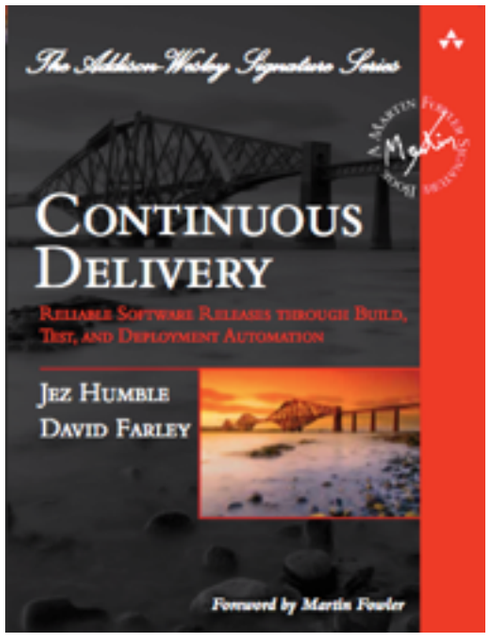
ThoughtWorks®

devops culture and practices
to create flow

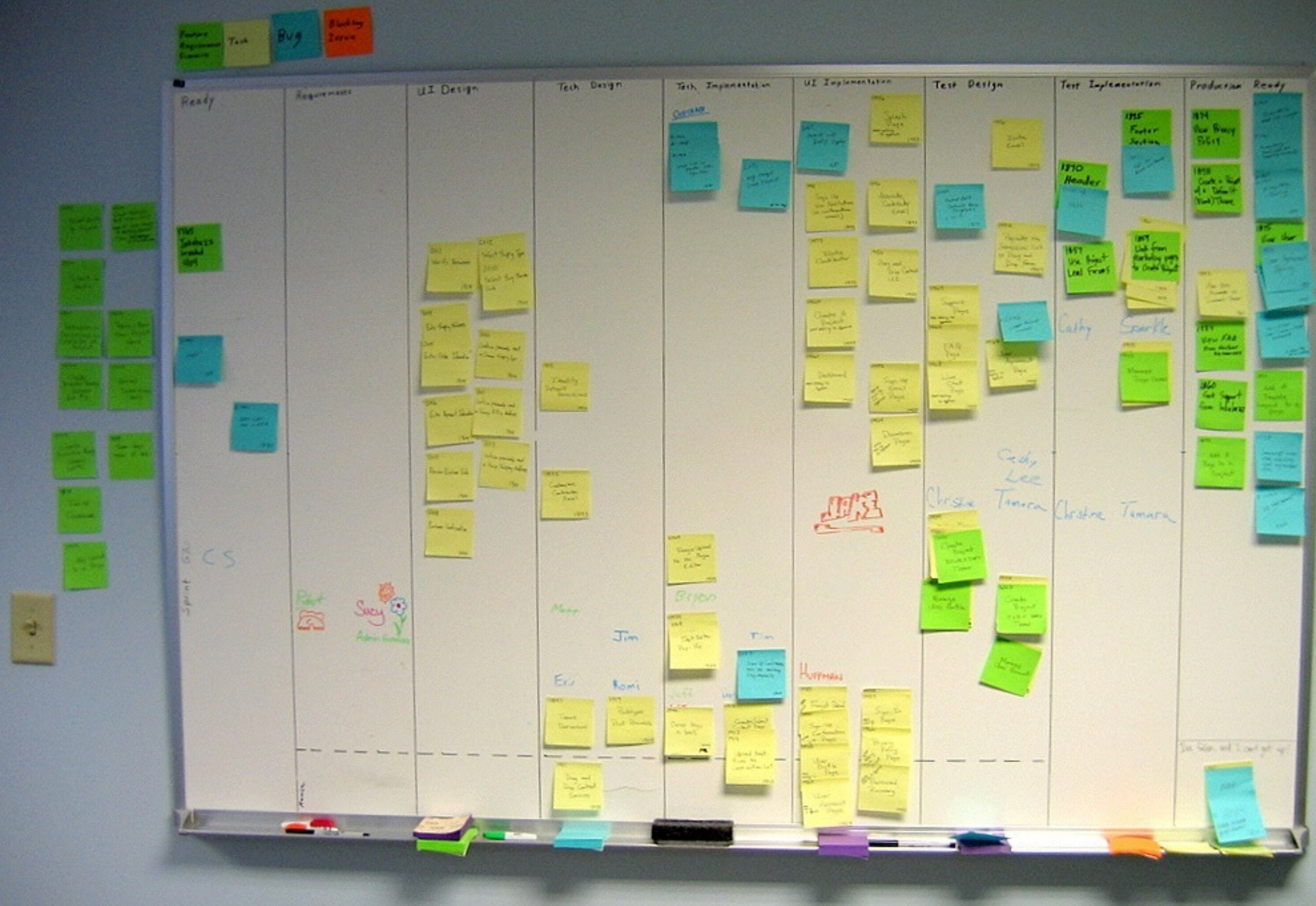


@jezhumble
#QConNewYork
11 june 2014

the production line

http://www.flickr.com/photos/toyotauk/4711057997/

the production line?



https://www.flickr.com/photos/chrishuffman/2336990347/

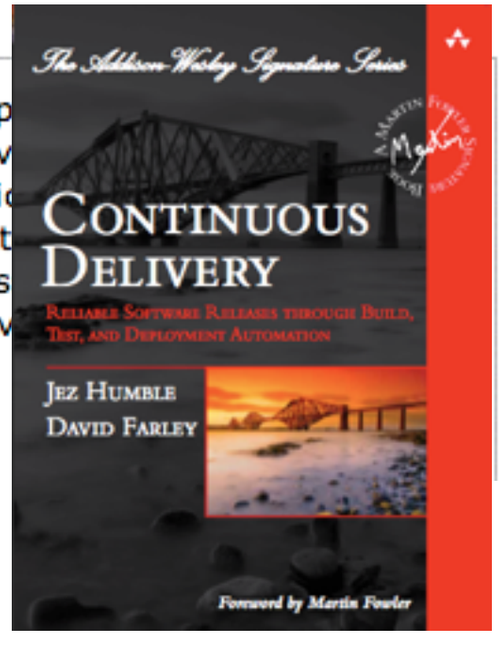
The Deployment Production Line

Authors: Jez Humble ThoughtWorks Limited
Chris Read ThoughtWorks Limited
Dan North ThoughtWorks Limited

Published in:
· Proceeding
AGILE '06 Proceedings of the conference on AGILE 2006
Pages 113 - 118
IEEE Computer Society Washington, DC, USA ©2006
table of contents ISBN:0-7695-2562-8 doi>10.1109/AGILE.2006.53

2006 Article

Testing and deployment can be a difficult and timeconsuming process in complex environments comprising app
messaging infrastructure and interfaces to external systems. We have seen deployments take several days, ev
have used automated builds to ensure their code is fully tested. In this paper we describe principles and practic
environments to be created, configured and deployed to at the click of a button. We show how to fully automat
deployment process using a multi-stage automated workflow. Using this "deployment production line", it is pos
tested code into production environments quickly and with full confidence that you can fall back to a previous v
problem occur.



toyoda automatic loom, type g

“Since the loom stopped when a problem arose, no defective products were produced. This meant that a single operator could be put in charge of numerous looms, resulting in a tremendous improvement in productivity.”

http://www.toyota-global.com/company/vision_philosophy/toyota_production_system/jidoka.html

