l e a n

software development

Organizational Agility

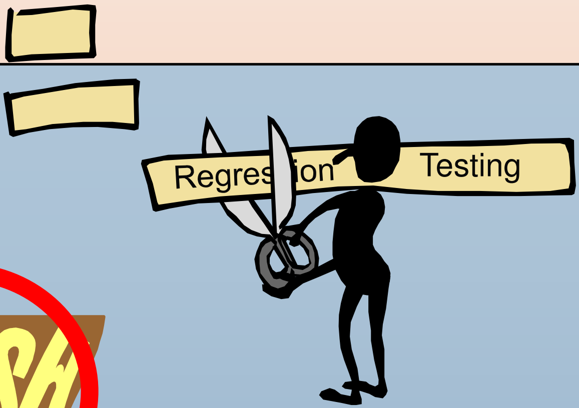
What's Not in the Book

mary@poppendieck.com

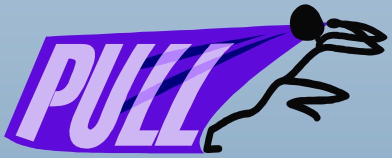
Mary Poppendieck

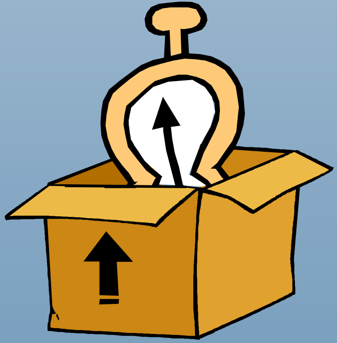
www.poppendieck.com

*What **is** in the Book*



1. Establish a Stop-the-Line Culture
   - ✓ Reduce Regression Deficit
   - ✓ A Challenging Discipline
2. Establish a Regular Cadence
   - ✓ Use Pull, not Push Scheduling
   - ✓ Very Counterintuitive
3. Limit Work to Capacity
   - ✓ Timebox, don't Scopebox
   - ✓ A Difficult Mental Model
4. Make Reliable Commitments
   - ✓ *No Partial Credit*
   - ✓ Requires True Teamwork

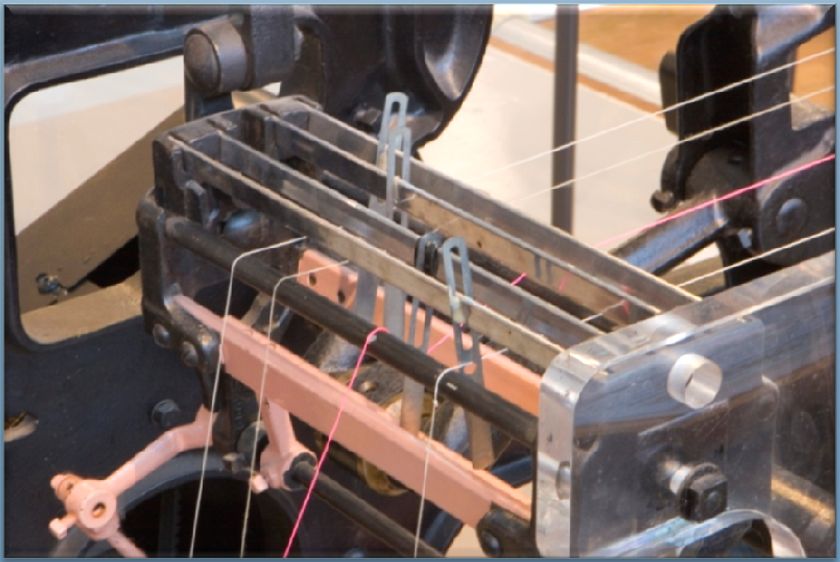




Stop-the-Line Culture

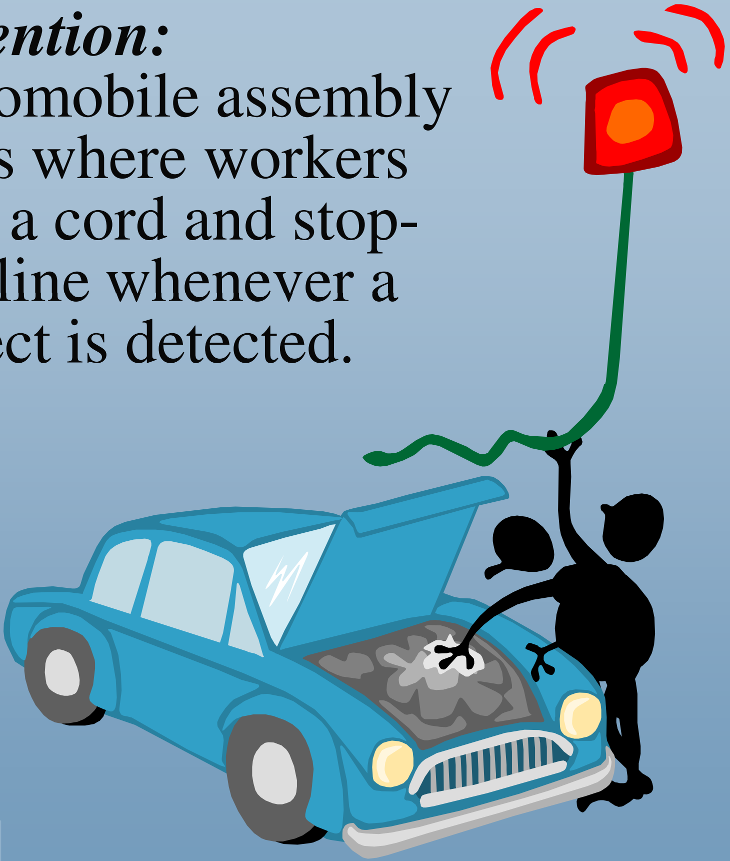
1920's:

- ✓ ***Invention:*** Looms that detect a broken warp thread when it breaks and immediately stop.

