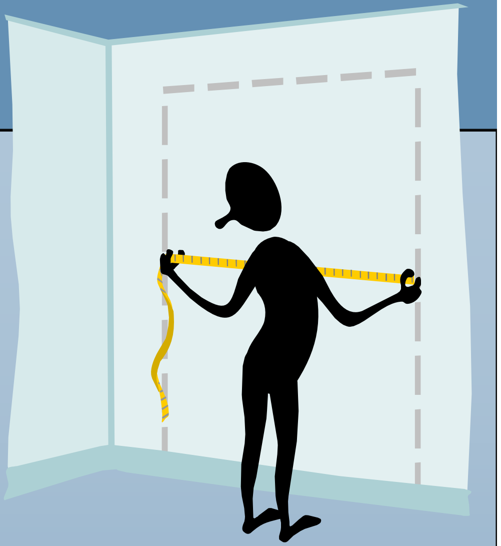


- ✓ Responsibilities changed:
  - × Testers: FIT tables & JIFFIE tests
  - × Developers: FIT fixtures, JUnit tests, and GUI test harness
- ✓ Now release monthly, pre-hardened!



lean

# *Principle 4: Defer Commitment*



## *The Goal:* ***Change-Tolerant Software***

✓ 60-80% of all software is developed after first release to production.

✓ A development process that anticipates change will result in software that tolerates change.

## *The Strategy:* ***Maintain Options***

✓ Make decisions ***reversible*** whenever possible.

✓ Make ***irreversible*** decisions as late as possible.

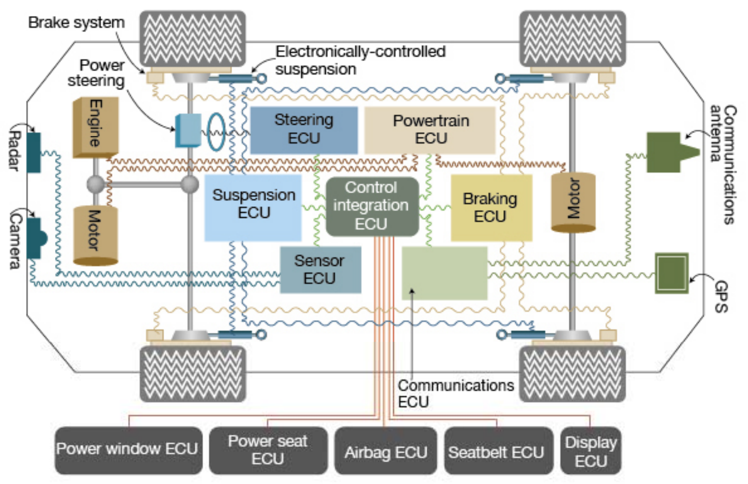
lean

Copyright©2006 Poppendieck.LLC

# *Case Study: Toyota Prius*

## A Computer on Wheels

4 million lines of code & growing exponentially….



- ✓ February 1, 1994 – First team meeting.
- ✓ November 1994 – team asked to make a hybrid concept car for the Motor Show in October 1995 (11 months later).

6 months

- ✓ November, 1994 - May 1995 – The team systematically narrowed eighty engine options down to ten, and narrowed these to four, which they simulated carefully.

5 months

- ✓ May, 1995 – Engine option chosen.
- ✓ October, 1995 – Concept car shown.
- ✓ December, 1995 – Launch date set for December, 1997 by president Okuda.

8 months

- ✓ Design competition held among Toyota's four design studios to establish concept.

15 months

- ✓ July, 1996 – Design from Calty Design Studio in Newport Beach, CA selected.
- ✓ October. 1997 – Prius officially unveiled.
- ✓ December, 1997 – Launched in Japan.

l e a n

Copyright©2006 Poppendieck.LLC

# *Principle 5: Deliver Fast*

DELL

Manufacturing

TOYOTA

Product Development

patientkeeper

Software Development

Companies that compete on the basis of speed:

- ✓ Enjoy a significant cost advantage relative to peers
  - A 25-30% cost advantage is typical.
- ✓ Have extremely low defect rates
  - It is impossible to go fast without superb quality.
- ✓ Acquire a deep customer understanding
  - Fast companies can take an experimental approach to product development.
- ✓ Have a sustainable competitive advantage.

