



LEAN DEVELOPMENT

WHEN YOU HAVE
ONLY ONE CHANCE TO
GET IT RIGHT

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

THE TOYOTA PRODUCTION SYSTEM



(1912-1990)

☀ Approach to Production

- ☀ Build only what is needed
- ☀ Stop if something goes wrong
- ☀ Eliminate anything which does not add value

☀ Philosophy of Work

- ☀ Respect for Workers
- ☀ Full utilization of workers' capabilities
- ☀ Entrust workers with responsibility & authority

CONCURRENT DEVELOPMENT

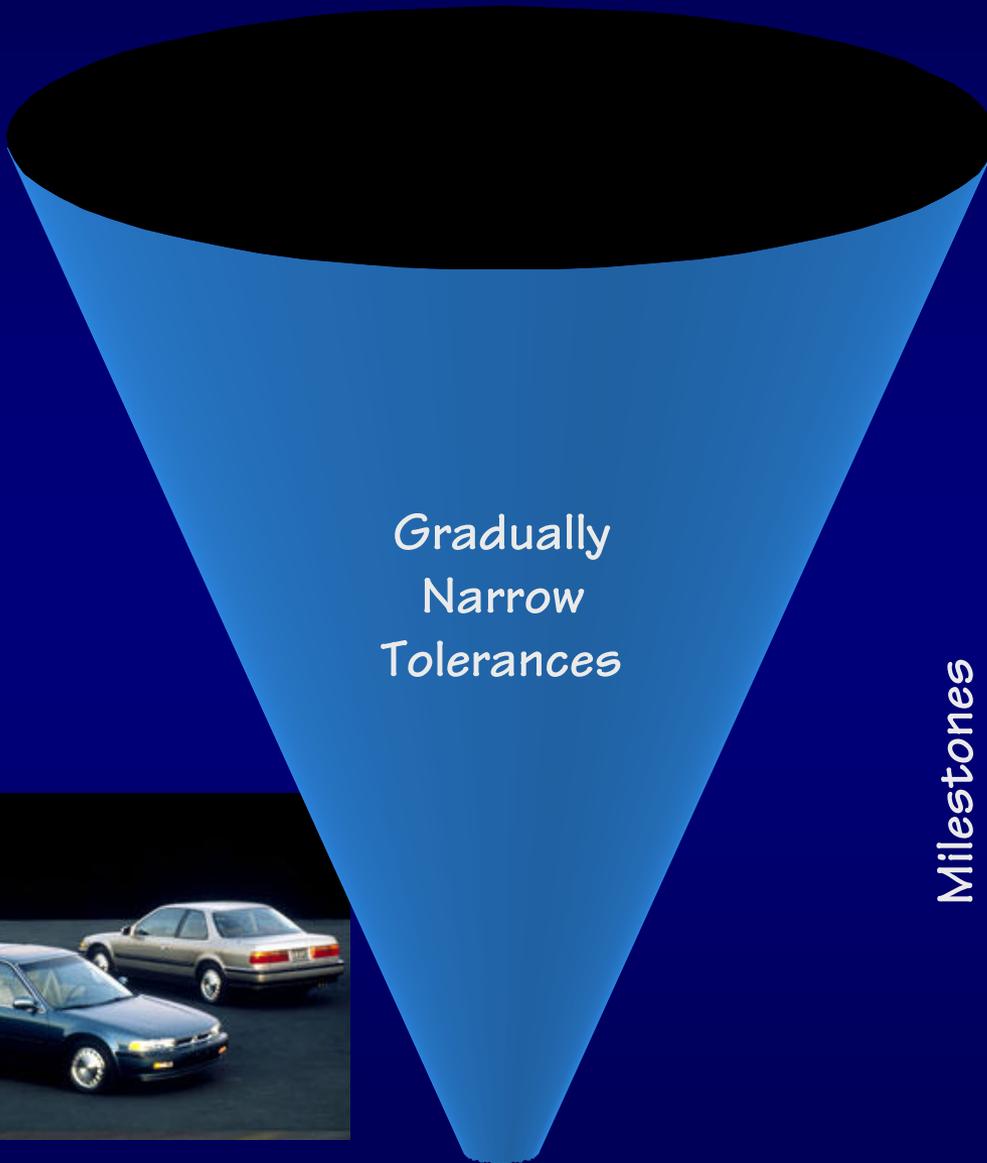
- ★ 1981 – GM Starts the G-10 Project
 - ★ 1988 – Buick Regal ★ 2 Years Late
 - ★ 1989 – Olds Cutlass & Pontiac Grand Prix
- ★ 1986 – Honda Starts the New Accord Project
 - ★ 1989 – Introduced as 1990 model
 - ★ 1990's – Largest-selling model in North America

The Machine That Changed The World, Womack, 1990

Late 80's	Japan	US	Advantage
Hours (millions)	1.7	3.1	45%
Months	46.2	60.4	24%
Staff	485	903	46%
Delayed Products	1 in 6	1 in 2	66%

Product Development Performance, Clark, 1991

CONCURRENT DEVELOPMENT



Milestones

- Vehicle concept
- Vehicle sketches
- Clay models
- Design structure plans
- First prototype
- Second prototype
- Production trials
- Release to production



PRINCIPLES OF LEAN THINKING

1. ELIMINATE WASTE
2. AMPLIFY LEARNING
3. DECIDE AS LATE AS POSSIBLE
4. DELIVER AS FAST AS POSSIBLE
5. EMPOWER THE TEAM
6. BUILD INTEGRITY IN
7. SEE THE WHOLE