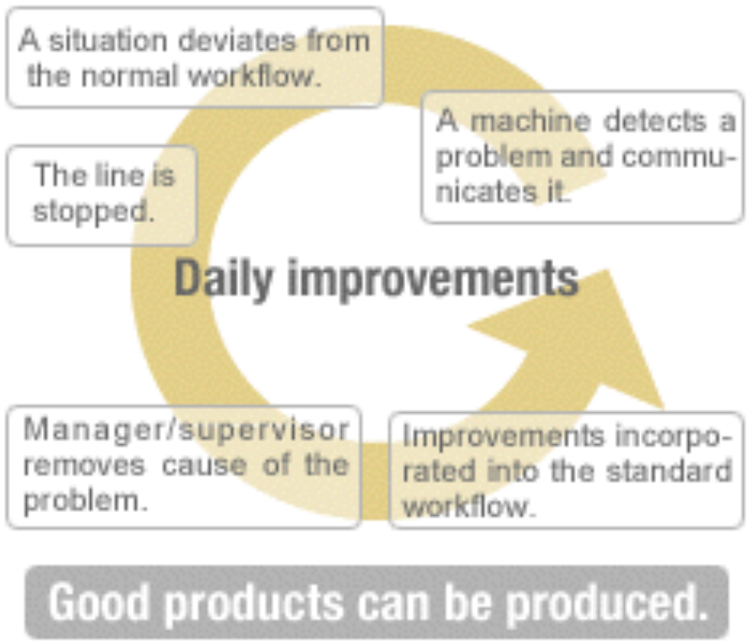
jidoka

自動化 + 人 = 自働化

automation + people
= autonomation

jidoka

Concept of jidoka



Visual Control using Andon

An operator communicating an abnormality

An andon problem display board that communicates abnormalities

http://www.toyota-global.com/company/vision_philosophy/toyota_production_system/jidoka.html

Local
Workstation

Mainline Server

pull

push

Done!

Local
Workstation

Mainline Server

Everyone Commits To the Mainline Every Day

pull

push

Done!

