DevOps Fundamentals

Will Angel

Linkedin:
https://www.linkedin.com/in/william-angel/
Twitter: @DataDrivenAngel

Overview

Presentation : DevOps Practices and Tooling

- DevOps Overview
- DevOps Dora Metrics
- Practices and Tools
- Challenges of DevOps for Data Engineering

DevOps Overview

What is DevOps?

DevOps is how you win.

DevOps is being good at **Dev**eloping and **Op**erating software.

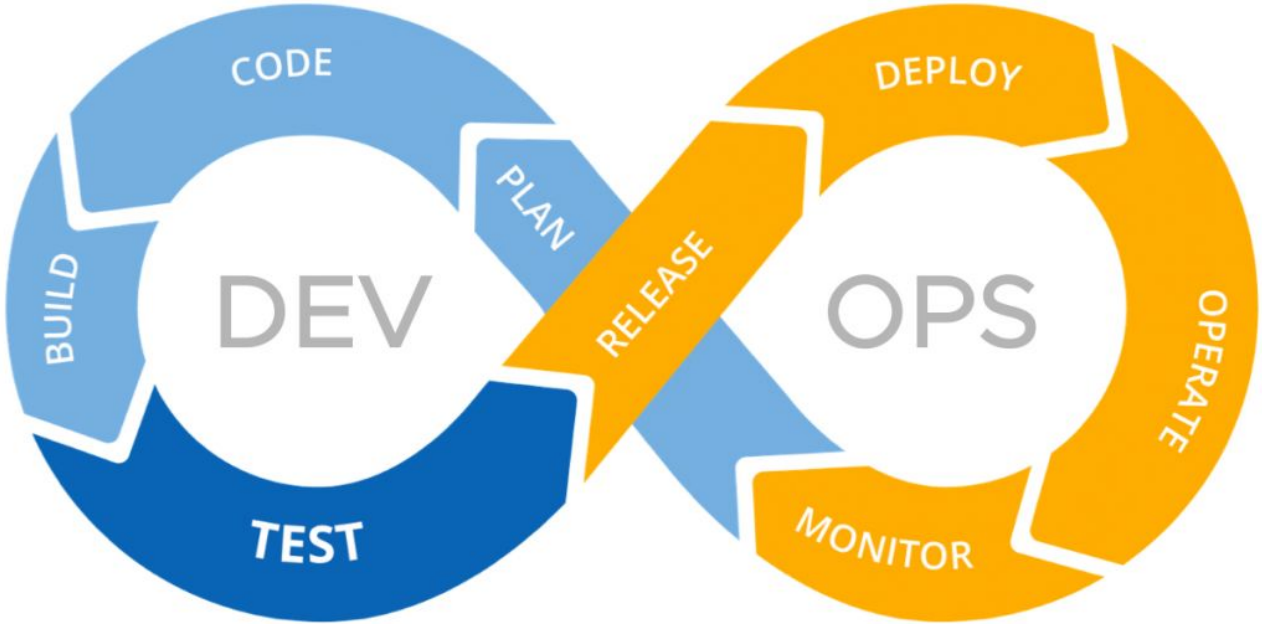
Developing and operating software is a continuous process, so DevOps is being good at continuously getting good at developing and operating software.

Why?

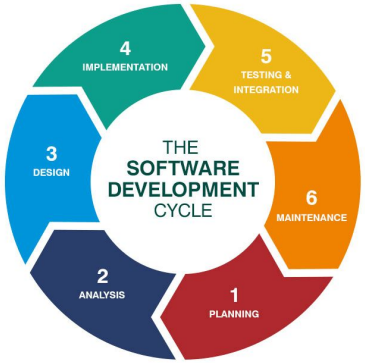
Reducing the number of handoffs required to deliver working software results in higher quality software.

Having developers care about how their software is running results in higher quality software in less time.