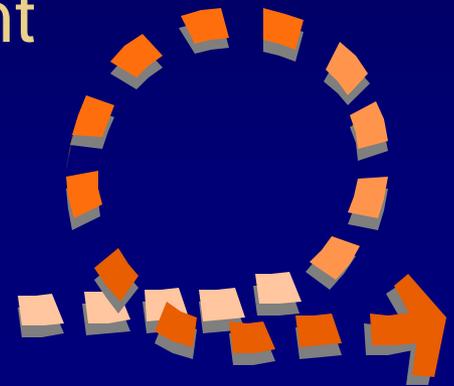
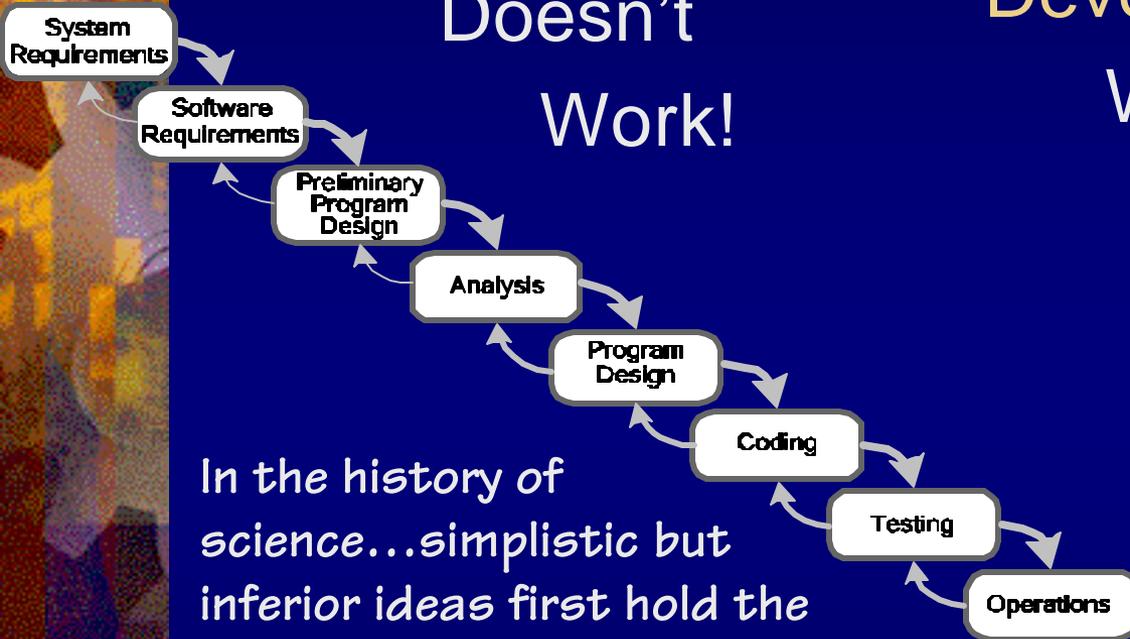
2. AMPLIFY LEARNING

✦ Waterfall

✦ Iterative Incremental Development

Doesn't
Work!

Works!



In the history of science...simplistic but inferior ideas first hold the dominant position, even in the absence of results. Medicine's "four humors" [come to mind].*

IID concepts have been and are recommended practice by leading software engineering thoughtleaders..., associated with many successful large projects and recommended by standards boards.*

* Craig Larman, "A History of Iterative and Incremental Development", IEEE Computer, June 2003

ITERATIVE DEVELOPMENT HISTORY

- ★ 1950's X-15 Hypersonic Jet (NASA)
 - ★ Success attributed to incremental development
- ★ 1960's Project Mercury (NASA)
 - ★ Very short, structured timeboxed iterations, Test First design
- ★ 1972 USA Trident submarine control system (IBM FSD)
 - ★ "Integration Engineering" was key to managing complexity
 - ★ Project 'could not be late'; used 6 month timeboxed iterations
- ★ Mid 1970's USA Navy helicopter-ship system LAMPS (IBM FSD)
 - ★ 45 - 1 month timeboxed iterations
 - ★ Every delivery was on time and under budget
- ★ 1987 Defense Science Board Task Force
 - ★ "Document-driven, specify-then-build approach lies at the heart of so many DoD software problems....Evolutionary development is best technically, and it saves time and money."
- ★ 1993 William J Spuck, JPL Internal Paper "Rapid Development Method"
 - ★ Describes Evolutionary Development as "a series of incremental deliveries. Each delivery contributes an operable, functionally viable, partial system. The overall system is developed and delivered to its users in small evolutionarily increments. The users employ the evolving system in the daily conduct of their mission."
- ★ 1994 Defense Science Board Task Force
 - ★ "DoD must manage programs using iterative development."