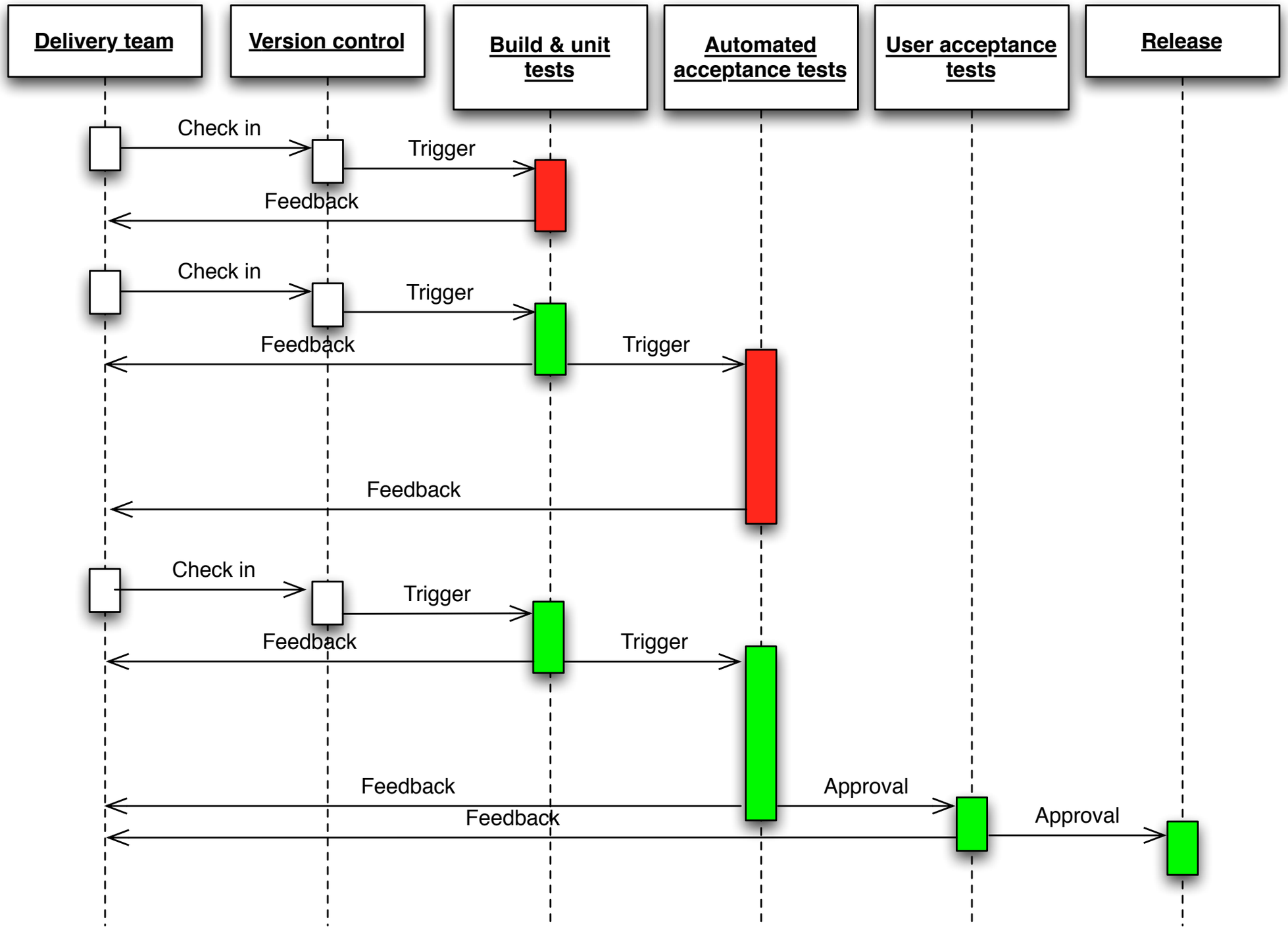
Google

Google Scale

- 10000+ developers in 40+ offices
- 2000+ projects under development
- Single monolithic code tree
- 20+ code changes submitted per minute - peaks at 60
- 50% of code changes every month
- Everyone develops and releases from head
- All builds from source
