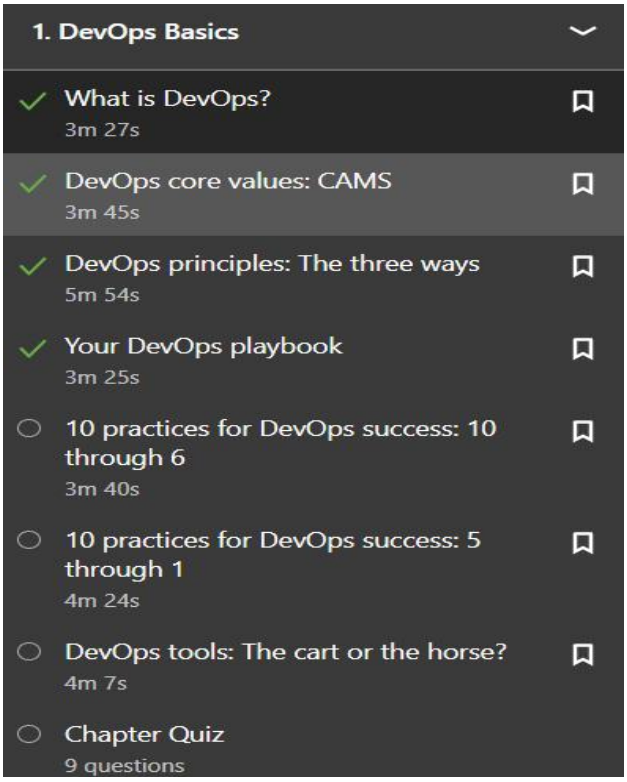
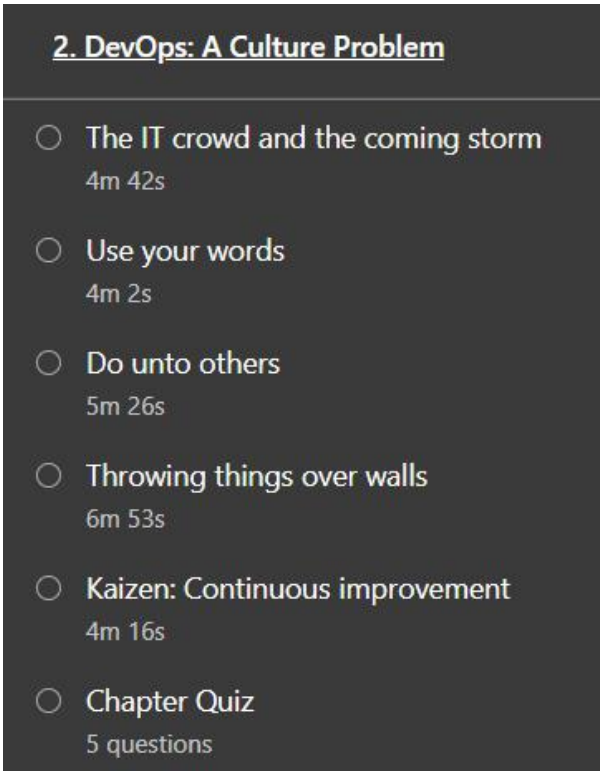
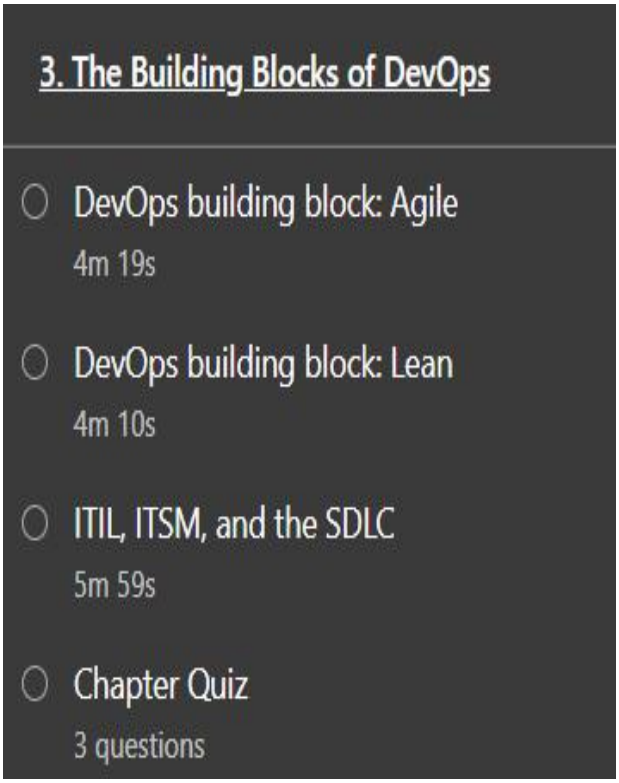
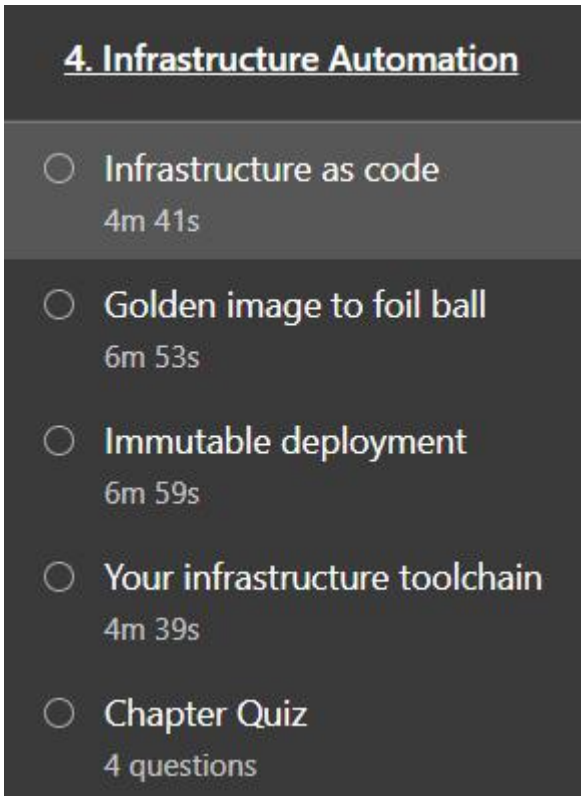
Content


1. DevOps Basics
What is DevOps?
3m 27s
DevOps core values: CAMS
3m 45s
DevOps principles: The three ways
5m 54s
Your DevOps playbook
3m 25s
10 practices for DevOps success: 10 through 6
3m 40s
10 practices for DevOps success: 5 through 1
4m 24s
DevOps tools: The cart or the horse?
4m 7s
Chapter Quiz
9 questions


2. DevOps: A Culture Problem
The IT crowd and the coming storm
4m 42s
Use your words
4m 2s
Do unto others
5m 26s
Throwing things over walls
6m 53s
Kaizen: Continuous improvement
4m 16s
Chapter Quiz
5 questions


3. The Building Blocks of DevOps
DevOps building block: Agile
4m 19s
DevOps building block: Lean
4m 10s
ITIL, ITSM, and the SDLC
5m 59s
Chapter Quiz
3 questions


4. Infrastructure Automation
Infrastructure as code
4m 41s
Golden image to foil ball
6m 53s
Immutable deployment
6m 59s
Your infrastructure toolchain
4m 39s
Chapter Quiz
4 questions

5. Continuous Delivery

- Small + fast = better
  4m 55s
- Continuous integration practices
  3m 51s
- The continuous delivery pipeline
  5m 13s
- The role Of QA
  4m 14s
- Your CI toolchain
  5m 22s
- Chapter Quiz
  5 questions

6. Reliability Engineering

- Engineering doesn't end with deployment
  3m 30s
- Design for operation: Theory
  4m 3s
- Design for operation: Practice
  5m 54s
- Operate for design: Metrics and monitoring
  5m 34s
- Operate for design: Logging
  3m 18s
- Your SRE toolchain
  4m 33s
- Chapter Quiz
  6 questions

7. Additional DevOps Resources

- Unicorns, horses, and donkeys, oh my
  5m 9s
- The 10 best DevOps books you need to read
  4m 35s
- Navigating the series of tubes
  3m 57s
- Chapter Quiz
  3 questions

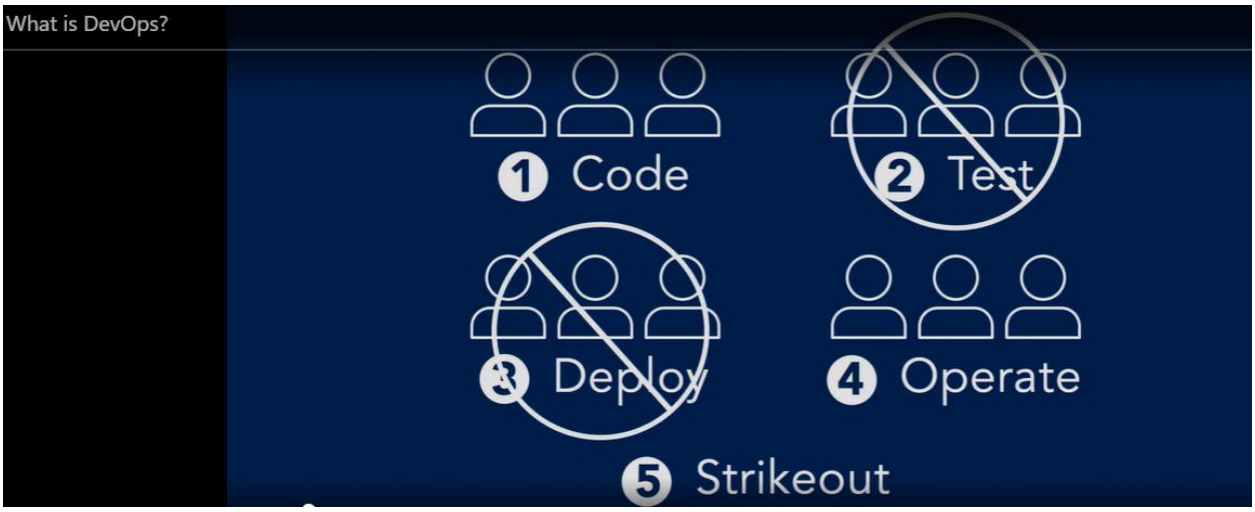
8. The Future of DevOps

- Cloud to containers to serverless solutions
  5m 29s
- The rugged frontier of DevOps: Security
  6m 9s
- Chapter Quiz
  3 questions

Conclusion

- Next steps: Am I a DevOp now?
  5m 13s


DevOps Foundations
What is DevOps?
7,327
DevOps
The practice of operations and development engineers participating together in the entire service lifecycle, from design through the development process to production support
https://theagileadmin.com/what-is-devops


What is DevOps?
1 Code
2 Test
3 Deploy
4 Operate
5 Strikeout


The Five Levels of DevOps
1 Values
2 Principles
3 Methods
4 Practices
5 Tools
values, principles, methods, practices, and tools.

What is DevOps?

1. High-performing IT organizations deploy more frequently, fail less, and recover faster.
2. Lean management and continuous delivery practices help deliver value faster.
3. High performance is achievable whether your apps are greenfield, brownfield, or legacy.

What is DevOps?

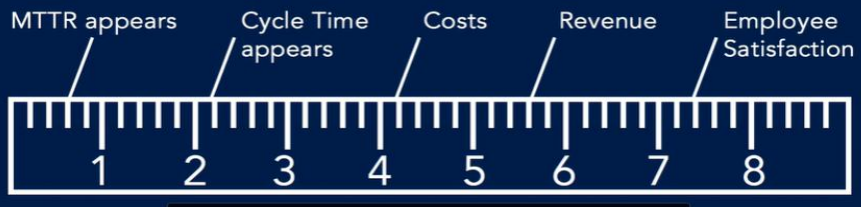
Dev → "code"

Ops ← "systems"

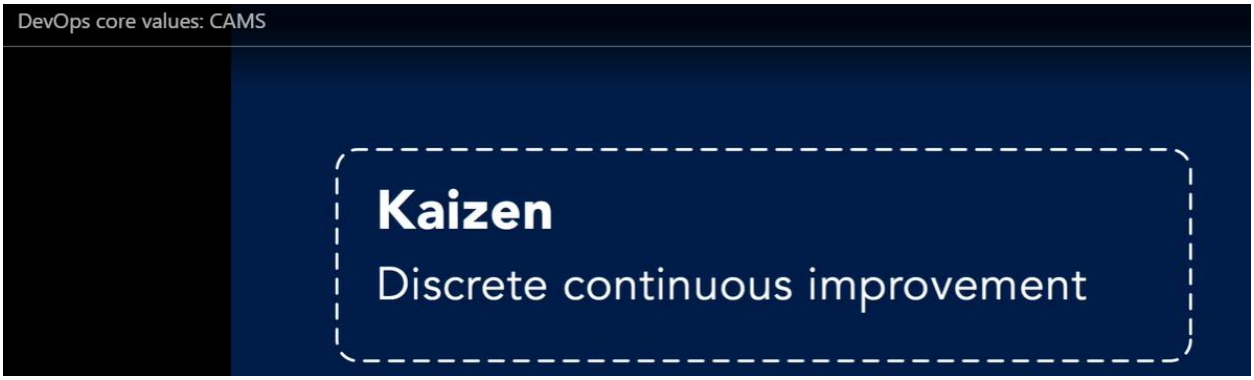
DevOps core values: CAMS

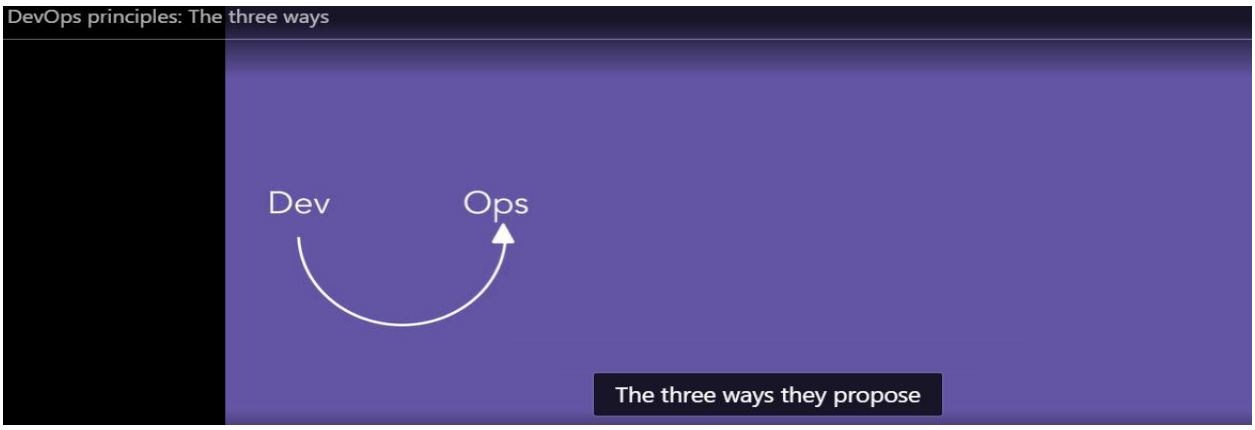
1. Culture
2. Automation
3. Measurement
4. Sharing

DevOps core values: CAMS

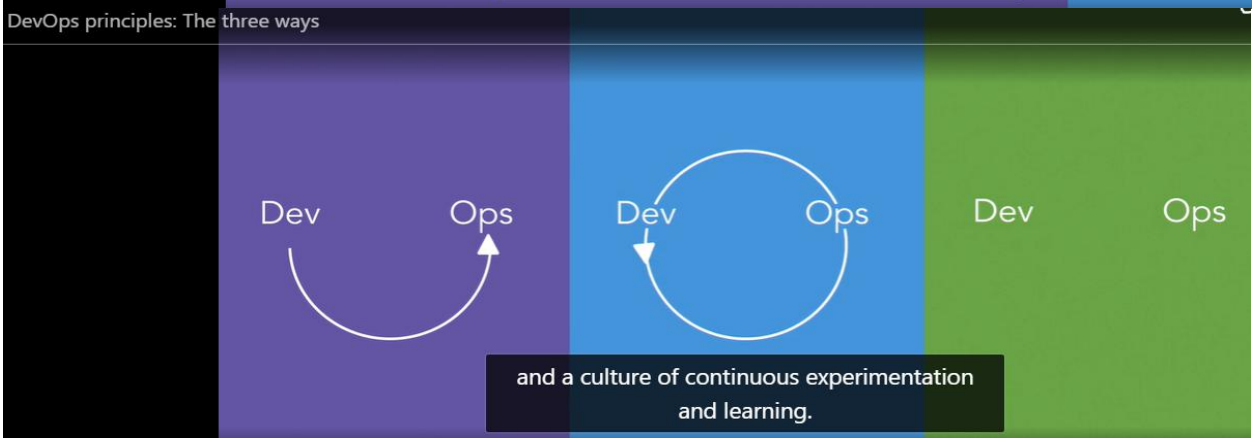


of generating a holistic insight across your system.

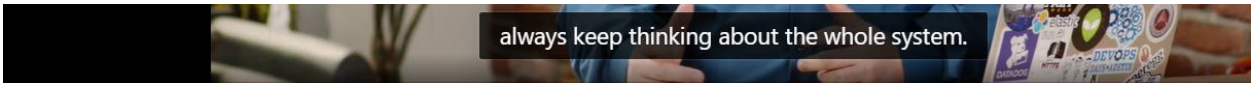

DevOps core values: CAMS
Kaizen
Discrete continuous improvement

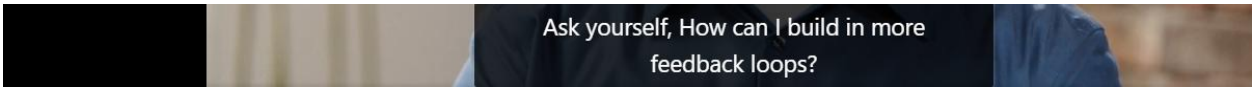

DevOps principles: The three ways
Dev
Ops
The three ways they propose

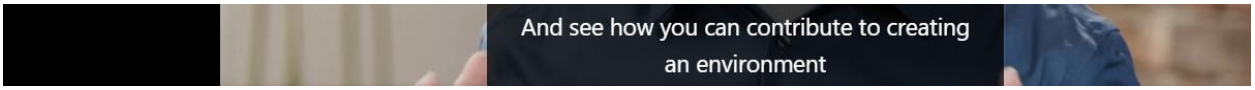

DevOps principles: The three ways
Dev
Ops
are systems thinking, amplifying feedback loops,

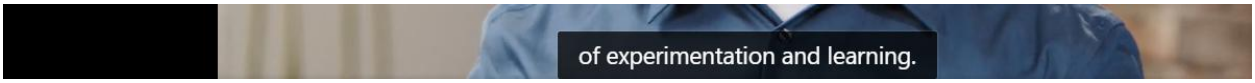

DevOps principles: The three ways
Dev
Ops
Dev
Ops
Dev
Ops
and a culture of continuous experimentation and learning.


DevOps principles: The three ways
Dev
Ops
Use
To create team processes
To create team standards
As part of management style

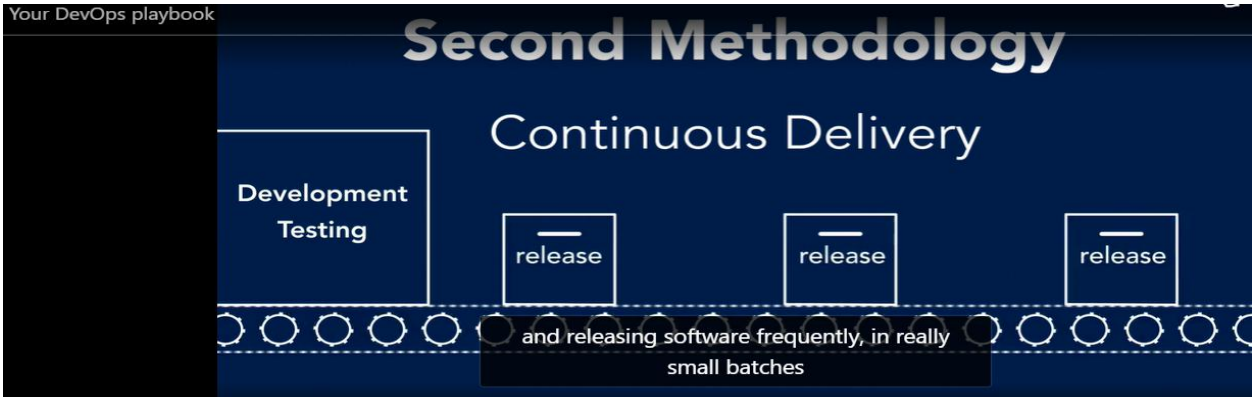

always keep thinking about the whole system.

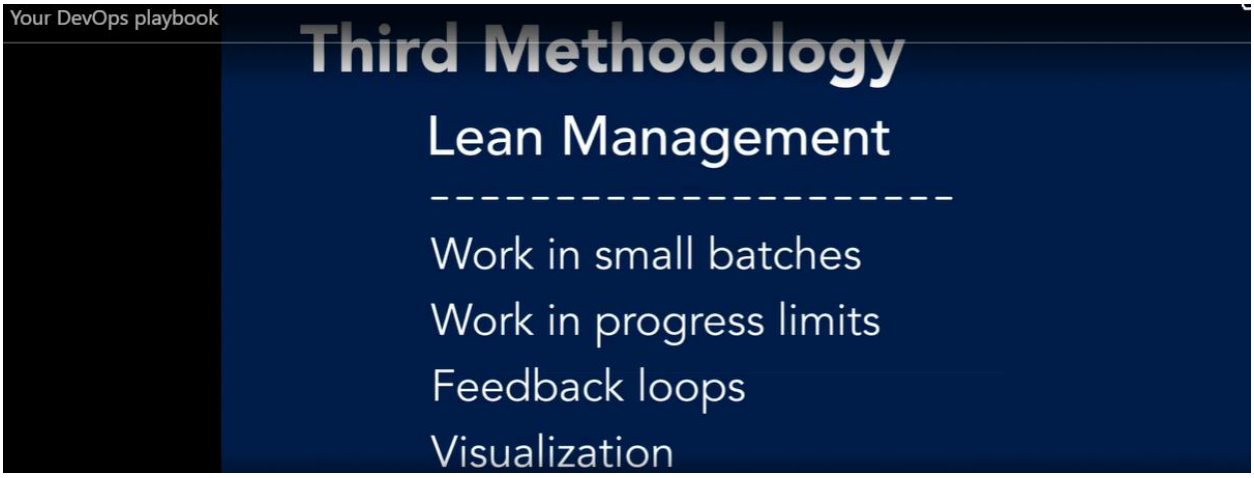

Ask yourself, How can I build in more feedback loops?