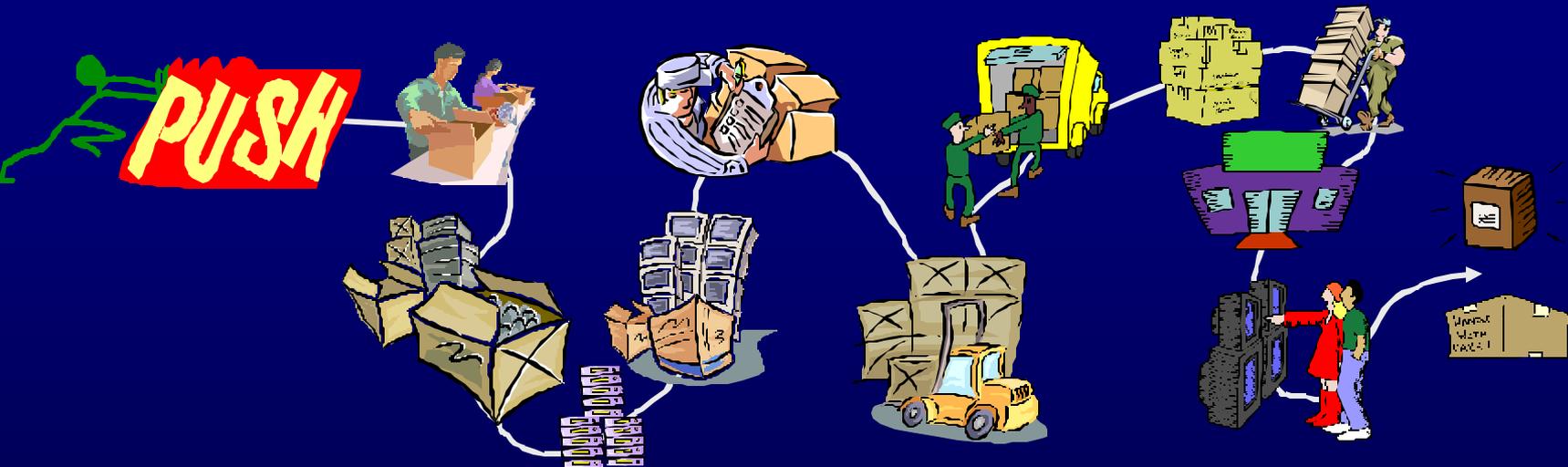
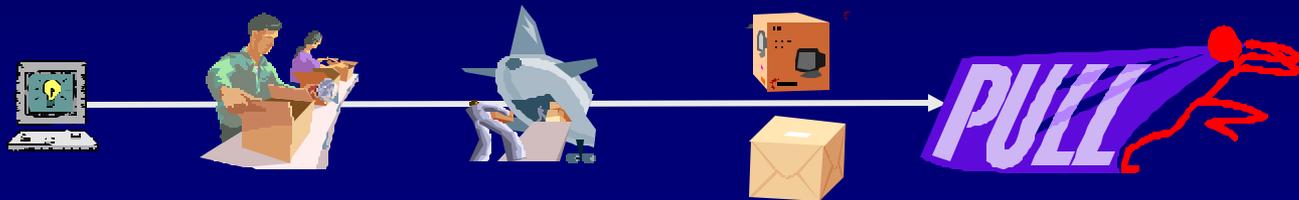
Craig Larman, "A History of Iterative and Incremental Development", IEEE Computer, June 2003



FOR TROUBLED PROJECTS

- ✱ Increase Feedback
 - ✱ Don't Toss Stuff Over-the-Wall
- ✱ Increase Learning
 - ✱ Use Rapid try-it-test-it-fix-it cycles
- ✱ Increase Communication
 - ✱ Develop Hardware & Software Together

1. ELIMINATE WASTE

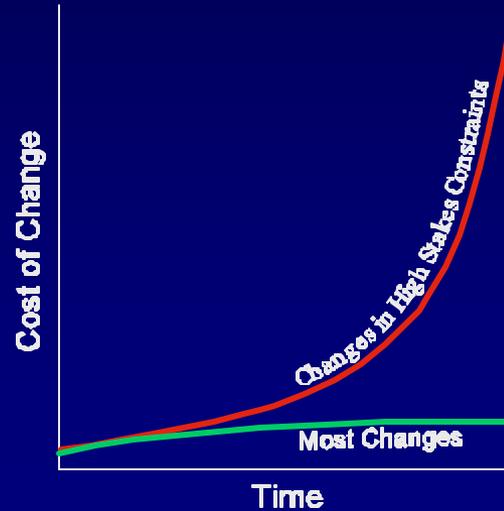


SEEING WASTE

Seven Wastes of Manufacturing*	Seven Wastes of Development
Inventory	Partially Done Work
Extra Processing	Paperwork
Overproduction	Extra Features
Transportation	Task Switching
Waiting	Waiting
Motion	Motion
Defects	Defects

* Shigeo Shingo, an engineer at Toyota and a noted authority on just-in-time techniques.

3. DECIDE AS LATE AS POSSIBLE



Manufacturing	Stamping Dies	Product Development
<p data-bbox="363 825 838 872"><i>Die Change is Expensive</i></p> <p data-bbox="291 896 865 943">Conventional Wisdom</p> <p data-bbox="363 972 768 1019">Don't Change Dies</p> <p data-bbox="291 1043 620 1090">Taiichi Ohno</p> <p data-bbox="363 1119 962 1215">Economics Requires Many Dies Per Stamping Machine</p> <p data-bbox="363 1243 838 1290"><i>One Minute Die Change</i></p>	<p data-bbox="1093 825 1566 872"><i>Mistakes are Expensive</i></p> <p data-bbox="1093 886 1476 933"><i>Changes Never End</i></p> <p data-bbox="1020 962 1591 1009">Conventional Wisdom</p> <p data-bbox="1093 1038 1688 1085">Wait to design – Wait to cut</p> <p data-bbox="1020 1105 1199 1152">Toyota</p> <p data-bbox="1093 1180 1624 1228">Early Design – Early Cut</p> <p data-bbox="1093 1242 1669 1289"><i>Half the Time – Half the Cost</i></p>	