- >100 million test cases executed per day

www.infoq.com/presentations/google-test-automation

deployment pipeline

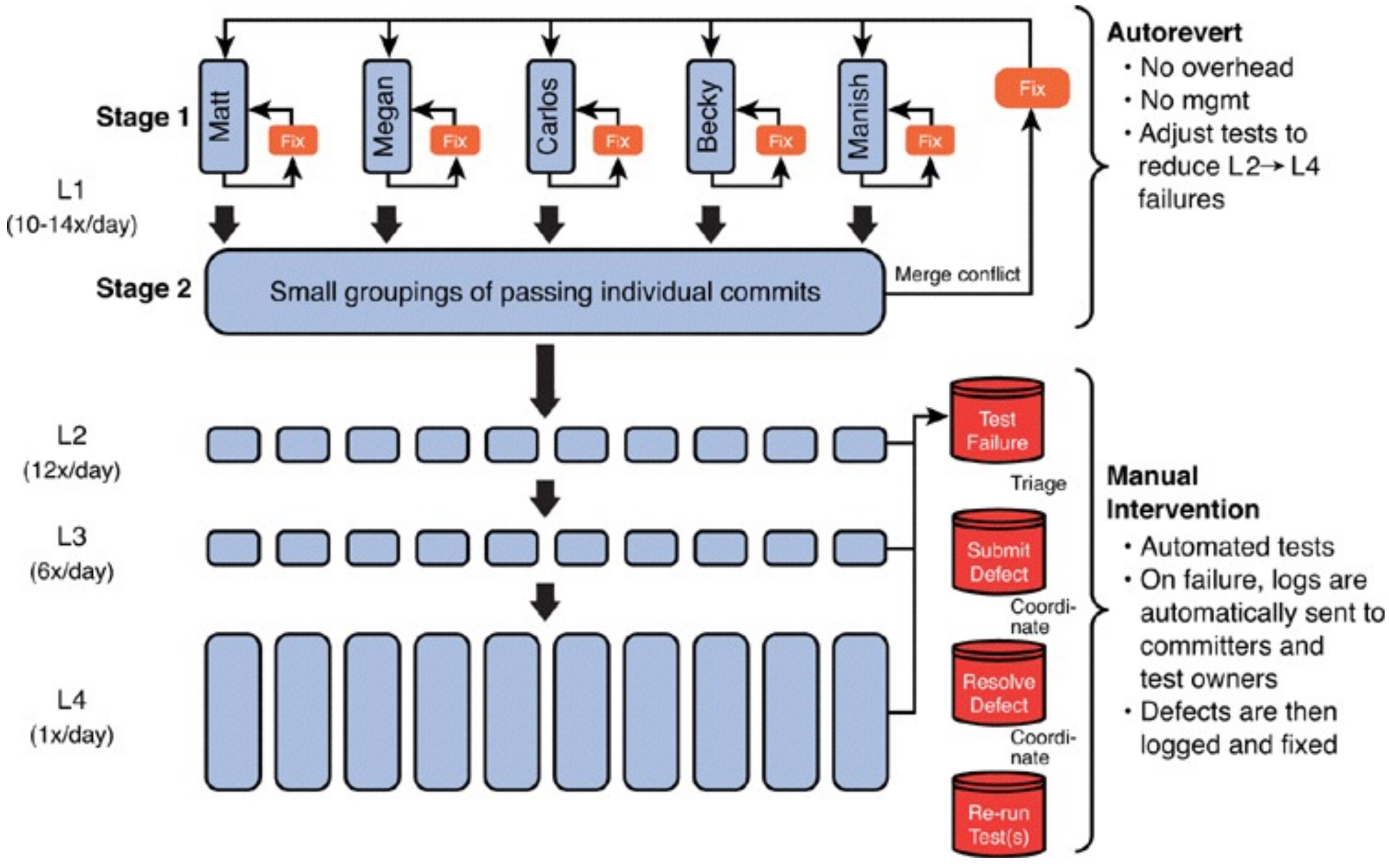


hp laserjet firmware team

2008

- 10% - code integration
- 20% - detailed planning
- 25% - porting code
- 25% - current product support
- 15% - manual testing
- **~5% - innovation**