And see how you can contribute to creating an environment

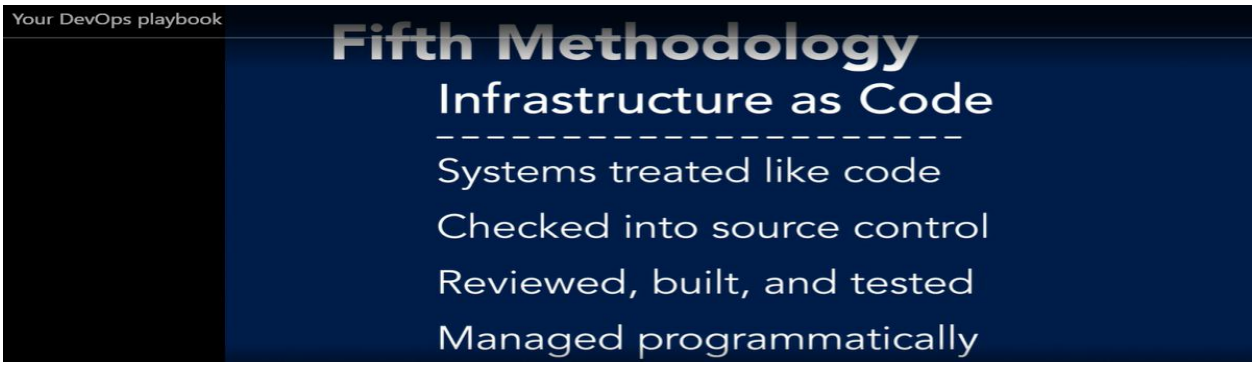

of experimentation and learning.

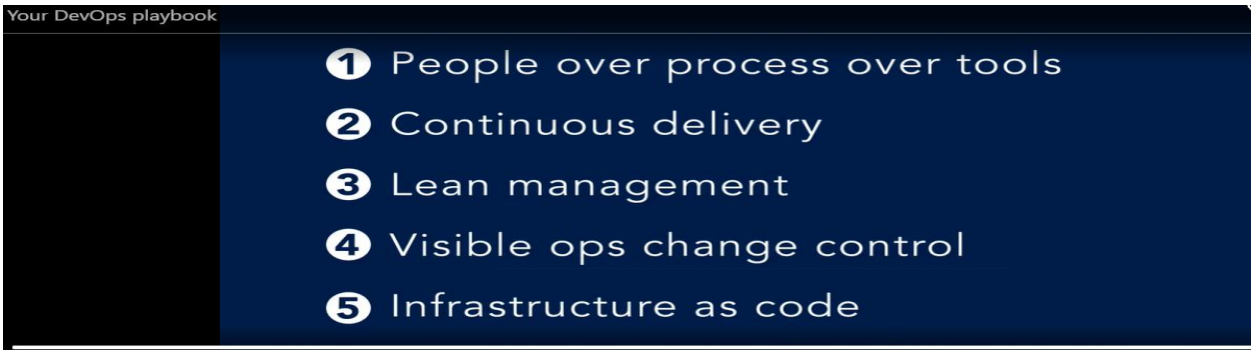

Your DevOps playbook
First Methodology
People
Process
Tools

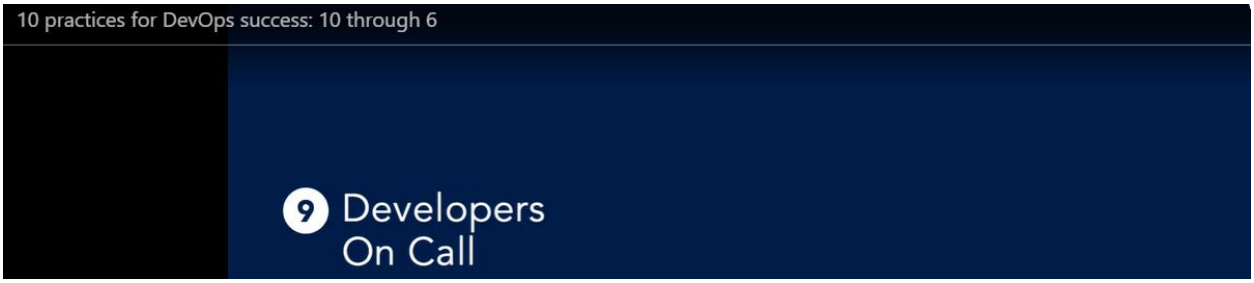

Your DevOps playbook
Second Methodology
Continuous Delivery
Development
Testing
release
release
release
and releasing software frequently, in really small batches

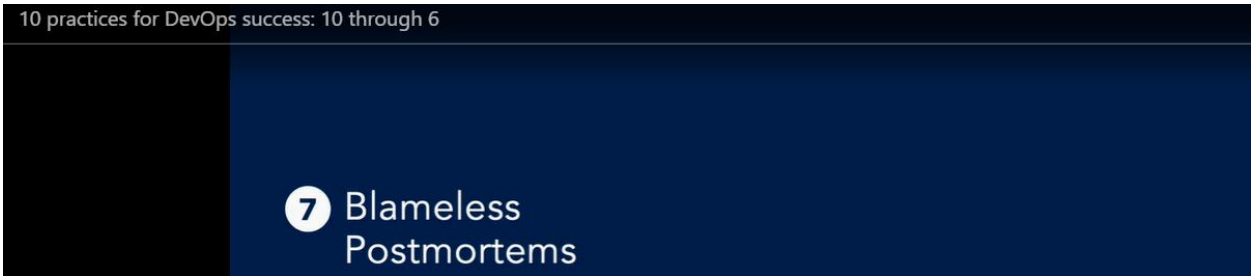

Your DevOps playbook
Third Methodology
Lean Management
Work in small batches
Work in progress limits
Feedback loops
Visualization


Your DevOps playbook
Fourth Methodology
Change Control
Eliminate fragile artifacts
Create a repeatable build process
Manage dependencies
Create an environment of continuous improvement

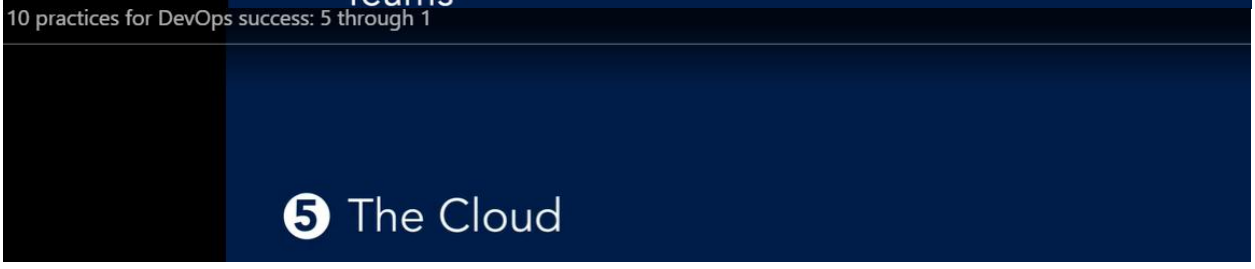

Your DevOps playbook
Fifth Methodology
Infrastructure as Code
Systems treated like code
Checked into source control
Reviewed, built, and tested
Managed programmatically

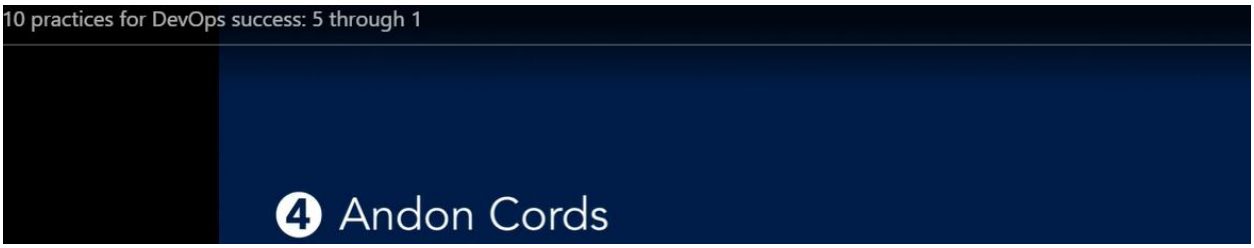

Your DevOps playbook
1 People over process over tools
2 Continuous delivery
3 Lean management
4 Visible ops change control
5 Infrastructure as code



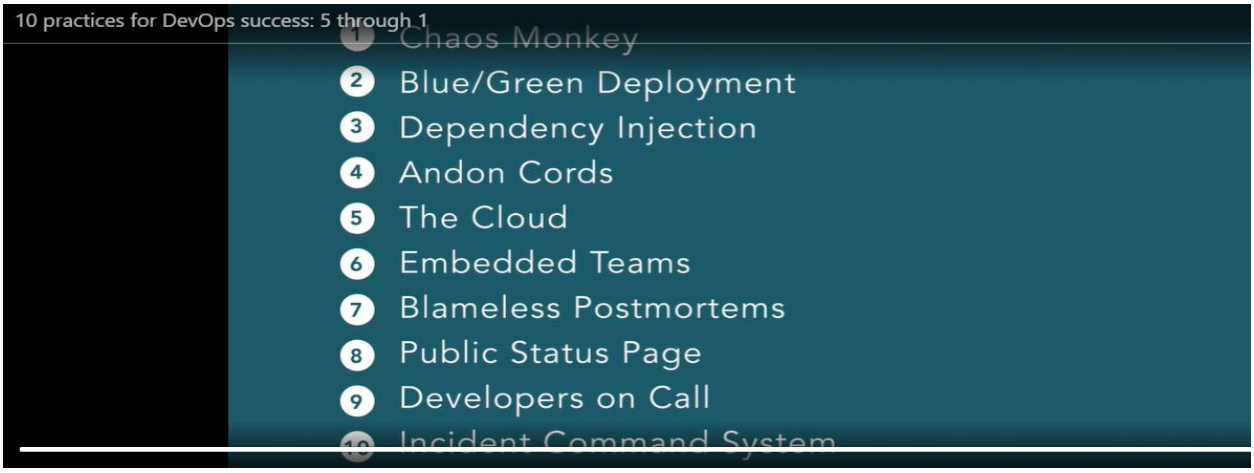


Postmortems means 事后调查



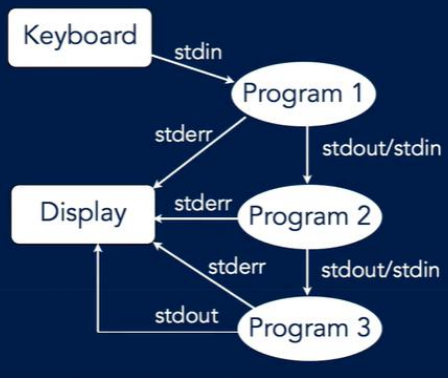


Andon Cords means Stop and fix the bugs.



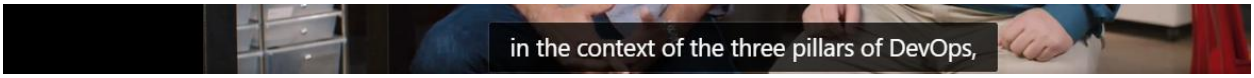
DevOps Tools: The Cart or the Horse?

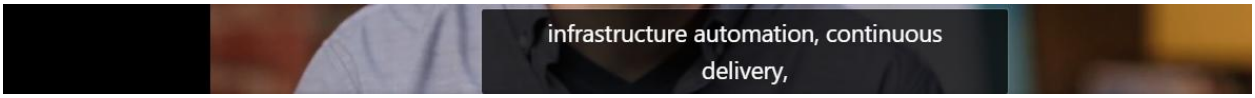
Weblogic Gradle Jenkins Maven Fai
Chef Solano memcache RabbitMQ
Ant Capistrano Cobbler Solaris Containers
Ansible Docker Linux Windows
nginx Unix Mac OS X JBoss IIS
Amazon Web Services Tomcat
Apache Cloud Foundry OpenStack
CFEngine Rackspace Jetty
VMware Azure Xen
ActiveMQ OpenStack Glassfish
varnish SaltStack LXC KVM VirtualBox Vagrant
squid Kickstart
RANCID Puppet / MCollective
Websphere Ubuntu Juju

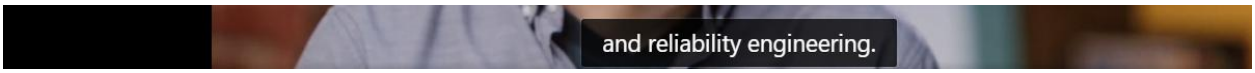


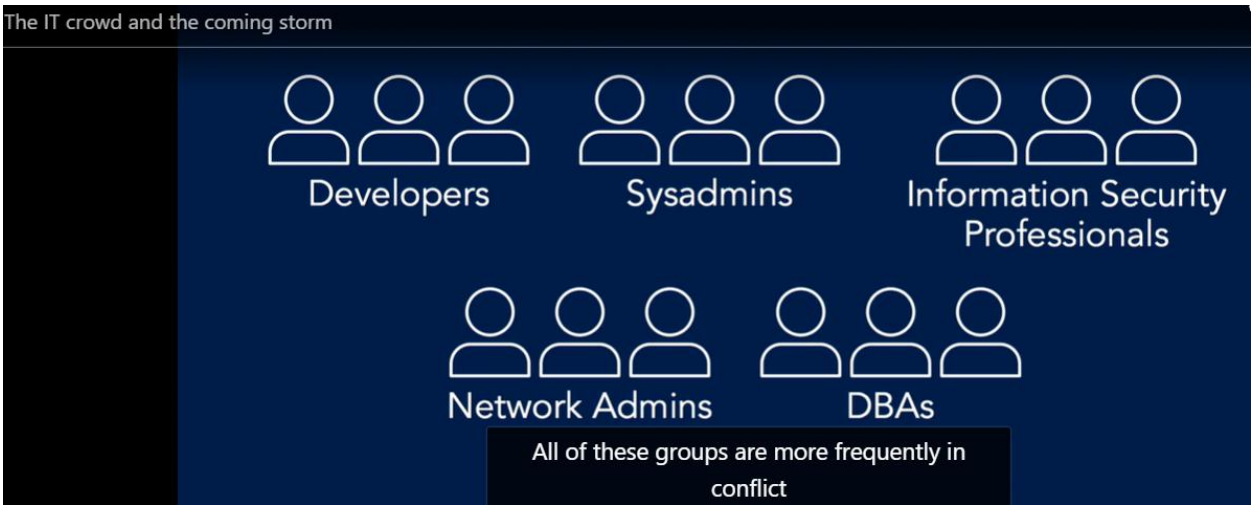
Tool Criteria

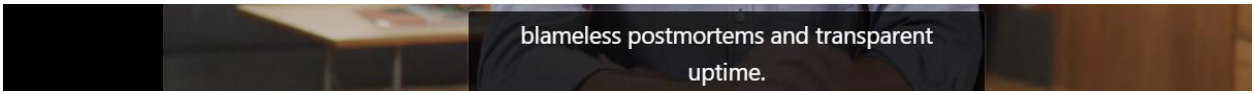
1. Programmable
2. Verifiable
3. Well behaved


in the context of the three pillars of DevOps,

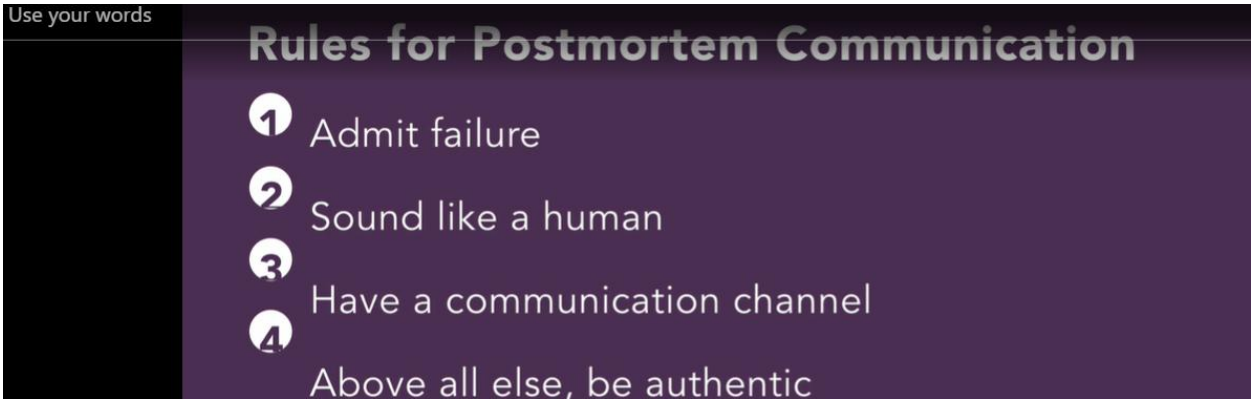

infrastructure automation, continuous delivery,

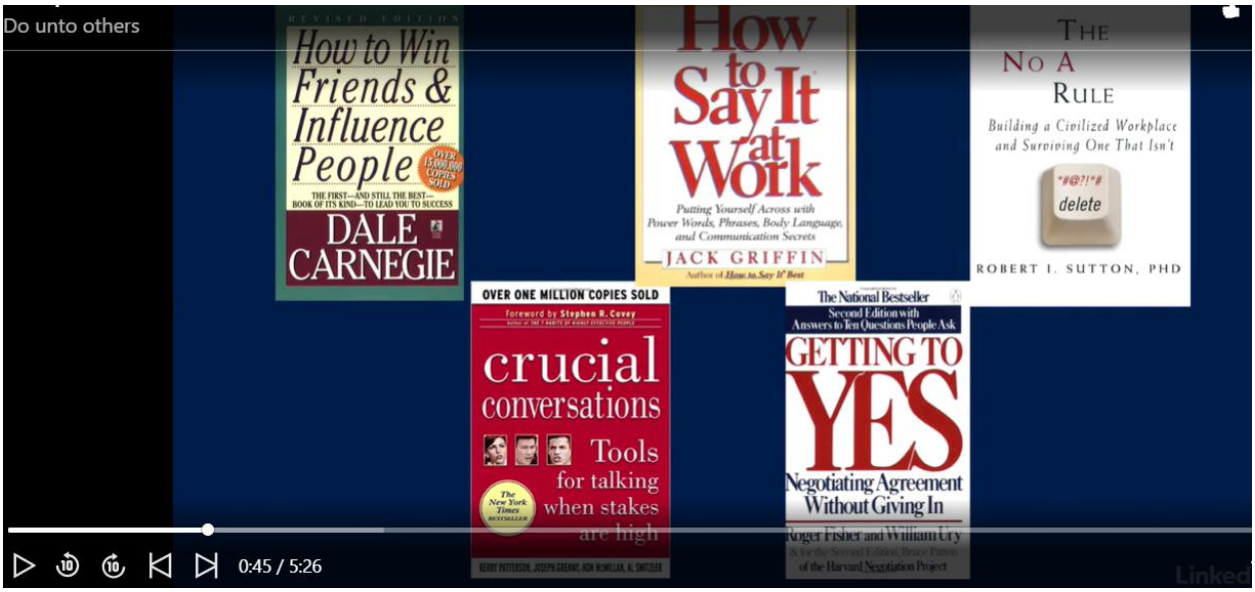

and reliability engineering.


The IT crowd and the coming storm
Developers
Sysadmins
Information Security Professionals
Network Admins
DBAs
All of these groups are more frequently in conflict


blameless postmortems and transparent uptime.