The DevOps Cycle



It’s the Software Development Lifecycle but AGILE



The DevOps triangle

Devops is a combination of:

- Development **Process**
- System **Architecture**
- Technical **Systems**

How do you build a complex sociotechnical system of people, process, and systems that can operate effectively, change to operate in new and more effective ways, and develop software

Devops is culture

“You build it, you run it”

Devops is not:

DevOps is not:

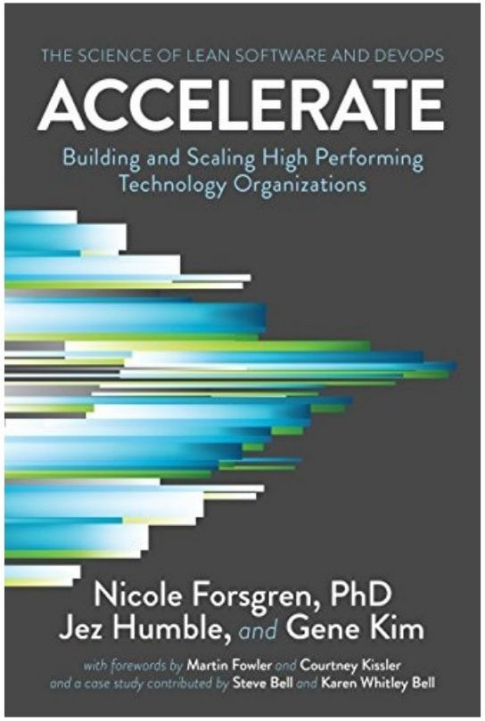
- Renaming your ops team to DevOps
- Hiring a ‘DevOps Engineer’
- Using Kubernetes everywhere.

So who defines what DevOps is?

DORA - DevOps Research and Assessment

- DevOps Research and Assessment Team: Google Cloud's devops research team.
  - Originally independent company started by Nicole Forsgren
    - She's the author of Accelerate: The Science of Devops and Lean Software

- Puts out Annual State of DevOps report
  - Industry survey of software development performance.



Dora Metrics

The core devops metrics from DORA (sometimes referred to as the ‘Four Keys’):

- Deployment Frequency
- Lead time for changes
- Time to restore service (MTTR)
- Change Failure Rate (CFR)

Deployment Frequency

How often does your organization deploy code to production or release it to end users?

- Elite: On-demand (multiple deploys per day)
- High: Between once per week and once per month
- Medium: Between once per month and once every 6 months
- Low: Fewer than once per six months

Lead time for changes

How long does it take to go from code committed to code successfully running in production?

- Elite: Less than one hour
- High: Between one day and one week
- Medium: Between one month and six months
- Low: More than six months

Time to restore service

Mean Time To Recovery (MTTR)
How long does it generally take to restore service when a service incident or a defect that impacts users occurs?

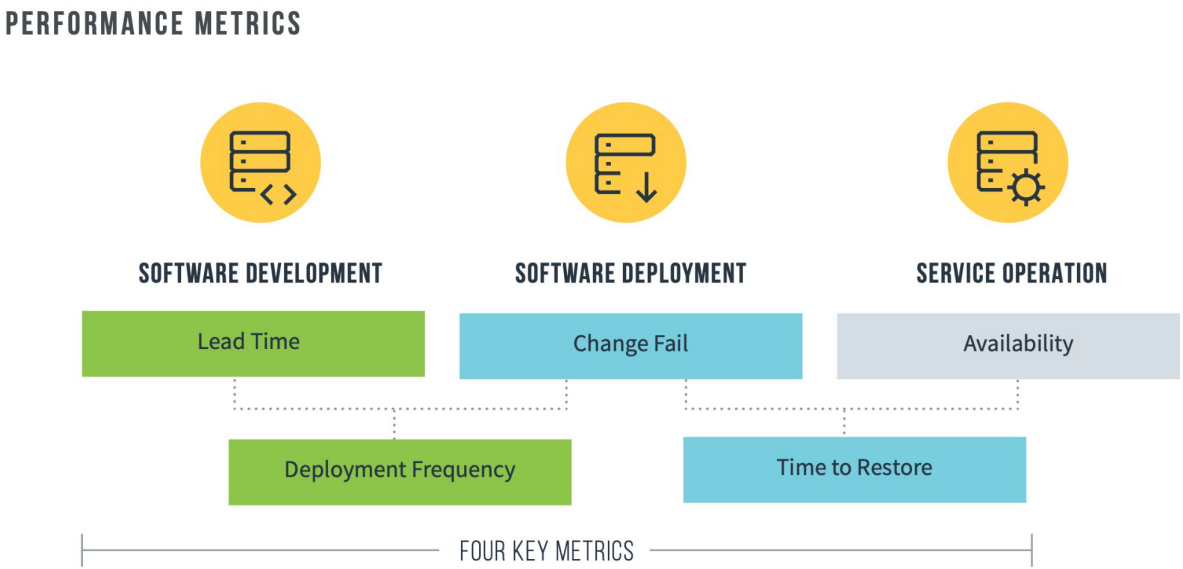
- Elite: Less than one hour
- High: Between one day and one week
- Medium: Between one month and six months
- Low: More than six months

Change Failure Rate

What percentage of changes to production or released to users result in degraded service and subsequently require remediation?

