



# lean

software development

## Beyond the Low Hanging Fruit

*What Lean Brings to Agile*

# *Lean in a Nutshell*



## **1. System Thinking**

- ✓ Customers don't want software.

## **2. Technical Excellence**

- ✓ Correctness can be proven at any time.

## **3. Reliable Delivery**

- ✓ Design the system to meet the constraints.

## **4. Relentless Improvement**

- ✓ Learn how to learn.



# Delighted Customers



## Customer

– Anyone who pays for, uses, supports, or derives value from the systems we create

### **Basic Needs**

- Quality
  - ✓ Assumed to be there

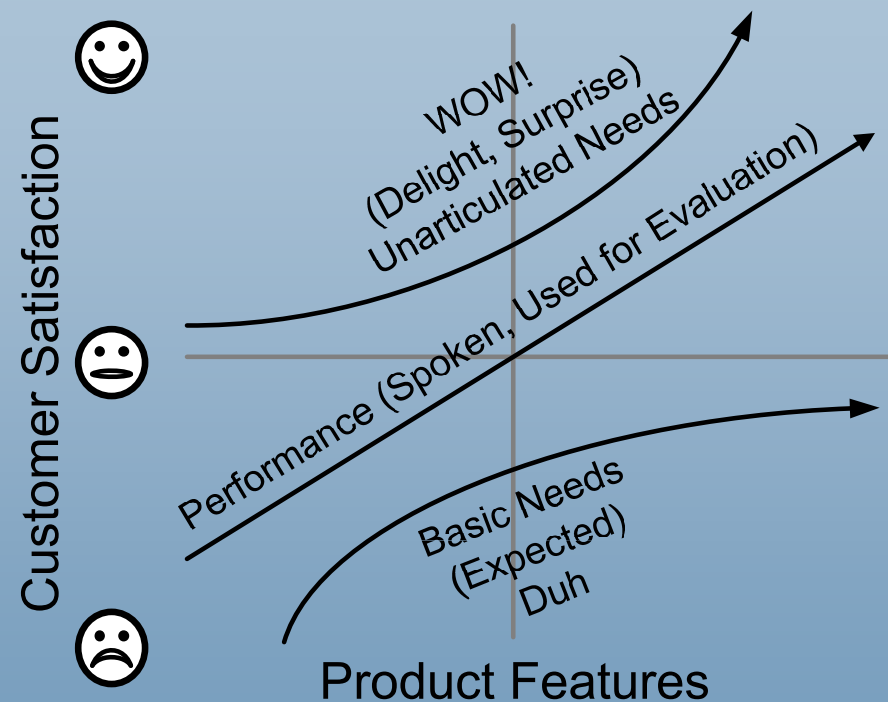
### **Performance Needs**

- Value
  - ✓ Spoken needs
  - ✓ Used for evaluation

### **Unarticulated Needs**

- Excitement
  - ✓ Unexpected features that delight & surprise

## The Kano Model



***Find unarticulated needs.***

# Customer Demand



## Value Demand

- ✓ Demand for work that adds value from a customer perspective
- ✓ The Goal: Delight Customers by Responding to Value Demand



## Failure Demand

- ✓ Demand on the resources caused by your failures
  - ✗ Eg. Support Calls
- ✓ The Goal: Eliminate Failure Demand
  - ✗ Meanwhile, respond as fast as possible



# *Lean in a Nutshell*



## 1. **System Thinking**

- ✓ Customers don't want software.

## 2. **Technical Excellence**

- ✓ Correctness can be proven at any time.

## 3. **Reliable Delivery**

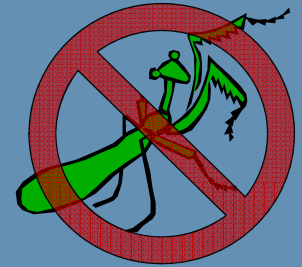
- ✓ Design the system to meet the constraints.

