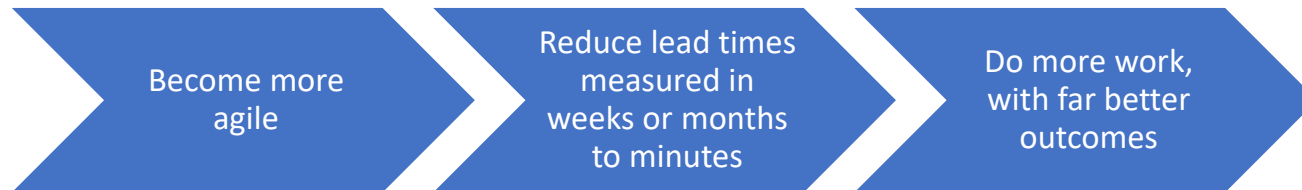


# DevOps

The Phoenix Project

# Why

- Because technology has become the dominant value creation process and an increasingly important (and often the primary) means of customer acquisition within most organizations.



Imagine

- Product owners, Development, QA, IT Operations, and InfoSec work together relentlessly to help each other and the overall organization win
  - enabling fast flow of planned work into production
  - preserving world-class stability, reliability, availability, and security

- ensure that work flows smoothly through the entire value stream,
- Speed up automated tests,
- improve deployment infrastructure,
- Ensure all applications create useful production telemetry.

---

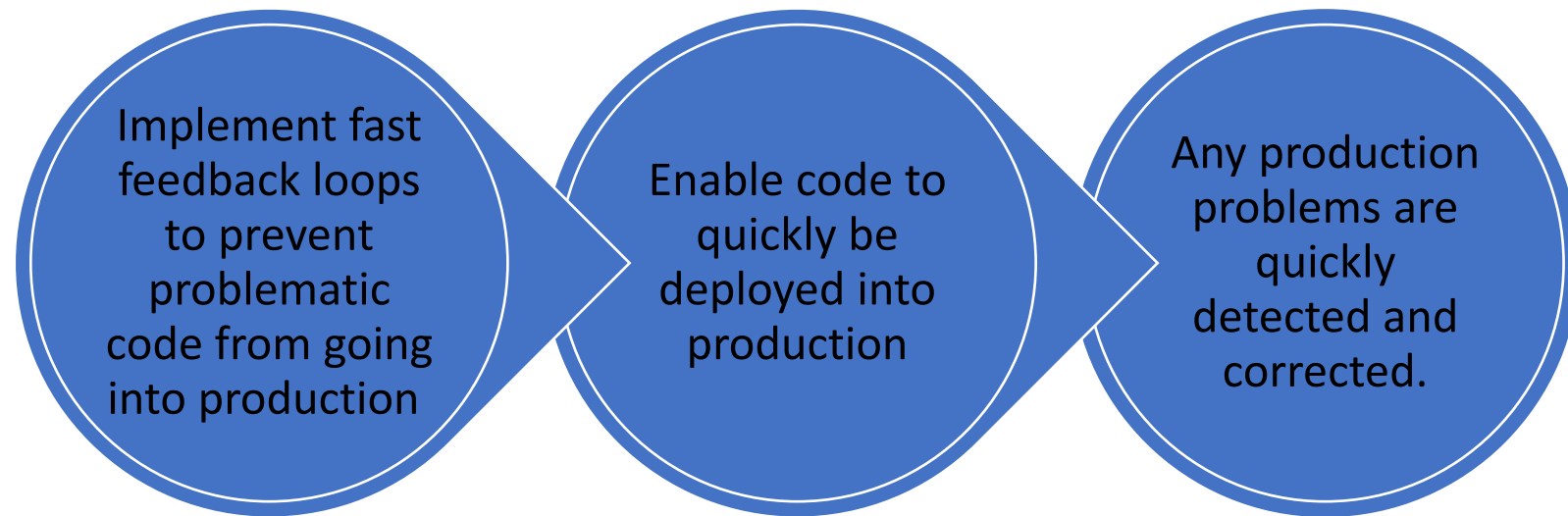
# Developers



20%

of time to...