- Elite: 0-15%
- High: 16-30%
- Medium: 16-30%
- Low: 16-30%

In short: going fast results in higher quality software



DevOps Practices & Tools

CAVEAT:

All practices and tools are context specific and may not be appropriate for your software development environment.

In many cases software development and operations may deviate from ‘best practices’ due to contractual or security constraints.

Constraints are okay...

Sometimes you get painful constraints

DevOps Practices & Tools

Practices & Tooling:

- Version Control
- Testing
- Deployment Automation
- Configuration Management / Infrastructure as code (IaC)
- Logging
- Monitoring

Version Control

Using a version control system (VCS) like git or SVN makes it easier to observe and revise code changes.

Do not ‘hot fix’ systems running in production (or if you do, please get those fixes in version control promptly).

Testing

You are testing your code.

Ideally that involves intentionally building and maintaining a set of tests that run automatically as you merge new code into your version control system as part of Continuous Integration (CI).

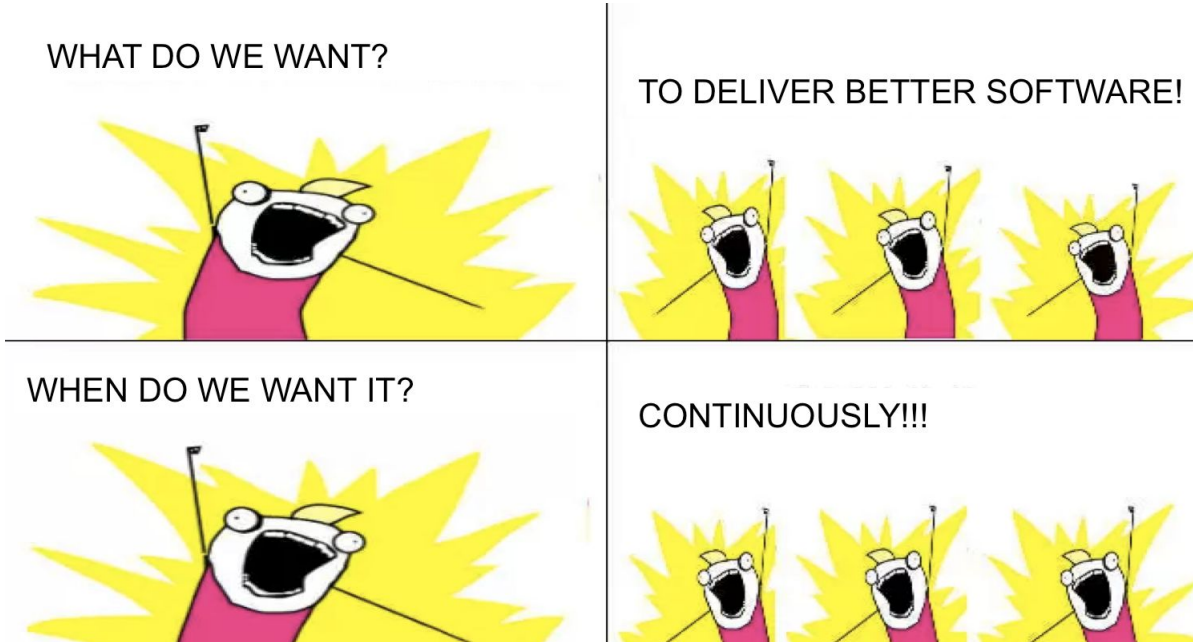
Otherwise angry users will do the testing for you.

Your language will have a testing framework or library. Use that.

Deployment Automation

Have a process to deploy your code.

Ideally your deployment process is fully automated and runs whenever there is new code in version control, resulting in Continuous Delivery (CD) of software.



Deployment Automation / CICD tools:

- Github Actions
- Gitlab CI
- Argo CD
- CircleCI
- Jenkins
- Many many more

Jenkins

Configuration Management / Infrastructure as Code

Putting your configuration in code makes deployment more repeatable.