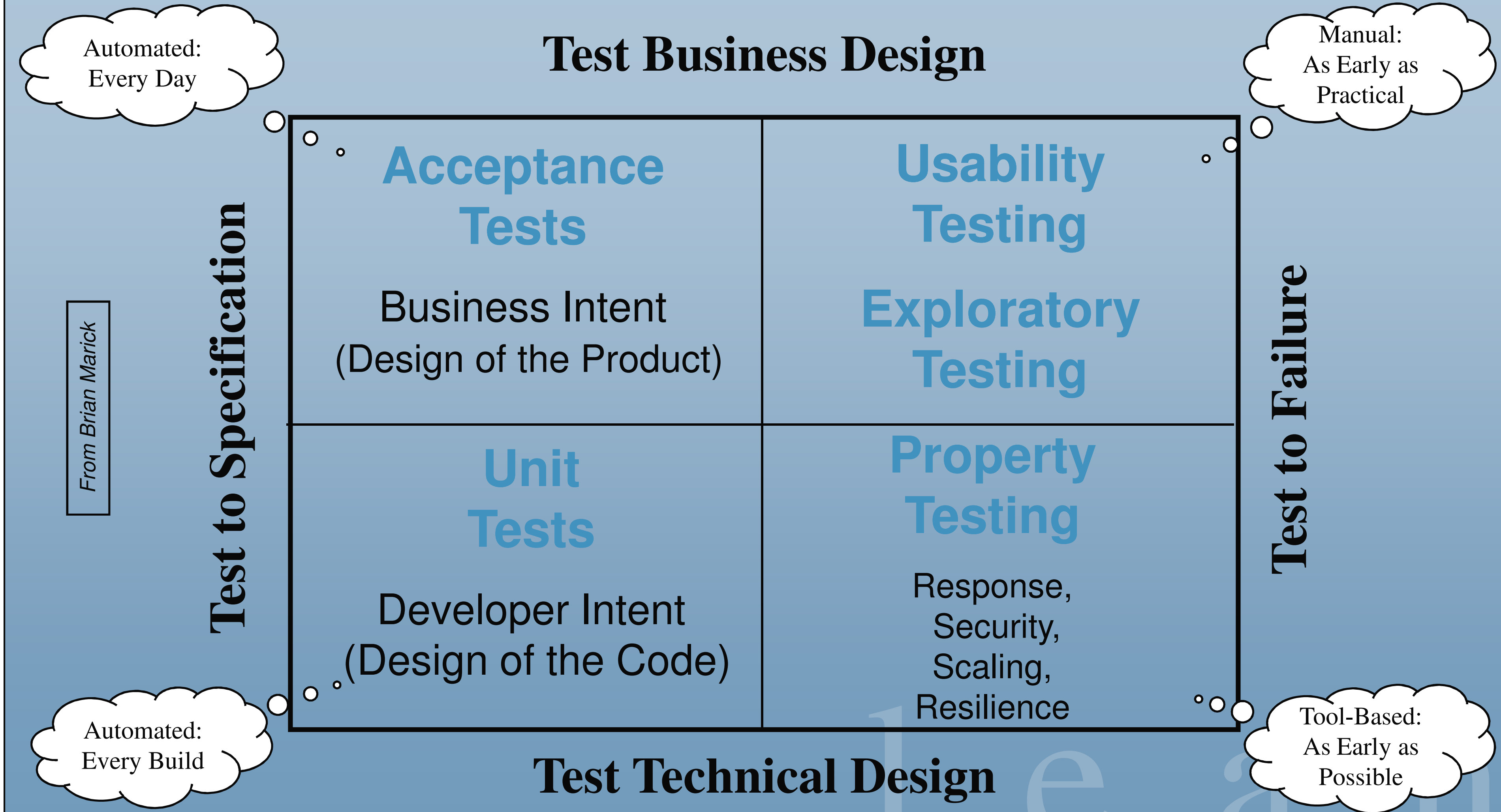


1950's:

- ✓ ***Invention:*** Automobile assembly lines where workers pull a cord and stop-the-line whenever a defect is detected.