# Feedback



# Nonfunctional value

---

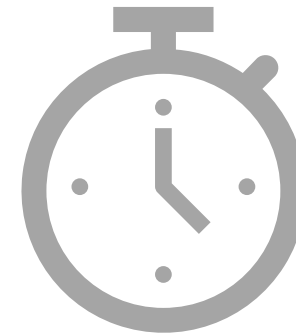


- Everyone values nonfunctional requirements as much as features. quality,
  - scalability,
  - manageability,
  - security,
  - operability

# hypothesis-driven culture



no assumptions for granted



doing nothing without  
measuring

“We don’t spend years building features that our customers don’t actually want, deploying code that doesn’t work, or fixing something that isn’t actually the problem”

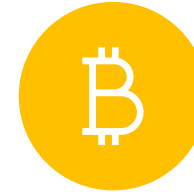
# Monitoring



developers are constantly getting fast feedback on their work: when they write code



automated unit, acceptance, and integration tests are constantly being run in production-like environments, giving us continual assurance that the code and environment will operate as designed



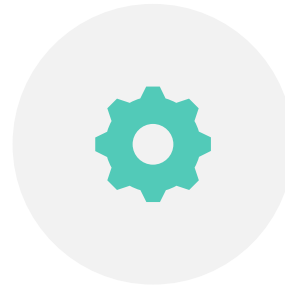
pervasive production metrics demonstrate to everyone that it is working, and the customer is getting value.



# Feature Flags



Months prior to the launch, Development has been deploying code into production, invisible to the customer, but enabling the feature to be run and tested by internal staff.



no new code is pushed into production. Instead, we merely change a feature toggle or configuration setting




Only when we have confidence that the feature is working as designed do we expose it to the next segment of customers,




experiments in production, testing our business hypotheses for every feature we build

# Origin

The principles behind DevOps work patterns are the same principles that transformed manufacturing



break the conflict by adopting Lean principles, such as reducing batch sizes, reducing work in process, and shortening and amplifying feedback loops. This resulted in dramatic increases in plant productivity, product quality, and customer satisfaction. (Toyota Production System, Lean Manufacturing, Theory of Constraints, Six Sigma, and so forth.)



optimize the IT value stream, converting business needs into capabilities and services that provide value for our customers

# Three Ways



left-to-right flow of work



flow of fast feedback from right-to-left



culture that fosters

continual experimentation  
repetition and practice is the prerequisite  
to mastery.

# First Way left-to-right flow of work

- To maximize flow
  - small batch sizes
  - intervals of work
  - never passing defects to downstream work centers
  - to constantly optimize for the global goals

## Second Way flow of feedback right-to-left

- amplifying it to ensure that we can prevent problems from happening again
- enable faster detection and recovery
- create quality at the source, creating or embedding knowledge where we need it
- elevating the improvement of daily work over daily work
- fast automated test suites to ensure that code is always in a potentially deployable state
- creating **shared goals** and shared pain
- creating pervasive production telemetry

# Third Way Culture

- Experimentation and risk taking are what enable us to relentlessly improve our system of work
  - do things very differently than how we've done it
- when things go wrong,
  - build skills and habits
    - constant repetition
    - daily practice
  - Enablement to retreat back to a place of safety
  - Resume normal operations
- high trust

# 4 Types of Work

## BUSINESS PROJECTS

- business initiatives, of which most Development projects encompass

## INTERNAL IT PROJECTS

- infrastructure or IT Operations projects & improvement projects
- problem when IT Operations is a bottleneck, because there is no easy way to find out how much of capacity is already committed to internal projects.