deployment pipeline



hp laserjet firmware team

2008	2011
10% - code integration	2% - continuous integration
20% - detailed planning	5% - agile planning
25% - porting code	15% - one main branch
25% - current product support	10% - one branch cpe
15% - manual testing	5% - most testing automated
~5% - innovation	**~40% - innovation**

The remaining 23% on RHS is spent on managing automated tests.

the economics

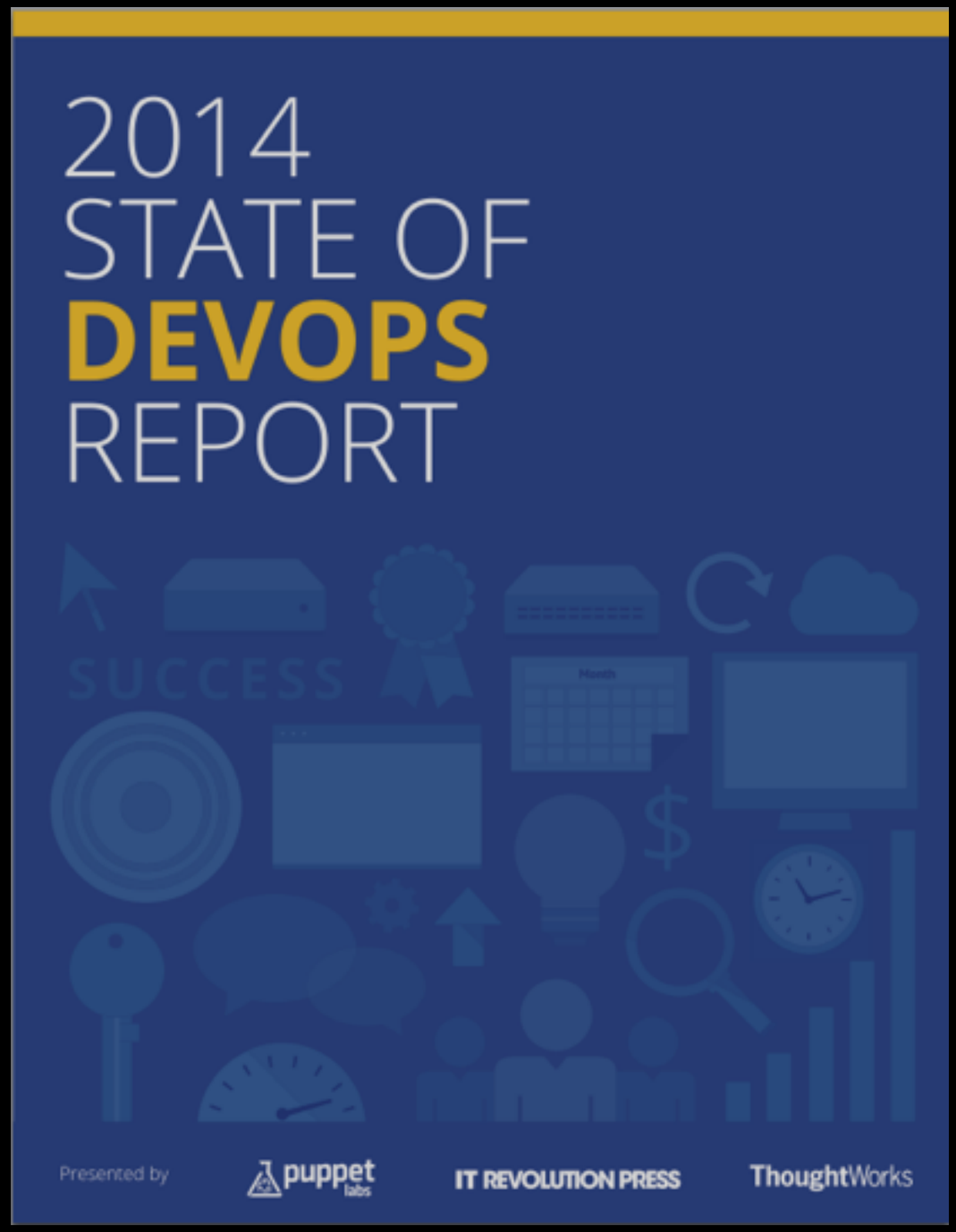
2008 to 2011

- overall development costs reduced by ~40%
- programs under development increased by ~140%
- development costs per program down 78%
- resources now driving innovation increased by 8X

A Practical Approach to Large-Scale Agile Development - Gruver, Young, Fulghum

