If You Don't Know Where You're Going, It Doesn't Matter How Fast You Get There

Nicole Forsgren, PhD @nicolefv
Jez Humble @jezhumble

Outline

Where am I going?

Why should I care?

How do I improve performance & quality?

How should I measure performance?

What is this culture thing (and how do I measure it)?

Where am I going?


World Map
Continent
Zone
Zoom Out
Right Click On Map To Zoom Out
Maturity models are for CHUMPS
KALIMDOR
EASTERN KINGDOMS

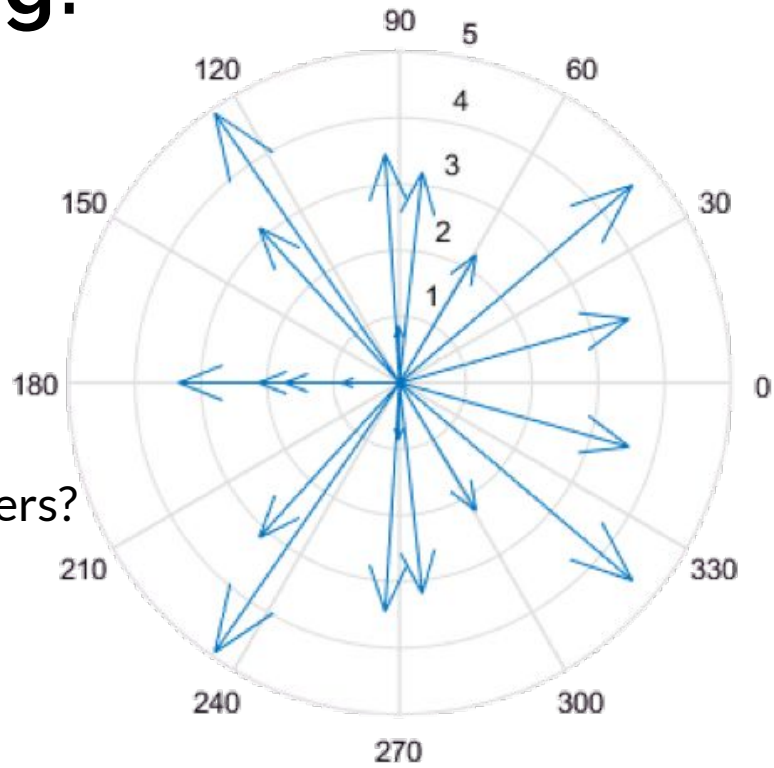

NORTHREND
BROKEN ISLES
KALIMDOR
EASTERN KINGDOMS
PANDARIA
Maturity models are for CHUMPS

Where am I going?

Direction. Not a destination.

But what direction?

Is there “one metric that matters?



IT performance

lead time for changes

release frequency

time to restore service

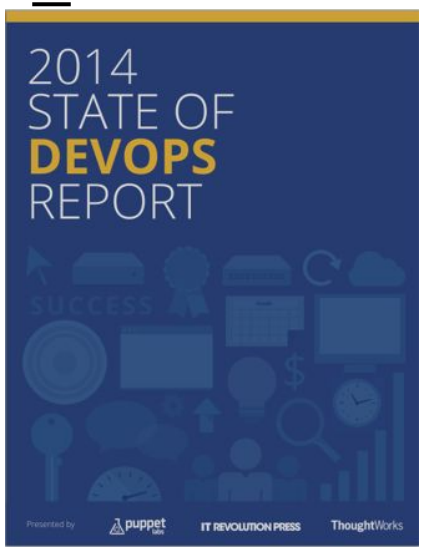
change fail rate

Survey questions	High IT performers	Medium IT performers	Low IT performers
Deployment frequency *For the primary application or service you work on, how often does your organization deploy code?*	On demand (multiple deploys per day)	Between once per week and once per month	Between once per week and once per month*
Lead time for changes *For the primary application or service you work on, what is your lead time for changes (i.e., how long does it take to go from code commit to code successfully running in production)?*	Less than one hour	Between one week and one month	Between one week and one month*
Mean time to recover (MTTR) *For the primary application or service you work on, how long does it generally take to restore service when a service incident occurs (e.g., unplanned outage, service impairment)?*	Less than one hour	Less than one day	Between one day and one week
Change failure rate *For the primary application or service you work on, what percentage of changes results either in degraded service or subsequently requires remediation (e.g., leads to service impairment, service outage, requires a hotfix, rollback, fix forward, patch)?*	0-15%	0-15%	31-45%

* *Note: Low performers were lower on average (at a statistically significant level), but had the same median as the medium performers.*

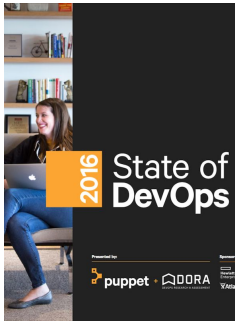
IT performance matters!