## CHANGES

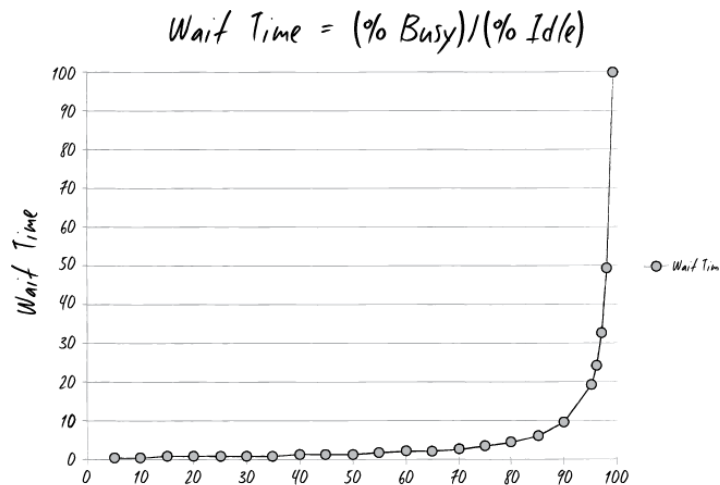
- generated from the previous two types of work and are typically tracked in a ticketing system
- two systems exist to track work for two different parts of the value stream can create problems

## UNPLANNED WORK OR RECOVERY WORK

- operational incidents and problem

# Wait Time

---



- on the x-axis is the percent busy for a given resource at a work center,
- the y-axis is the approximate wait time (or maybe more precisely stated, the queue length).
- resource **utilization goes past eighty percent, wait time goes through the roof**





# Wait Time

---

- 'percentage of time busy' divided by the 'percentage of time idle'
- $\frac{\textit{percentage of time busy}}{\textit{percentage of time idle}}$
- if a resource is ninety percent busy
  - the wait time is 'ninety percent divided by ten percent'
  - $\frac{90\% \textit{ busy}}{10\% \textit{ idle}} = 9 \textit{ unit of time}$
  - 1 unit of time = 1 hour
  - **nine hours**