## 4. **Relentless Improvement**

- ✓ Learn how to learn.



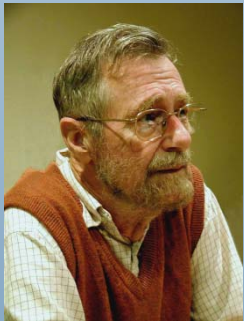
# Technical Excellence



Correctness can be proven at any time.

Every system development process ever invented has had as its primary goal:  
**Find and remove defects as early in the process as you possibly can.**

**Edsger Dijkstra – 1968**



*Those who want really reliable software will discover that they must find a means of avoiding the majority of bugs to start with, and as a result the programming process will become cheaper. If you want more effective programmers, you will discover that they should not waste their time debugging – they should not introduce bugs to start with.*

**Harlan Mills – 1972**

*My principle criterion for judging whether top down programming is actually used is [the] absence of any difficulty at integration.*



**Agile Development is nothing more (and nothing less) than being able to prove correctness at any time, at any level of the system.**

# *Quality by Construction*



## *A Quality Process Builds Quality IN.*

- ✓ Rather than trying to test quality in later.
- ✓ If you find defects at the end of your process...
- ✓ Your process is defective!

## *Quality by Construction*

- ✓ Code that reveals its intentions
- ✓ Design/code reviews
- ✓ Immediate, automated testing
- ✓ **STOP** if the tests don't pass!
- ✓ Continuous, nested integration
- ✓ Escaped defect analysis & feedback



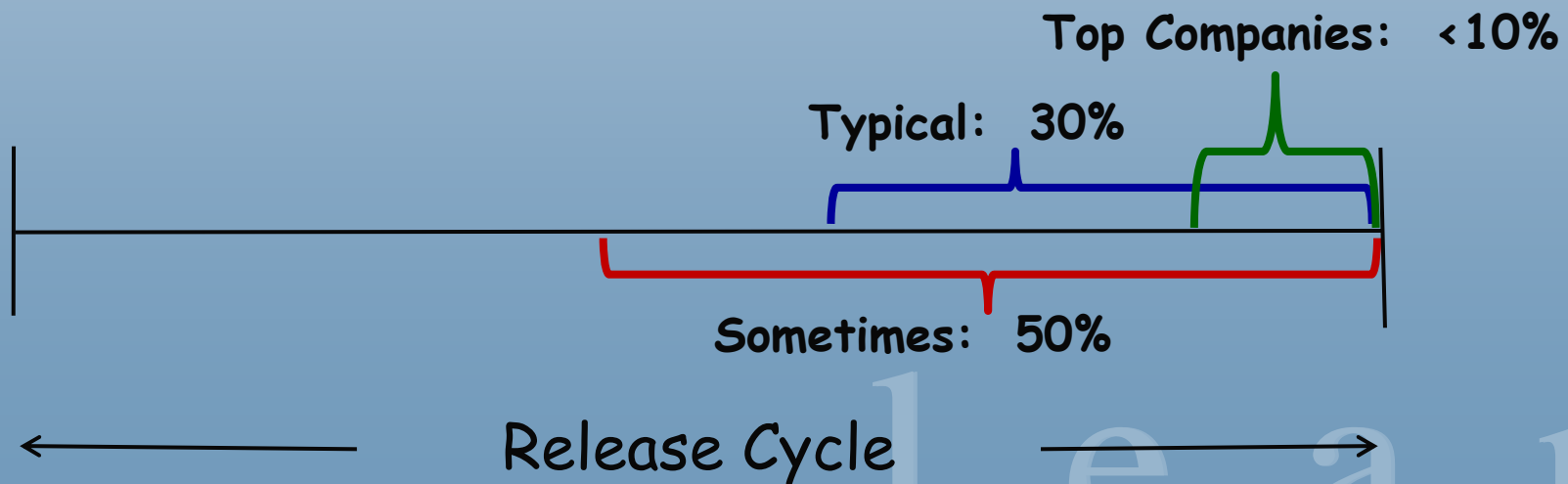
# Dijkstra's Challenge



*If you want more effective programmers, you will discover that they should not waste their time debugging – they should not introduce bugs to start with.*

## *How good are you?*

When in your release cycle do you try to freeze code and test the system?  
What percent of the release cycle remains for this “hardening”?



