Double the Awesome

Dr. Nicole Forsgren
Research & Strategy
Today!

- Is this DevOps thing even “A Thing”?
- Getting better (aka Choose your own adventure)
- Performance
- Productivity
- Culture
- Open source
- Fin! (and more!)
DevOps?

Let’s get on the same page.
We have movement

- Look at the elite performers. Yay!
- Now look at the low performers. Yay!
- Check out medium performers…
- Now look at the medium vs. high. Oooo

...But what does it all mean?
<table>
<thead>
<tr>
<th>Aspect of Software Delivery Performance*</th>
<th>Elite</th>
<th>High</th>
<th>Medium</th>
<th>Low</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Deployment frequency</strong></td>
<td>On-demand (multiple deploys per day)</td>
<td>Between once per day and once per week</td>
<td>Between once per week and once per month</td>
<td>Between once per month and once every six months</td>
</tr>
<tr>
<td><strong>Lead time for changes</strong></td>
<td>Less than one day</td>
<td>Between one day and one week</td>
<td>Between one week and one month</td>
<td>Between one month and six months</td>
</tr>
<tr>
<td><strong>Time to restore service</strong></td>
<td>Less than one hour</td>
<td>Less than one day&lt;sup&gt;a&lt;/sup&gt;</td>
<td>Less than one day&lt;sup&gt;a&lt;/sup&gt;</td>
<td>Between one week and one month</td>
</tr>
<tr>
<td><strong>Change failure rate</strong></td>
<td>0-15%&lt;sup&gt;b,c&lt;/sup&gt;</td>
<td>0-15%&lt;sup&gt;b,d&lt;/sup&gt;</td>
<td>0-15%&lt;sup&gt;c,d&lt;/sup&gt;</td>
<td>46-60%</td>
</tr>
</tbody>
</table>

*Data from DORA (Developers Research and Assessment) and Google Cloud.
How do they compare? (And what does it all mean?)

- **208 TIMES MORE** frequent code deployments
- **106 TIMES FASTER** lead time from commit to deploy
- **2,604 TIMES FASTER** time to recover from incidents
- **7 TIMES LOWER** change failure rate (changes are 1/7 as likely to fail)
Availability

Availability is about promises we make and keep to our customers and end users…

The measure: How well teams

- Define their availability targets
- Track their current availability
- Learn from any outages
Availability

Availability is about promises we make and keep to our customers and end users…

The measure: How well teams

- Define their availability targets
- Track their current availability
- Learn from any outages

Day one is short, day two is **long**
So... We do tech because it’s fun?
So... We do tech because it’s fun?

yes, and...
Technology delivers value

Elite performers are 2x as likely to meet or exceed their organizational performance goals.
Okay… so it matters. How do we get better?
Okay… so it matters. *How do we get better?*
girl. whoa.
Improving Performance

- CLEAR CHANGE PROCESS
- HEAVYWEIGHT CHANGE PROCESS
- DISASTER RECOVERY TESTING
- CODE MAINTAINABILITY
  - Loosely coupled architecture
  - Monitoring
  - Trunk-based development
  - Deployment automation
- Continuous delivery
- Cloud
- Continuous integration
- Automated testing
- SDO Performance
  - Software delivery performance
  - Availability
- Organizational performance
- Industry (control)
  - Enterprise (control)
- Burnout

Legend:
- Construct
- Second-order construct
- Common goal for team or organization
- Control variable
- Predictive relationship
- Mixed results
- Negative predictive relationship
- Bold: Newly investigated this year
Cloud is a differentiator for elite performance...

- On-demand self-service
- Broad network access
- Resource pooling
- Rapid elasticity
- Measured service
Cloud is a differentiator for elite performance…

But **only 29% of respondents** met all five characteristics of cloud computing:

- On-demand self-service
- Broad network access
- Resource pooling
- Rapid elasticity
- Measured service
Cloud is a differentiator for elite performance...