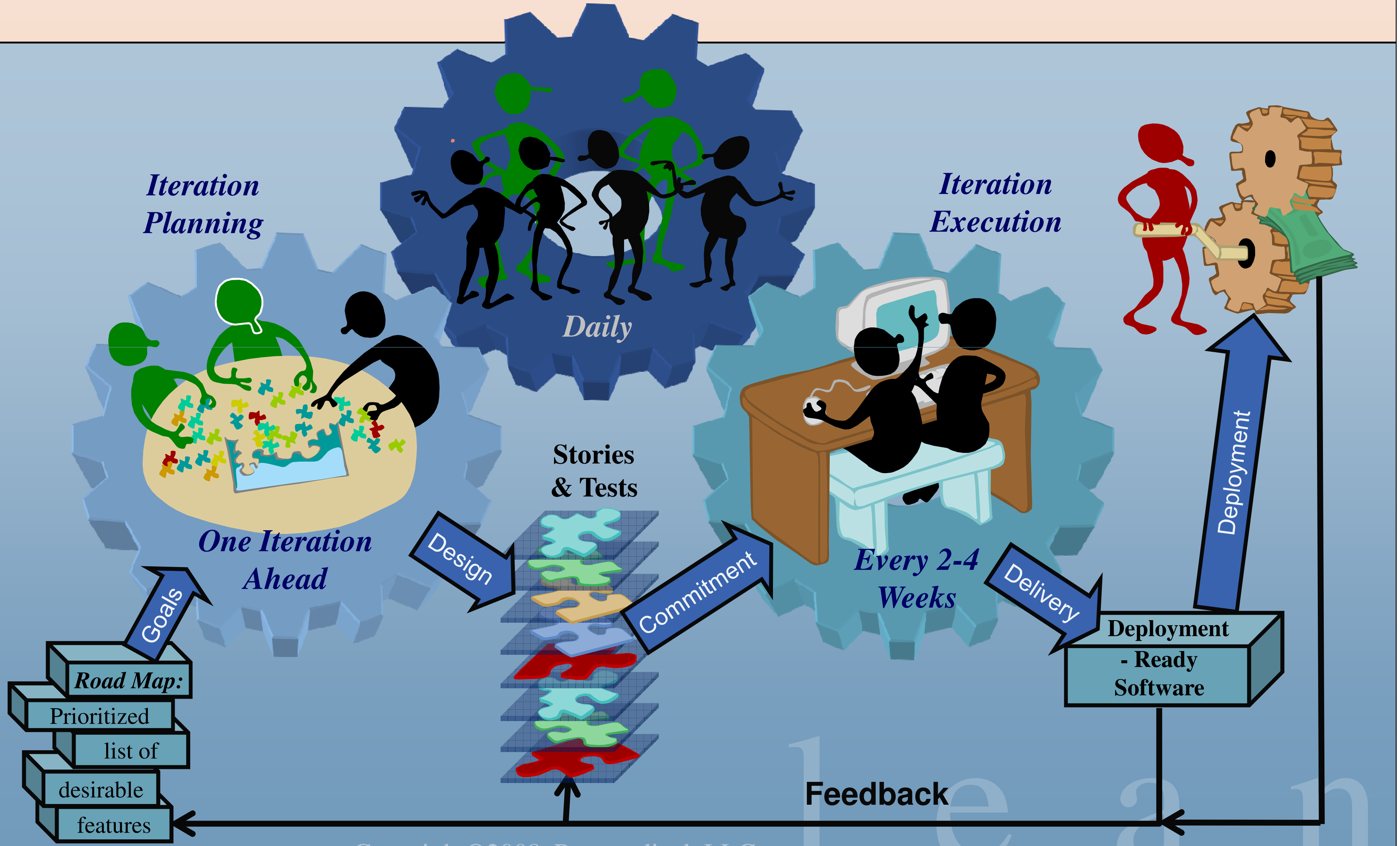


Test Harnesses

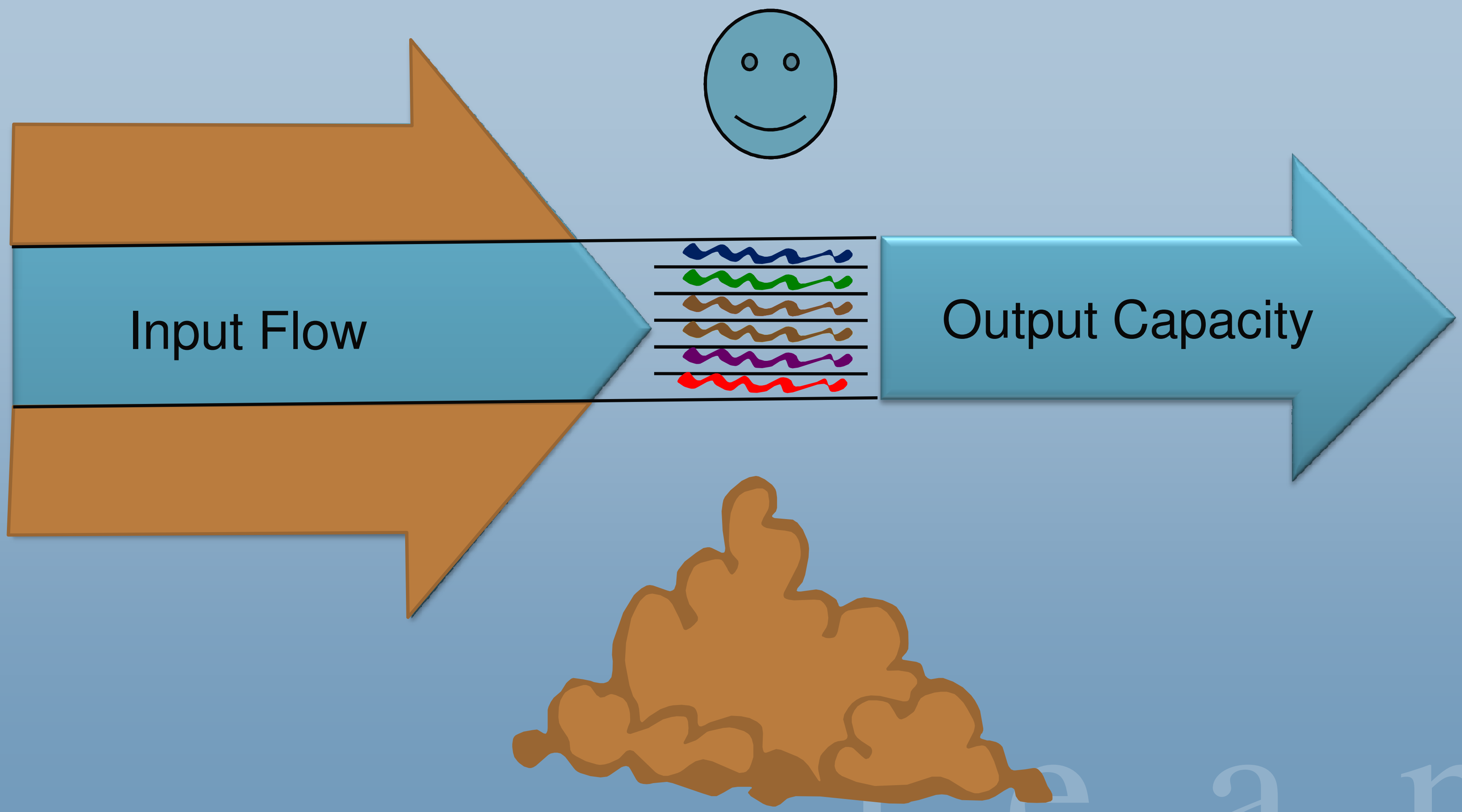


From Brian Marick

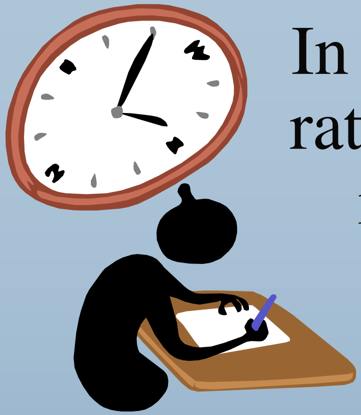
Establish a Regular Cadence



Limit Work to Capacity



Timebox, Don't Scopebox

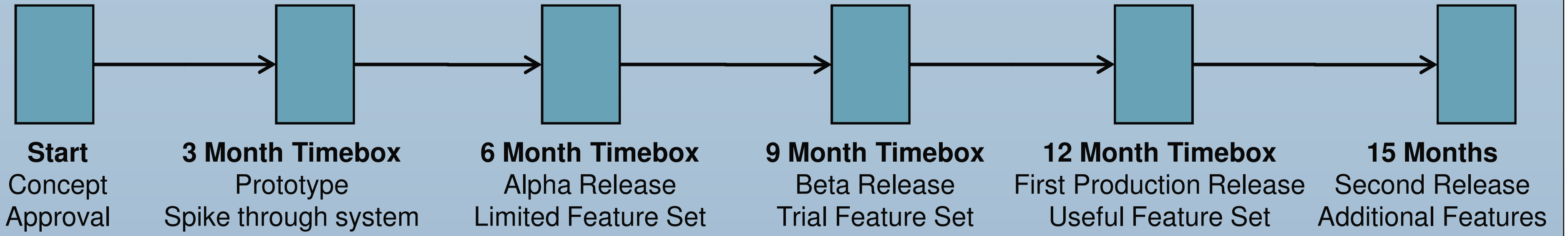


In software development, meeting a timebox, rather than completing pre-determined scope, is ***almost always*** the preferred approach.

Why?

1. The biggest waste in software development is *Extra Features* – and scopeboxing drives extra features. Their cost is exponential.
2. Staged delivery decreases work-in-process, reduces risk, gives earlier feedback, and results in faster return on investment.
3. Timeboxed development is *almost always* more productive.
4. The code – or the process/product – can *almost always* be simplified to capabilities that will fit in the timebox.

Timebox Scheduling



Responsibility-based Planning and Control:

- ✓ Timeboxed target events are scheduled