# *Lean in a Nutshell*



## 1. **System Thinking**

- ✓ Customers don't want software.

## 2. **Technical Excellence**

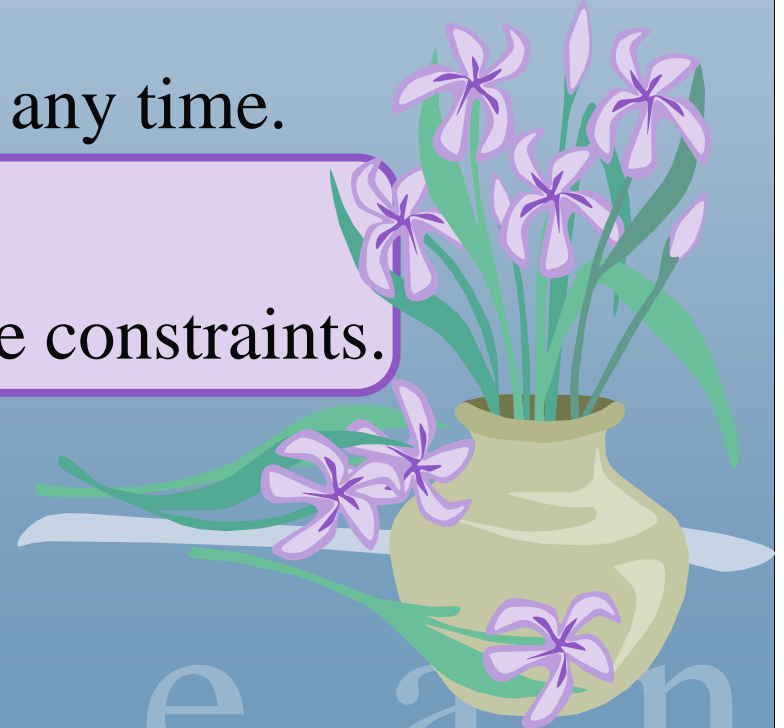
- ✓ Correctness can be proven at any time.

## 3. **Reliable Delivery**

- ✓ Design the system to meet the constraints.

## 4. **Relentless Improvement**

- ✓ Learn how to learn.



# *Empire State Building*



September 22, 1929  
Demolition started  
January 22, 1930  
Excavation started  
March 17, 1930  
Construction started  
November 13, 1930  
Exterior completed  
May 1, 1931  
Building opened  
Exactly on time  
18% under budget



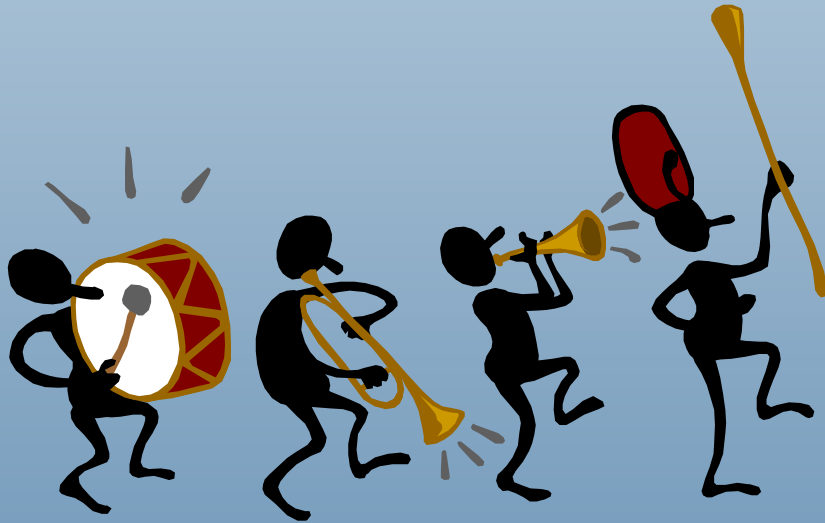
One Year Earlier:



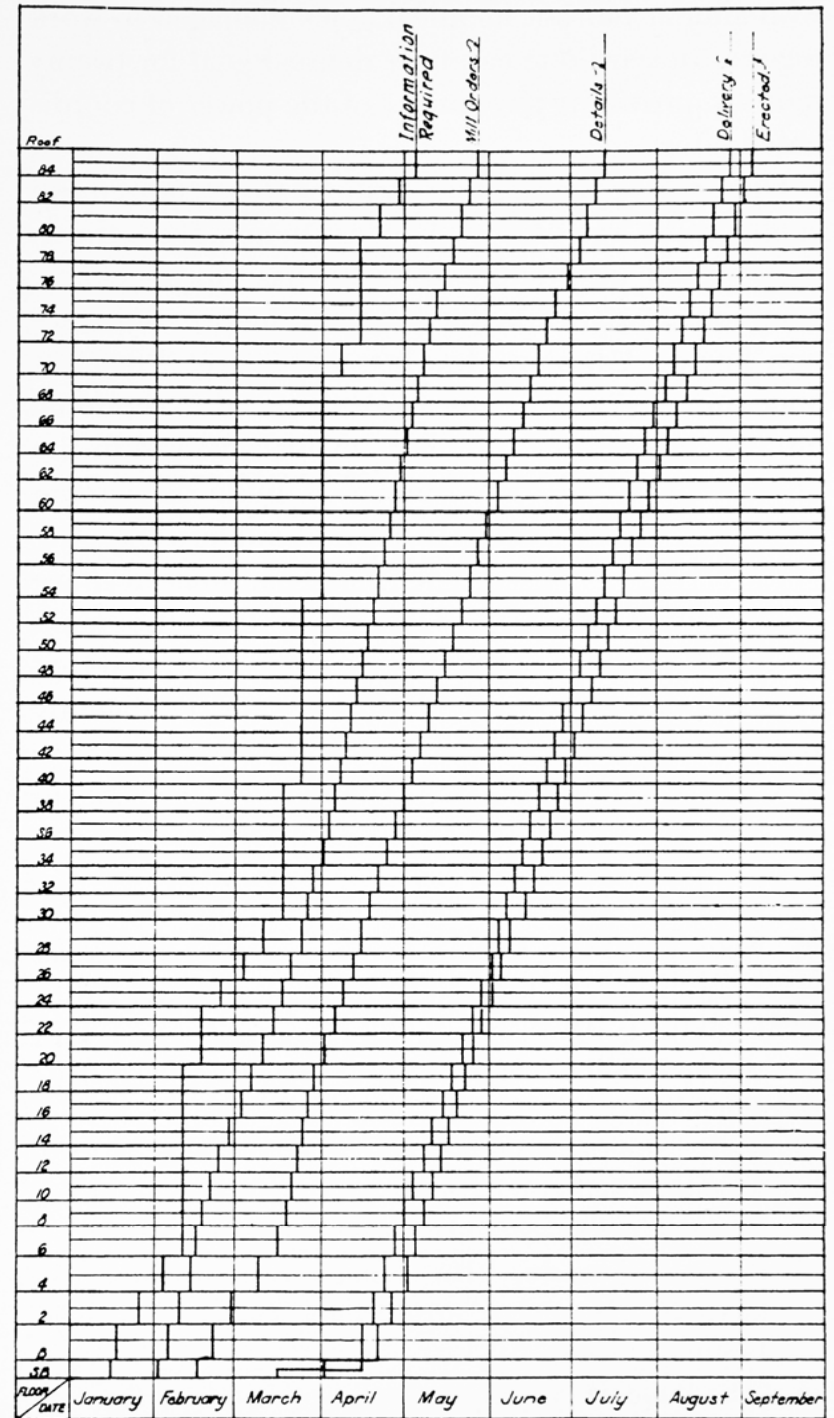
**How did they do it? The key: Focus on FLOW.**

# Steel Schedule

We thought of the work as if it were a band marching through the building and out the top.