DELAY COMMITMENT

Make Decisions at the Last Responsible Moment

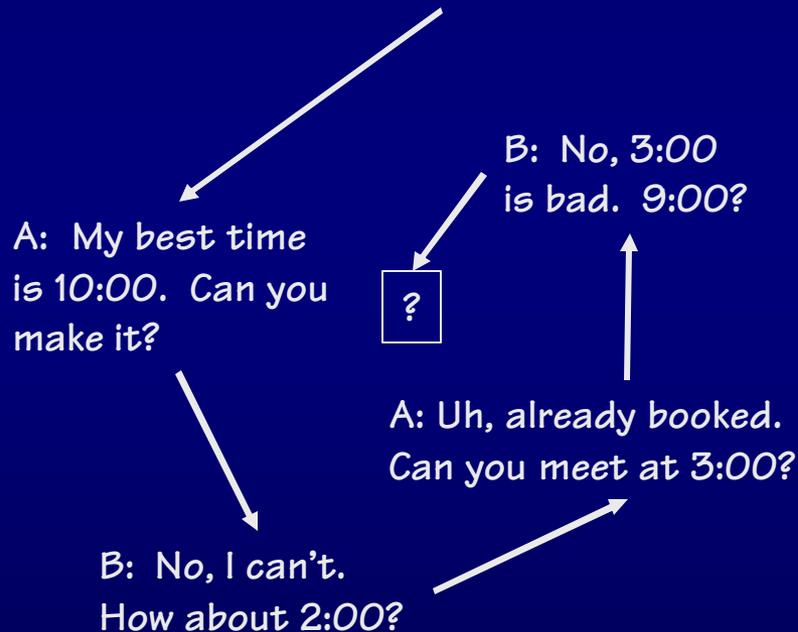
The moment when failing to make a decision eliminates an important alternative

- ✱ Share partially complete design information.
- ✱ Develop a sense of when to make decisions.
- ✱ Develop a sense of how to absorb changes.
- ✱ Develop a quick response capability.
- ✱ Avoid extra features.

POINT-BASED VS. SET-BASED

Point Based Design

Set up a meeting using the point-based model.



Set Based Design

Now set up the meeting by communicating about sets.

A: I can meet 10:00 - 1:00 or 3:00 - 5:00. Can you make any of these times?

B: Let's meet 12:00 - 1:00.

The diagram illustrates a set-based negotiation process. A proposes two time sets: 10:00 - 1:00 and 3:00 - 5:00. B suggests a meeting time of 12:00 - 1:00, which overlaps with the first set proposed by A.

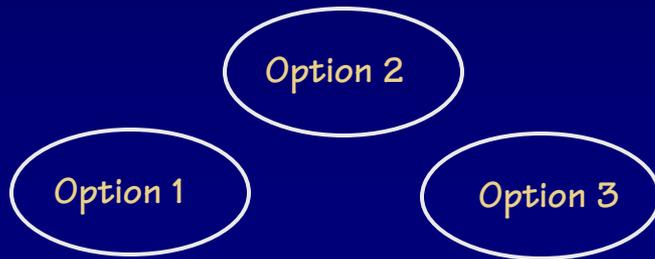
You already understand this!

based on dissertation by Durward K. Sobek, II, 1997

SET-BASED DEVELOPMENT

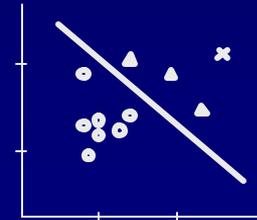
Develop Multiple Options

[Yes, It's really cheaper!]



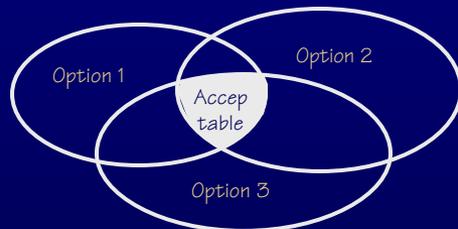
Communicate Constraints

[Not Solutions]



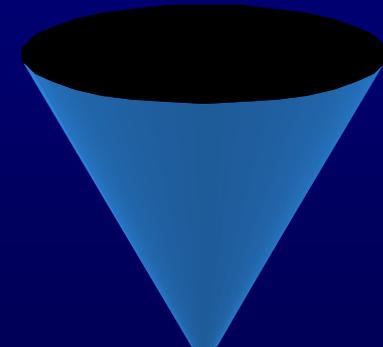
Look for Intersections

[Stick With Decisions]



Narrow Options Gradually

[The Solution Emerges]



4. DELIVER AS FAST AS POSSIBLE

The Dell logo, featuring the word "DELL" in white, bold, sans-serif capital letters on a blue rectangular background. The letter "E" is stylized with a diagonal slash.The Compaq logo, featuring the word "COMPAQ" in white, bold, italicized sans-serif capital letters on a red rectangular background.The L.L.Bean logo, featuring the text "L.L.Bean" in a black, serif font on a white rectangular background.The Sears logo, featuring the word "SEARS" in a blue, serif font on a white rectangular background.The LensCrafters logo, featuring a stylized eye icon with a red needle-like element and the word "LENSCRAFTERS" in a blue, serif font on a light blue background.The Eye Clinic logo, featuring a stylized eye icon and the words "EYE CLINIC" in a blue, serif font on a white background.The Airborne Express logo, featuring the words "AIRBORNE EXPRESS" in white, bold, sans-serif capital letters on a red rectangular background with white horizontal stripes at the bottom.The United States Postal Service logo, featuring the eagle head icon and the words "UNITED STATES POSTAL SERVICE" in a blue, sans-serif font on a white background.