  - × Purpose: Review design, synchronize, make scheduled decisions.
  - × Decisions are scheduled at the "***Last Responsible Moment***."
- ✓ Skilled teams know what is expected at each target event.
- ✓ The timebox is always met – without tracking!
  - × Skilled workers pull information as required to deal with dependencies
  - × Multiple options are brought for each scheduled decision
  - × Functional leaders assume responsibility for deadlines

Reliable Commitment

Is the team committed?

- ✓ Small team
- ✓ Short timeframe
- ✓ Well-defined goal
- ✓ Used to working together
- ✓ Basic disciplines
- ✓ Necessary skills
- ✓ Clear priorities
- ✓ Good leadership
- ✓ ***Commitment of everyone to work together to achieve a common goal***

If commitments are not met:

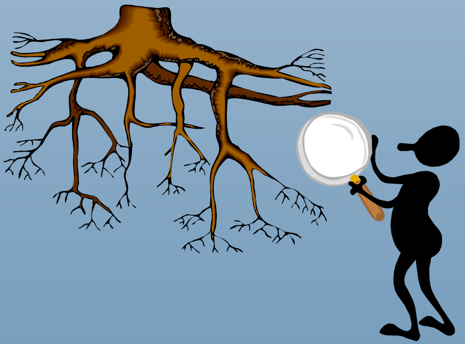
- ✓ Is it a team or a work group?
- ✓ Are tasks assigned or self-selected?
- ✓ Is the team ***expected*** to meet commitments?
- ✓ ***Who gets Partial Credit?***