We put our code in version control to make running it more repeatable.

By putting our infrastructure provisioning and configuration in version control our infrastructure becomes more repeatable. If it’s just something

Configuration Management / Infrastructure as Code Tools:

- Terraform
- Chef
- Puppet
- Pulumi
- Ansible
- GCP/AWS/Azure tools
- many many more

Terraform

Logging

We log events in applications to help with:

- Debugging
  - Identify errors and unexpected behavior
  - Confirm correct behavior
  - Measure performance

- Auditing
  - Business & legal audits
  - Data Analytics

Monitoring

We monitor deployed software applications so that we can:

- Sleep at night
  - Having insight into system performance increases confidence in reliability
  - Monitoring usually includes alerting as well
- Improve system performance
  - Identifying performance bottlenecks allows for optimization
- Improve user satisfaction / system value
  - Users don’t like errors and software bugs

Monitoring Ecosystem

- Prometheus
- Grafana
- Elastic/ ELK
- Splunk
- New Relic
- Sentry
- Data Dog
- Many many many more

This seems like a lot...

“Shifting left” on operations means eventually needing 10x unicorn developers who understand everything, which will not happen.

DevOps is a team endeavor that requires support from the whole business.

DevOps for Data Engineering

DevOps for Data Engineering

It’s the same!... But also different.*

Differences (non exhaustive list):

1. Data has mass.
   DevOps & IaC work better as you put more state in VCS. Data is inherently stateful and gets expensive.
2. Large (larger?) number of third party vendors
   Lots of great vendors make running it all yourself less appealing, DevOps with significant third party dependencies gets harder.

* This depends a lot on what kind of data engineering you’re doing. Custom built Data Intensive applications will benefit more obviously from ‘conventional’ DevOps practices than data warehouse / orchestration data engineering.

Data has Mass

In application development, the core artifact delivered is an application.

In data engineering, the core artifact delivered is (often) an application that interacts with data.

- Larger volumes of data are heavy and stateful
  - Many CICD tools work best with low/no state.
  - Tests are harder and more expensive with large data.
    - Maintaining a dev/qa/prod environment means 3x your data storage if you duplicate data.
- Application often becomes commingled with data:
  - Harder to test code in isolation without data

Larger number of third party tools

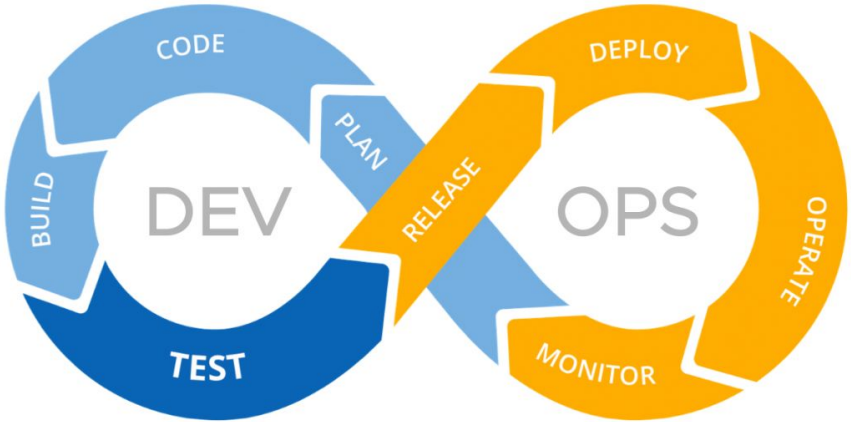
Build vs buy decisions in data engineering often heavily tilted towards buy:

- Don’t try and develop your own data warehouse unless you really really need it. BigQuery and snowflake are great.
- Off the shelf ETL like Stitch and Fivetran are great affordable options.

Harder to do CICD, Monitoring, and Infrastructure as Code on third party SaaS if they do not offer that as a feature. Many don’t.

You can go full open source, in most analytics focused data engineering roles that will not be a cost effective option unless you’re super good at operations.

Conclusion: Data makes DevOps harder, but DevOps makes data engineering easier



Questions

Linkedin: https://www.linkedin.com/in/william-angel/
Twitter: @DataDrivenAngel
Github: DataDrivenAngel
Email: will@williamangel.net

Additional Resources:

12 Factor App Principles: https://12factor.net/

Google SRE book: https://sre.google/sre-book/