DO, RE, Me
Measuring the effectiveness of Site Reliability Engineering

Dave Stanke @davidstanke
Developer Relations Engineer, Google Cloud
SRE implements DevOps
SRE implements DevOps

okay... prove it.
Dave Stanke is a Developer Relations Engineer at Google, where he advises customers on best practices for adopting DevOps and SRE. Throughout his career, he has worn all the hats, including startup CTO, product manager, customer support, software developer, sysadmin, and graphic designer. He holds an MS in Technology Management from Columbia University.

Dave wrote about DevOps and SRE in the 2021 State of DevOps report.
What’s DevOps?
Cool, but...

What do we do with that?
Who do we study?

With seven years of research and more than 32,000 survey responses from industry professionals, the State of DevOps Report showcases the software development and DevOps practices that make teams and organizations most successful.

For this year’s report, 1200 working professionals from a variety of industries around the globe shared their experiences to help grow our understanding of the factors that drive higher performance.
Predictive model

DO, Re, Me: Measuring the effectiveness of Site Reliability Engineering

CAPABILITY → OUTCOME
CAPABILITY → OUTCOME
Commercial Outcomes
(e.g. market share, profitability, employee retention)
That's why

DO, Re, Me: Measuring the effectiveness of Site Reliability Engineering

Software Delivery and Operations Performance

Predict

Commercial Outcomes
(e.g. market share, profitability, employee retention)
Measuring Software Delivery

DO, Re, Me: Measuring the effectiveness of Site Reliability Engineering

- **Speed**
  - Deployment frequency
  - Lead time for changes

- **Stability**
  - Change fail rate
  - Time to restore service
Four Keys to Software Delivery success

SOFTWARE DELIVERY PERFORMANCE

- Lead time for changes
- Deployment frequency
- Time to restore service
- Change failure rate
# Software delivery performance at a glance

<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 week and once per month</td>
<td>Between once per month and once every 6 months</td>
<td>Fewer than once per six months</td>
</tr>
<tr>
<td><strong>Lead time for changes</strong></td>
<td>Less than one hour</td>
<td>Between one day and one week</td>
<td>Between one month and six months</td>
<td>More than six months</td>
</tr>
<tr>
<td><strong>Time to restore service</strong></td>
<td>Less than one hour</td>
<td>Less than one day</td>
<td>Between one day and one week</td>
<td>More than six months</td>
</tr>
<tr>
<td><strong>Change failure rate</strong></td>
<td>0%-15%</td>
<td>16%-30%</td>
<td>16%-30%</td>
<td>16%-30%</td>
</tr>
</tbody>
</table>
What's so special about Elite performers?

- 973x more frequent code deployments
- 6570x faster lead time to deploy
- 3x lower change failure rate (changes are 1/3 as likely to fail)
- 6570x faster time to recover from incidents

Yes, you read correctly. This is not an error.
Software Delivery and Operations Performance

As measured by

- **Velocity**
  - lead time for changes
  - deployment frequency

- **Stability**
  - time to restore service
  - change failure rate

- (another thing, too... stay tuned!)

Commercial Outcomes
(e.g. market share, profitability, employee retention)

Predict
**Capabilities**

**Technical**
- Trunk-based development
- Cloud infrastructure
- Shifting left on security

**Process**
- Work in small batches
- Streamlined change approval
- Visibility of work in value stream

**Cultural**
- Generative, trust-based
- Learning culture
- Transformational leadership

---

**Software Delivery and Operations Performance**

- **As measured by**
  - **Velocity**
    - lead time for changes
    - deployment frequency
  - **Stability**
    - time to restore service
    - change failure rate
  - (another thing, too... stay tuned!)

---

**Commercial Outcomes**
(e.g. market share, profitability, employee retention)
State of DevOps 2021
What’s SRE?
Site Reliability Engineering

- A framework for modern technical operations
- Originated at Google in the early 2000s
- Designed to help distributed services scale sustainably to global user bases and 24/7 uptime

SRE Methodology

Error budgets
Rational alerting
Disaster preparedness
Toil reduction

SRE Principles

1. The Most Important Feature of Any System is its Reliability.
2. Our Monitoring Doesn’t Decide Our Reliability... Our Users Do.
3. Meeting our reliability goals requires:
   a. Well-executed software
   b. Well-executed operations

SRE Language

Service Level Objectives
The values we adopt help us meet our objectives.
SRE vs(?) DevOps
DevOps’ original scope

Business - Product - Dev - Test - Deploy - Operate

DevOps originated here
DevOps’ original scope

Business - Product - Dev - Test - Deploy - Operate

DevOps originated here

SRE lives mostly around here
DevOps scope creep
(BizDevOps, DevSecOps, etc)

Business - Product - Dev - Test - Deploy - Operate

Most of the things
Is DevOps all the things?

Business - Product - Dev - Test - Deploy - Operate

All the things...

@davidstanke
2021: DORA tackles SRE
Four + 1
The keys to Software Delivery and Operations

SOFTWARE DELIVERY PERFORMANCE

- Lead time for changes
- Deployment frequency
- Time to restore service
- Change failure rate

OPERATIONAL PERFORMANCE

- Availability
Availability → Reliability
Four + 1
The keys to Software Delivery and Operations

SOFTWARE DELIVERY PERFORMANCE
- Lead time for changes
- Deployment frequency
- Time to restore service
- Change failure rate

OPERATIONAL PERFORMANCE
- Reliability !!!
How do we know if a team is doing SRE?

Actually, it's only SRE if it's from the cloud division of Google, otherwise it's just sparkling devops.
Alerts are generated based on service degradation from the perspective of end users or dependent systems (as opposed to system metrics like CPU %, ICMP latency, or disk IO).

My team actively identifies and eliminates, automates, or streamlines repetitive manual work.

My team is empowered to maintain reliability standards throughout the product development process.

Alerts are configured so that humans are only paged when an immediate response is needed.

My team has well-defined protocols and tools for responding to production incidents.

Reliability reviews are performed throughout the development process for all major features on the applications I work on.

My team regularly practices incident response procedures.

My team regularly reviews and revises reliability targets based on evidence.

My team proactively plans for capacity demand using data from multiple sources.

When we miss our availability targets, we perform improvement work and/or re-prioritize.
What did we learn?
1. SRE is widely practiced
52% of respondents reported the use of SRE practices (to some degree)
2. SRE is good for humans and systems
Humans
- SRE mitigates burnout
- SRE enables balance between coding and “ops” work

Systems
- “Shared responsibility” for operations predicts better r9y outcomes
- SRE predicts higher reliability

Business
- Higher reliability predicts better business outcomes
3. Reliability is a force multiplier
Software Delivery Performance
Software Delivery Performance

Reliability

Business outcomes
4. There’s room for growth
Dave’s Hot Takes
Dave’s Hot Take #1:

SRE implements part of DevOps
Dave’s Hot Take #2:

DevOps culture ≈ SRE culture ≈ TPS ≈ psych safety ≈ [...]
Dave’s Hot Take #3:

Ops is still Ops
Thank you.

What questions do you have?
Want more?

bit.ly/dora-sodr
The Accelerate State of DevOps report 2021

sre.google
Books, articles, resources

bit.ly/r9y-discuss
Reliability Engineering Discussion Group