it performance

- deployment frequency
- lead time for changes
- mean time to recover



http://bit.ly/2014-devops-report

highest correlation with it perf

“Our app configurations are in a version control system”

“Our system configurations are in a version control system”

“Our app code is in a version control system”

“We get failure alerts from logging and monitoring systems”

“Developers merge their code into trunk daily”

top predictors of it perf

peer-reviewed change approval process

version control everything

proactive monitoring

high trust organizational culture

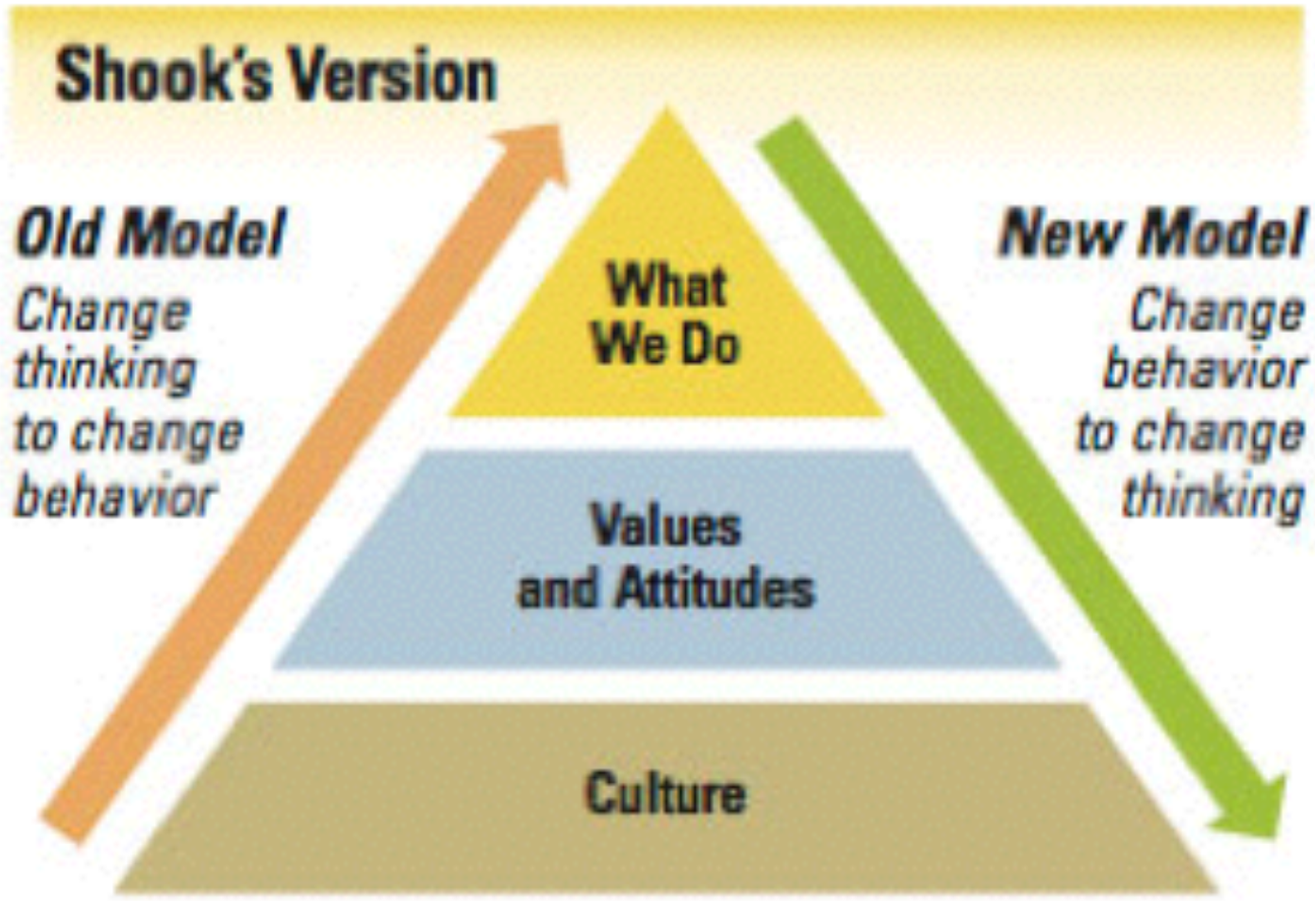
win-win relationship between dev and ops