What ***isn't*** *in the Book*

1. Whole Team
   - ✓ Beyond Product Owners
   - ✓ Eliminate Handoffs!

2. Leadership
   - ✓ Beyond Self-organizing Teams
   - ✓ Entrepreneurial Systems Designer

3. Deep Knowledge
   - ✓ Beyond Testing
   - ✓ Robust Systems



4. Relentless Improvement
   - ✓ Beyond Retrospectives
   - ✓ Classic Process Improvement Processes

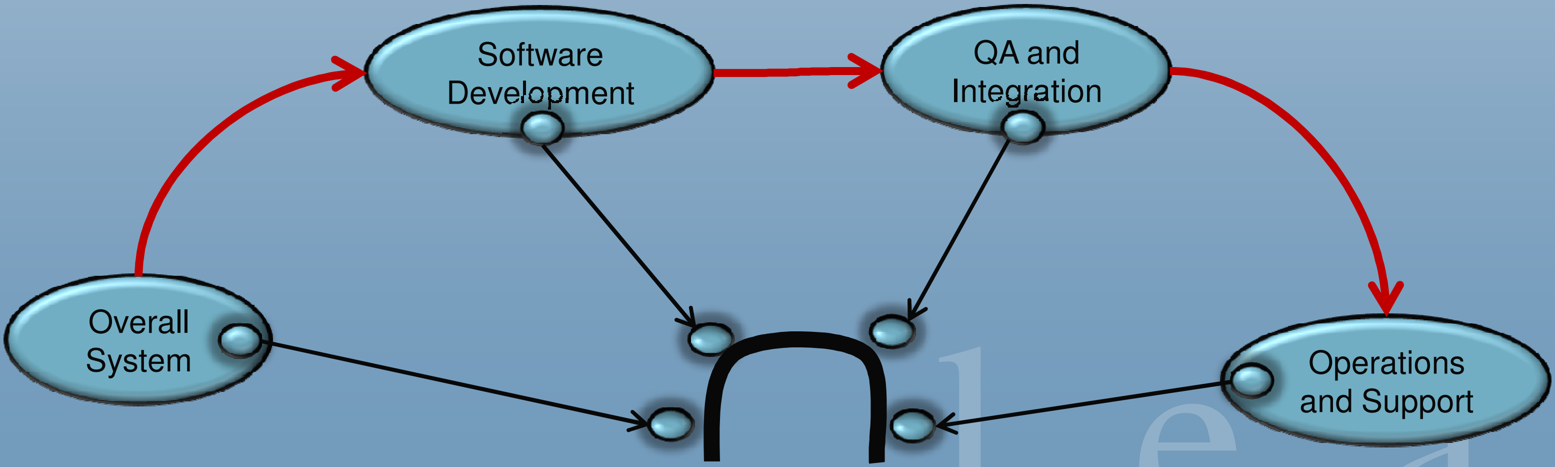
Whole Team

Handoffs: The Biggest Waste in Product Development *

A hand-off occurs whenever we separate:*

*Allen Ward "Lean Product and Process Development"

- ✓ Responsibility – One person decides what to do.
- ✓ Knowledge – Another person decides how to do it.
- ✓ Action – Still other people actually do the work.
- ✓ Feedback – Feedback has a long torturous road back!



lean

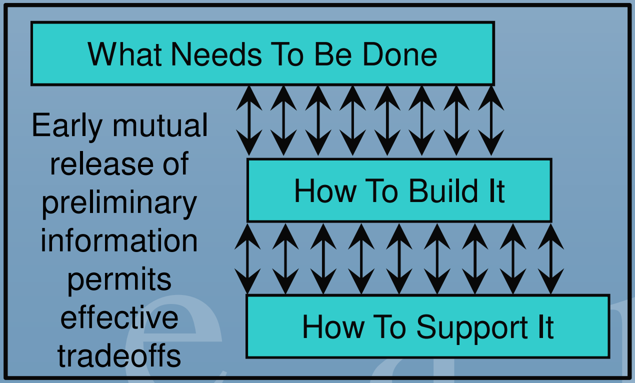
The Roots of Product Integrity*

Deep, Shared Understanding of the Job

- ✓ Who is involved?
- ✓ Exactly what is the job to be done?
- ✓ How is that job getting done today?
- ✓ What problems do people struggle with every day?

Deep, Shared Knowledge of the Technology

- ✓ Complete Products are developed by Complete Teams
  - Product Manager
  - Business Analysts
  - Technical Writers
  - Testers / QA
  - Support Desk
  - Technical Lead
  - Engineers
  - Developers
  - Operations
  - Etc.


What Needs To Be Done
Early mutual release of preliminary information permits effective tradeoffs
How To Build It
How To Support It