THE MEASURE OF MATURITY

FAST & SELF-DIRECTING WORK

- ★ Story Cards or Iteration Feature List

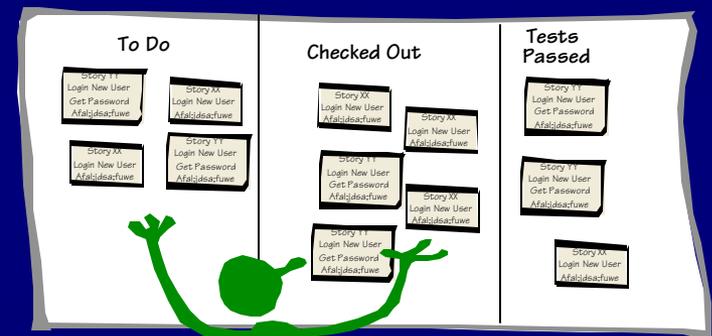
- ★ How do developers know what to do?

- ★ Information Radiators

- ★ White Boards
- ★ Charts on the Wall

- ★ Daily Meetings

- ★ Status
- ★ Commitment
- ★ Need



Software Kanban

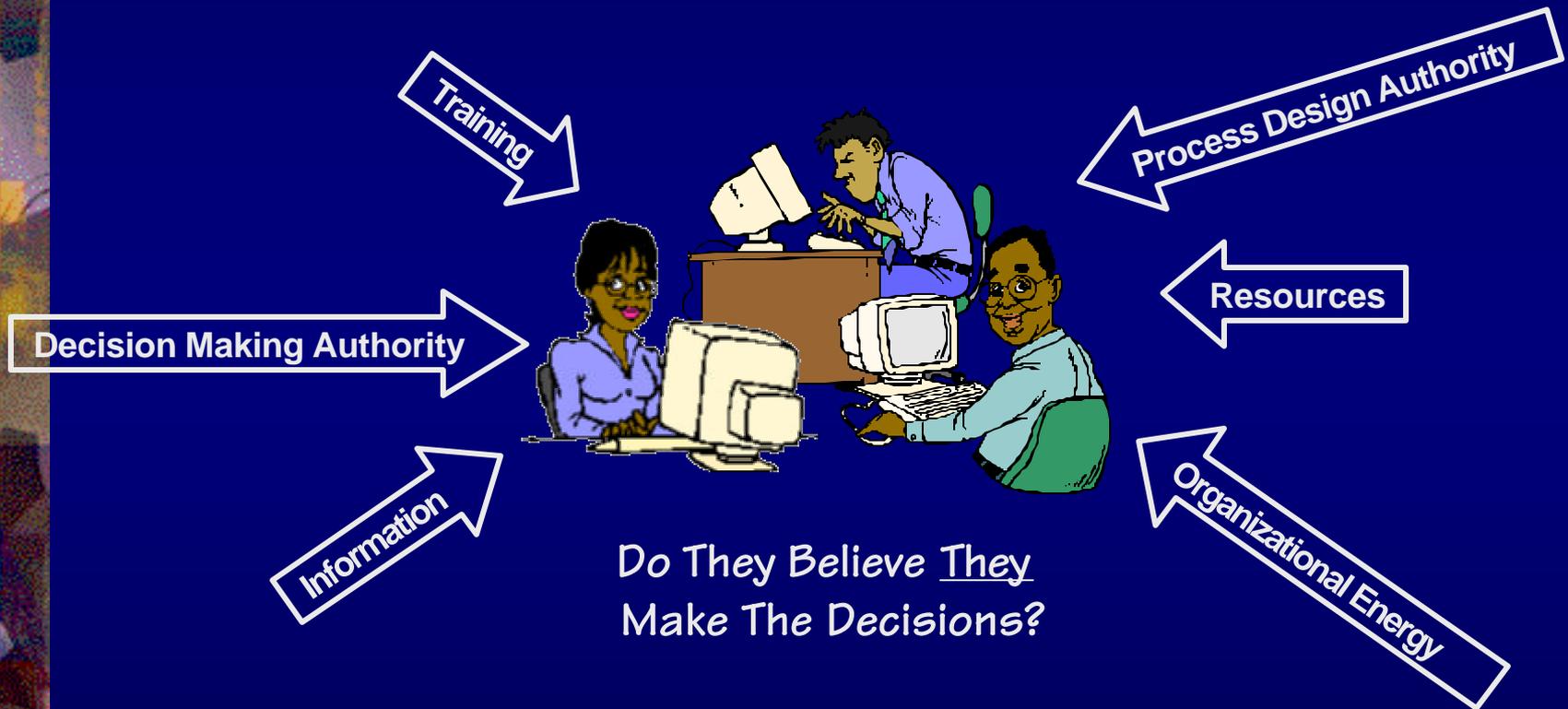
5. EMPOWER THE TEAM

- ✦ 1982 – GM Closed the Fremont, CA Plant
 - ✦ Lowest Productivity
 - ✦ Highest Absenteeism
- ✦ 1983 – Reopened as NUMMI (Toyota & GM)
 - ✦ Same work force
 - ✦ White-collar jobs switch from directing to support
 - ✦ Small work teams trained to design, measure, standardize and optimize their own work
- ✦ 1985
 - ✦ Productivity & quality doubled, exceeded all other GM plants
 - ✦ Drug and alcohol abuse disappeared
 - ✦ Absenteeism virtually stopped
 - ✦ Time to expand the plant



DECIDE AS LOW AS POSSIBLE

★ Who Designs Your Processes?