Use your words
Blameless Postmortems
1 It's a meeting!
2 Do it within 48 hours of the incident, if possible
3 Have a third party run it
Use your words
Blameless Postmortems
1 A description of the incident
2 A description of the root cause
3 How the incident was stabilized or fixed
4 A timeline of events, including all actions taken
5 How the incident affected customers
6 Remediations and corrective


Use your words
Rules for Postmortem Communication
1 Admit failure
2 Sound like a human
3 Have a communication channel
4 Above all else, be authentic


Do unto others
How to Win Friends & Influence People
DALE CARNEGIE
How to Say It at Work
JACK GRIFFIN
THE NO A RULE
Building a Civilized Workplace and Surviving One That Isn't
ROBERT I. SUTTON, PHD
OVER ONE MILLION COPIES SOLD
crucial conversations
Tools for talking when stakes are high
The National Bestseller
GETTING TO YES
Negotiating Agreement Without Giving In
0:45 / 5:26


Do unto others
Open It Up
1 Chat rooms
2 Wiki pages
3 Source code (read)
4 Infrastructure
5 Monitoring tools
6 Ticket tracker

Do unto others

Throwing things over walls

The Westrum Model

Pathological (power-oriented)

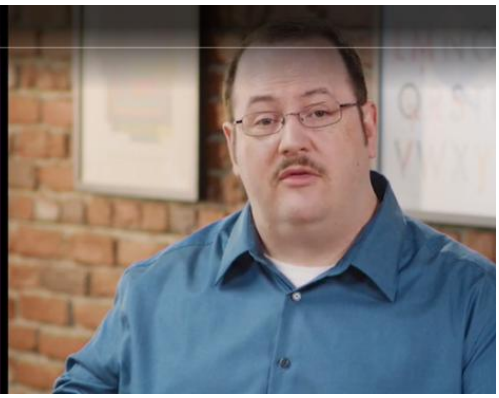
Bureaucratic (rule-oriented)

Generative (performance-oriented)

Throwing things over walls

Management Best Practices

1. Independent, cross-functional teams
2. People first
3. Agile, lean processes

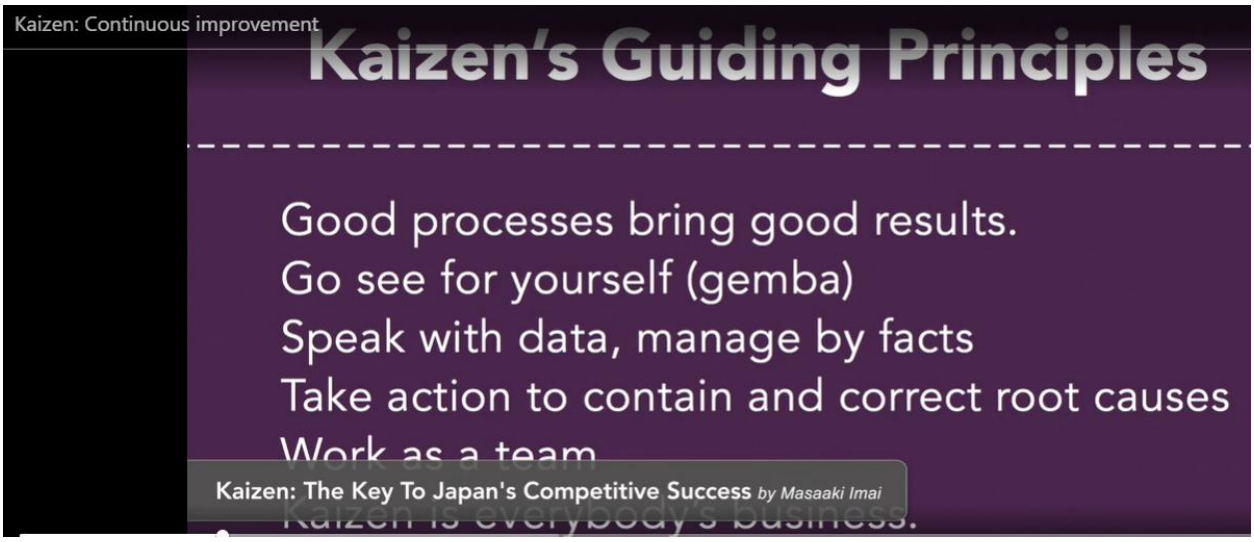


Kaizen: Continuous improvement

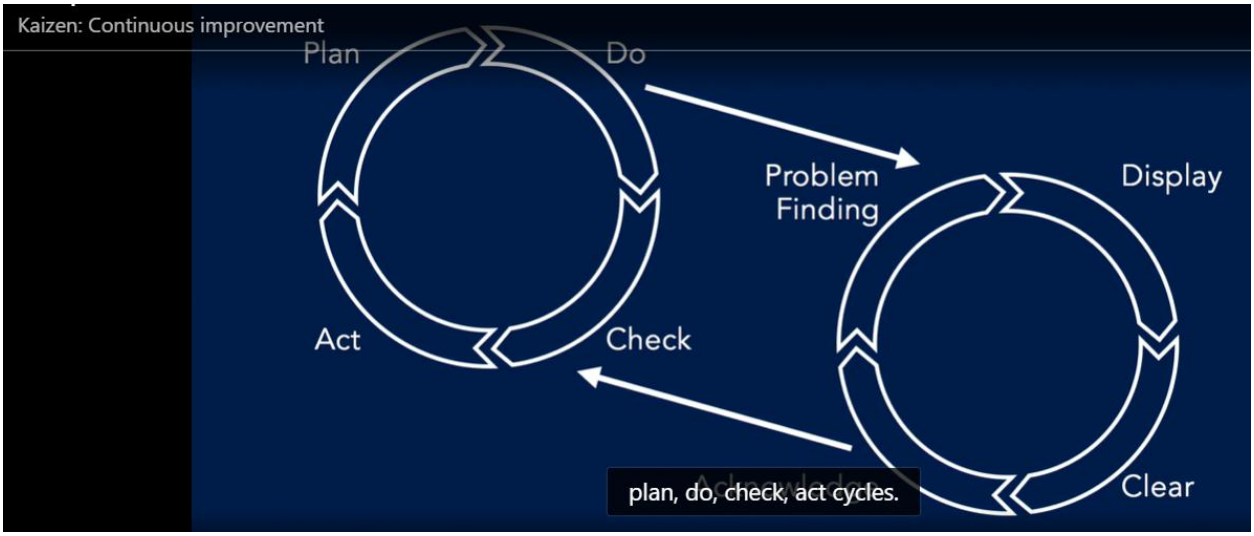
改 善

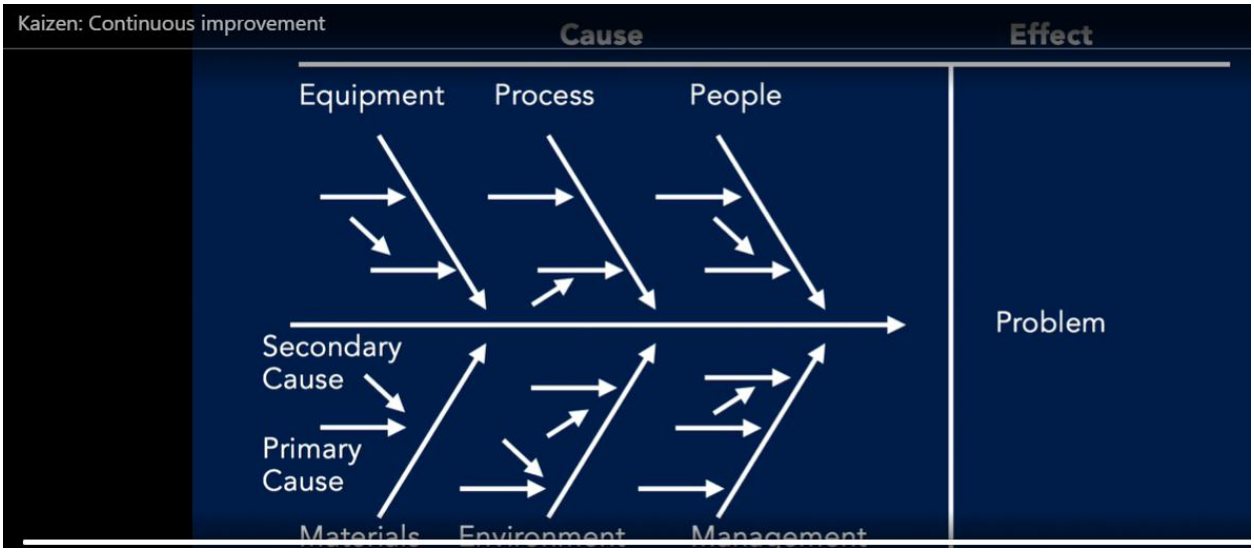
Kai Zen

Continuous Improvement


Kaizen: Continuous improvement
Kaizen's Guiding Principles
Good processes bring good results.
Go see for yourself (gemba)
Speak with data, manage by facts
Take action to contain and correct root causes
Work as a team
Kaizen: The Key To Japan's Competitive Success by Masaaki Imai

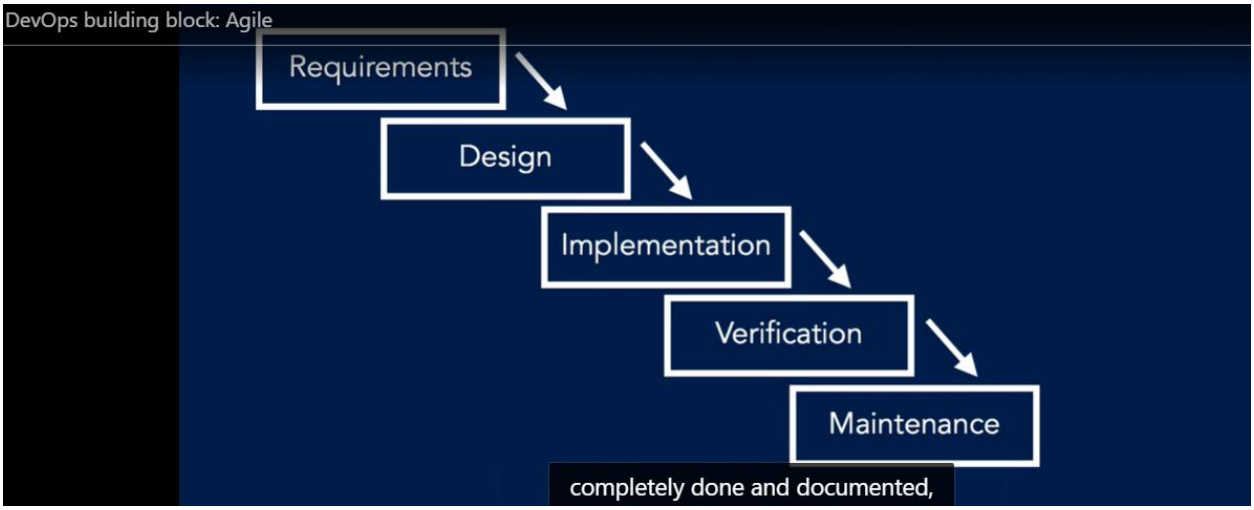

Kaizen: Continuous improvement
現 場
Gem Ba
Locus
We might best translate it as locust.


Kaizen: Continuous improvement
Plan
Do
Check
Act
Problem Finding
Display
Clear
plan, do, check, act cycles.

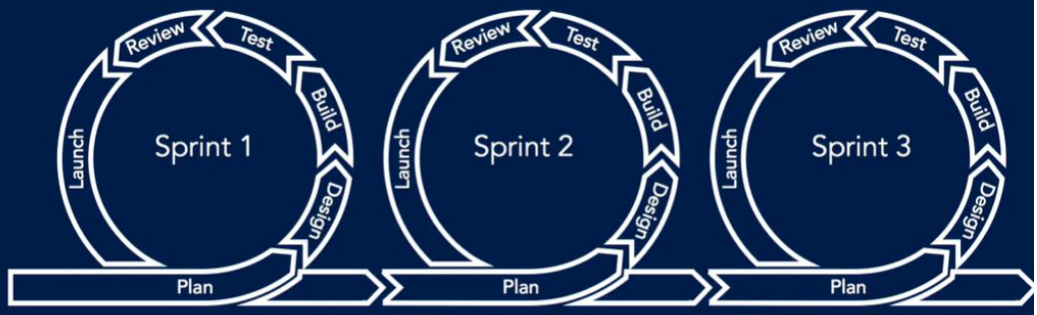

Kaizen: Continuous improvement
Cause
Effect
Equipment
Process
People
Problem
Secondary Cause
Primary Cause
Materials
Environment
Management


DevOps building block: Agile
Manifesto for Agile Software Development
Through this work we have come to value…
• Individuals and interactions over processes and tools
• Working software over comprehensive documentation
• Customer collaboration over contract negotiation
• Responding to change over following a plan

waterfall


DevOps building block: Agile
Requirements
Design
Implementation
Verification
Maintenance
completely done and documented,

DevOps building block: Agile



- And Agile has proven its benefits.

DevOps building block: Agile

Agile Manifesto

We follow these principles:

- Our highest priority is to satisfy the customer through early and continuous delivery of valuable software.
- Business people and developers must work together daily throughout the project.

however you'll see what's missing:

DevOps building block: Agile

Agile Manifesto

- Working software is the primary measure of progress.
- Agile processes promote sustainable development.
- The sponsors, developers, and users should be able to maintain a constant pace indefinitely.

but it wasn't customary to bring system administrators

- Also the manifesto doesn't mention anything

about the last part of the software delivery pipeline,

where infrastructure is built and the apps

are deployed and maintained in production.

DevOps building block: Agile

DevOps Manifesto

We follow these principles:

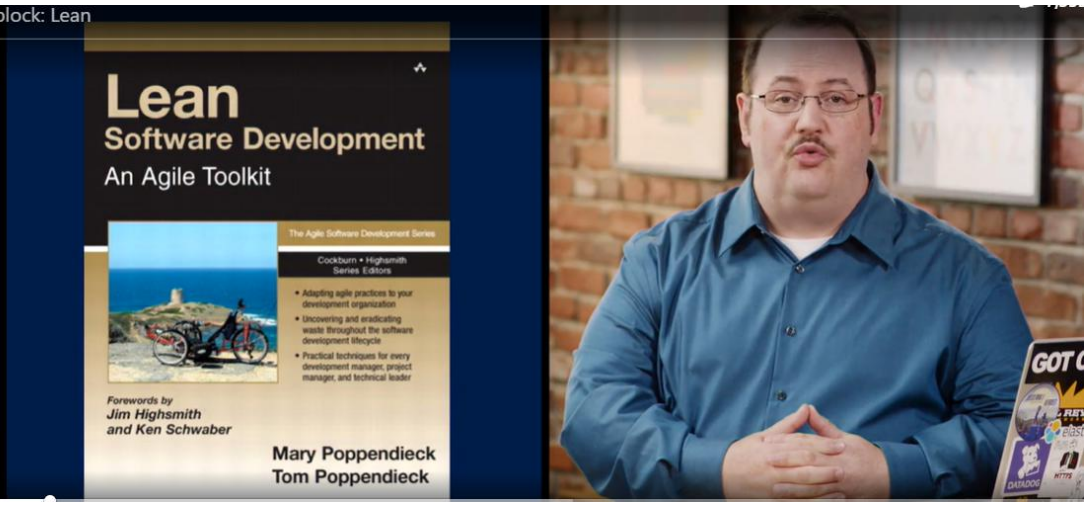
- Our highest priority is to satisfy the customer through early and continuous delivery of valuable functionality.
- Business people, operations, and developers must work together daily throughout the project.

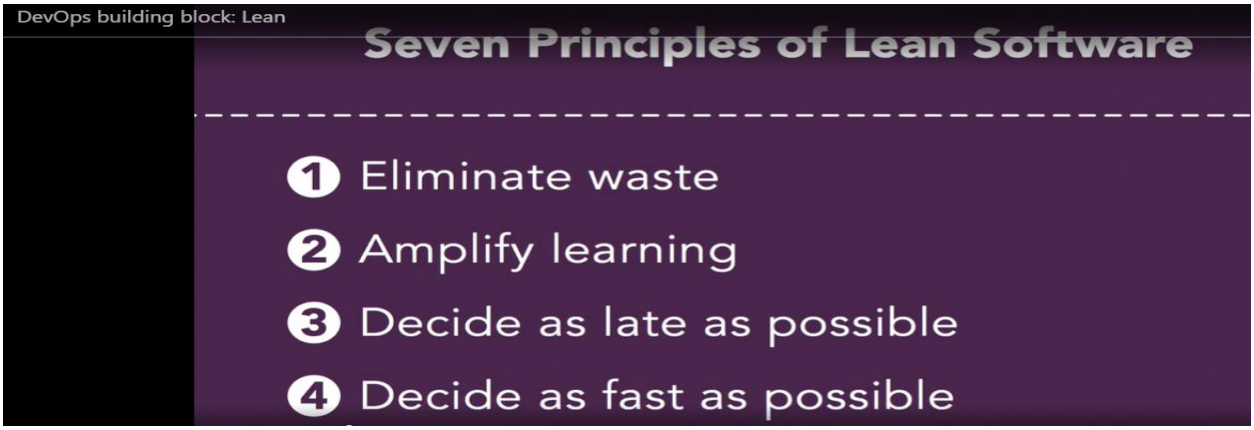
DevOps building block: Agile

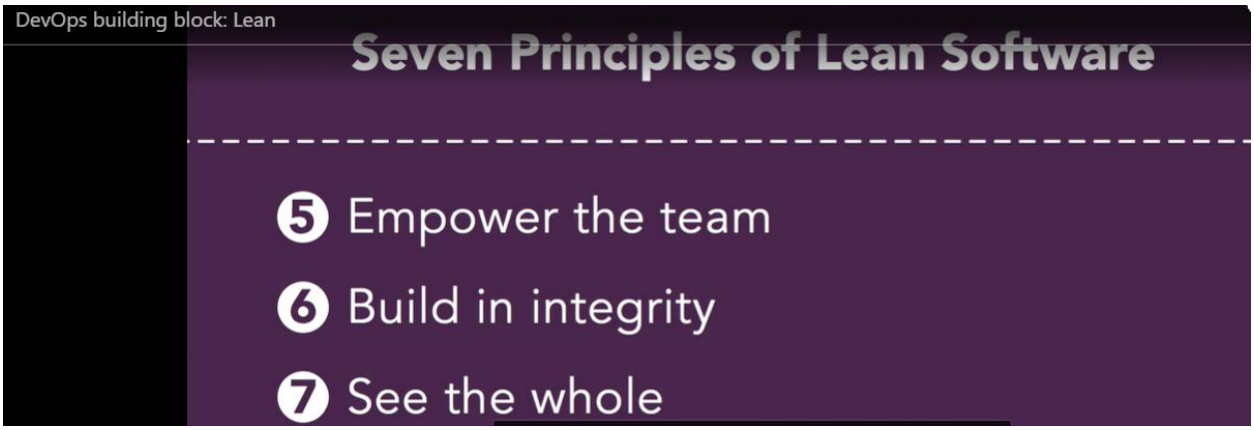
DevOps Manifesto

- Working software successfully delivered by sound systems is the primary measure of progress.
- Agile processes promote sustainable development. The sponsors, developers, operations, and users should be able to maintain a constant pace indefinitely.