* Clark & Fujimoto – Product Development Performance

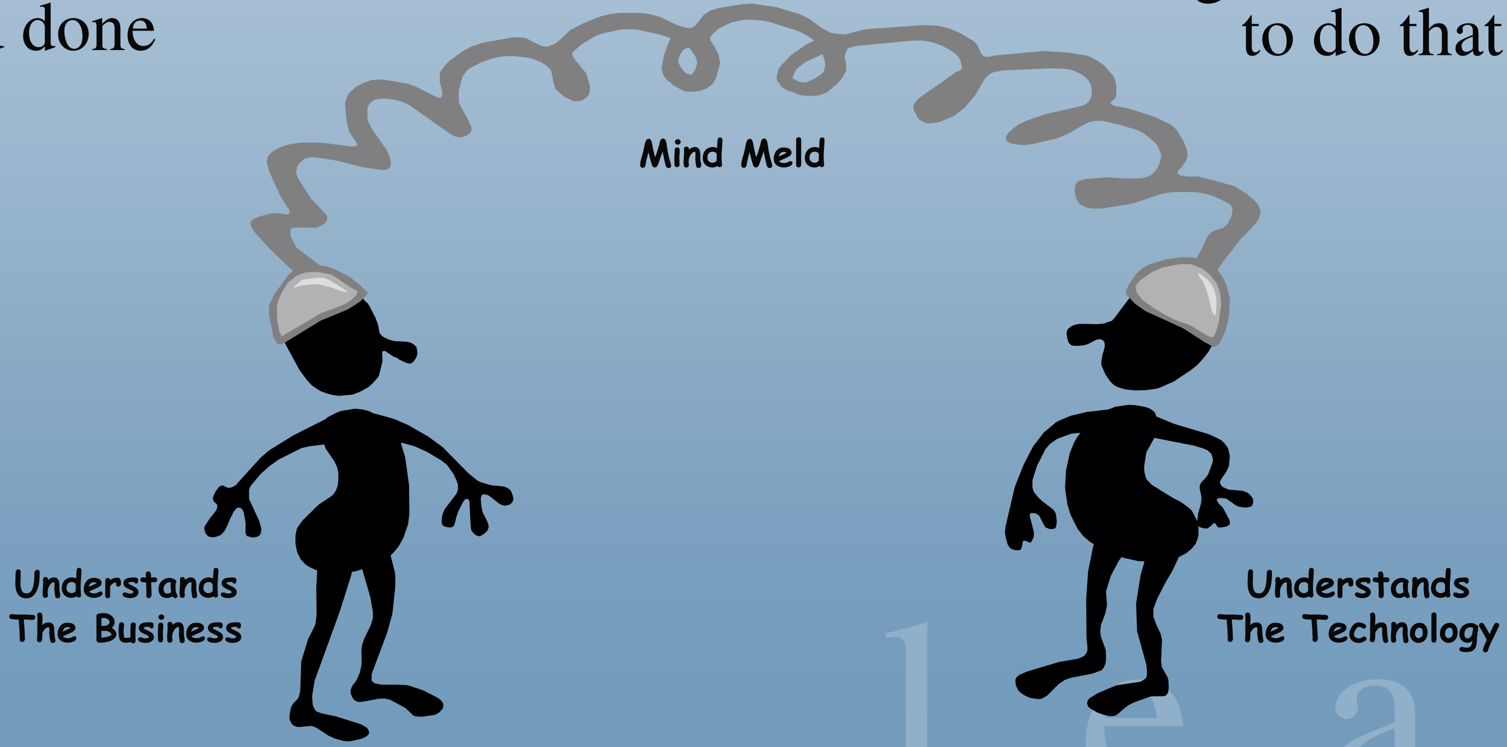
Leadership

Brilliant Products

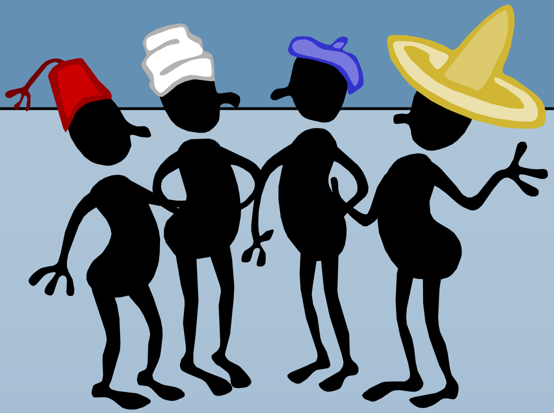
Embody a Deep Understanding of:

The job that customers need done

The right technology to do that job

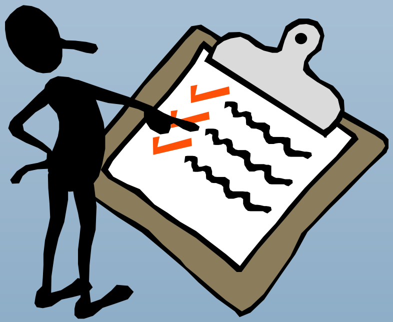


Who Decides?



Priority by Committee

- ✓ Everyone gets a vote
- ✓ No one is responsible for the outcome



Marketing by Checklist

- ✓ We want whatever the competition has
- ✓ The best way to get a me-too product

Behind every great product is a person with:

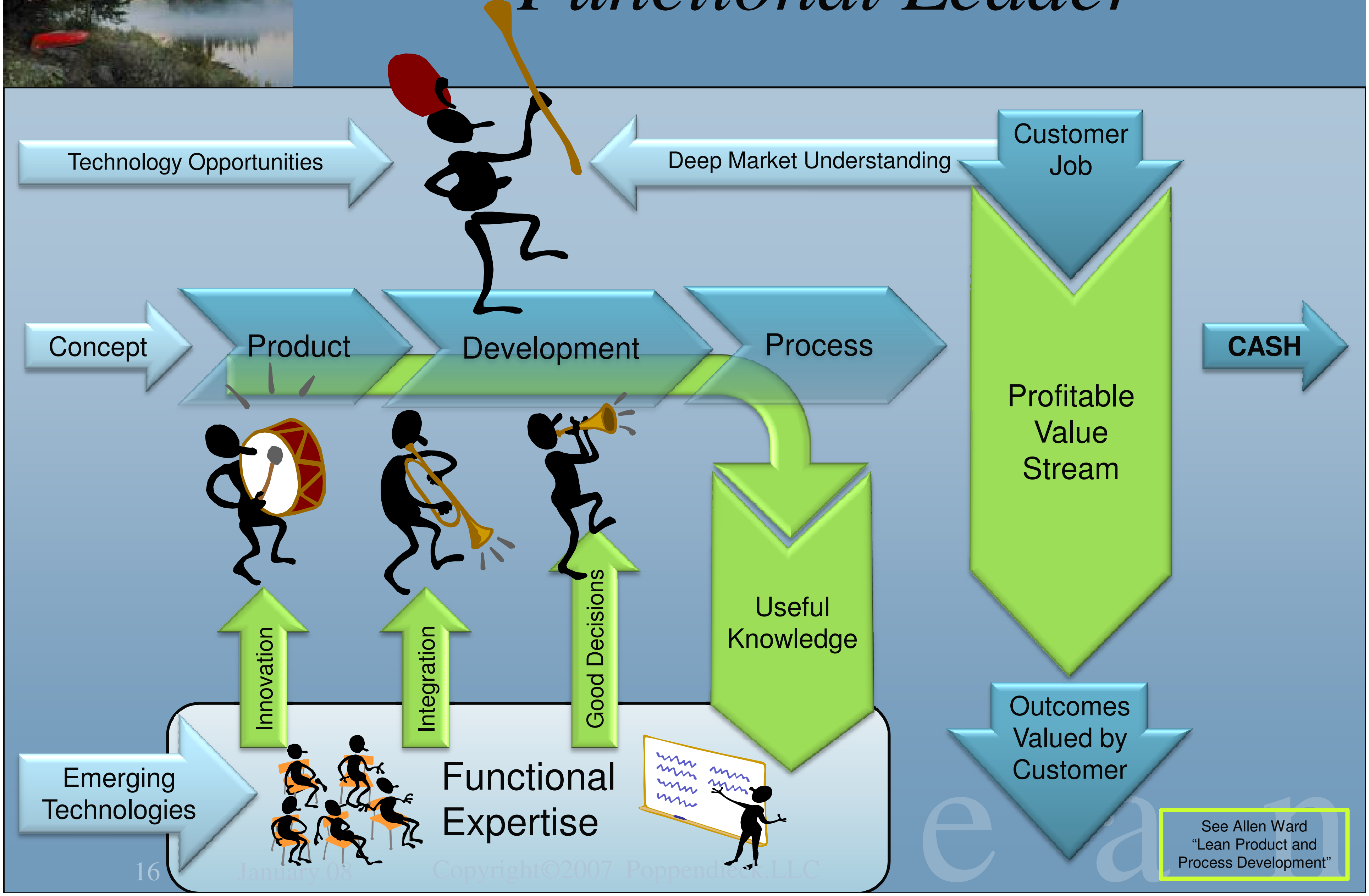
- ✓ Great empathy for the customer
- ✓ Insight into what is technically possible
- ✓ The ability to see what is essential and what is incidental

The Champion

Example: Chief Engineer at Toyota

- ✓ Responsible for Business Success
- ✓ Develops Deep Customer Understanding
- ✓ Develops the Product Concept
- ✓ Creates The High Level System Design
- ✓ Sets the Schedule
- ✓ Understands what customers will value and conveys this to the engineers making day-to-day tradeoffs
- ✓ Arbitrates trade-offs when necessary
- ✓ Defends the Vision

Functional Leader



See Allen Ward "Lean Product and Process Development"

Leadership Roles

Marketing Leader

Business Responsibility
Customer Understanding
Release Planning
Tradeoffs

Technical Leader

System Architecture

- ✓ At a high level
- ✓ Work daily with those developing the details

Technical Guidance

- ✓ Integration
- ✓ Tradeoffs

C H A M P I O N

Functional Leader

Preserve Knowledge

- ✓Towering Technical Expertise

Solve Problems

- ✓Relentless Improvement

Grow People

- ✓Full Potential

Process Leader

New Process Coach

Project Leader

Funding
Scheduling
Tracking

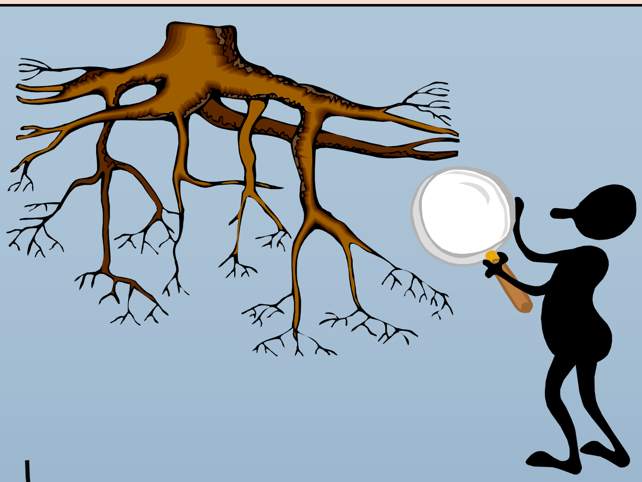
Deep Knowledge

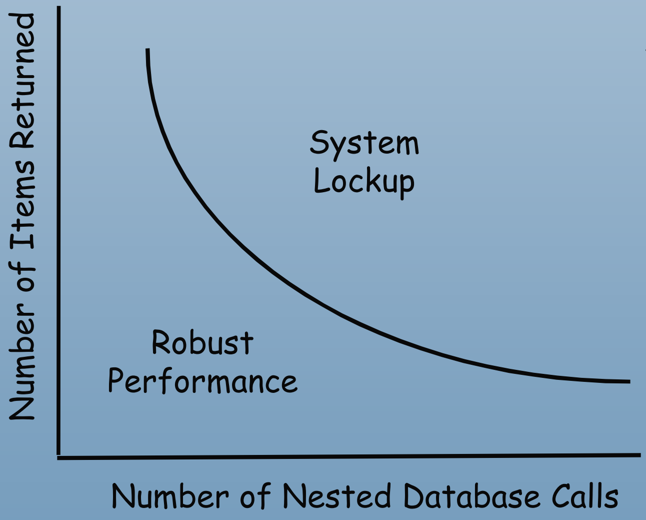
Deep Technical Expertise

- ✓ In-depth skill development
- ✓ Technical career paths

Robust Systems

- ✓ Capture knowledge from every failure incident
  - ➢ Root Cause Analysis
- ✓ Record the knowledge
  1. Add a Test
  2. Document a Pattern
  3. Create a Trade-off Curve
- ✓ A3 Documentation





Tradeoff curve of nested calls plotted against time-spent-in-nest, with a line showing where lockups do/don't occur.