6. BUILD INTEGRITY IN

★ Perceived Integrity

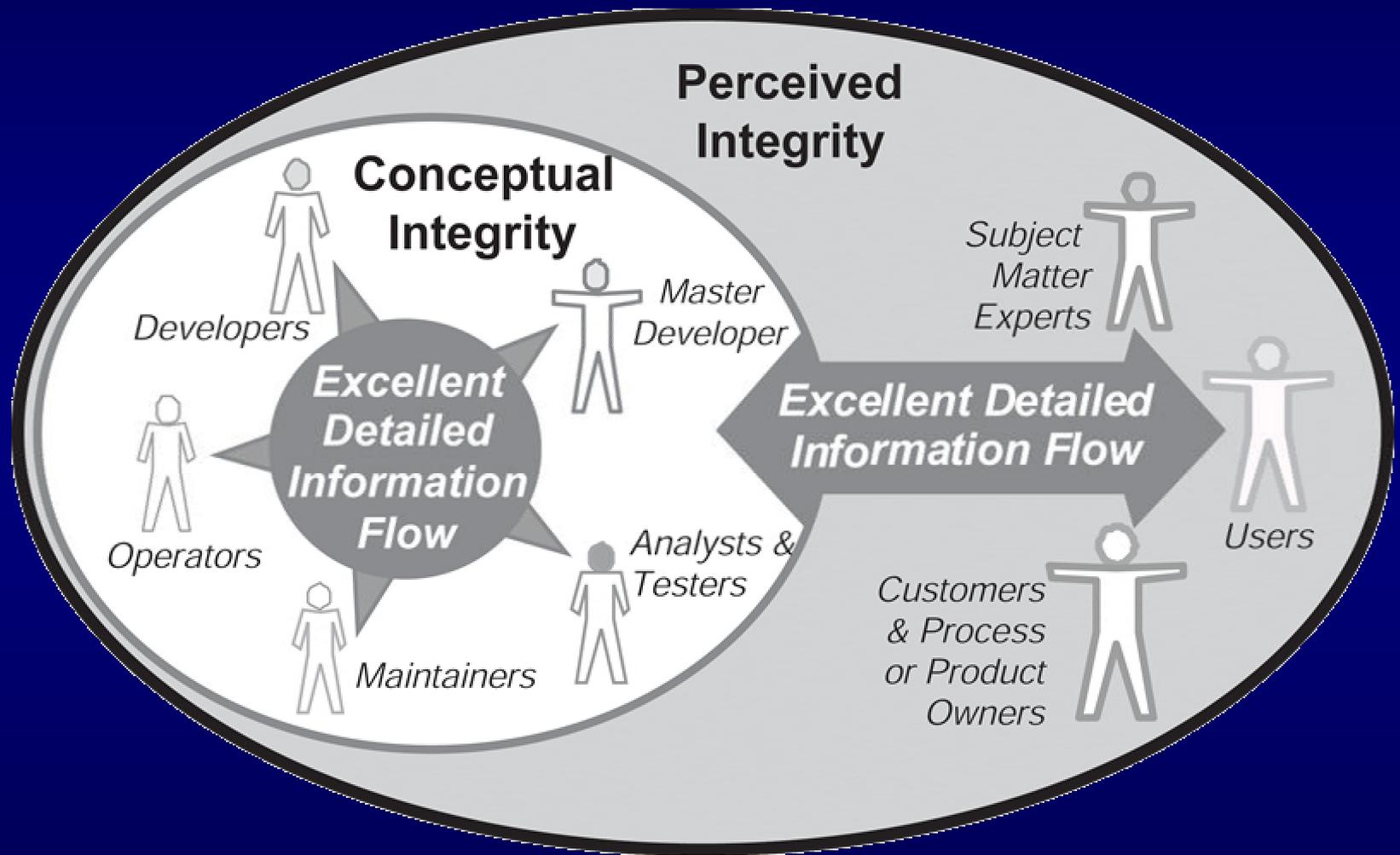
The totality of the system achieves a balance of function, usability, reliability and economy that delights customers.

★ Conceptual Integrity

The system's central concepts work together as a smooth, cohesive whole.

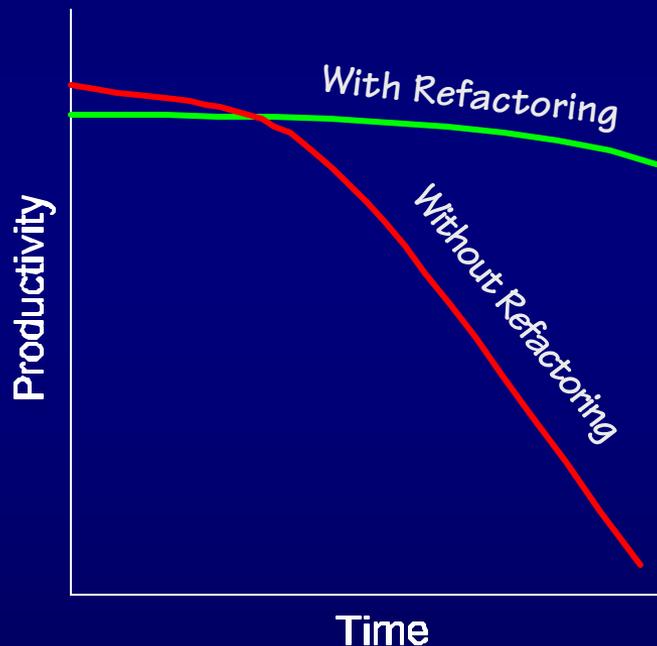
Clark & Fujimoto, *Product Development Performance*, Harvard Business School Press, 1991