But **only 29% of respondents** met all five characteristics of cloud computing.

Elite performers were **24 times more likely** to have met cloud characteristics than low performers.

- On-demand self-service
- Broad network access
- Resource pooling
- Rapid elasticity
- Measured service
**Code maintainability**

Contributes to CD and helps reduce technical debt (stay tuned!)

**Systems and tools that make it easy to:**

- Change code maintained by other teams
- Find code in the codebase
- Reuse other people’s code
- Add, upgrade, and migrate to new versions of dependencies without breaking code
Yeah, yeah. Read the report. What else is new?
Improving Productivity

- Culture of Psychological Safety
- Years of experience (control)
- Useful, Easy-to-Use Tools
- Internal Search
- External Search
- Technical Debt
- Code Maintainability
- Loosely coupled architecture
- Monitoring
- Work Recovery
- Burnout

Symbols:
- Construct
- Predictive relationship
- Negative predictive relationship
- Common goal for team or organization
- Control variable
- n.s.: Not significant
- BOLD: Newly investigated this year

@nicolefv
What is productivity?

Productivity is the ability to get complex, time-consuming tasks completed with minimal distractions and interruptions.
What is productivity?

Productivity is the ability to get complex, time-consuming tasks completed with minimal distractions and interruptions. This kind of productivity helps us leave work at work and reduce burnout.
Technical debt was introduced in 1992 by Ward Cunningham to describe what happens when we fail to maintain “immature code”.

It is a problem for many of us and includes code or systems with:

- Known bugs that go unfixed in favor of new features
- Insufficient test coverage
- Problems related to low code quality or poor design
- Code or artifacts that aren’t cleaned up when no longer used
- Implementations the team doesn’t have expertise in, and therefore can’t debug or maintain
- Incomplete migration
- Obsolete technology
- Incomplete or outdated documentation or missing comments

@nicolefv
Reducing Technical Debt

This helps us maintain a mental model of our systems, something Ward Cunningham suggested we would need in his original article.

- Maintainable code
- Loosely coupled architecture
- Monitoring
I keep hearing culture matters.
What does that even mean?
1. **Psychological Safety**: Team members feel safe to take risks and be vulnerable in front of each other.

2. **Dependability**: Team members get things done on time and meet Google's high bar for excellence.

3. **Structure and Clarity**: Team members have clear roles, plans, and goals.

4. **Meaning**: Work is personally important to team members.

5. **Impact**: Team members think their work matters and creates change.
Culture is important

- A culture of trust and psychological safety has a positive impact on:
  - Software delivery performance
  - Organizational performance
  - Productivity
- These results indicate teams with this culture see significant benefits in teams in many contexts
So what does this all mean for open source?
Community is an asset

- High and Elite performers continue to use open source software most
  - Elite performers **1.75 more likely to make extensive use of open source components, libraries, and platforms** than low performers (2018 ASODR)
  - This also **benefits recruiting**
  - Low performers use fully proprietary software most

- External search contributes to productivity
  - Engineers who used external information **1.67 times more likely to be productive.**
Faster is better - even in open source

Open source projects are community-driven, with contributors from around the world, with different skill sets, contributing when their schedule allows. What this means for open source:

- **Committing code sooner is better.** Merging patches faster to prevent rebases helps developers move faster.

- **Working in small batches is better.** Large “patch bombs” are harder and slower to merge into a project than smaller, more readable patchsets since maintainers need more time to review the changes.
TL;DR

- Is this DevOps thing even “A Thing”?
- Getting better (aka Choose your own adventure)
- Performance
- Productivity
- Culture
- Open source
- Fin! (there’s so much more!)
  - Disaster Recovery Testing
  - Change Approvals (the “right” way)
  - More open source!
  - More cloud (and costs!)
  - Scaling transformations successfully

Go check it out! cloud.google.com/devops
Thank you!
nicolefv.com
@nicolefv