Copyright©2006 Poppendieck.LLC

l e a n

# *Case Study: PatientKeeper*

patientkeeper

Jeff Sutherland
CTO PatientKeeper

**Speed to market**

- ✓ 45 cycles while competition does one
  - Maintenance releases once or twice a week
  - New feature releases every month
  - New applications released every quarter

**Predictable Delivery**

- ✓ Never a late release
- ✓ Problems are seen long before the release date
- ✓ The company self-organizes around the problems

**Pull from Demand**

- ✓ Priorities reorganized on a weekly basis by CEO, sales, and account management
- ✓ Customer impact and schedule impacts are dealt with at the time of the decision

**No Abnormalities because rapid cycle time:**

- ✓ Eliminates buggy software because you die if you don't fix this
- ✓ Fixes the install process because you have to install 45 releases a year
- ✓ Improves the upgrade process because of a constant flow of mandatory upgrades

Copyright©2006 Poppendieck.LLC

# *Don’t Batch & Queue*

## ***Lists***

- ✓ How long is your defect list?
- ✓ How far apart are your releases?
- ✓ How many weeks (years?) of work do you have in your backlog?

## ***Churn***

- ✓ If you have requirements churn, you are specifying too early.
- ✓ If you have test-and-fix cycles, you are testing too late.

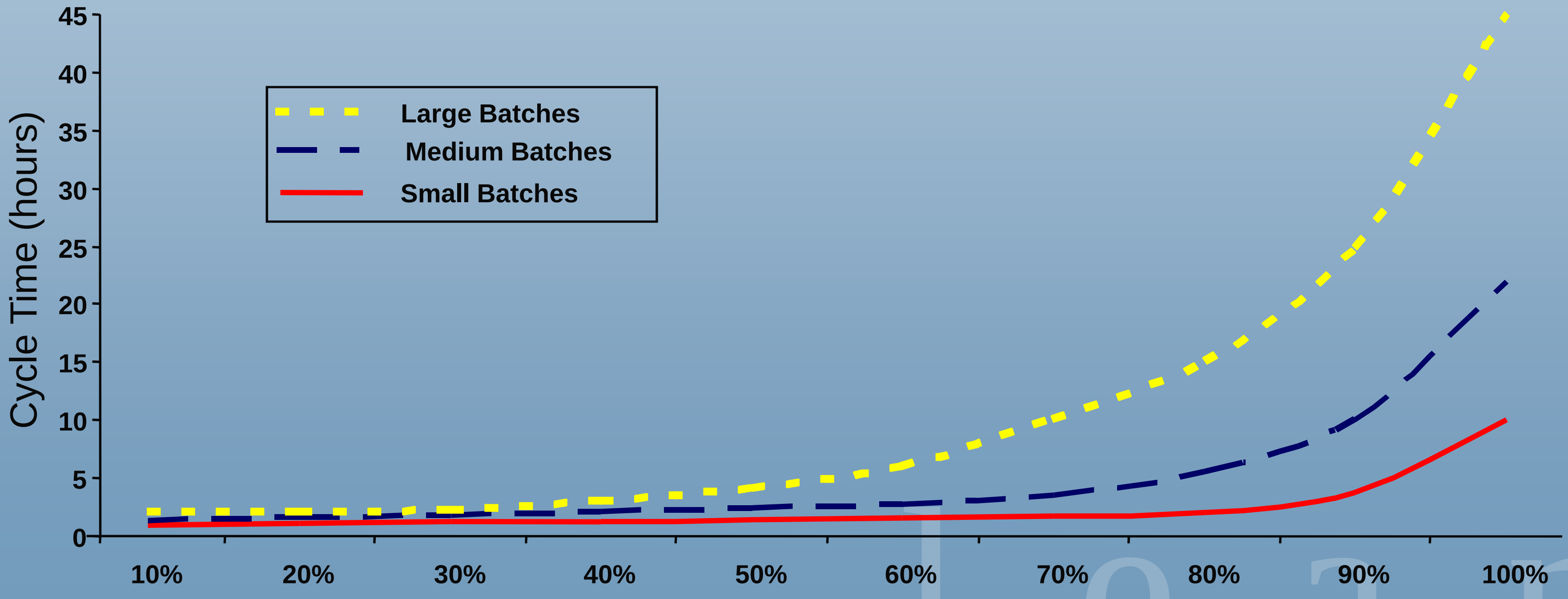
Copyright©2007 Poppendieck.LLC

l e a n

# *Throughput Trumps Utilization*

Little’s Law: $$\textbf{Total Cycle Time} = \frac{\textbf{Number of Things in Process}}{\textbf{Average Completion Rate}}$$