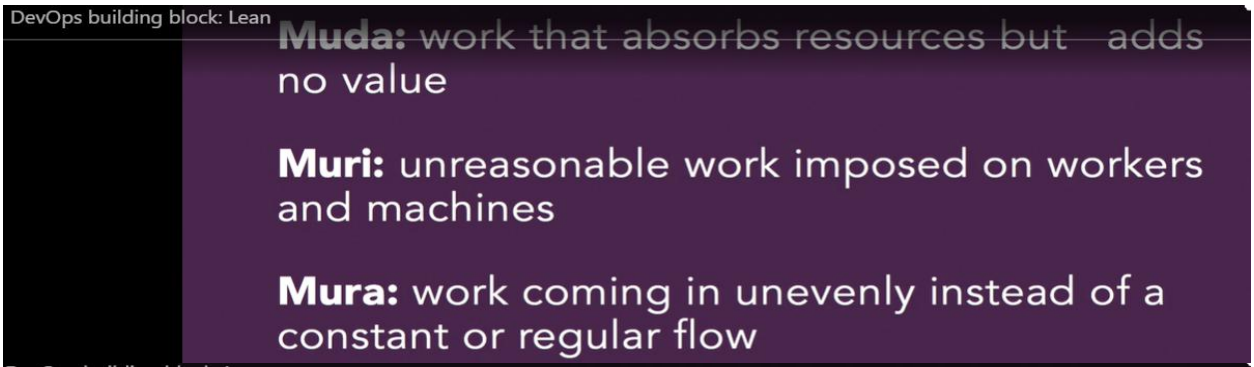
DevOps building block: Lean



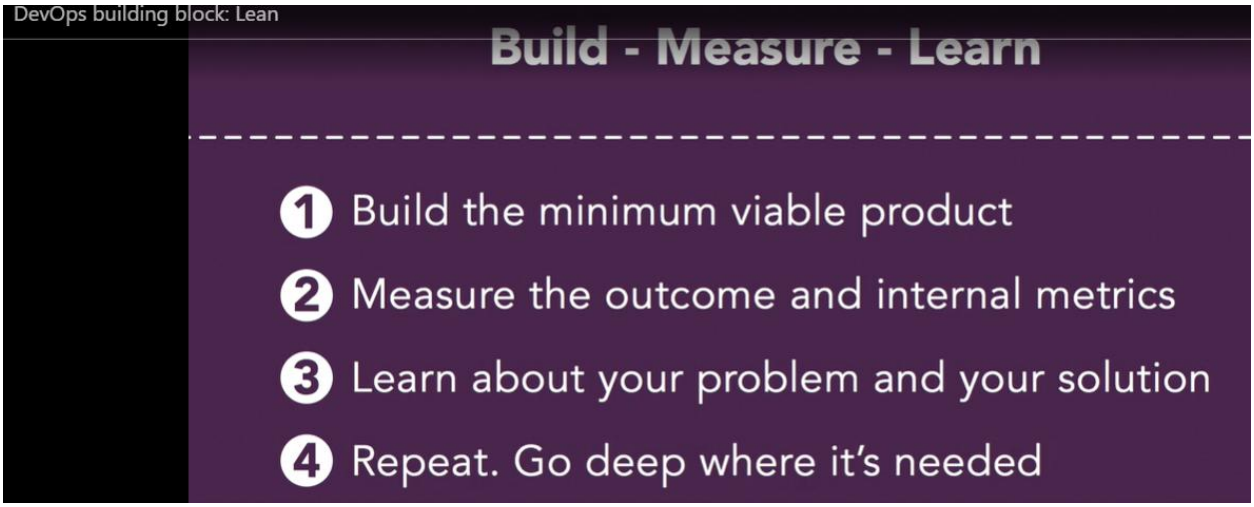


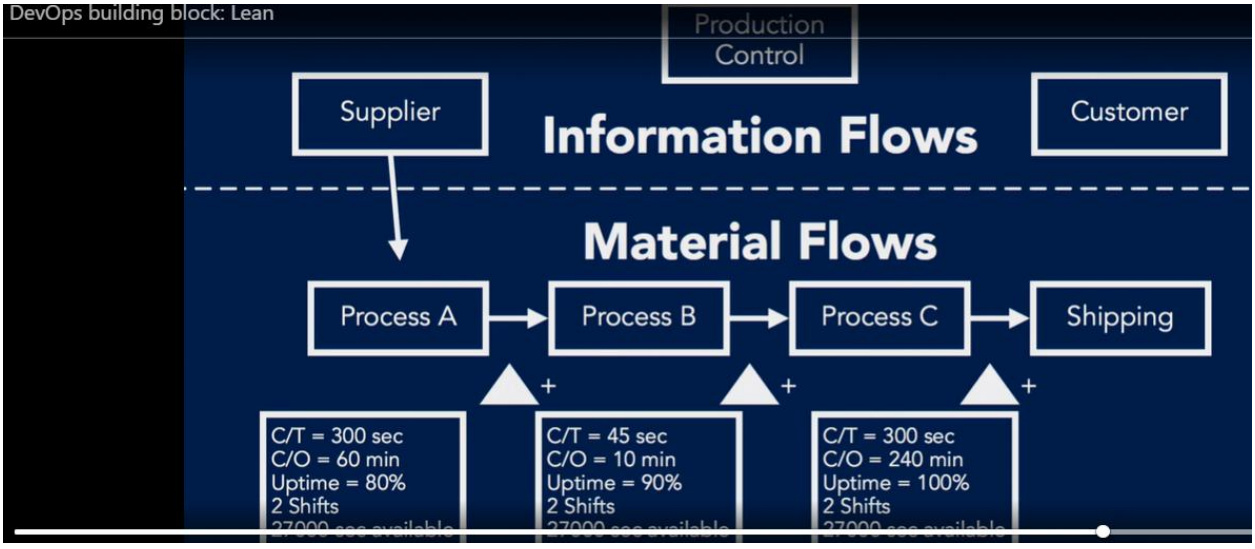


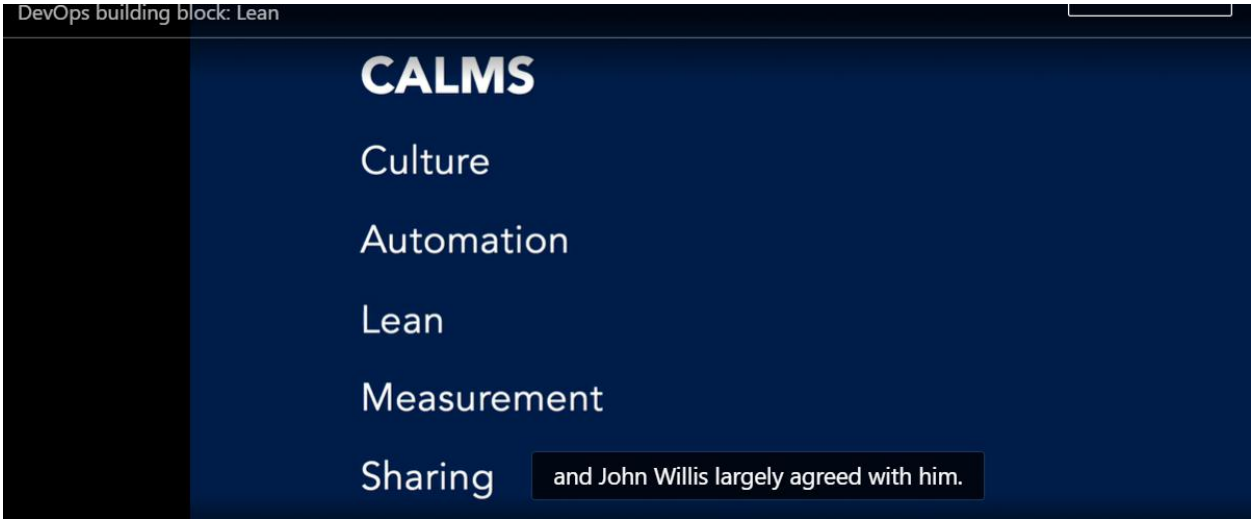
Three types of waste



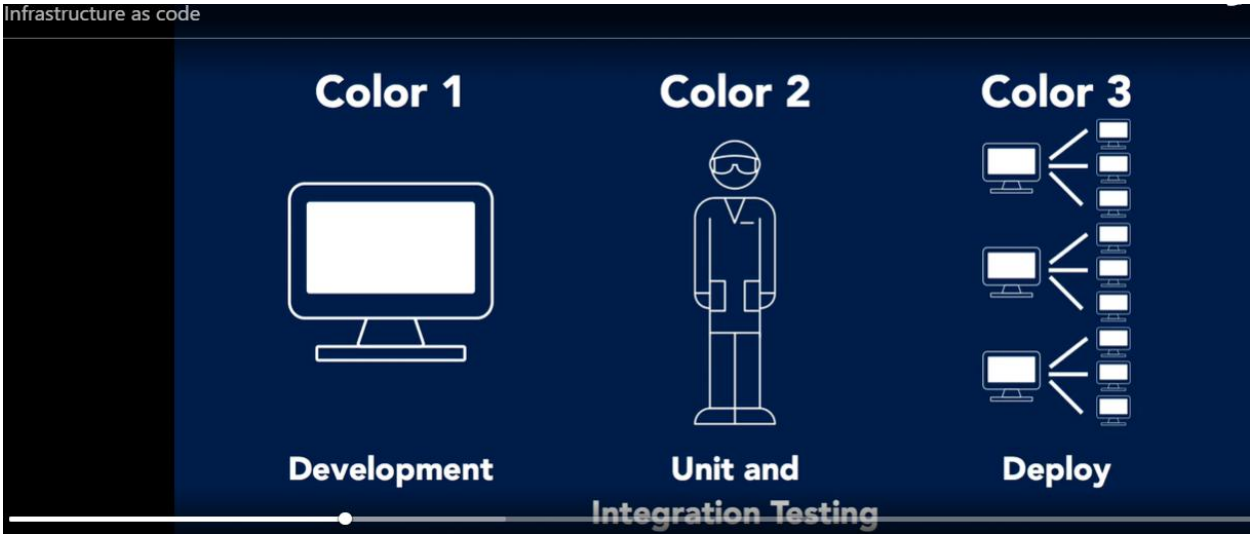

DevOps building block: Lean
The Seven Wastes of Software
5 Waste #5 - Delays
6 Waste #6 - Task switching
7 Waste #7 - Defects


DevOps building block: Lean
Build - Measure - Learn
1 Build the minimum viable product
2 Measure the outcome and internal metrics
3 Learn about your problem and your solution
4 Repeat. Go deep where it's needed

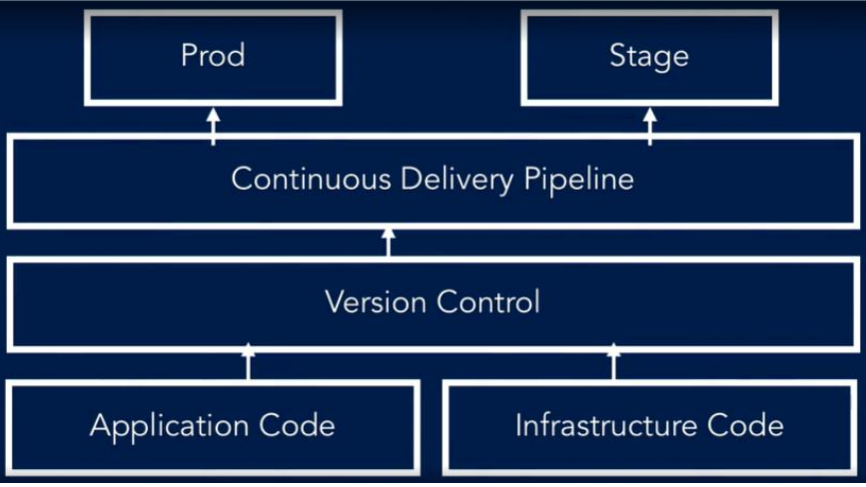

DevOps building block: Lean
Production Control
Supplier
Information Flows
Customer
Material Flows
Process A
Process B
Process C
Shipping
C/T = 300 sec
C/O = 60 min
Uptime = 80%
2 Shifts
C/T = 45 sec
C/O = 10 min
Uptime = 90%
2 Shifts
C/T = 300 sec
C/O = 240 min
Uptime = 100%
2 Shifts


DevOps building block: Lean
CALMS
Culture
Automation
Lean
Measurement
Sharing
and John Willis largely agreed with him.


ITIL, ITSM, and the SDLC
ITIL Phases
1 Service strategy
2 Service design
3 Service transition
4 Service operation


Infrastructure as code
Color 1
Color 2
Color 3
Development
Unit and
Integration Testing
Deploy

Infrastructure as code



Golden image to foil ball

Provisioning

Making a server ready for operation, including hardware, OS, system services, and network connectivity

Golden image to foil ball

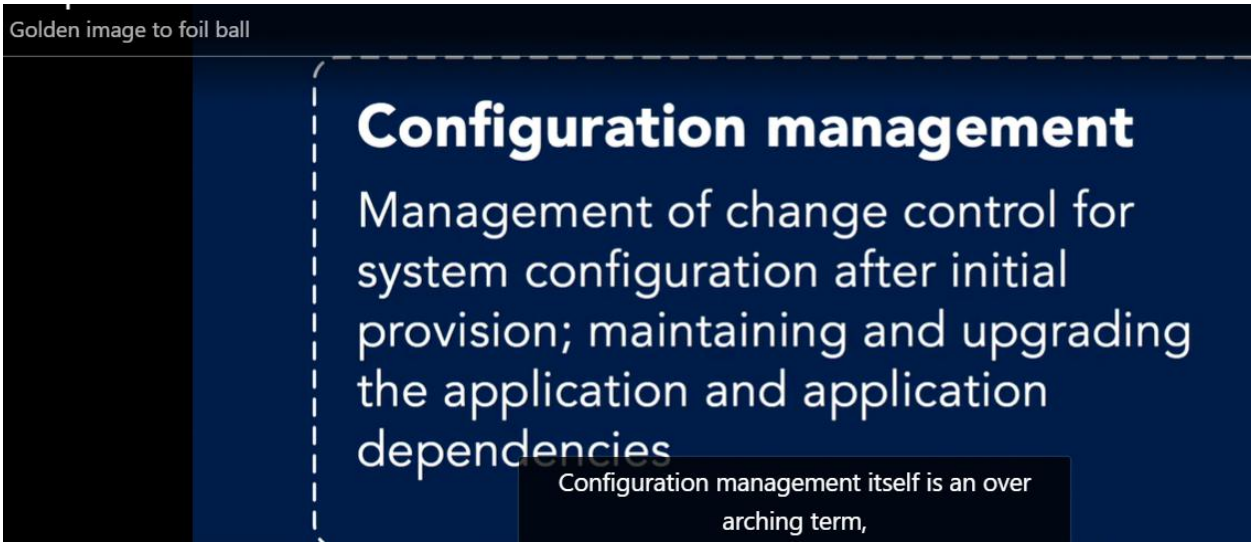
Deployment

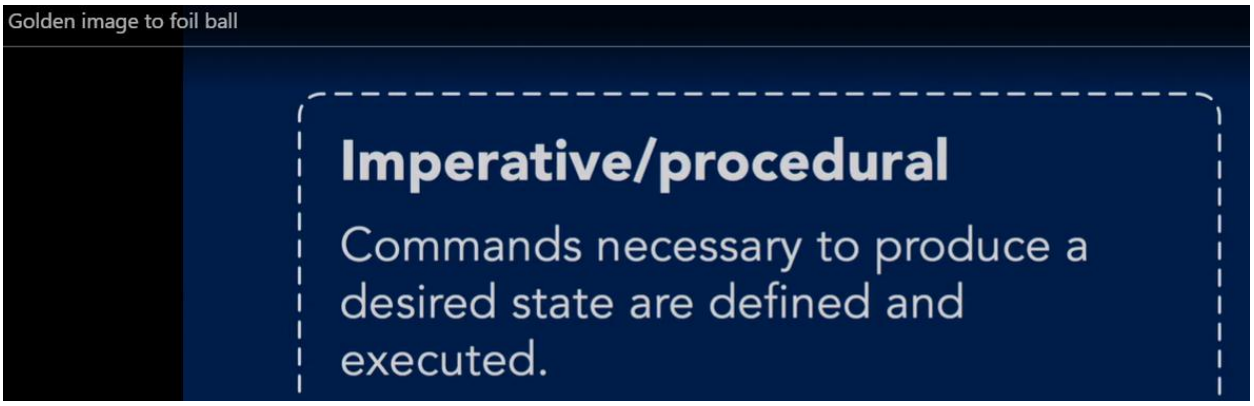
Automatically deploying and upgrading applications on a server

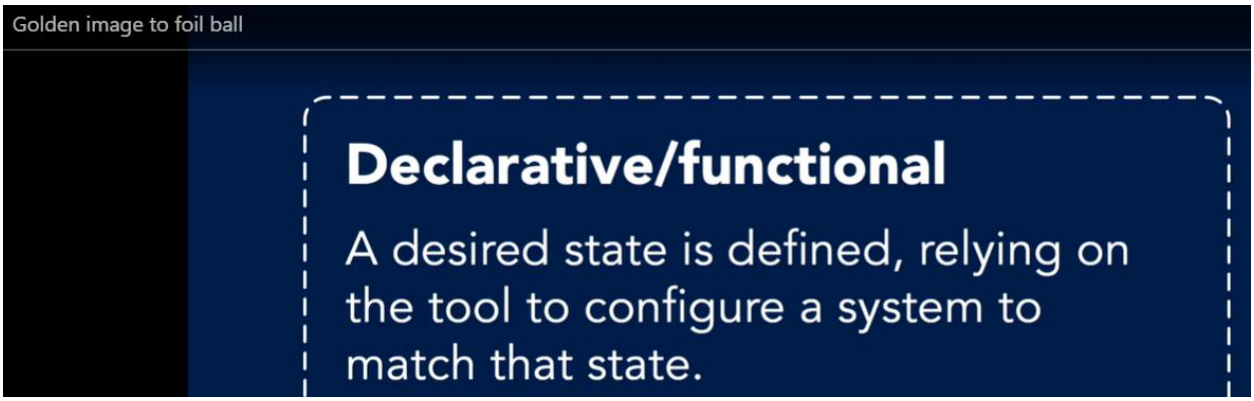
Golden image to foil ball

Orchestration

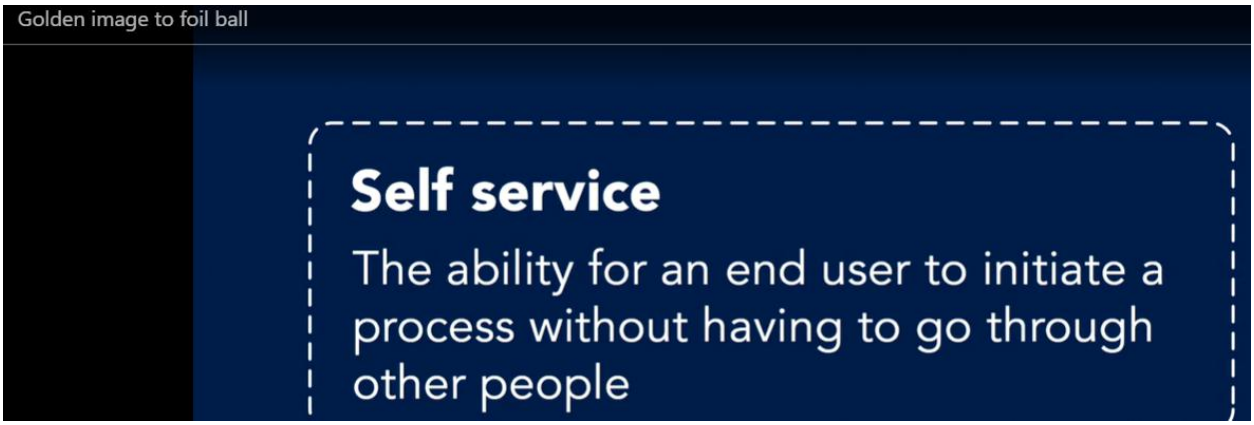
Performing coordinated operations across multiple systems

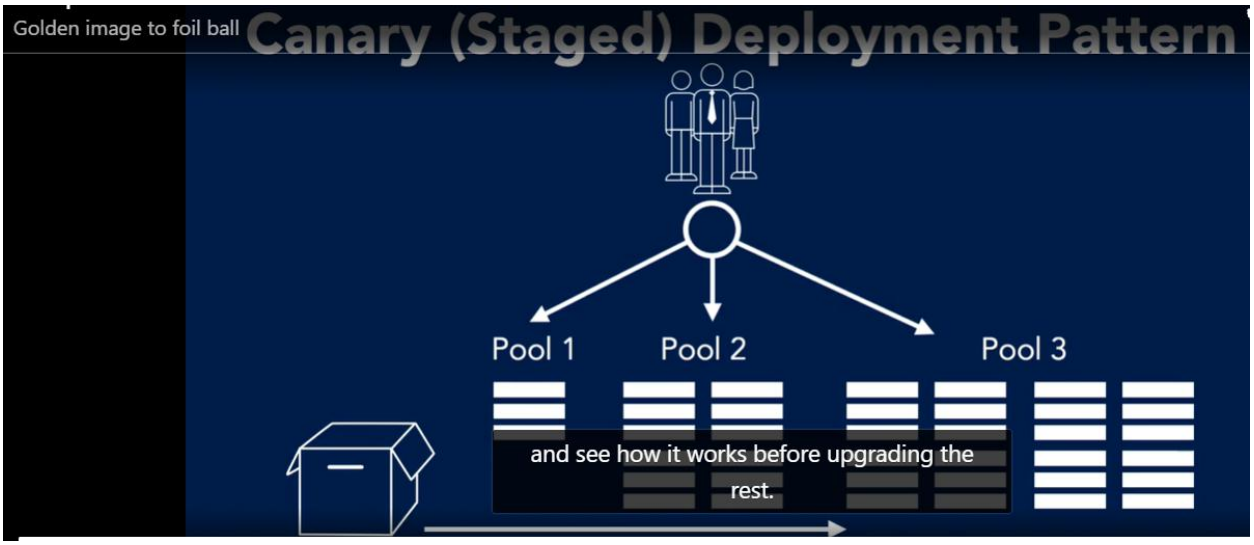

Golden image to foil ball
Configuration management
Management of change control for system configuration after initial provision; maintaining and upgrading the application and application dependencies
Configuration management itself is an over arching term,

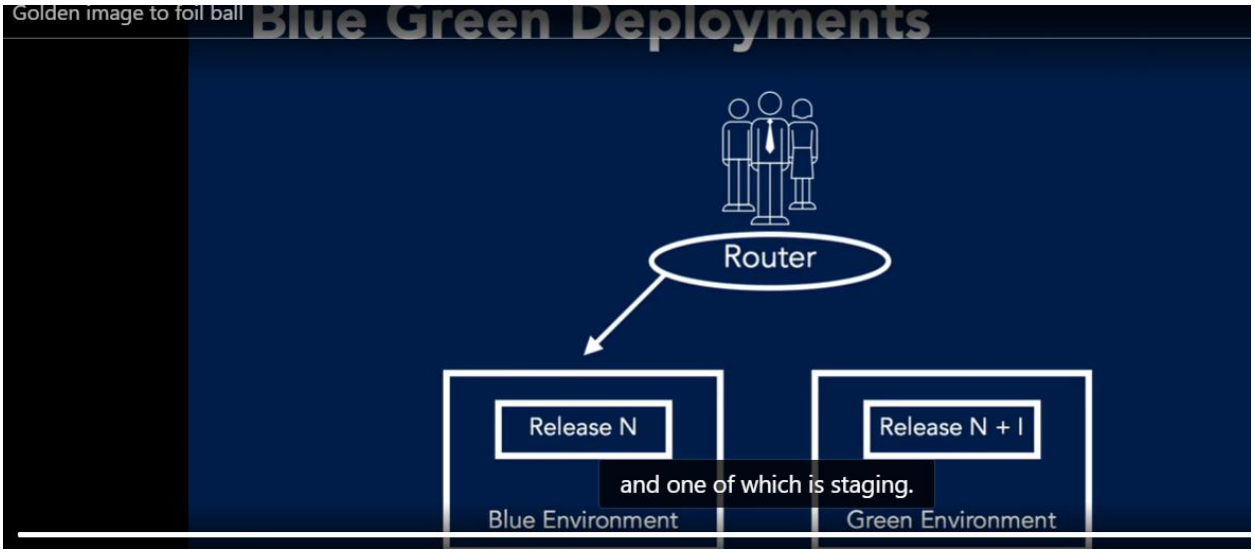

Golden image to foil ball
Imperative/procedural
Commands necessary to produce a desired state are defined and executed.

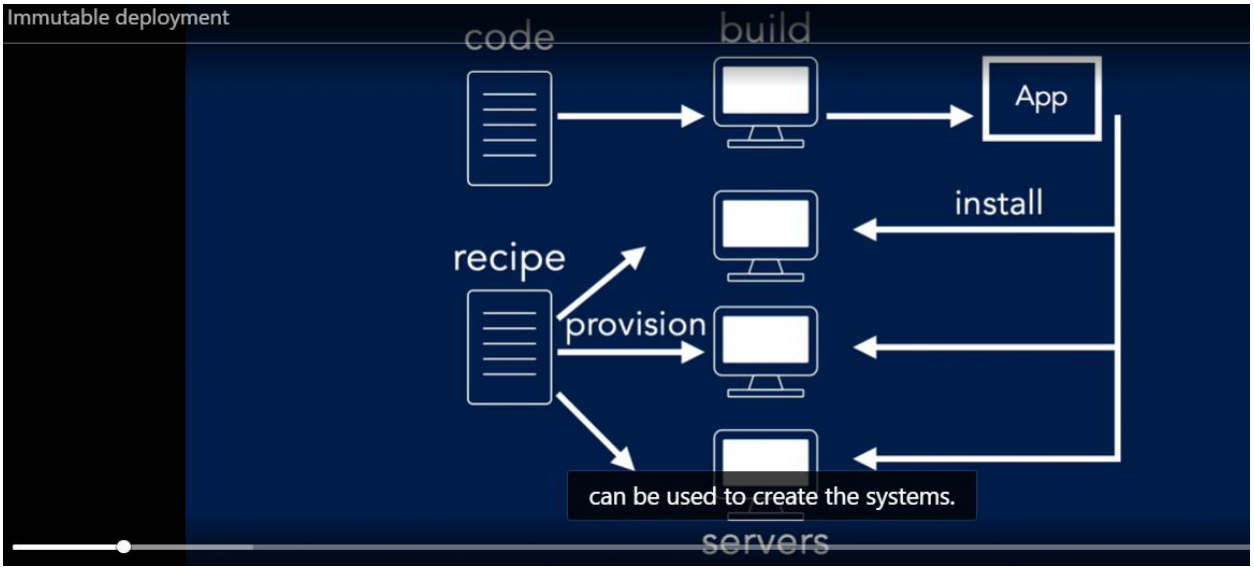

Golden image to foil ball
Declarative/functional
A desired state is defined, relying on the tool to configure a system to match that state.