Kai 改 Change

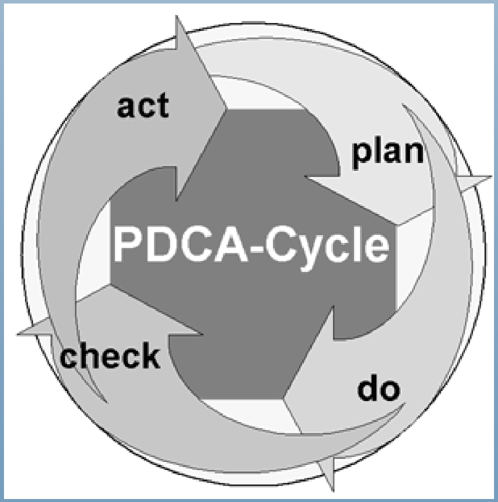
Zen 善 Good

Relentless Improvement

KAIZEN

Regular Team Meetings

- ✓ Every week or
- ✓ Every iteration



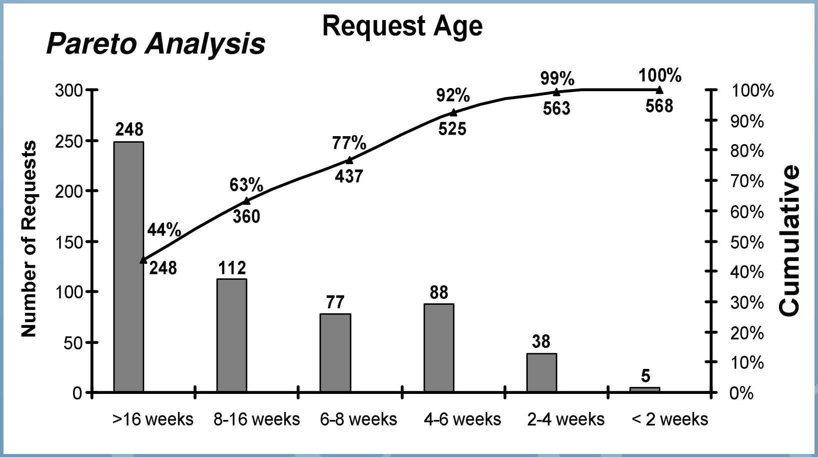
Deming Cycle (Sheward Cycle)

Data-Based Problem Analysis

- ✓ Pareto Analysis
  - What's Important?
- ✓ Ishikawa (Fishbone) Diagram
  - Cause & Effect Diagram
- ✓ Five Why's?
  - Root Cause Analysis

Many Quick Experiments

Solve One Problem at a Time



Request Age	Number of Requests	Cumulative	Cumulative %
>16 weeks	248	248	44%
8-16 weeks	112	360	63%
6-8 weeks	77	437	77%
4-6 weeks	88	525	92%
2-4 weeks	38	563	99%
< 2 weeks	5	568	100%

Respect People Case Study: Two Distribution Centers

(Same city, same type of service, employees with the same educational level, roughly equal wages.)

- ***Center 1:*** Management controlled the workforce through individual metrics. Employees had a specific amount of work to get done but were given considerable latitude on just how to do it. They were judged at the end of the day, week, month, and quarter on whether they achieved the target results, using data collected by a computerized tracking system. Front-line managers were engaged in working around current problems rather than in actually solving these problems at the root cause in collaboration with the employees. This was a task for higher-level managers and staff experts as time permitted, usually without the involvement of the production associates.
  70 percent associate turnover, significant management turnover.

- ***Center 2:*** Management had worked with employees to create standard work for every task. Visual control with status boards enabled everyone to see how everyone else was proceeding with their work. Therefore employees could help each other with any problems which emerged. The work process was very stable due to strict adherence to standardized work, so line managers could devote most of their energy to problem solving by engaging production associates in dialogues to get to root causes and implement sustainable solutions. Every associate spent ***four hours every week*** on improvement activities.
  1 percent associate turnover, practically no management turnover.
  Why? "The work here is always challenging because we are always solving problems using a method we all understand. And we all respect each other's contribution."
  This is a Toyota parts distribution center.

From: Jim Womack's e-letter: December 20, 2007
See the Lean Enterprise Institute (www.lean.org)

What are You Building?

Three Stonecutters were asked:

"What are you doing?"

I'm cutting stones!

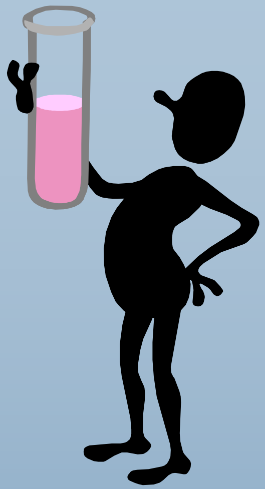
I'm building a cathedral.

I'm earning a living.

Cathedral Builders

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

Stone Cutters or Cathedral Builders?



The Litmus Test:

When workers are annoyed by their job –

Do they complain, ignore it, or fix it?

l e a n
software development

Thank You!

More Information: www.poppendieck.com