high trust culture => information flow

Pathological	Bureaucratic	Generative
Information is hidden	Information may be ignored	Information is actively sought
Messengers are “shot”	Messengers are tolerated	Messengers are trained
Responsibilities are shirked	Responsibility is compartmented	Responsibilities are shared
Bridging is discouraged	Bridging is allowed but discouraged	Bridging is rewarded
Failure is covered up	Organisation is just and merciful	Failure causes enquiry
New ideas are crushed	New ideas create problems	New ideas are welcomed

Westrum, “A Typology of Organizational Cultures”, *Qual Saf Health Care* 2004; 13 (Suppl II):ii22-ii27

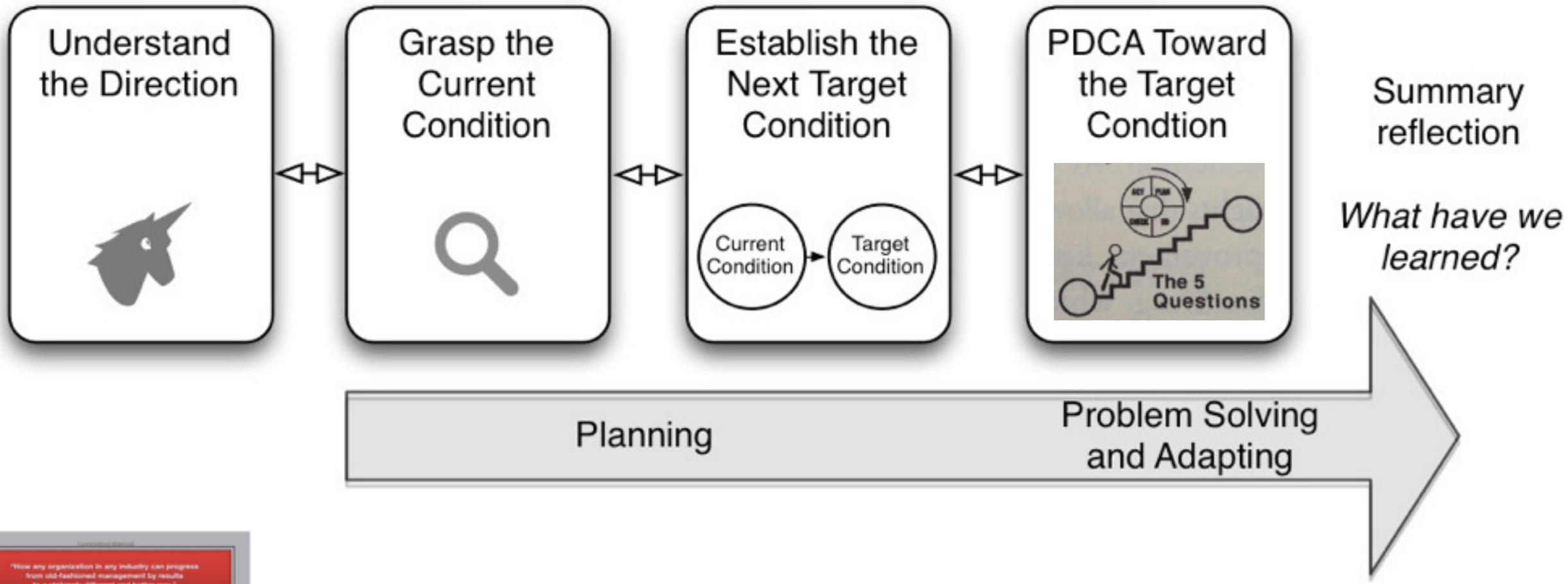
changing culture

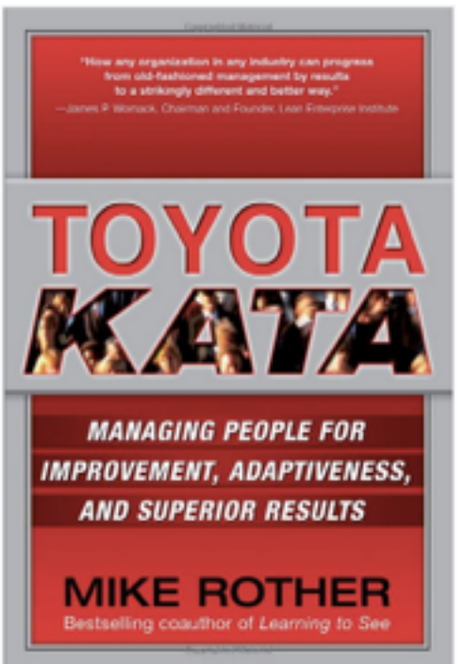


http://sloanreview.mit.edu/article/how-to-change-a-culture-lessons-from-nummi/

Original model: Schein, *The Corporate Culture Survival Guide*

improvement kata





improvement kata

What is the target condition? (*The challenge*)

What is the actual condition now?

What obstacles are preventing you from reaching it? which one are you addressing now?

What is your next step? (*Start of PDCA cycle*)

When can we go and see what we learned from taking that step?

improvement kata

Table 5.1. Sample Mini-Milestone Objectives (MM30 Objectives)

Rank	Theme	Exit Criteria: Objective Met/*Objective not met*
0	Quality threshold	P1 issues open < 1week L2 test failure 24-hour response
1	Quarterly bit release	*A) Final P1 change requests fixed* B) Reliability error rate at release criteria