Cycle Time as a Function of Utilization and Batch Size*



Copyright©2006 Poppendieck.LLC

l e a n

# *Principle 6: Respect People*

## *Decide as Low as Possible:*

- ✓ *Move responsibility and decision-making to the lowest possible level.*

People Thrive on

- ✓ **Pride**
  - Passion
  - Deep Expertise
- ✓ **Commitment**
  - A team is a group of people who have committed to each other to work together to achieve a common purpose.
- ✓ **Trust**
  - Consistent, Reliable Performance
  - Responsibility-Based Planning & Control
- ✓ **Applause**

Copyright©2006 Poppendieck.LLC

# Case Studies: Engaged People

## Quicken Renal Property Manager

- ✓ **The Goal:** Convert the Quicken team from V20+ thinking to an entrepreneurial mindset
- ✓ The Quicken team was challenged to develop a new product and a new development process
  - × Solve the customers' problem
  - × Design the process

    …no more…no less
- ✓ The team spanned all functions
  - × Not just software development
- ✓ Worked together like a startup
  - × Interviewed customers together
- ✓ Everything was an experiment
  - × Focused on learning
- ✓ **The Result:** ***Excited, engaged team***
  - × 1 yr to release a great new product

'Moving from V20+ to V1 Thinking; A case study in applying Lean Principles' Soni Meckem, Intuit; Lean Design & Development 2005

## Mortgage Company

- ✓ Files a LOT of Paperwork
- ✓ Moved from paper to electronic filing
- ✓ Used "Classic Lean" in Operations
- ✓ Great Success over three years
- ✓ But Could not get any software done
  - × Small Jobs with Low Priority
- ✓ Lean Software Development Class for Management Team
  - × Along with additional agile training
- ✓ Assigned Developers to work with Lean Operations Teams
- ✓ Now: The team ALWAYS figures out how to accomplish its goals
- ✓ ***Very engaged people***
  - × "They can't be stopped."
- ✓ The only challenge is spreading the enthusiasm throughout the company.

Copyright©2006 Poppendieck.LLC
l e a n

# *Principle 7: Optimize the Whole*

Drive cost out of each department

- ✓ Easy
- ✓ Often interferes with overall cost reduction

Eliminate waste between departments

- ✓ Difficult
- ✓ May not result in the lowest department costs

ZARA

| RESULTS | Zara | Industry |
|---|---|---|
| New Items introduced / year | 11,000 | 3,000 |
| Items sold at full price | 85% | 60-70% |
| Unsold Items | <10% | 17-20% |
| % sales spent on advertising | 0.3% | 3-4% |
| % sales spent on IT | 0.5% | 2% |

*Zara: Women's fashion clothing*

- ✓ Design-to-Store in 2 weeks.
- ✓ Twice-weekly orders.
  - Delivers globally 2 days after order
    - On hangers, priced, ready to sell
    - Shipping prices are not optimized!
- ✓ Manufactures in small lots
  - Mostly at co-ops in Western Spain
    - At Western European labor rates…

Copyright©2007 Poppendieck.LLC

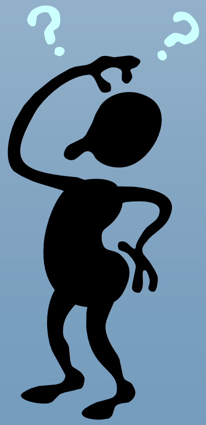
l e a n

# *Alignment*

In order for organizations to perform brilliantly, there are two prerequisites:*

First, everyone has to agree on *what they want*, Second everyone has to agree on *cause and effect*.

Does everyone on the management team speak the same language?