From: "Building the Empire State"  
Builders Notebook: Edited by Carol Willis



# *The Four Pacemakers*



1. Structural Steel Construction
  - ✓ Completed September 22, 12 days early
2. Concrete Floor Construction
  - ✓ Completed October 22, 6 days early
3. Exterior Metal Trim & Windows
  - ✓ Completed October 17, 35 days early
4. Exterior Limestone
  - ✓ Completed November 13, 17 days early



From: "Building the Empire State"  
Builders Notebook: Edited by Carol Willis

# *Key Success Factors*

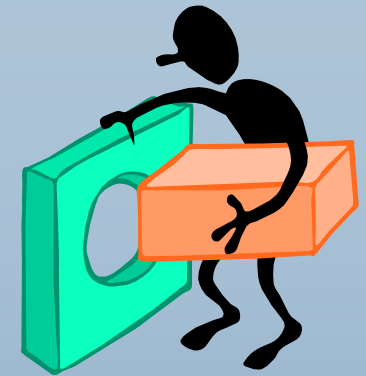


- 1. Teamwork of owner, architect, and builder***
  - ✓ Eliminated design loops by consulting experts early.
- 2. Deeply Experienced Builders***
  - ✓ Fixed Price Contract!
- 3. Focus on the key constraint: Material Flow***
  - ✓ 500 trucks a day – no storage on site
- 4. Decoupling***
  - ✓ The pacemakers (and other systems) were *designed* to be independent
- 5. Cash Flow Thinking***
  - ✓ Every day of delay cost \$10,000 (\$120,000 today).
- 6. Schedule was not laid out based on the details of the building design, the building was designed based on the constraints of the situation.***
  - ✓ Two acres of land in the middle of New York City, zoning ordinances, \$35,000,000 of capital, the laws of physics, and a May 1, 1931 deadline.

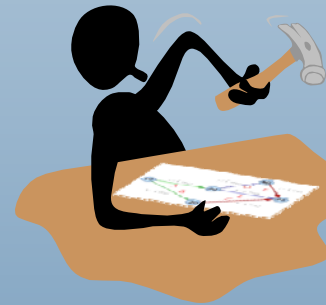
# *Lessons*



Design the system to meet the constraints;  
do not derive constraints from the design.



Decouple workflows;  
break dependencies!



Workflows are easier to control &  
more predictable than a schedules.



# *Lean in a Nutshell*



## 1. **System Thinking**

- ✓ Customers don't want software.

## 2. **Technical Excellence**

- ✓ Correctness can be proven at any time.

## 3. **Reliable Delivery**

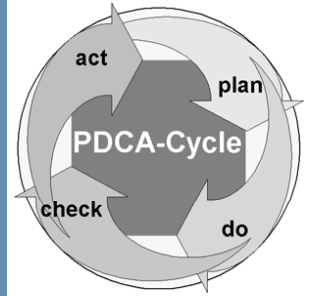
- ✓ Design the system to meet the constraints.

## 4. **Relentless Improvement**

- ✓ Learn how to learn.