“Firms with high-performing IT organizations were twice as likely to exceed their profitability, market share and productivity goals.”



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

http://bit.ly/2015-devops-report/



http://bit.ly/2016-devops-report/

http://bit.ly/2017-devops-report/

...for nonprofits too

high performers were also twice as likely to exceed objectives in:

- quantity of goods and services
- operating efficiency
- customer satisfaction
- quality of products or services
- achieving organization or mission goals.

The DevOps Movement

A cross-functional community of practice dedicated to the study of building, evolving and operating rapidly changing, secure, resilient systems at scale.

Transformational Leadership
Vision
Intellectual Stimulation
Inspirational Communication
Supportive Leadership
Personal Recognition

Lean Management
Limit Work in Process
Visual Management
Feedback from Production
Lightweight Change Approvals

Lean Product Development
Work in Small Batches
Make Flow of Work Visible
Gather & Implement Customer Feedback
Team Experimentation

Software Development Practices
Test Automation
Deployment Automation
Trunk-Based Development
Shift Left on Security
Loosely Coupled Architecture
Empowered Teams
Continuous Integration
Version Control
Test Data Management
Monitoring
Proactive Notifications

Continuous Delivery

Identity

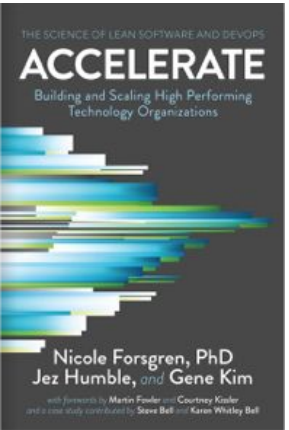
Software Delivery Performance

Westrum Organizational Culture

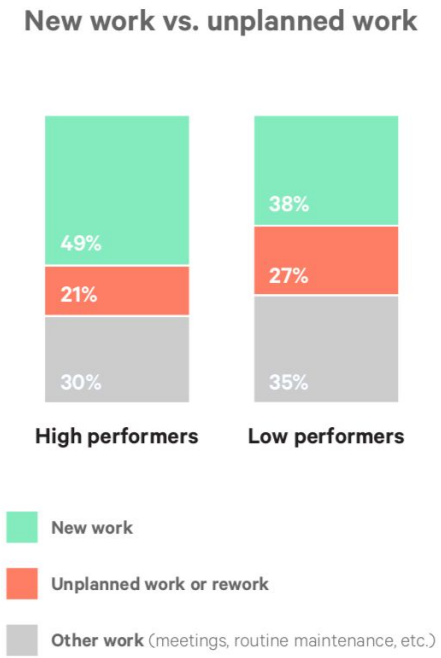
Job Satisfaction

Less Burnout

Organizational Performance



Quality


New work vs. unplanned work
49%
21%
30%
38%
27%
35%
High performers
Low performers
New work
Unplanned work or rework
Other work (meetings, routine maintenance, etc.)

How should I measure performance?

Common Mistakes

- Outputs vs. Outcomes
- Individual/local vs. Team/global
- Some common examples:
    - Lines of code
    - Velocity
    - Utilization

Common Mistakes: Lines of Code

- More is better?
  - Bloated software
  - Higher maintenance costs
  - Higher cost of change
- Less is better?
  - Cryptic code that no one can read
- Ideal: solve business problems with most efficient code

Common Mistakes: Velocity

- Agile: problems are broken down into stories, which are assigned “points” of estimated effort to complete
- At end of sprint, total points signed off by customer is recorded = velocity
- Velocity is a capacity planning tool. NOT a productivity tool.
- Why doesn’t this work for productivity?
  - Velocity is a relative measure, not absolute. So: bad for comparing teams
  - Gaming by inflating estimates
  - Focus on team completion at the expense of collaboration (a global goal)

Common Mistakes: Utilization

- Utilization is only good up to a point
- Higher utilization is better?
  - High utilization doesn't allow slack for unplanned work
  - Queue theory: as utilization approaches 100%, lead times approach infinity
  - Once you hit higher and higher levels of utilization (a poor goal of productivity), teams will take longer and longer to get work done