KEYS TO PRODUCT INTEGRITY



REFACTORING

Continuous Improvement of the Design



1. Simplicity

- The goal of most patterns

2. Clarity

- Common language
- Encapsulation
- Self-documenting code

3. Suitable for Use

- Usability
- Performance

4. No Repetition

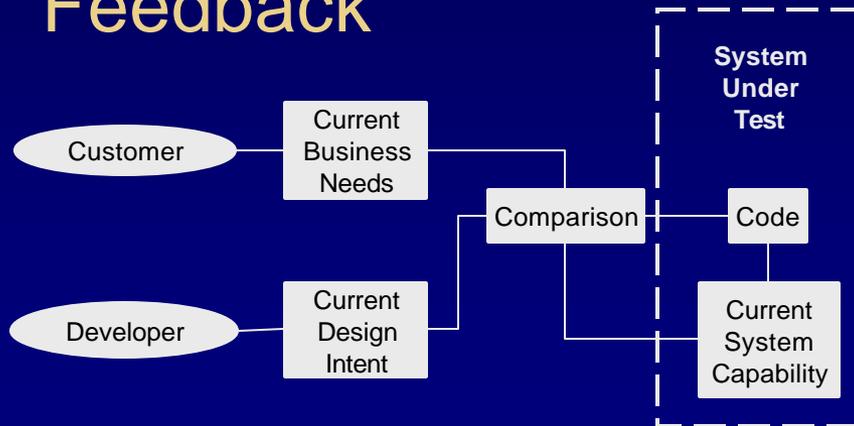
- NO REPITITION!

5. No Extra Features

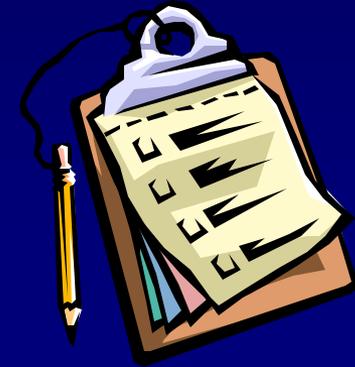
- No Code Before its Time
- No Code After its Time