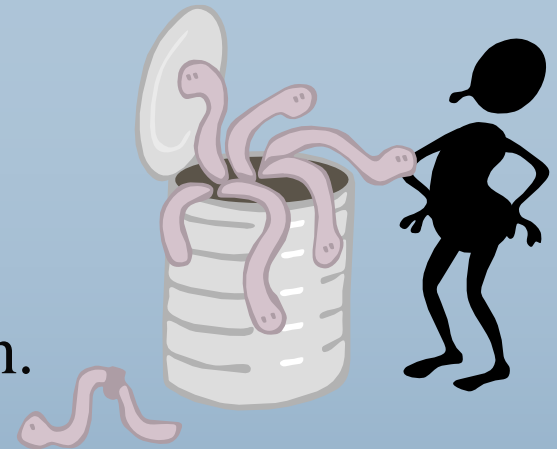


# Failure is a Learning Opportunity



Complex systems will always fail

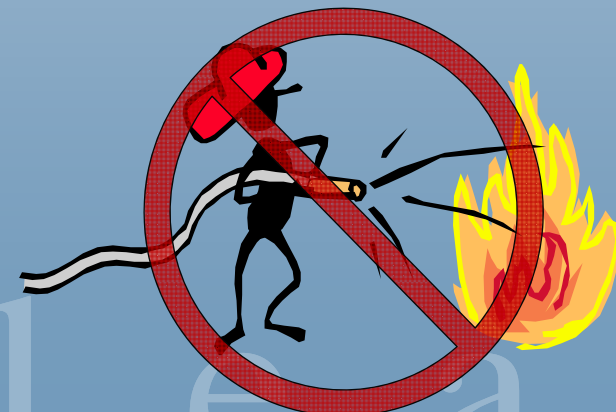
- ✓ A failure demonstrates a lack of understanding of the system.
- ✓ Failure is the system talking to you.
- ✓ **Listen** – It's not noise; it's information.



*No Workarounds*



*No Firefighting*



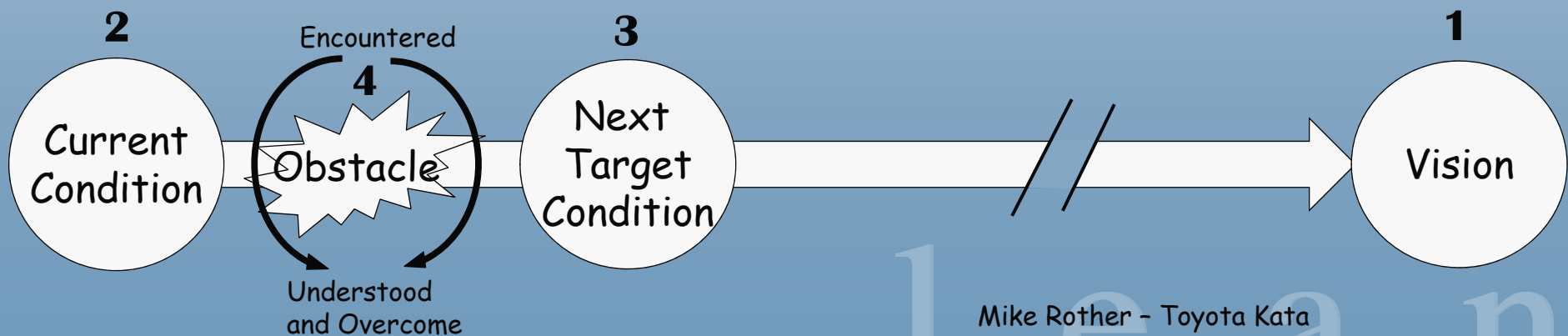
*Learn!*

Learn

# *Toyota Improvement Kata*



1. Visualize perfection
2. Have a first hand grasp of the situation
3. Define a target condition on the way to perfection
4. Strive to move step-by-step to the target
5. As obstacles are encountered, they are systematically understood and overcome