High Trust Culture

How Organizations Process Information

Pathological (*power oriented*)	Bureaucratic (*rule oriented*)	Generative (*performance oriented*)
Low cooperation	Modest cooperation	High cooperation
Messengers shot	Messengers neglected	Messengers trained
Responsibilities shirked	Narrow responsibilities	Risks are shared
Bridging discouraged	Bridging tolerated	Bridging encouraged
Failure leads to scapegoating	Failure leads to justice	Failure leads to enquiry
Novelty crushed	Novelty leads to problems	Novelty implemented

Westrum, “A Typology of Organizational Cultures” | http://bmj.co/1BRGh5q

Likert-type scale

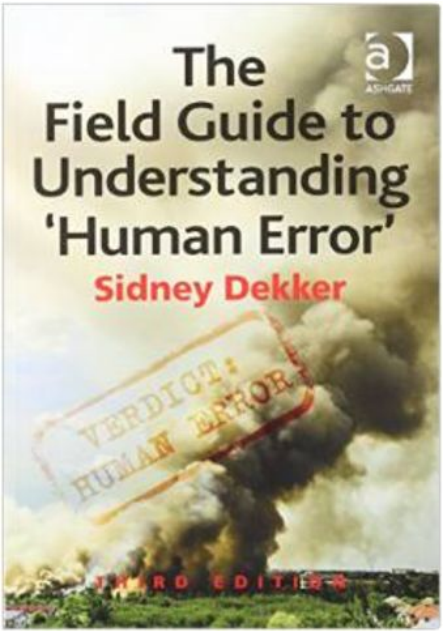
How strongly do you agree or disagree with the following? On my team...

	Strongly disagree	Disagree	Somewhat disagree	Neither agree nor disagree	Somewhat agree	Agree	Strongly agree
Information is actively sought.	○	○	○	○	○	○	○
Messengers are not punished when they deliver news of failures or other bad news.	○	○	○	○	○	○	○
Responsibilities are shared.	○	○	○	○	○	○	○
Cross-functional collaboration is encouraged and rewarded.	○	○	○	○	○	○	○
Failure causes inquiry.	○	○	○	○	○	○	○
New ideas are welcomed.	○	○	○	○	○	○	○
Failures are treated primarily as opportunities to improve the system.	○	○	○	○	○	○	○

Effective Teams

https://rework.withgoogle.com/blog/five-keys-to-a-successful-google-team/

Dealing with Failure



- In a complex, adaptive system failure is inevitable
- when accidents happen, human error is the starting point of a blameless post-mortem
- ask: how can we get people better information?
- ask: how can we detect and limit failure modes?

@rynchantress | https://ryn.works/2017/06/17/on-failure-and-resilience/

Disaster Recovery Testing

“For DiRT-style events to be successful, an organization first needs to accept system and process failures as a means of learning… We design tests that require engineers from several groups who might not normally work together to interact with each other. That way, should a real large-scale disaster ever strike, these people will already have strong working relationships”

-Kripa Krishnan, Director, Cloud Operations, Google

Kripa Krishnan | http://queue.acm.org/detail.cfm?id=2371297

Conclusions

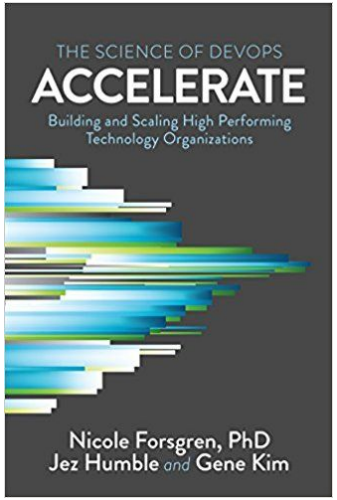
We CAN have it all, or at least tempo AND stability.

DevOps culture & practices have a measurable impact on IT & org perf & quality

Culture can be measured and changed

Technology and agility do matter - but it’s not enough

Want more Measurement Goodness?



To receive the following:

- A 93-page excerpt of ***Accelerate: The Science of DevOps***
- This presentation
- DORA's ROI whitepaper: Forecasting the Value of DevOps Transformations
- Metrics Guidance whitepaper
- Tactics for Leading Change whitepaper
- My ACM Queue article on DevOps Metrics with Mik Kersten: Your Biggest Mistake Might Be Collecting the Wrong Data

Just grab your phone and send an email:

- To: nicolefv@sendyourslides.com
- Subject: devops