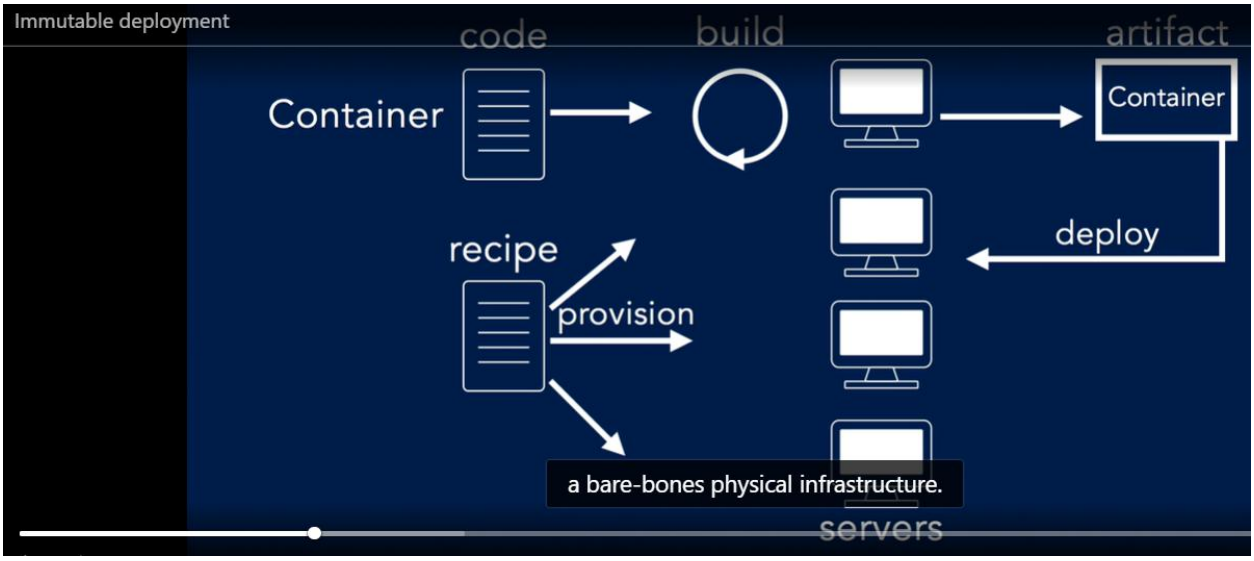

Golden image to foil ball
Idempotent
The ability to execute repeatedly, resulting in the same outcome

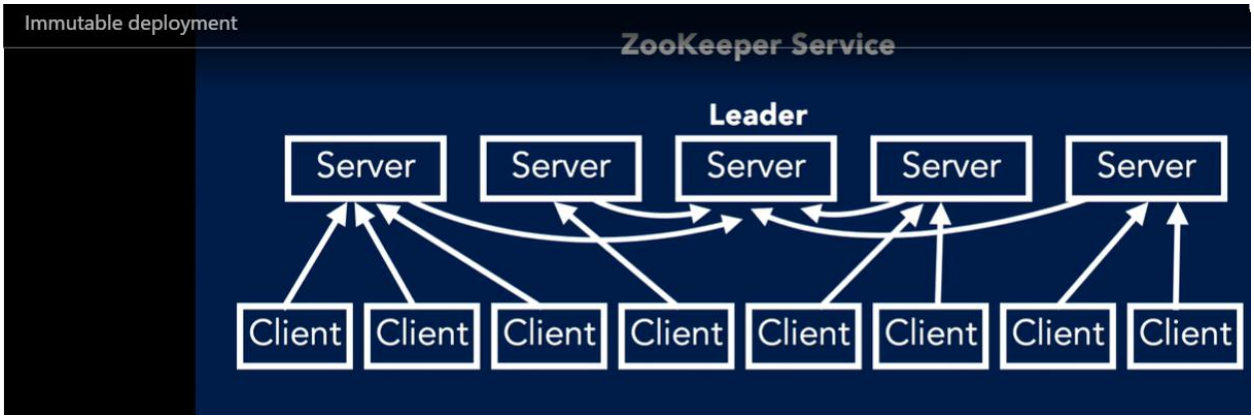

Golden image to foil ball
Self service
The ability for an end user to initiate a process without having to go through other people

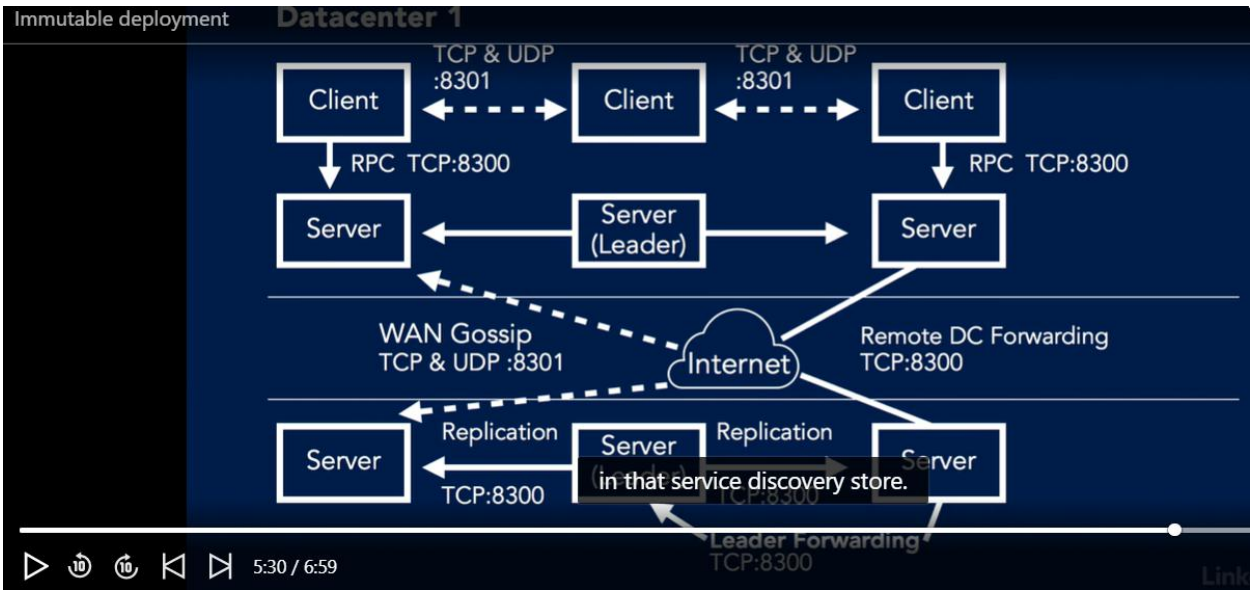

Golden image to foil ball
Canary (Staged) Deployment Pattern
Pool 1
Pool 2
Pool 3
and see how it works before upgrading the rest.

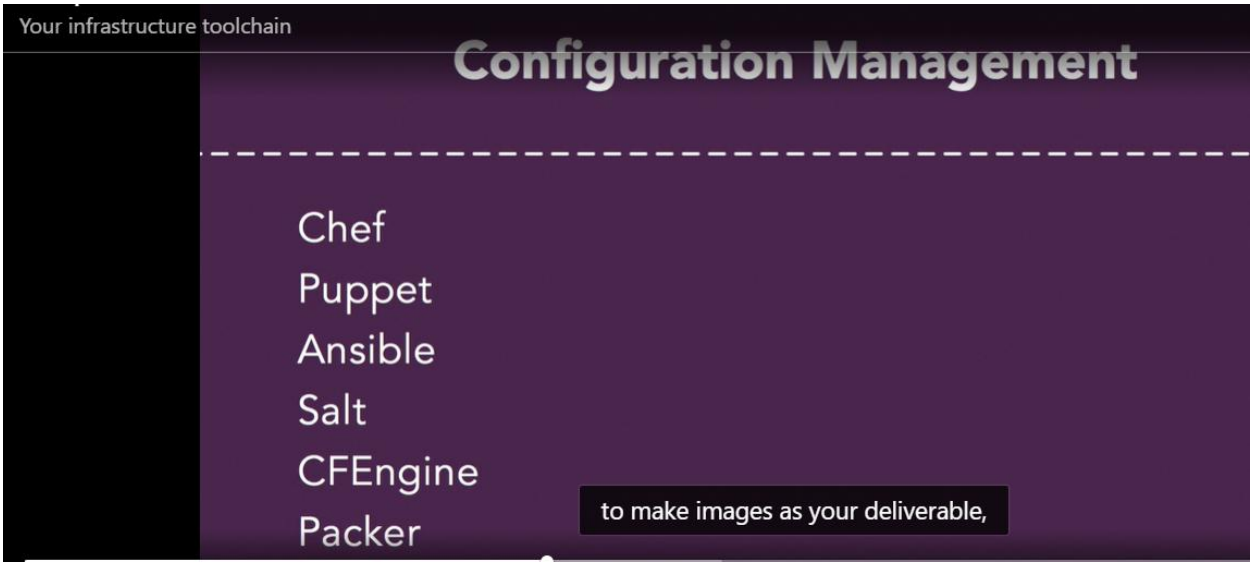

Golden image to foil ball
Blue Green Deployments
Router
Release N
Release N + 1
and one of which is staging.
Blue Environment
Green Environment

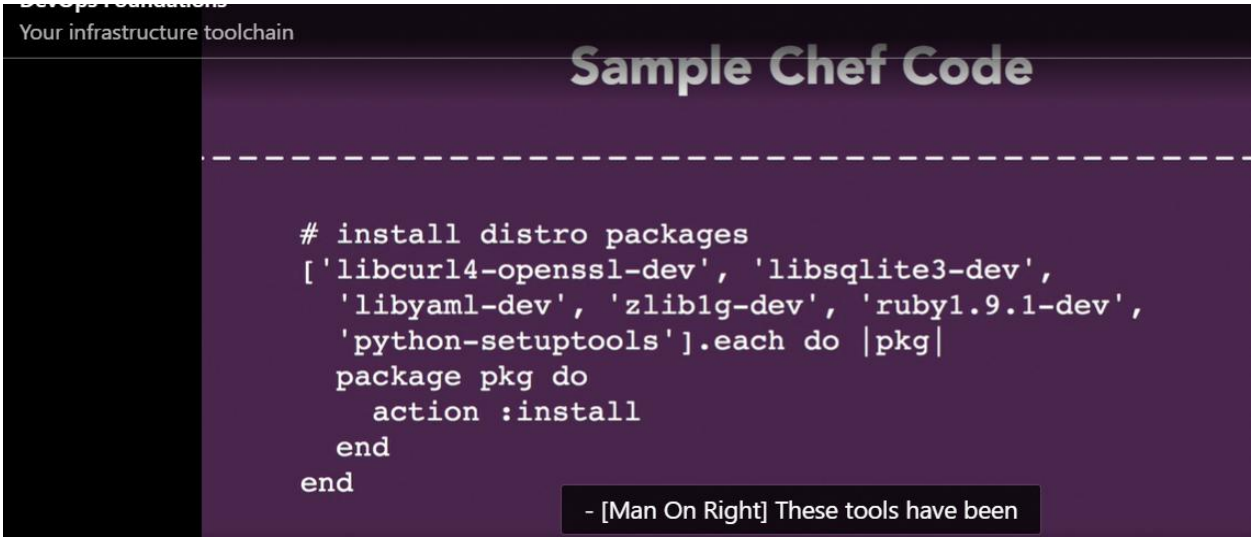

Immutable deployment
code
build
App
install
recipe
provision
can be used to create the systems.
servers


Immutable deployment
code
build
artifact
Container
Container
deploy
recipe
provision
a bare-bones physical infrastructure.
servers


Immutable deployment
ZooKeeper Service
Leader
Server
Server
Server
Server
Server
Client
Client
Client
Client
Client
Client
Client
Client


Immutable deployment
Datacenter 1
Client
TCP & UDP :8301
Client
TCP & UDP :8301
Client
RPC TCP:8300
RPC TCP:8300
Server
Server (Leader)
Server
WAN Gossip TCP & UDP :8301
Internet
Remote DC Forwarding TCP:8300
Server
Replication
Server
Replication
Server
TCP:8300
in that service discovery store.
Leader Forwarding TCP:8300
5:30 / 6:59


Your infrastructure toolchain
Configuration Management
Chef
Puppet
Ansible
Salt
CFEngine
Packer
to make images as your deliverable,


Your infrastructure toolchain
Sample Chef Code
install distro packages
['libcurl4-openssl-dev', 'libsqlite3-dev',
'libyaml-dev', 'zlib1g-dev', 'ruby1.9.1-dev',
'python-setuptools'].each do |pkg|
package pkg do
action :install
end
end
- [Man On Right] These tools have been

Your infrastructure toolchain

Chef Cookbook

Rubocop | Foodcritic | Chefspec | KitchenCI

Your infrastructure toolchain

Services Directory Tools

etcd
ZooKeeper
Consul

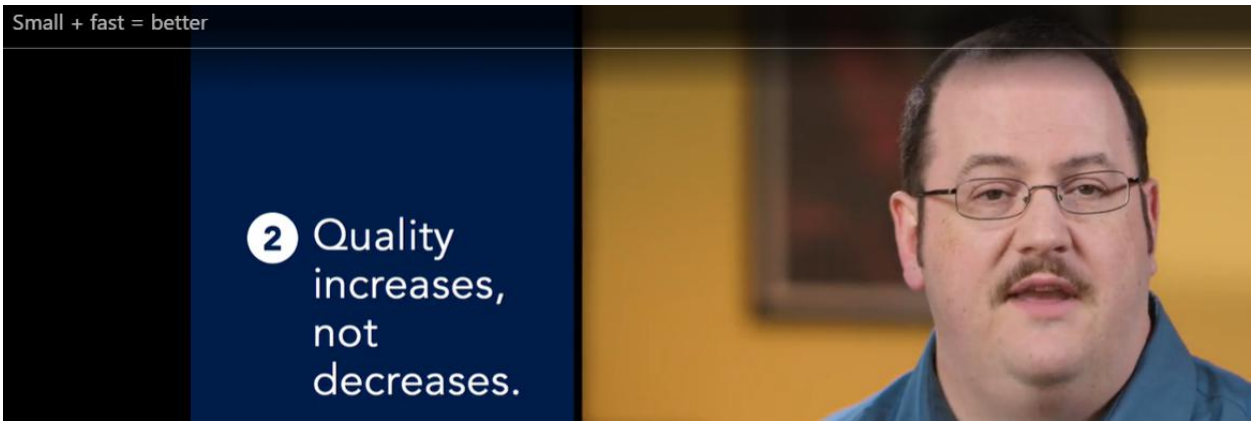
Your infrastructure toolchain

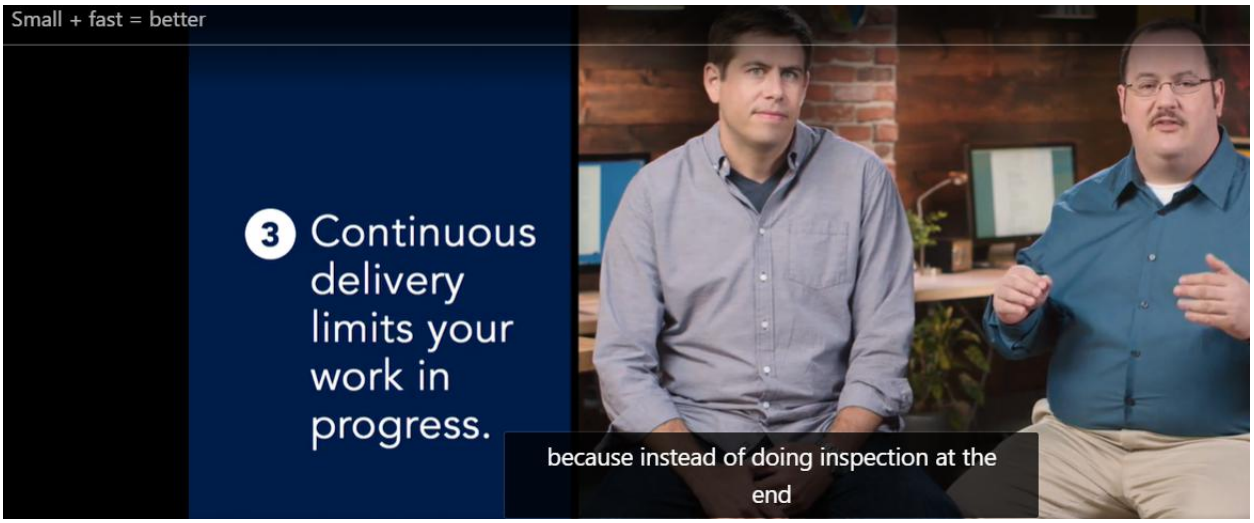
Private Container Services

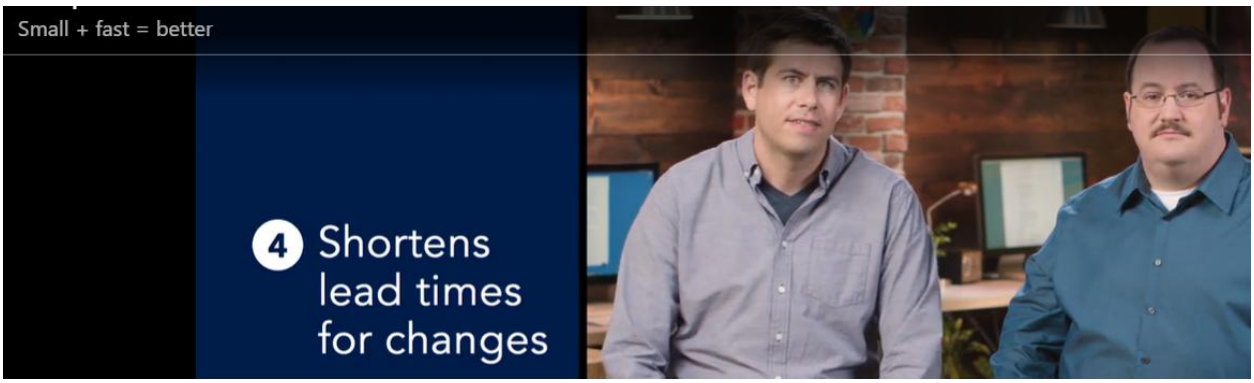
Rancher
Google Cloud Platform
Amazon Web Services ECS

Small + fast = better

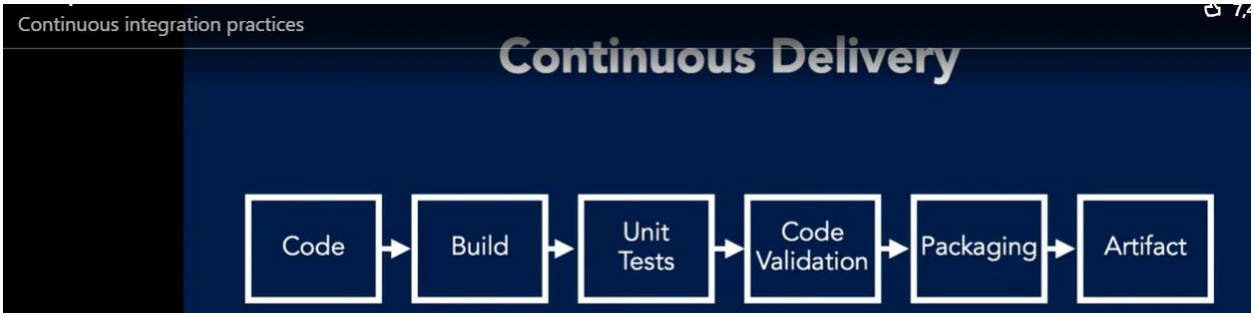
1 Time to market goes down.

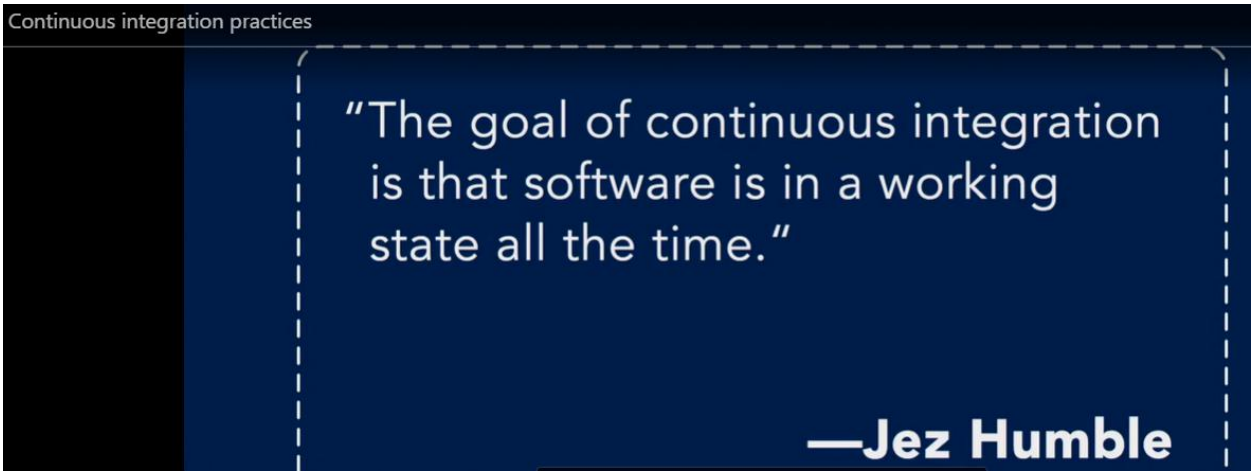

Small + fast = better
2 Quality increases, not decreases.


Small + fast = better
3 Continuous delivery limits your work in progress.
because instead of doing inspection at the end

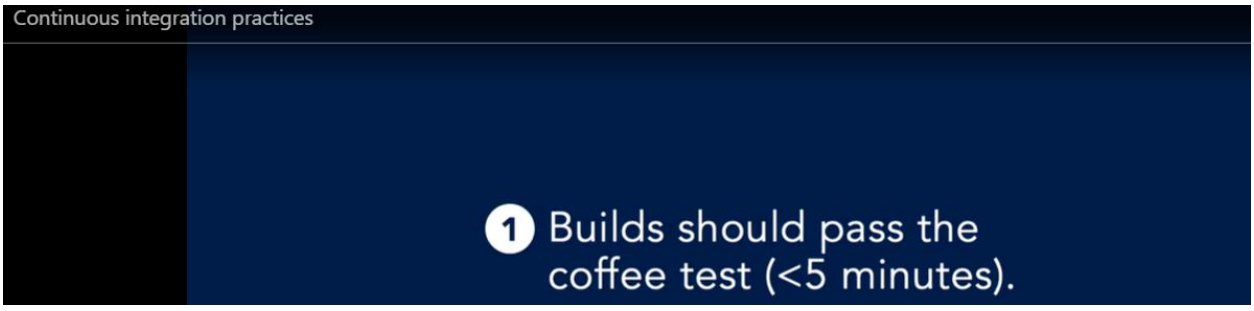

Small + fast = better
4 Shortens lead times for changes


Small + fast = better
5 Improves mean time to recover


Continuous integration practices
Continuous Delivery
Code
Build
Unit Tests
Code Validation
Packaging
Artifact


Continuous integration practices
"The goal of continuous integration is that software is in a working state all the time."
—Jez Humble


Continuous integration practices
Continuous Deploy
Continuous Delivery
Continuous Integration


Continuous integration practices
1 Builds should pass the coffee test (<5 minutes).

Continuous integration practices

2 Commit really small bits

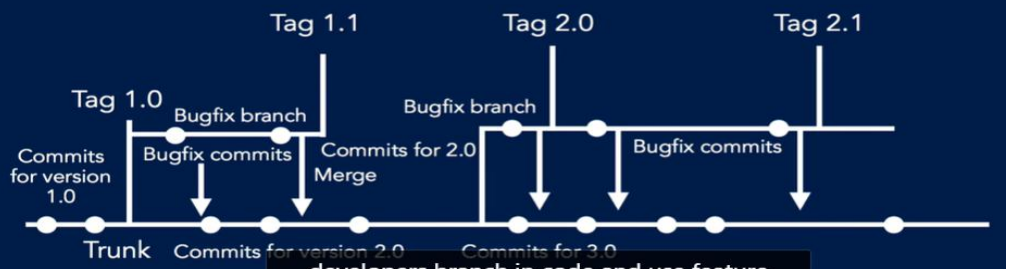
Continuous integration practices

3 Don't leave the build broken

Continuous integration practices

4 Use a trunk-based development flow

Continuous integration practices



developers branch in code and use feature flags.