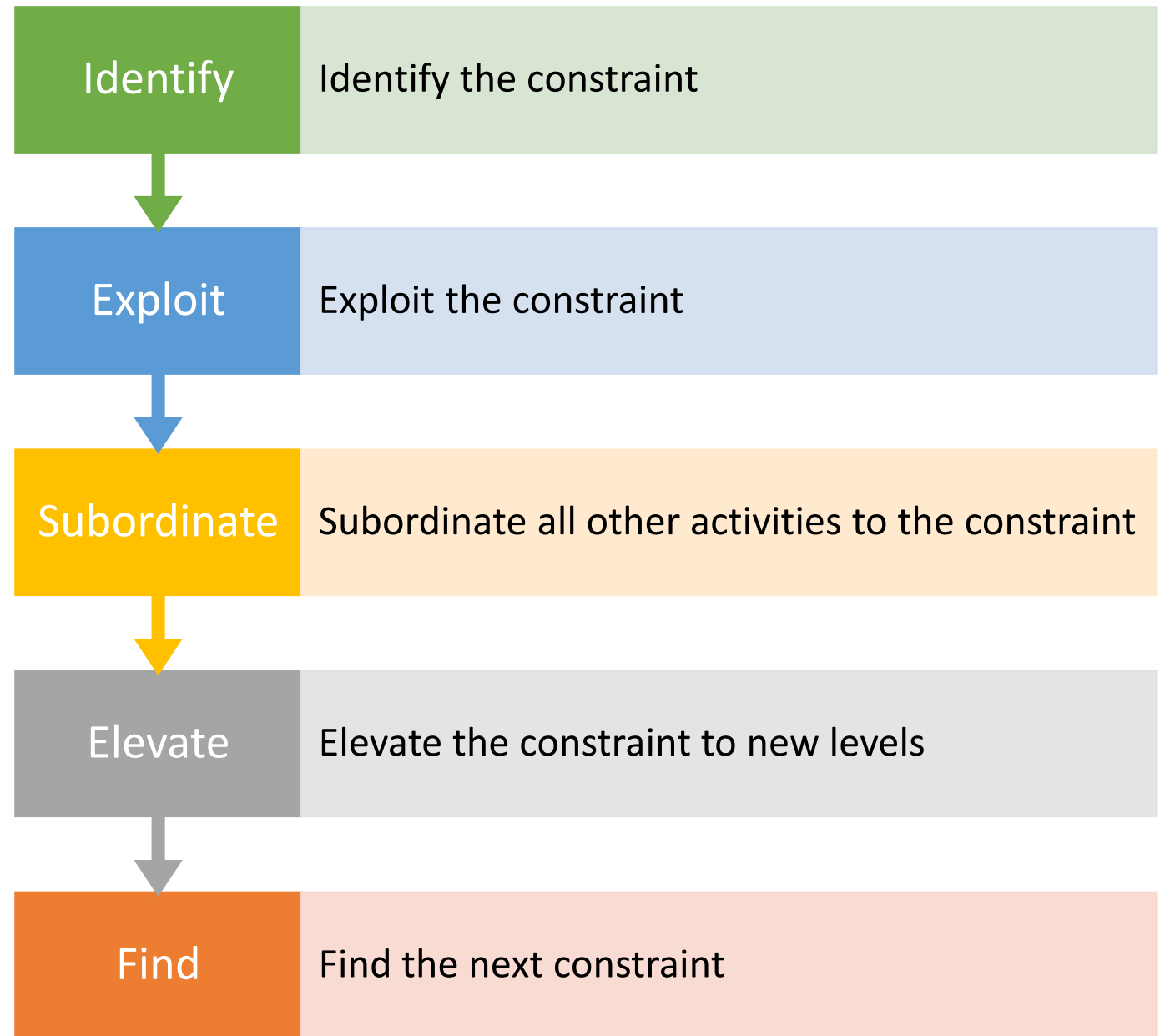


# touch time

---

- % of value added time
  - 'task time divided by total wait time'
  - $\frac{\text{task time unit}}{\text{total wait time}}$
- Assuming
  - seven handoffs
  - thirty-minute task
  - each resources is busy ninety percent of the time
  - the tasks would spend in queue a total of **63 hours**
    - nine hours times the seven steps
  - touch time of 0.79%
    - thirty minutes divided by sixty-three hours
    - $\frac{0.5 \text{ hrs}}{63 \text{ hrs}} = 0.0079 \approx 0.79\%$
  - 99.2% of total lead time
- the work sits in queue 99 % of the time

# Theory of Constraints



# Five Dysfunctions of a Team: A Leadership Fable

one of the core contributors to a team's inability to achieve goals is due to lack of trust

- Absence of trust

unwilling to be vulnerable within the group

- Fear of conflict

seeking artificial harmony over constructive passionate debate

- Lack of commitment

feigning buy-in for group decisions creates ambiguity throughout the organization

- Avoidance of accountability

ducking the responsibility to call peers on counterproductive behavior, which sets low standards

- Inattention to results

focusing on personal success, status, and ego before team success

# Managing People for Improvement, Adaptiveness and Superior Results

- frames the thought process and culture that must exist to enable the Lean PDCA cycle (Plan, Do, Check, Act)
- two-week improvement cycle
- daily repetition in order to create habits