*"The Tools of Cooperation and Change," by Clayton Christensen and others, *Harvard Business Review*, Oct 2006

Copyright©2007 Poppendieck.LLC

lean

# Financial Perspectives

## *Balance Sheet Thinking*

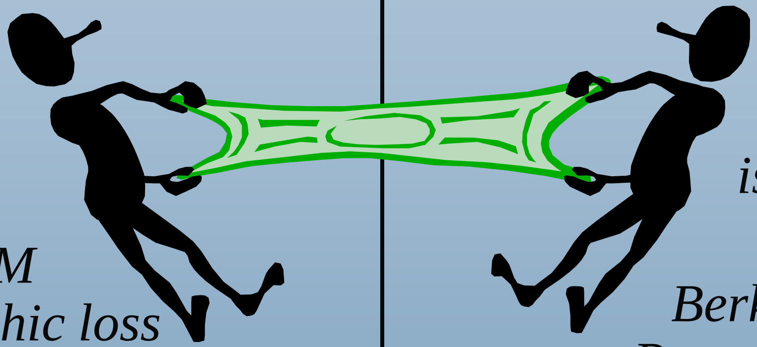
- What is the break-up value of the company?

*"I look at the bottom line. It tells me what to do." Roger B. Smith*

*"This metric guided GM into the most catastrophic loss of market share in business history."**

- ✓ Delay doesn't matter
- ✓ Just-in-case is wise
- ✓ Work-in-process has value
- ✓ Queues support better decisions

*"Conquering Complexity in your Business,"
by Michael George & Stephen Wilson, p 53



## *Cash Flow Thinking*

- How long does it take to convert capital into cash?

*"The value of any stock, bond, or business today is determined by the cash inflows and outflows…"*

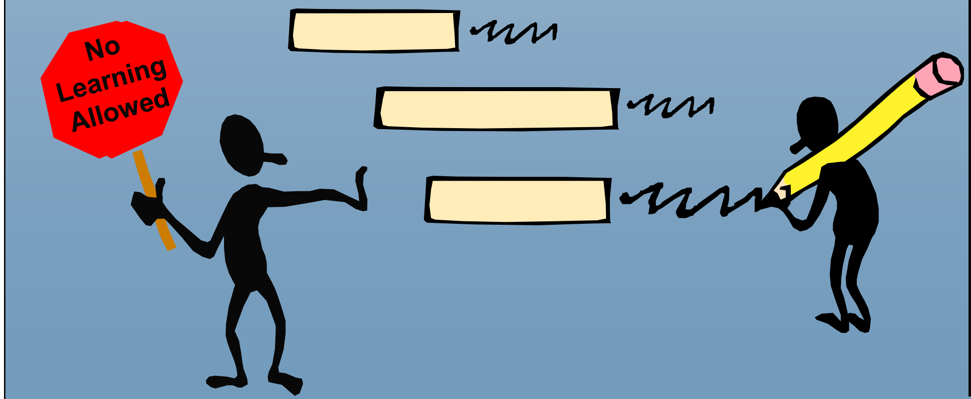
*Berkshire Hathaway Annual Report, 1992 (Warren Buffett)*

- ✓ Delay creates waste
- ✓ Just-in-time is wiser
- ✓ Work-in-process is waste
- ✓ Queues gum up the works and slow things down

# *Conformance to Plan*

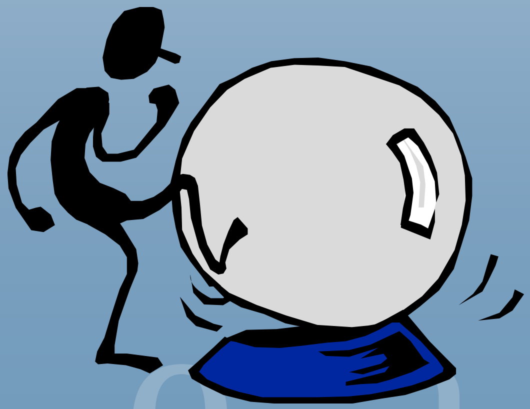
## A Plan is a Commitment

- ✓ Predictability comes from conformance to plan.
- ✓ The plan is always right, even though it was made when we had the least information.



## Planning is indispensable, but plans are useless. *

- ✓ The most predictable performance comes from maintaining options until we have the most information.



* Dwight Eisenhower

l e a n

Copyright©2007 Poppendieck.LLC

# Utilization

We need full utilization of expensive resources.

- ✓ It is impossible to have intact teams because this decreases utilization.
- ✓ Large queues of work help keep everyone busy.

It is impossible to move rapidly without slack.

- ✓ Intact teams increase overall productivity by preserving team learning.
- ✓ Batch and queue mentality is the biggest detriment to system-wide performance.