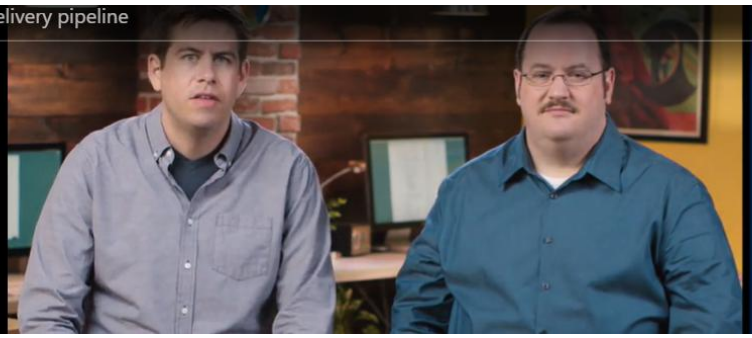
Continuous integration practices

5 Don't allow flaky tests. Fix them!

Continuous integration practices

6 The build should return a status, a log, and an artifact.

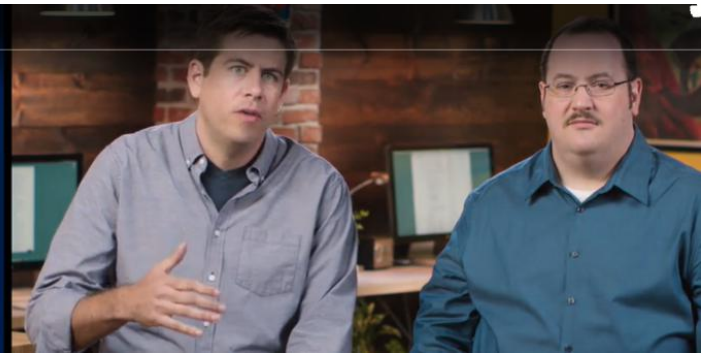
The continuous delivery pipeline



1 Only build artifacts once

The continuous delivery pipeline

2 Artifacts should be immutable.



The continuous delivery pipeline

1 Code

2 Version continued

3 CI system creates build

4 Builds going into shelf

The deploy goes to stage

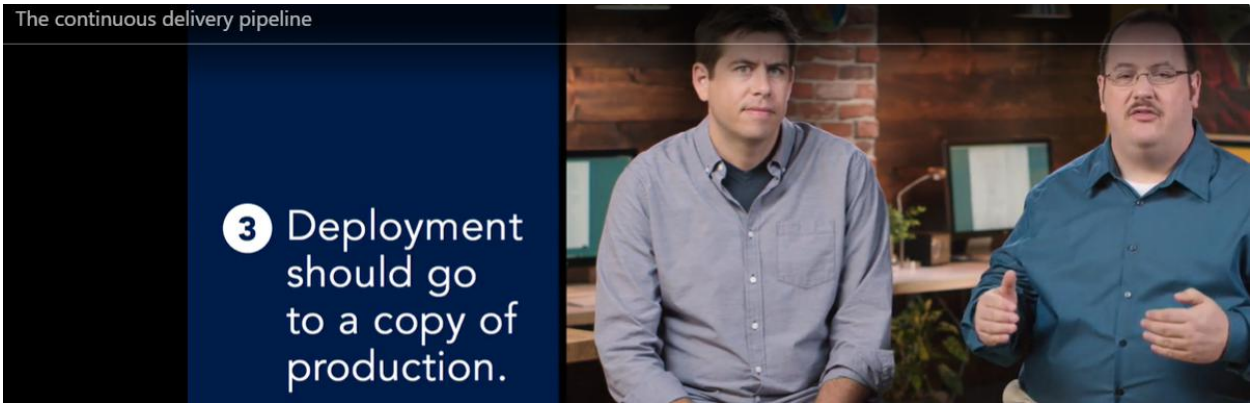
5 Staging

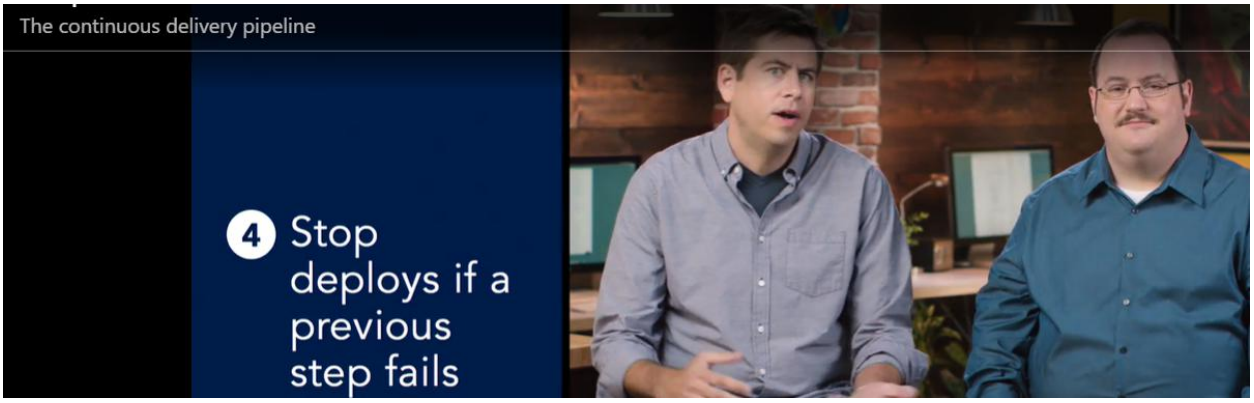
6 Testing

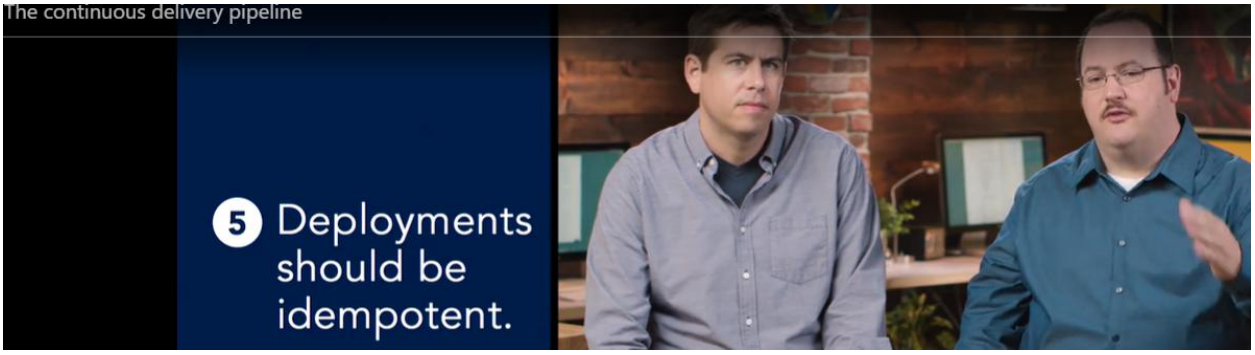
7 Ready for Prod

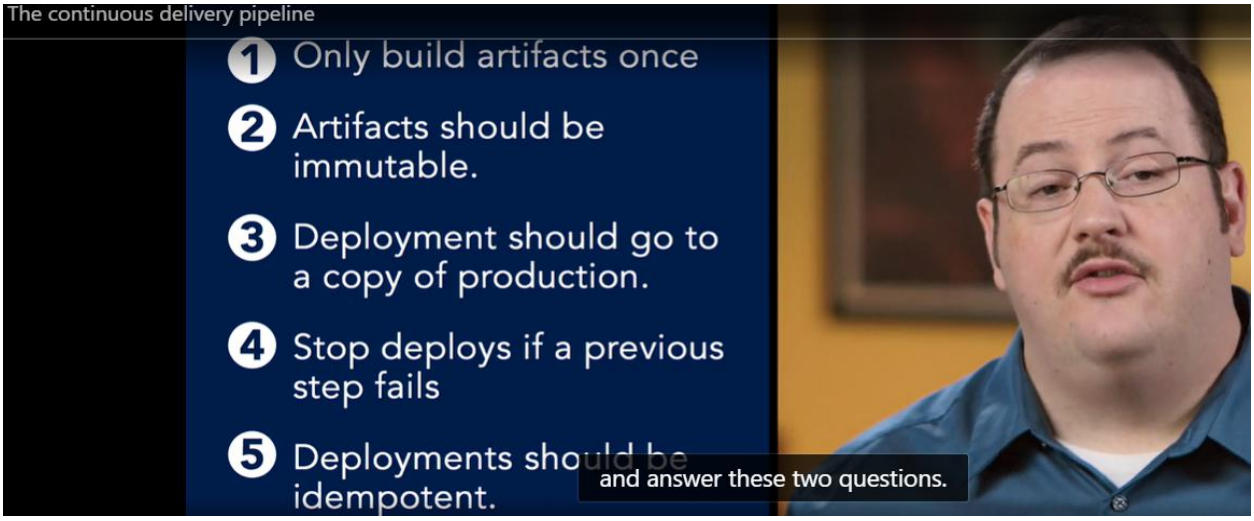
8 Deploy to Prod

When you want, you deploy the artifacts


The continuous delivery pipeline
3 Deployment should go to a copy of production.

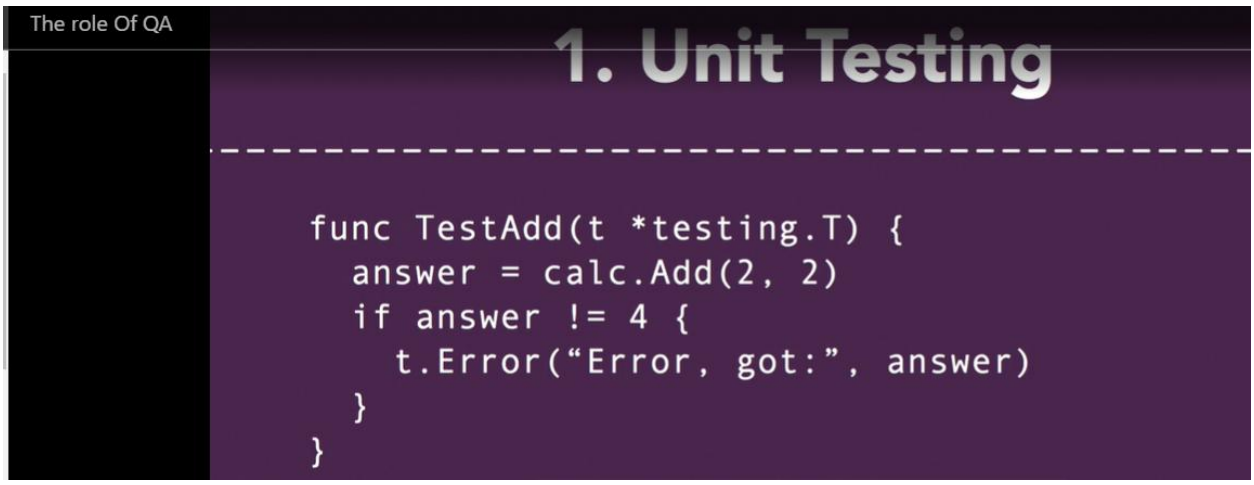

The continuous delivery pipeline
4 Stop deploys if a previous step fails

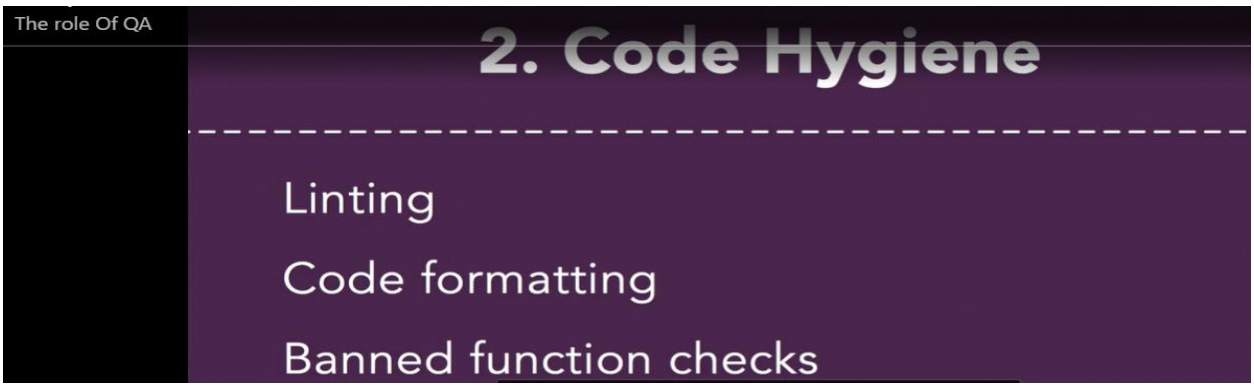

The continuous delivery pipeline
5 Deployments should be idempotent.


The continuous delivery pipeline
1 Only build artifacts once
2 Artifacts should be immutable.
3 Deployment should go to a copy of production.
4 Stop deploys if a previous step fails
5 Deployments should be idempotent.
and answer these two questions.

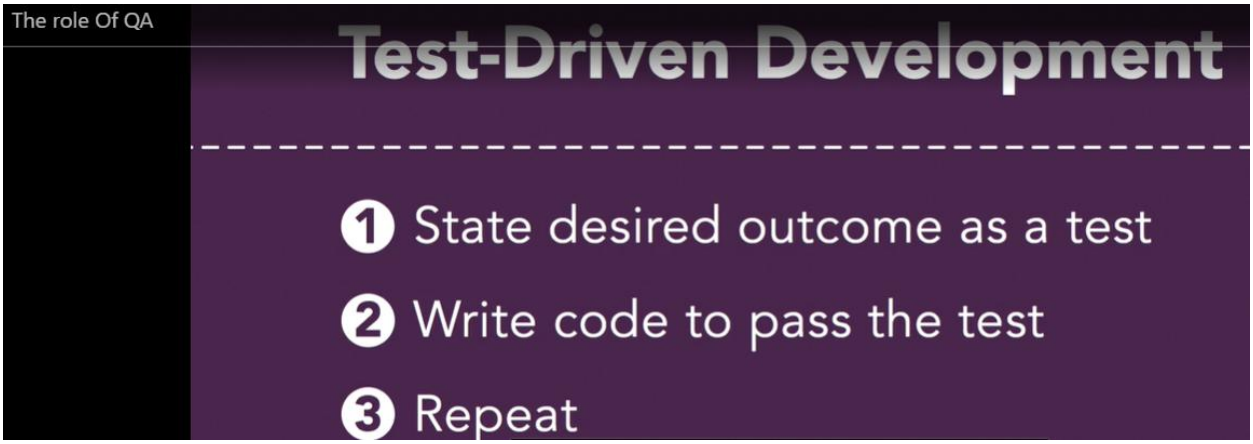

The role Of QA
Types of Testing
1 Unit testing
2 Code hygiene
3 Integration testing
4 Security testing


The role Of QA
Types of Testing
5 TDD/BDD/ATDD
6 Infrastructure testing
7 Performance testing

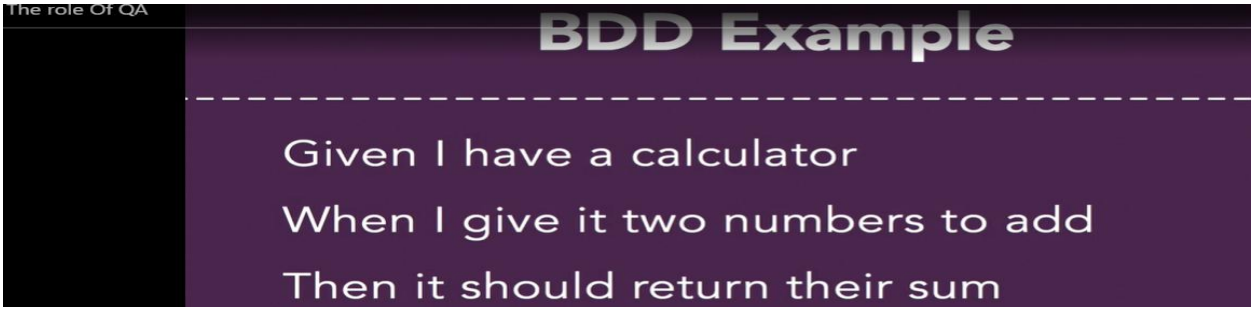

The role Of QA
1. Unit Testing
func TestAdd(t *testing.T) {
answer = calc.Add(2, 2)
if answer != 4 {
t.Error("Error, got:", answer)
}
}

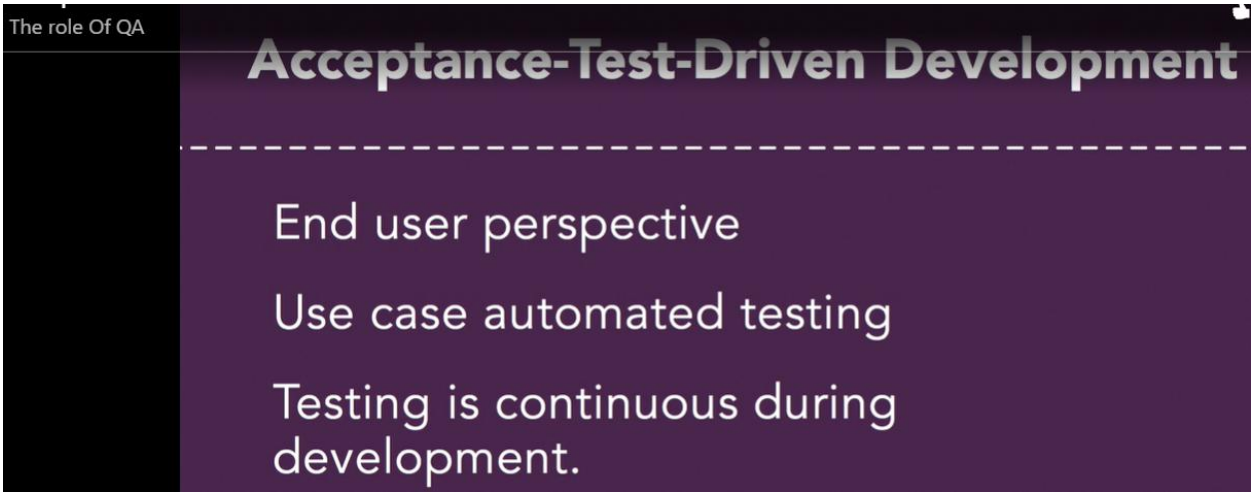

The role Of QA
2. Code Hygiene
Linting
Code formatting
Banned function checks

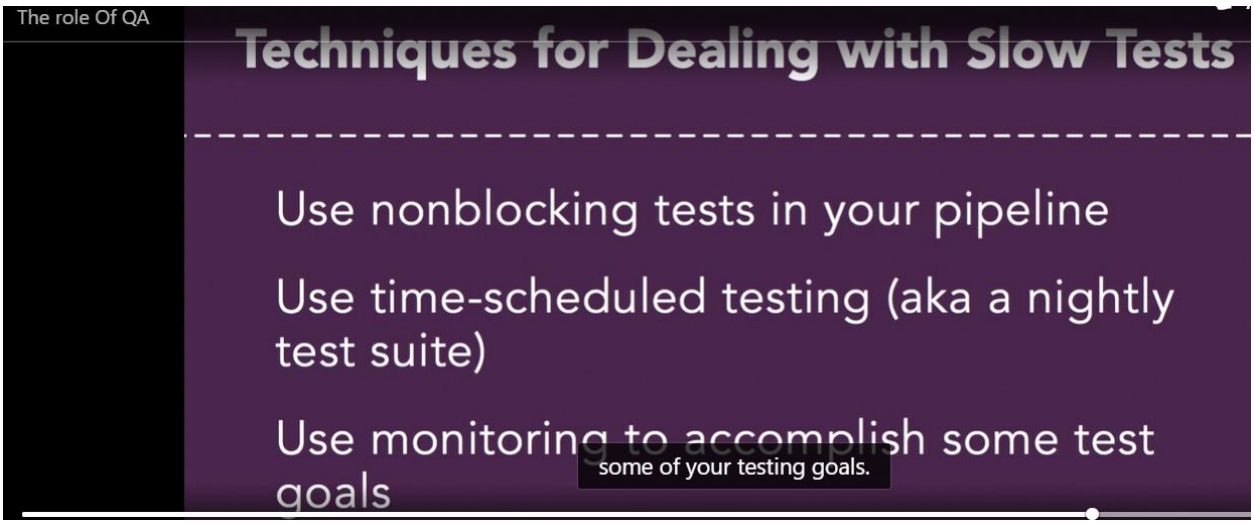

The role Of QA
4. TDD/BDD/ATDD
Test-driven development
Behavior-driven development
Acceptance-test-driven development

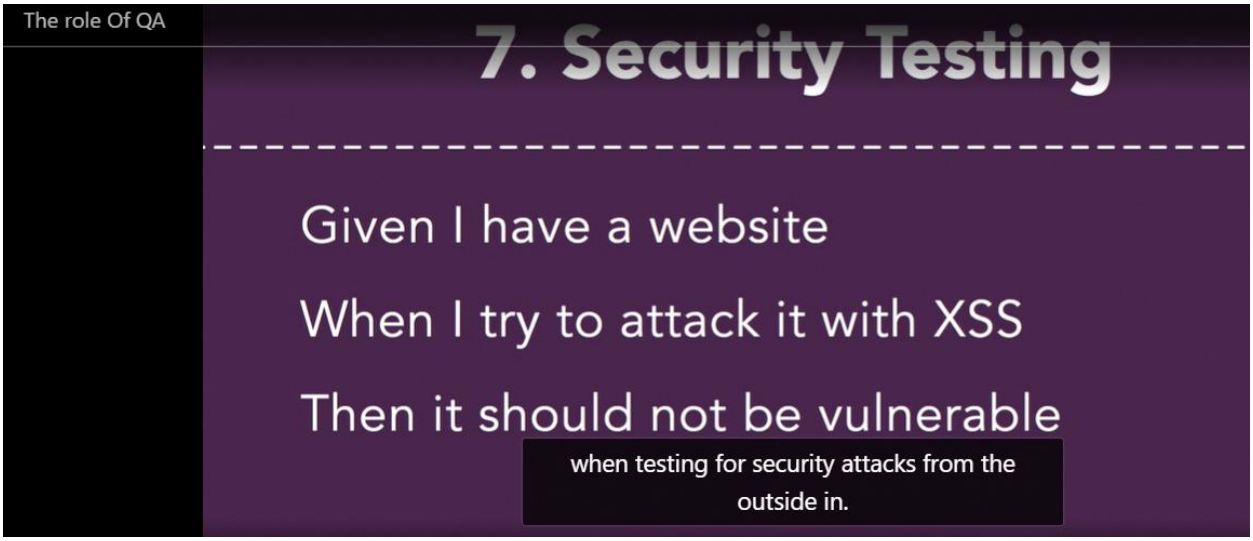

The role Of QA
Test-Driven Development
1 State desired outcome as a test
2 Write code to pass the test
3 Repeat


The role Of QA
Behavior-Driven Development
Work with stakeholders
Describe business functionality
Tests are based on natural language descriptions.
with the business stakeholder to describe


The role Of QA
BDD Example
Given I have a calculator
When I give it two numbers to add
Then it should return their sum


The role Of QA
Acceptance-Test-Driven Development
End user perspective
Use case automated testing
Testing is continuous during development.