## *Toyota Kata*

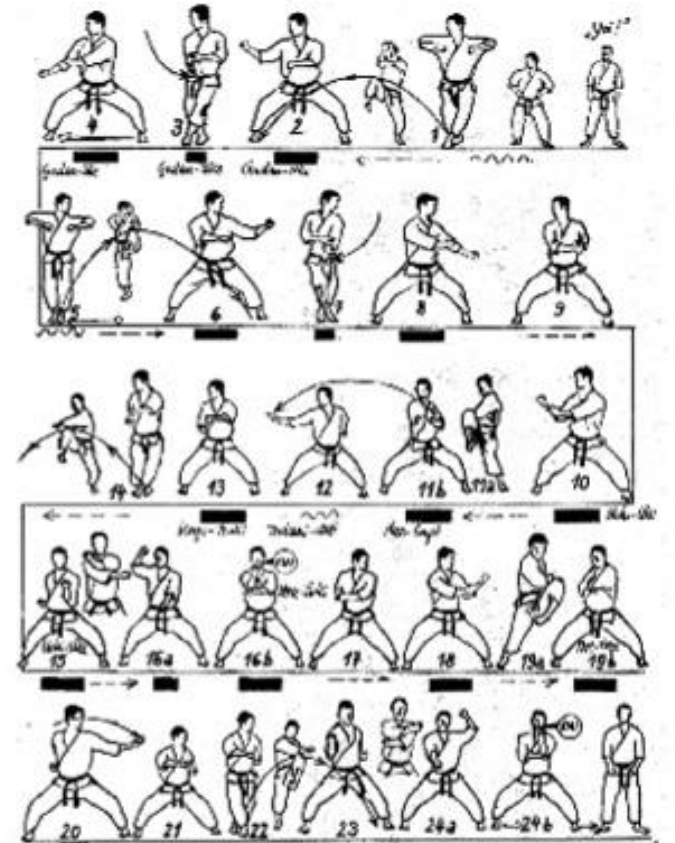
Mobilizing our ingenuity through good management

Mike Rother, February 2010

The practice of kata is the act of practicing a pattern so it becomes second nature.

In its day-to-day management Toyota teaches a way of working -- a *kata* -- that has helped make it so successful over the last six decades.

Toyota's Improvement Kata is something we overlook in benchmarking and should learn more about in order to understand Toyota's story.



Copyright © 2010 Mike Rother, all rights reserved

1217 Baldwin Avenue / Ann Arbor, MI 48104, USA / tel: (734) 265-5411 / mrother@umich.edu

# Continuous Delivery: Reliable Software Releases through Build, Test, and Deployment Automation



importance of the performance of the entire system, as opposed to the performance of a specific silo of work or department



Continuous delivery is the extension of continuous integration,

which are the Development practices that include continuous builds, continuous testing, daily integration of branches back into trunk, testing in a clone of the production environment, etc.



Continuous delivery techniques extend these processes all the way into the production environment.

# Release It

- How to build applications that can be deployed and managed and survive in even the most hostile production environments.

## a Drum-Buffer-Rope Solution

- Too long to finish work requested from the business:
  - average lead time was 155 days
- Dissatisfaction with lateness and long lead times forced IT management to do “more work estimation,” forcing managers to spend all their time building PowerPoints (because business conclusion was that they didn’t do a good job estimating), instead of doing real work
- Whenever the business asked for anything, the response was “it’ll take 5 months”
- Every task was estimated at 20 days, but no one knew where the other 135 days went
- “Waiting For Dragos,”
  - capture when any work was blocked.
  - seventy percent of all the team’s time was blocked on other people
  - **seventy percent of the time the work was in queue.**



<b>Title</b>	<b>Author</b>	<b>ISBN 13</b>
The Logical Thinking Process: A Systems Approach to Complex Problem Solving	H. William Dettmer	978-0873897235
The Goal: A Process of Ongoing Improvement	Eliyahu M. Goldratt	978-0884271956
Continuous Delivery: Reliable Software Releases Through Build, Test, and Deployment Automation	Jez Humble, David Farley	978-0321601919
Toyota Kata: Managing People for Improvement, Adaptiveness and Superior Results	Mike Rother	978-0071635233
Release It!: Design and Deploy Production-Ready Software	Michael T. Nygard	978-1680502398
The Five Dysfunctions of a Team: A Leadership Fable	Patrick Lencioni	978-0787960759