2	New platform stability and test coverage	A) Customer Acceptance Test 100% passing B) All L2 test pillars 98% passing C) L4 test pillars in place D) L4 test coverage for all Product Turn On requirements E) 100% execution of L4 tests on new products
3	Product Turn On dependencies and key features	A) Print for an hour at speed to finisher with stapling B) Copy for an hour *at speed* C) *Enable powersave mode* D) Manufacturing nightly test suite execution E) Common Test Library support for four-line control panel display
4	Build for next-gen products	*A) End-to-end system build on new processor* *B) High-level performance analysis on new processor*
5	Fleet integration plan	Align on content and schedule for "slivers" of end-to-end agile test with system test lab

leadership

“highly aligned, loosely coupled”

leaders decide *outcomes* — with short horizon

people doing work discover how to achieve outcomes

job of managers is to enable their reports

update vision, outcomes, metrics based on learnings

takeaways

use automation to *detect* problems quickly

work in small batches

measure and improve customer outcomes

use continuous improvement to get better

jesse's rule

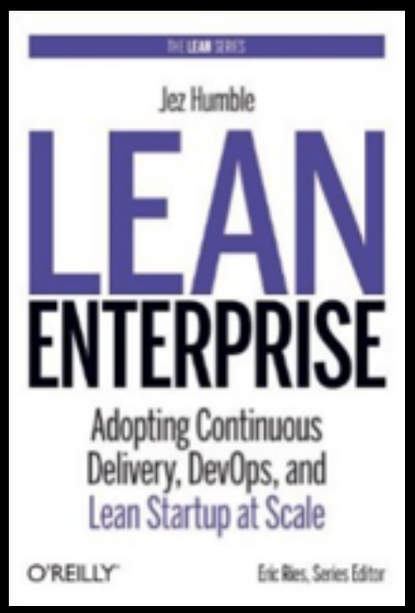
"don't fight stupid, make more awesome"

Jesse Robbins, Co-founder, Opscode @jesserobbins

questions

@jezhumble | jez@thoughtworks.com
http://continuousdelivery.com/

Pre-order my new book!
http://amzn.to/1f7UkbV



Attend FlowCon! http://flowcon.org/

ThoughtWorks is hiring!
http://join.thoughtworks.com/

Australia | Brazil | Canada | China
Ecuador | Germany | India | Italy
Singapore | South Africa | Turkey
Uganda | UK | USA

ThoughtWorks®