AGGRESSIVE, AUTOMATED TESTING

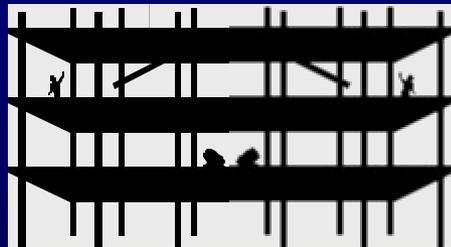
Feedback



Requirements



Scaffolding

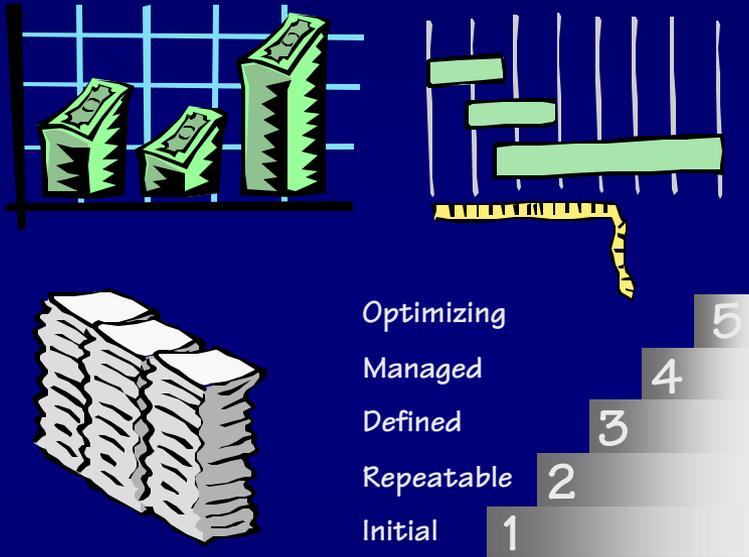


As-Built

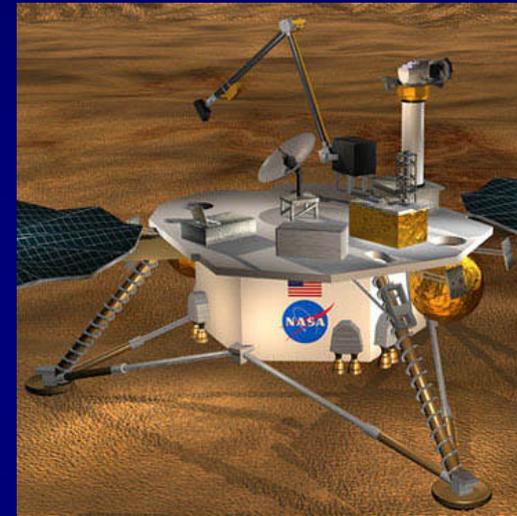


7. SEE THE WHOLE

Sub-Optimizing



What Matters



YOU GET WHAT YOU MEASURE
MEASURE WHAT MATTERS

CERTIFICATION ¹ IMPROVEMENT

CMM

ISO
9000

Malcolm
Baldrige

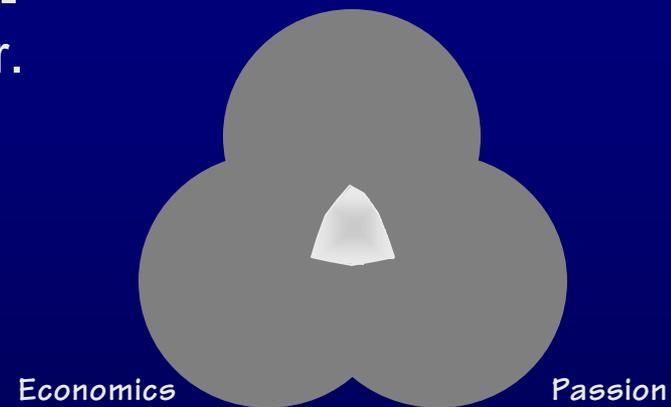
★ Zeos / Micron

- ★ PC Maker that went public in 1990
- ★ Finalist for Malcolm Baldrige Award
- ★ 1995 Merged into Micron Electronics
- ★ Stuck to their intense focus on build-to-forecast processes... a fatal error.

★ Dell

- ★ Focus on high value customers
- ★ View inventory as the greatest risk
- ★ Strong partnerships with suppliers

Best in the World



Jim Collins, *Good to Great*, Harper Business, 2001



THANK YOU!

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

BIBLIOGRAPHY – LEAN THINKING

- Austin**, Robert D. *Measuring and Managing Performance in Organizations*. Dorset House, 1996.
- Christensen**, Clayton M. *The Innovator's Dilemma*. Harvard Business School Press, 2000.
- Clark**, Kim B., & Takahiro Fujimoto. *Product Development Performance: Strategy, Organization, and Management in the World Auto Industry*. Harvard Business School Press, 1991.
- Collins**, Jim. *Good to Great: Why Some Companies Make the Leap...and Others Don't*. HarperBusiness, 2001.
- Dyer**, Jeffrey H. *Collaborative Advantage: Winning Through Extended Enterprise Supplier Networks*. Oxford University Press; 2000.
- Freedman**, David H. *Corps Business*. HarperBusiness, 2000.
- Goldratt**, Eliyahu M. *The Goal: A Process of Ongoing Improvement*, 2nd rev. ed. North River Press, 1992.
- Klein**, Gary. *Sources of Power: How People Make Decisions*. MIT Press, 1999.
- O'Reilly**, Charles A., III, & Jeffrey Pfeffer. *Hidden Value: How Great Companies Achieve Extraordinary Results with Ordinary People*. Harvard Business School Press, 2000.
- Ohno**, Taiichi. *The Toyota Production System: Beyond Large-Scale Production*. Productivity Press, 1988.
- Reinertsen**, Donald G. *Managing the Design Factory: A Product Developer's Toolkit*. Free Press, 1997.
- Smith**, Preston G., & Donald G. Reinertsen. *Developing Products in Half the Time: New Rules, New Tools*, 2nd ed. John Wiley & Sons, 1998.
- Ward**, Allen, Jeffrey K. Liker, John J. Cristaino, & Durward K. Sobek, II. "The Second Toyota Paradox: How Delaying Decisions Can Make Better Cars Faster." *Sloan Management Review* 36(3): Spring 1995, 43–61.
- Womack**, James P., & Daniel T. Jones. *Lean Thinking, Banish Waste and Create Wealth in your Corporation*. Simon and Schuster, 1996.
- Womack**, James P., Daniel T. Jones, & Daniel Roos. *The Machine That Changed the World: The Story of Lean Production*. HarperPerennial, 1991.

BIBLIOGRAPHY – SOFTWARE DEVELOPMENT

- Beck, Kent, & Martin Fowler.** *Planning Extreme Programming*. Addison-Wesley, 2001.
- Beck, Kent.** *Extreme Programming Explained: Embrace Change*. Addison-Wesley, 1999.
- Cockburn, Alistair.** *Agile Software Development*. Addison-Wesley, 2002.
- Cockburn, Alistair.** *Writing Effective Use Cases*. Addison-Wesley, 2000.
- Constantine, Larry, & Lucy Lockwood.** *Software for Use: A Practical Guide to the Models and Methods of Usage-Centered Design*. Addison-Wesley, 1999.
- Cusumano, Michael A., & Richard W. Selby.** *Microsoft Secrets: How the World's Most Powerful Software Company Creates Technology, Shapes Markets, and Manages People*. Simon & Schuster, 1998.
- Evans, Eric.** *Domain Driven Design*. In press, 2003.
- Highsmith, Jim.** *Agile Software Development Ecosystems*. Addison-Wesley, 2002.
- Highsmith, James A.** *Adaptive Software Development: A Collaborative Approach to Managing Complex Systems*. Dorset House, 2000.
- Hohmann, Luke.** *Beyond Software Architecture: Creating and Sustaining Winning Solutions*. In press, 2003.
- Hunt, Andrew, & David Thomas.** *The Pragmatic Programmer: From Journeyman to Master*. Addison-Wesley, 2000.
- Jeffries, Ron, Ann Anderson, & Chet Hendrickson.** *Extreme Programming Installed*. Addison-Wesley, 2001.
- Johnson, Jeff.** *GUI Bloopers: Don'ts and Do's for Software Developers and Web Designers*. Morgan Kaufmann Publishers, 2000.
- Poppendieck, Mary & Tom Poppendieck.** *Lean Software Development: An Agile Toolkit*. Addison-Wesley, 2003.
- Schwaber, Ken, & Mike Beedle.** *Agile Software Development with Scrum*. Prentice Hall, 2001.
- Thimbleby, Harold.** "Delaying Commitment." *IEEE Software* 5(3): May 1988.