The role Of QA
Techniques for Dealing with Slow Tests
Use nonblocking tests in your pipeline
Use time-scheduled testing (aka a nightly test suite)
Use monitoring to accomplish some test goals
some of your testing goals.


The role Of QA
7. Security Testing
Given I have a website
When I try to attack it with XSS
Then it should not be vulnerable
when testing for security attacks from the outside in.

Your CI toolchain

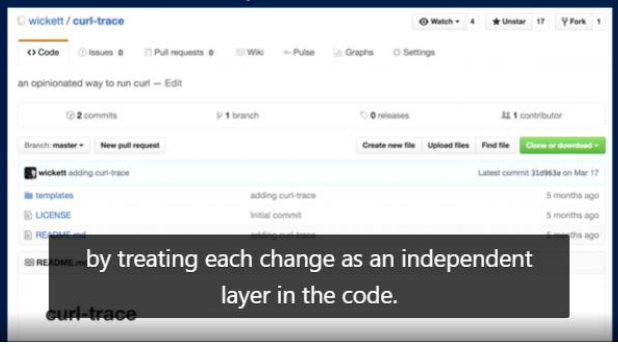
1. Version control
2. CI systems
3. Build
4. Test
5. Artifact repository
6. Deployment

you'll need an Artifact repository and Deployment.

Your CI toolchain

1. Version control

GitHub example

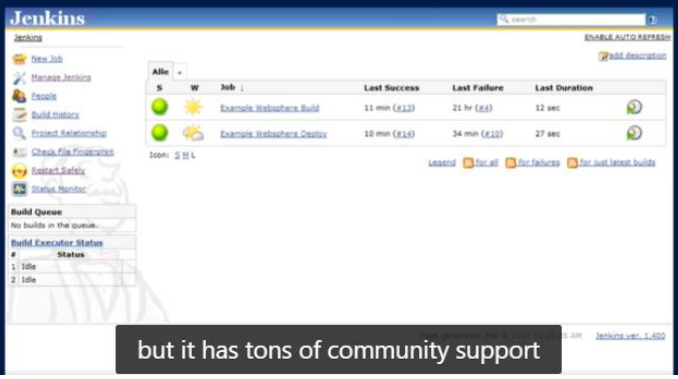


by treating each change as an independent layer in the code.

curl-trace

Your CI toolchain

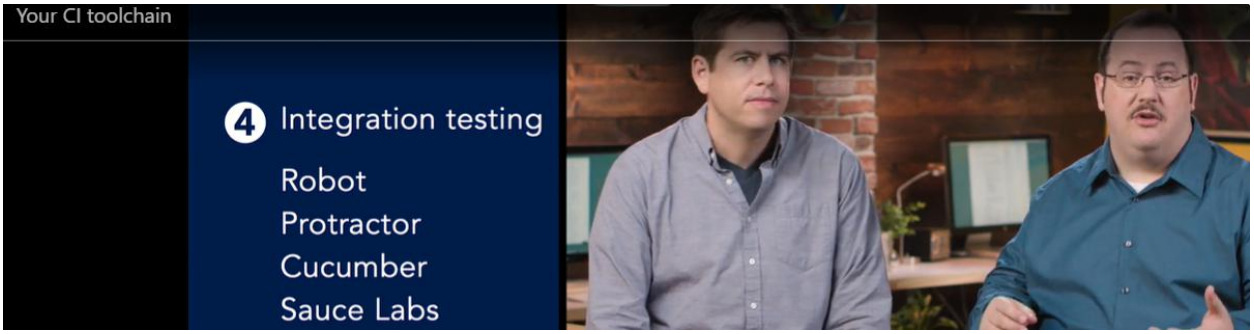
2. CI systems



but it has tons of community support


Your CI toolchain
3 Build
Make/Rake
Maven
Gulp
Packer


Your CI toolchain
4 Unit Testing
JUnit
golint/gofmt
RuboCop


Your CI toolchain
4 Integration testing
Robot
Protractor
Cucumber
Sauce Labs


Your CI toolchain
4 Infrastructure Testing
KitchenCI
Home
Blog
Guide
Your infrastructure deserves tests too.
GET STARTED >
driver:
name: vagrant
provisioner:
name: chef_zero
platforms:
- name: centos-7.1
- name: ubuntu-14.04
- name: windows-2012r2
suites:
- name: client
run_list:
- recipe[postgresql::client]
Kitchen CI for Chef.

Your CI toolchain

4 Performance Testing

ApacheBench
Meter

Your CI toolchain

4 Security Testing

Brakeman
Veracode

Your CI toolchain

5 Artifact repository

Artifactory
Nexus
Docker Hub
AWS S3

Your CI toolchain

6 Deployment

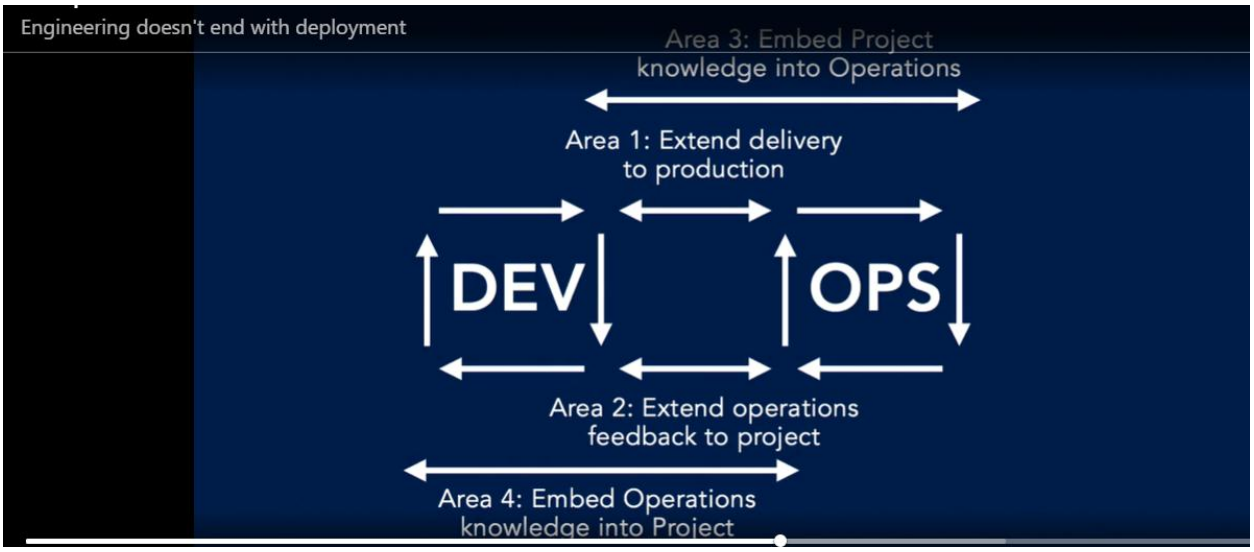
Rundeck
UrbanCode
ThoughtWorks
Deployinator

Engineering doesn't end with deployment

Key Success Metrics

- Deployment frequency
- Lead time for changes
- Change failure rate
- Mean time to recovery

Engineering doesn't end with deployment



Operate for design: Metrics and monitoring

How Complex Systems Fail

Change introduces new forms of failure.

Complex systems contain changing mixtures of failures latent within them.

All complex systems are always running in degraded mode.

Operate for design: Metrics and monitoring

Lean Approach

1. Build
2. Measure
3. Learn
4. Repeat

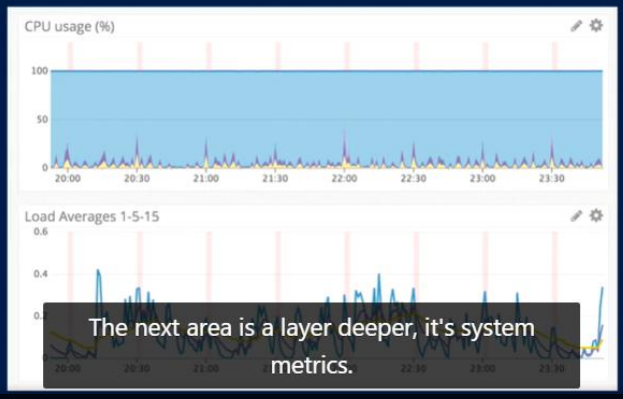
1. Service Performance and Uptime

14 OK	elasticsearch.can_connect
14 OK	elasticsearch.cluster_health

2. Software Component Metrics

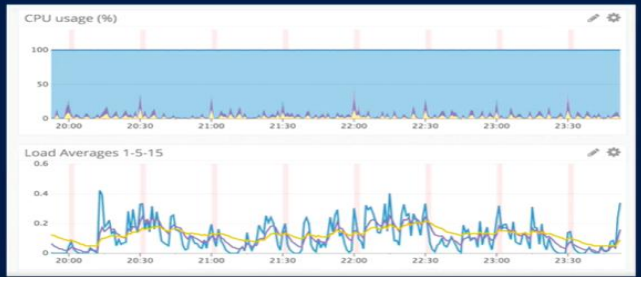
OK	port:9200	server:localhost	host:prod-esdata04
OK	port:9200	server:localhost	host:prod-esdata05
OK	port:9200	server:localhost	host:prod-esdata06
OK	port:9200	server:localhost	host:prod-esdata07
OK	port:9200	server:localhost	host:prod-esdata08
OK	port:9200	server:localhost	host:prod-esdata09

3. System Metrics



The next area is a layer deeper, it's system metrics.

4. App Metrics



Operate for design: Metrics and monitoring

5. Performance

Linting

Code formatting

Banned function checks

Tied through all of the previous types of metrics

Operate for design: Metrics and monitoring

6. Security: System Security

Operate for design: Metrics and monitoring

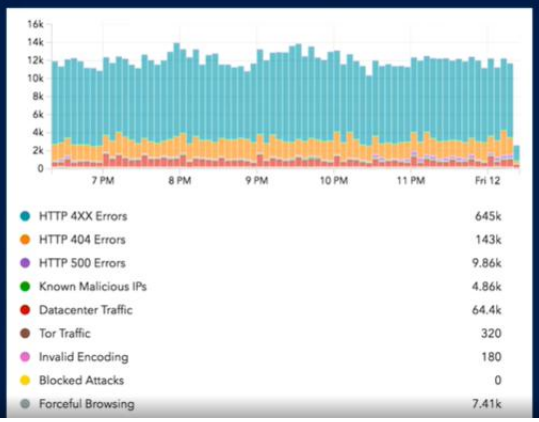
6. Security: System Security

6. Security: System Security

Custom events in the application, things like

password resets, invalid logins, or new account creations.

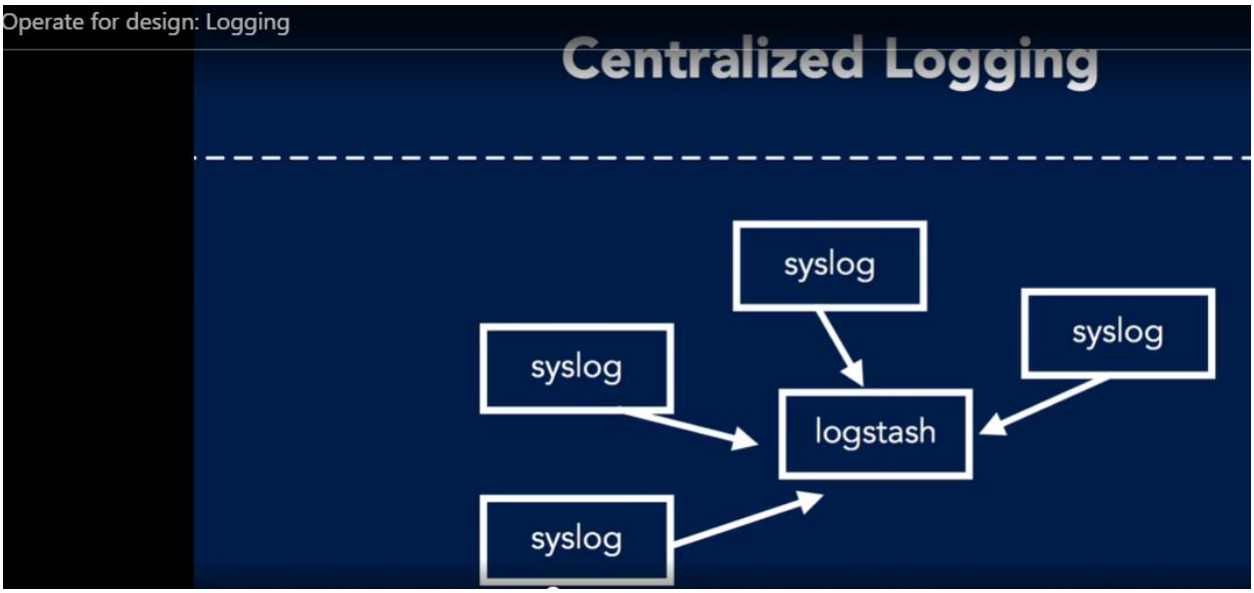
6. Security: Anomalies



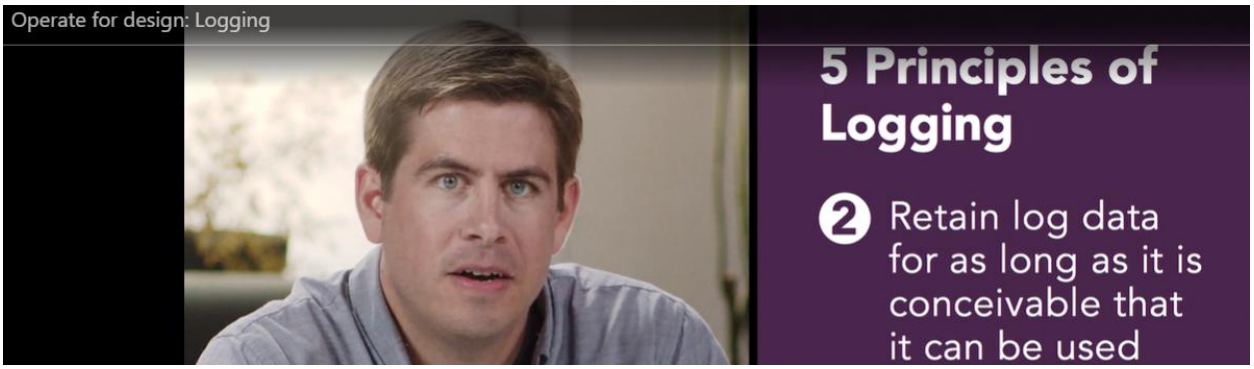
5 Ws of Logging

What happened?
When did it happen?
Where did it happen?
Who was involved?
Where did that entity come from?

Logging can be used for lots of purposes,

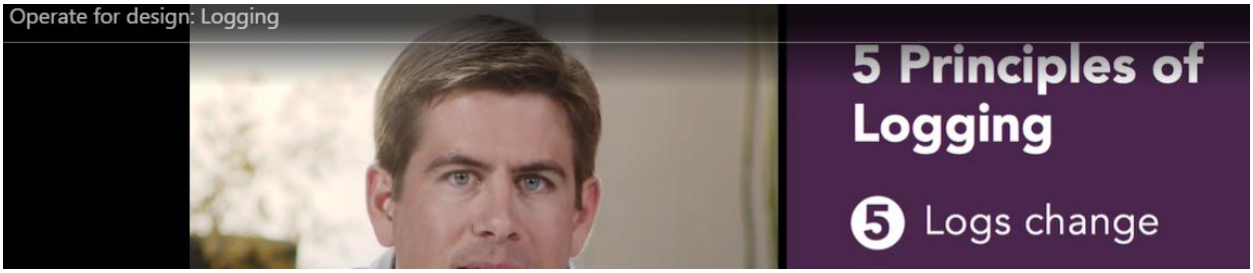

Operate for design: Logging
Centralized Logging
syslog
syslog
syslog
logstash
syslog


Operate for design: Logging
5 Principles of Logging
1 Do not collect log data if you never plan to use it


Operate for design: Logging
5 Principles of Logging
2 Retain log data for as long as it is conceivable that it can be used

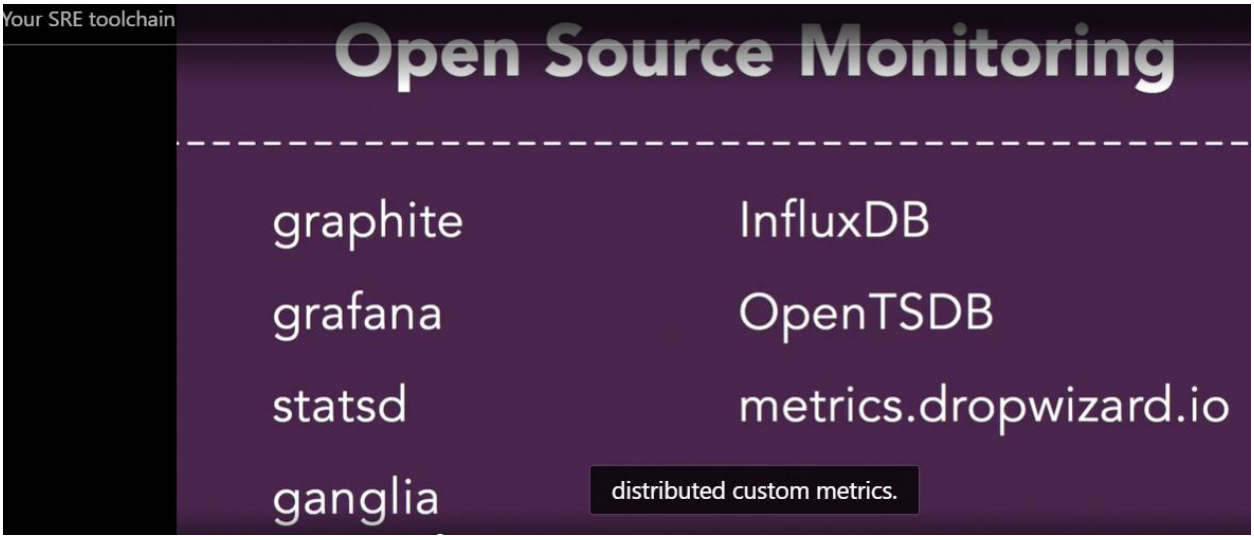

Operate for design: Logging
5 Principles of Logging
3 Log all you can, but alert only on what you must respond to

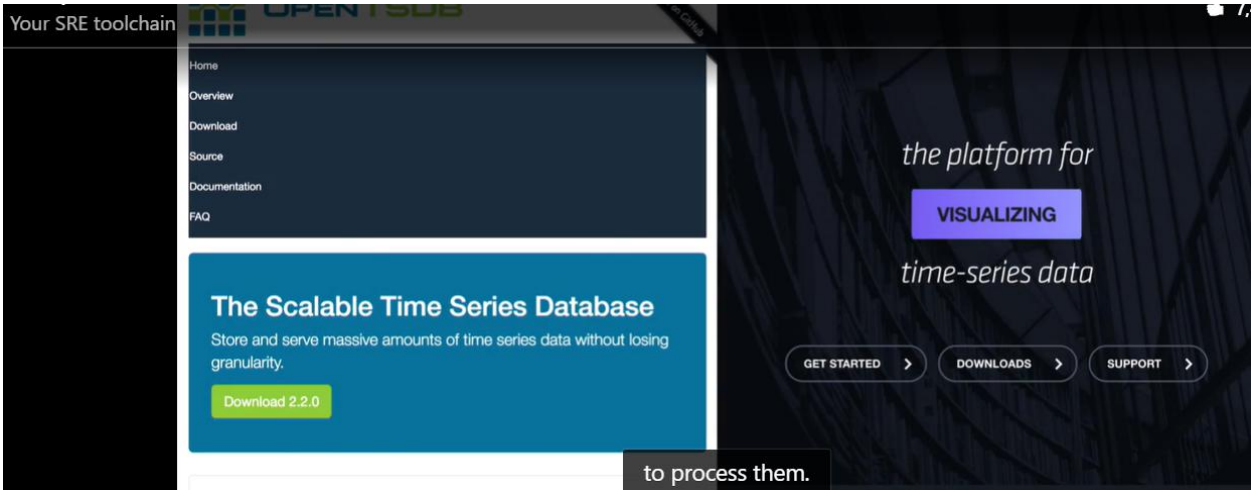

Operate for design: Logging
5 Principles of Logging
4 Don't try to make your logging more available or more secure than your production stack