Mike Rother - Toyota Kata

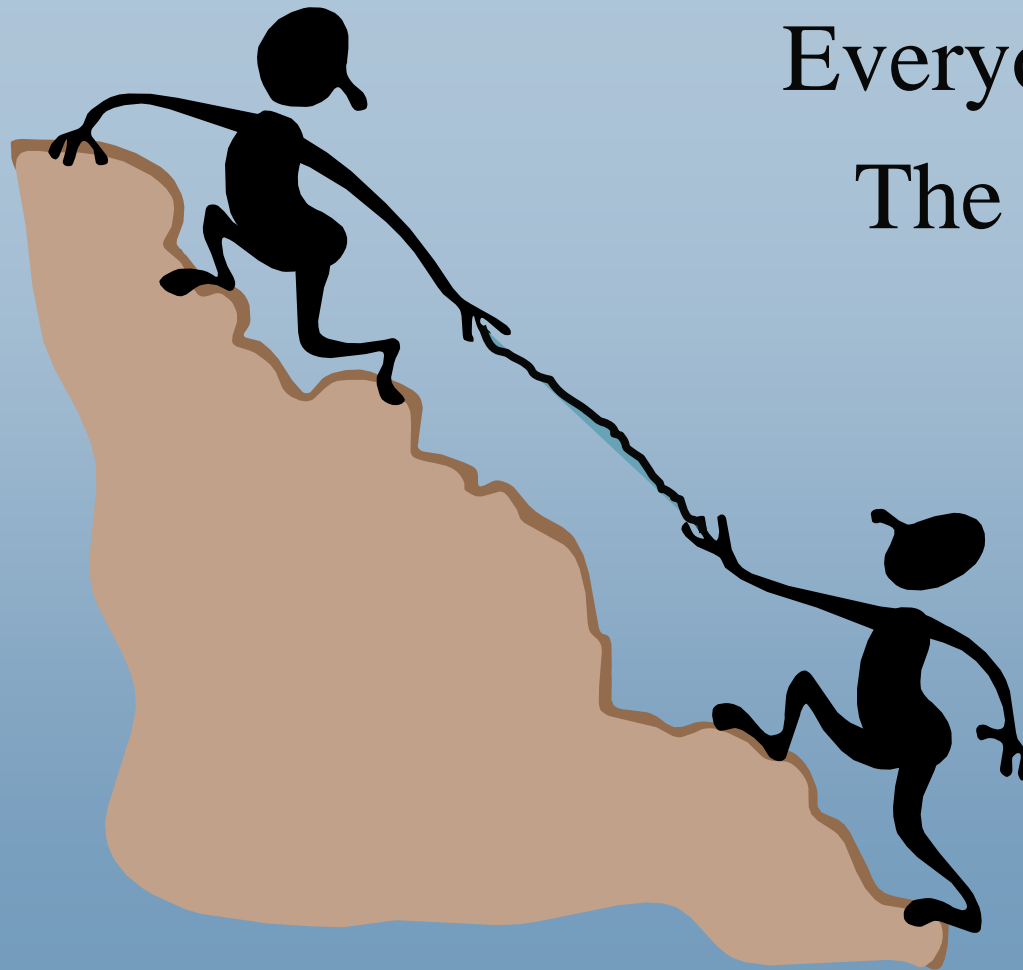
# *Toyota Coaching Kata*



Everyone has a Mentor

The Mentor

- ✓ Knows the details
- ✓ Asks questions
- ✓ Teaches the kata
- ✓ Focuses on learning
  - ✗ Not results



# Results are Not the Point



- ✓ Developing people so that they can achieve successful results is the point.
- ✓ *The Toyota Way has two main pillars: continuous improvement and respect for people.*
- ✓ *Only after American carmakers had exhausted every other explanation for Toyota's success – an undervalued yen, a docile workforce, Japanese culture, superior automation – were they finally able to admit that Toyota's real advantage was its ability to harness the intellect of 'ordinary' employees.*

Katsuaki Watanabe  
Toyota President  
2005 - 2009

"Management Innovation" by Gary Hamel,  
*Harvard Business Review*, February, 2006



# l e a n

software development

## Thank You!

*More Information: [www.poppendieck.com](http://www.poppendieck.com)*