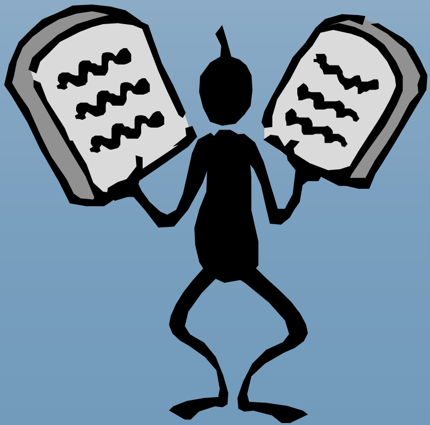
lean

Copyright©2007 Poppendieck.LLC

# *Work Standards*

The purpose of standards is to make it possible for any one to do any job.

- ✓ Standards are initiated by process groups.
- ✓ Written standards are to be followed, not changed.



The purpose of standards is to provide a baseline for the team to change.

- ✓ *If you believe that standards are writ in stone, you will fail. You have to believe that standards are there to be changed.**

* Yoshio Shima, Director, Toyoda Machine Works

lean

Copyright©2007 Poppendieck.LLC

# Accountability

## Span of Control

Hold people accountable for what they can ***control***

Measure at the individual level

Fosters competition

## Example

The development team should be responsible for ***technical success***

The product manager should be responsible for ***business success***

## Span of Influence

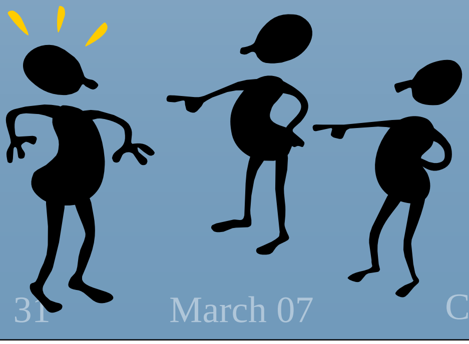
Hold people accountable for what they can ***influence***

Measure at the team level

Fosters collaboration

## Example

The team includes technical and business people, and the whole team assumes responsibility for ***business success***



> ***"There is no such thing as "Technical Success"***
>
> Kent Beck, XP 2004

Copyright©2007 Poppendieck.LLC

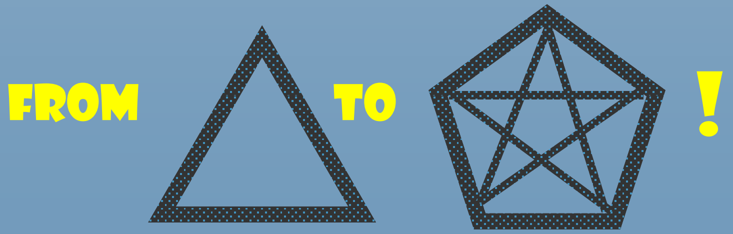
# Measure UP

## Decomposition

- ✓ You get what you measure
- ✓ You can’t measure everything
- ✓ Stuff falls between the cracks
- ✓ You add more measurements
- ✓ You get local sub-optimization

## Example

- ✓ Measure Cost, Schedule, & Scope
  - × Quality & Customer Satisfaction fall between the cracks
  - × Measure these too!



## Aggregation

- ✓ You get what you measure
- ✓ You can’t measure everything
- ✓ Stuff falls between the cracks
- ✓ You measure UP one level
- ✓ You get global optimization

## Example

- ✓ Measure Cost, Schedule, & Scope
  - × Quality & Customer Satisfaction fall between the cracks
  - × Measure Business Case Realization instead!

l e a n

Copyright©2007 Poppendieck.LLC

# *Three System Measurements*

## *Average Cycle Time*

- ✓ From Product Concept
- ✓ To First Release

or

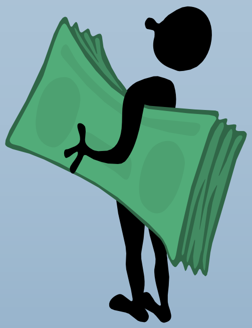
- ✓ From Feature Request
- ✓ To Feature Deployment

or

- ✓ From Defect
- ✓ To Patch

## *The Business Case*

- ✓ P&L or
- ✓ ROI or
- ✓ Goal of the Investment



## *Customer Satisfaction*

- ✓ A measure of sustainability

Copyright©2007 Poppendieck.LLC

lean

lean

software development

# Thank You!

*More Information: www.poppendieck.com*