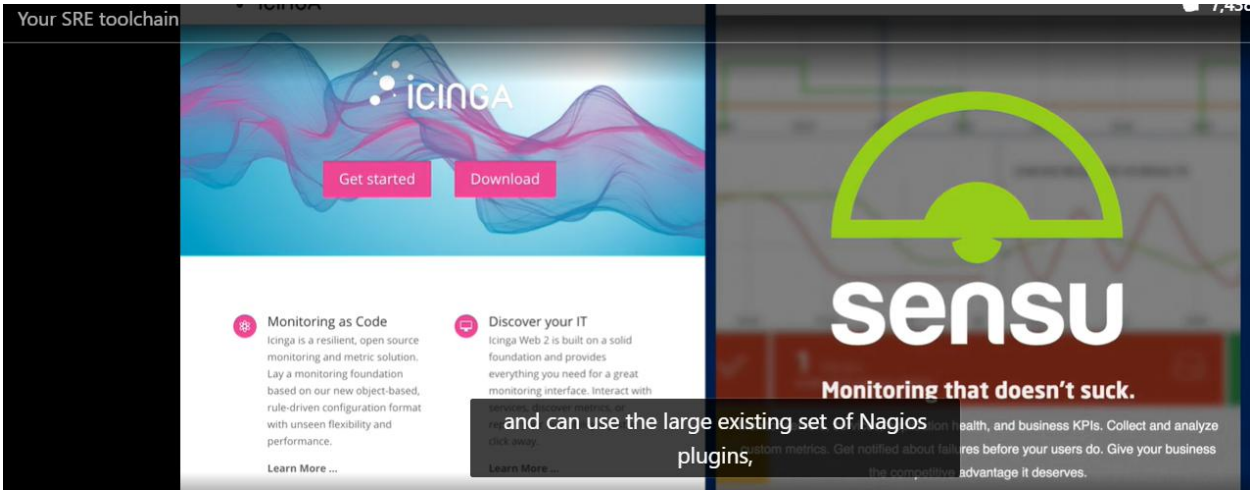

Operate for design: Logging
5 Principles of Logging
5 Logs change

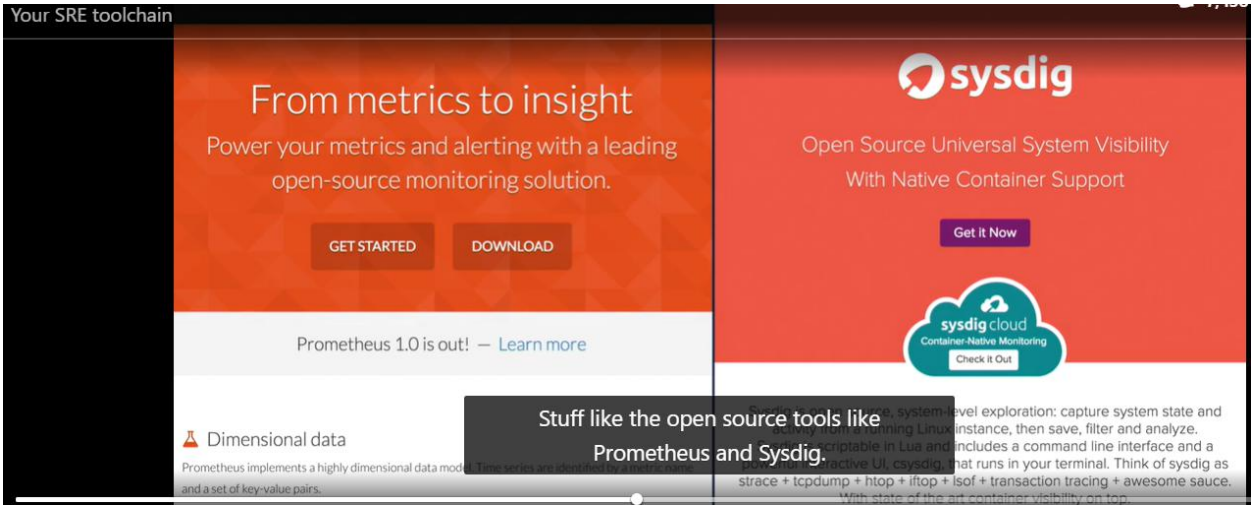

Operate for design: Logging
SYNGRESS
LOGGING AND LOG MANAGEMENT
The Authoritative Guide to Understanding the Concepts Surrounding Logging and Log Management
Dr. Anton A. Chuvakin
Kevin J. Schmidt
Christopher Phillips
VOLUME 2
THE PRACTICE OF CLOUD SYSTEM ADMINISTRATION
DESIGNING AND OPERATING LARGE DISTRIBUTED SYSTEMS
YOU
THOMAS A. LIMONCELLI • STRATA R. CHALUP • CHRISTINA J. HOGAN
Web Operations
Keeping the Data on Time
O'REILLY
John Allspaw & Jesse Robbins


Your SRE toolchain
Monitoring
Pingdom
Librato
Datadog
New Relic
Netuitive
AppDynamics
Ruxit
like New Relic and AppDynamics.

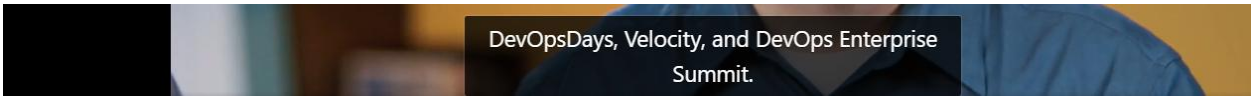

Your SRE toolchain
Open Source Monitoring
graphite
grafana
statsd
ganglia
InfluxDB
OpenTSDB
metrics.dropwizard.io
distributed custom metrics.


Your SRE toolchain
Home
Overview
Download
Source
Documentation
FAQ
The Scalable Time Series Database
Store and serve massive amounts of time series data without losing granularity.
Download 2.2.0
the platform for
VISUALIZING
time-series data
GET STARTED
DOWNLOADS
SUPPORT
to process them.

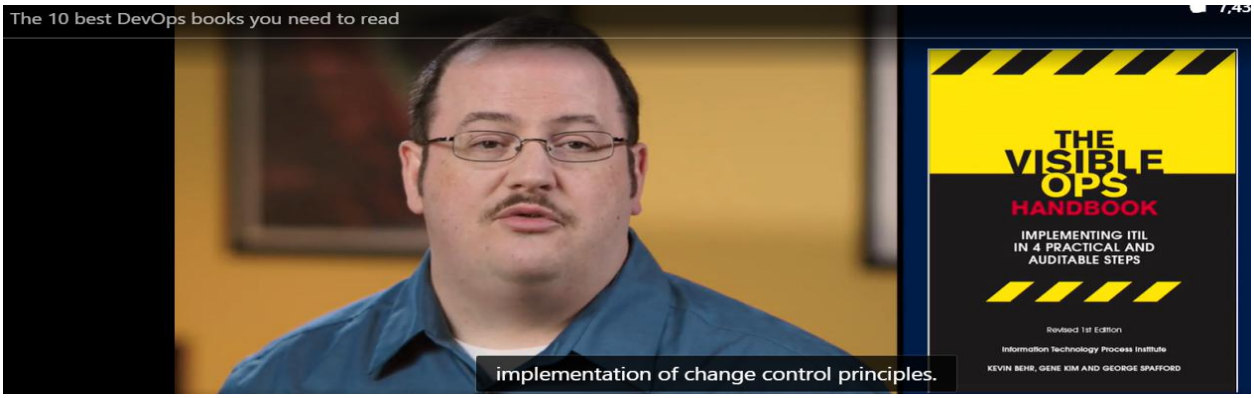

Your SRE toolchain
ICINGA
Get started
Download
Monitoring as Code
Icinga is a resilient, open source monitoring and metric solution. Lay a monitoring foundation based on our new object-based, rule-driven configuration format with unseen flexibility and performance.
Learn More ...
Discover your IT
Icinga Web 2 is built on a solid foundation and provides everything you need for a great monitoring interface. Interact with
Learn More ...
sensu
Monitoring that doesn't suck.
and can use the large existing set of Nagios plugins,

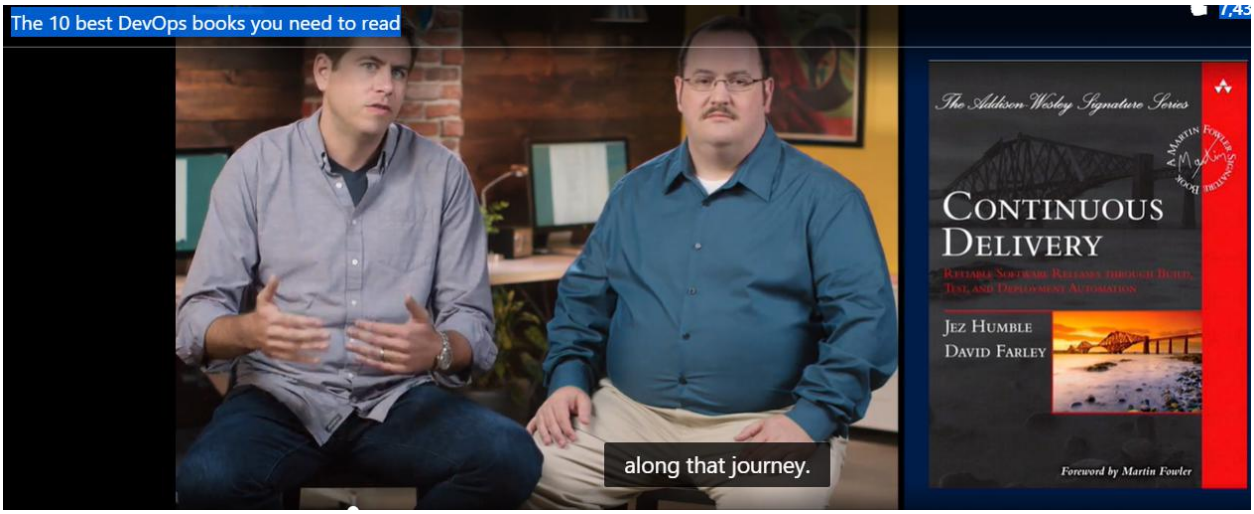

Your SRE toolchain
From metrics to insight
Power your metrics and alerting with a leading open-source monitoring solution.
GET STARTED
DOWNLOAD
Prometheus 1.0 is out! — Learn more
Dimensional data
Prometheus implements a highly dimensional data model. Time series are identified
and a set of key-value pairs.
sysdig
Open Source Universal System Visibility With Native Container Support
Get it Now
sysdig cloud
Container-Native Monitoring
Check it Out
Stuff like the open source tools like Prometheus and Sysdig.

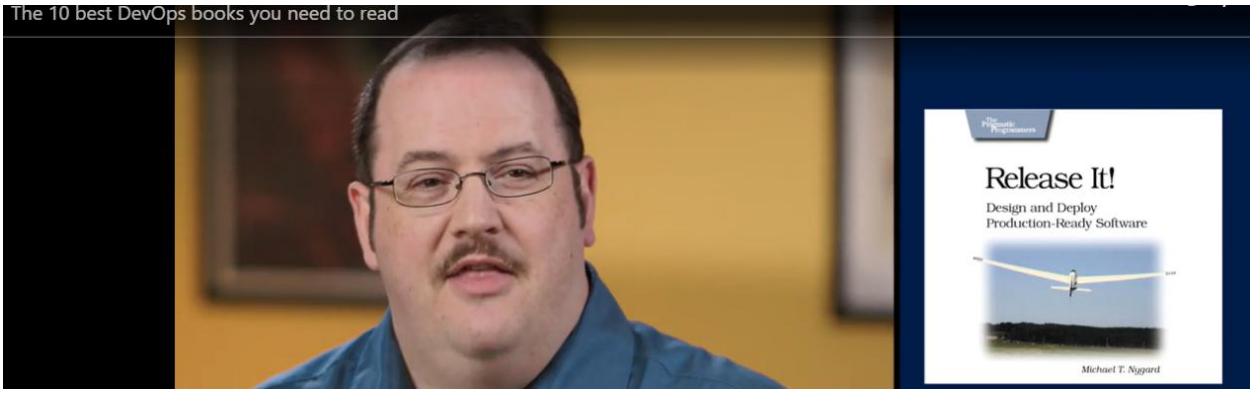

Your SRE toolchain
flapjack
monitoring notification routing + event processing system
Composable
Routing + Rollup
API Driven
at flapjack.io, that can help you do that yourself,
http://flapjack.io
Get going with the quickstart guide.
Start now!

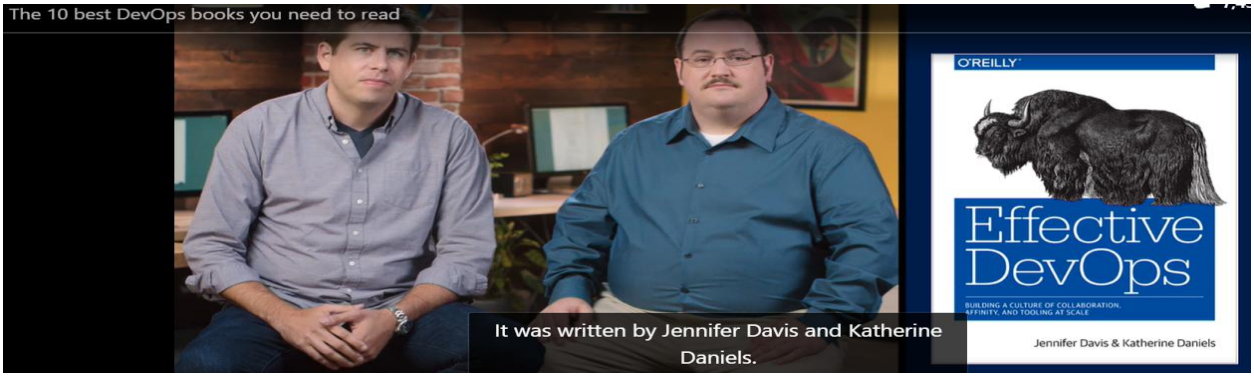

DevOpsDays, Velocity, and DevOps Enterprise Summit.


Unicorns, horses, and donkeys, oh my
DevOpsDays
Velocity
DevOps Enterprise Summit
ChefConf
PuppetConf
AWS re:Invent
Monitorama
Surge
ScaleConf
Structure
They all focus on certain aspects of DevOps.


The 10 best DevOps books you need to read
THE VISIBLE OPS HANDBOOK
IMPLEMENTING ITIL IN 4 PRACTICAL AND AUDITABLE STEPS
Revised 1st Edition
Information Technology Process Institute
KEVIN BEHR, GENE KIM AND GEORGE SPAFFORD
implementation of change control principles.


The 10 best DevOps books you need to read
The Addison-Wesley Signature Series
CONTINUOUS DELIVERY
JEZ HUMBLE
DAVID FARLEY
Foreword by Martin Fowler
along that journey.

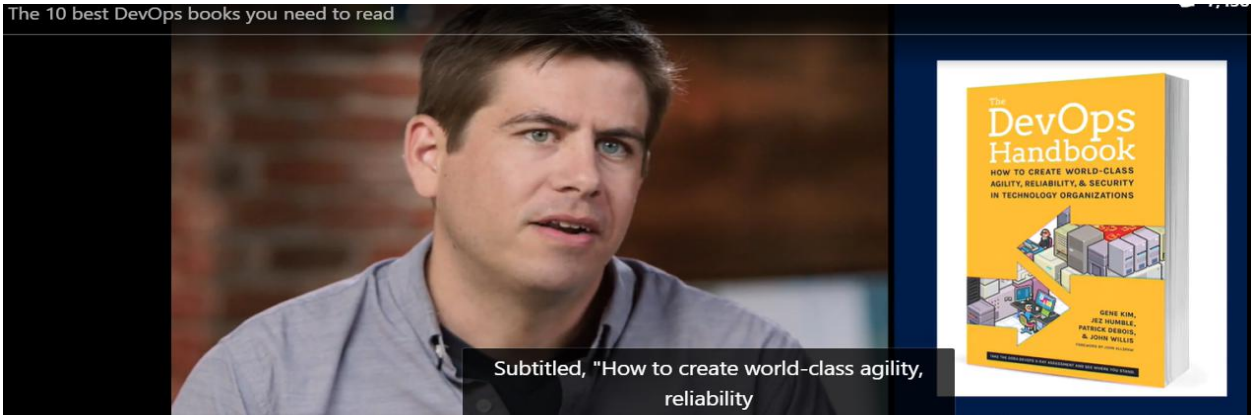

The 10 best DevOps books you need to read
Release It!
Design and Deploy Production-Ready Software
Michael T. Nygard

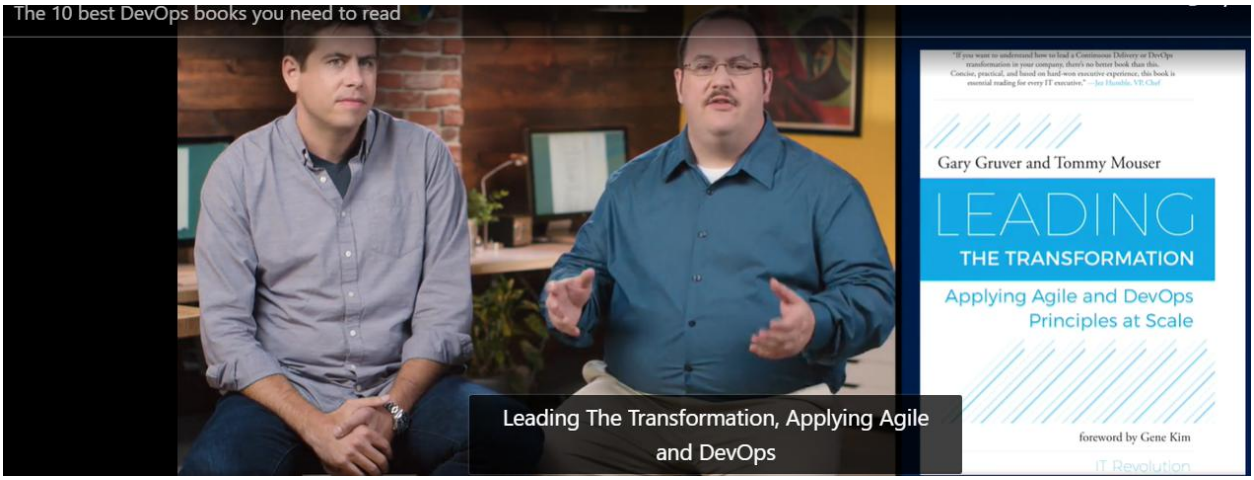

The 10 best DevOps books you need to read
O'REILLY
Effective DevOps
Jennifer Davis & Katherine Daniels
It was written by Jennifer Davis and Katherine Daniels.

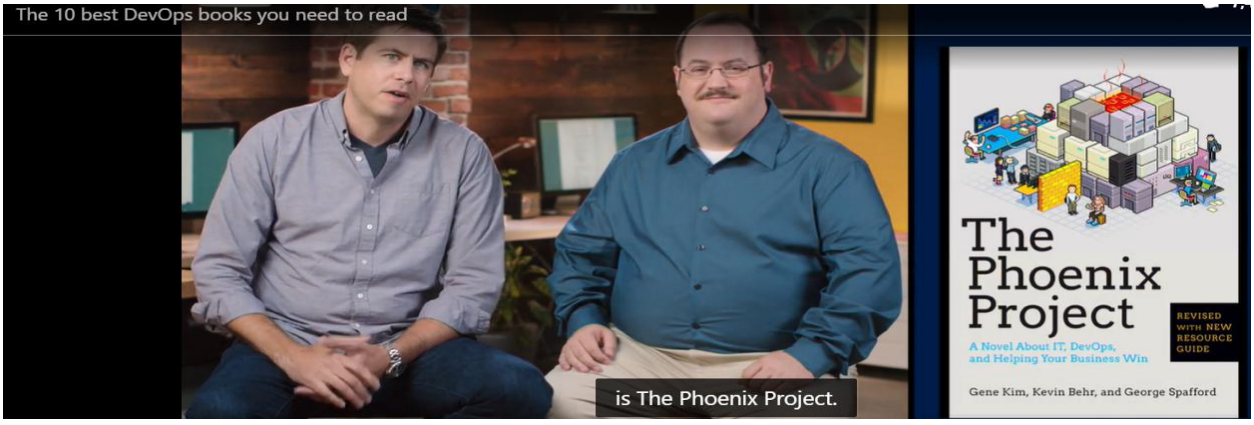

The 10 best DevOps books you need to read
- Number six, Lean Software Development, An Agile Toolkit.
Lean
Software Development
An Agile Toolkit
Mary Poppendieck
Tom Poppendieck


The 10 best DevOps books you need to read
- Book five, Web Operations, this book is edited
Web Operations
Keeping the Data On Time
John Allspaw
& Jesse Robbins

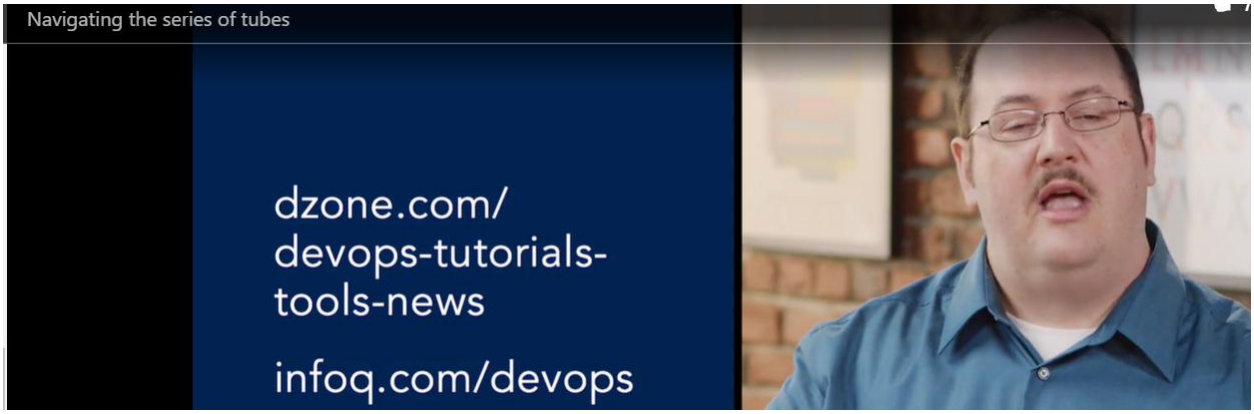

The 10 best DevOps books you need to read
- Book four, The Practice of Cloud System Administration.
VOLUME 2
THE PRACTICE OF
CLOUD SYSTEM
ADMINISTRATION
DESIGNING AND OPERATING
LARGE DISTRIBUTED SYSTEMS
YOU
THOMAS A. LIMONCELLI • STRATA R. CHALUP • CHRISTINA J. HOGAN


The 10 best DevOps books you need to read
Subtitled, "How to create world-class agility, reliability
The
DevOps
Handbook
HOW TO CREATE WORLD-CLASS
AGILITY, RELIABILITY, & SECURITY
IN TECHNOLOGY ORGANIZATIONS
GENE KIM,
JEZ HUMBLE,
PATRICK DEBOIS,
& JOHN WILLIS


The 10 best DevOps books you need to read
Gary Gruver and Tommy Mouser
LEADING
THE TRANSFORMATION
Applying Agile and DevOps Principles at Scale
foreword by Gene Kim
IT Revolution
Leading The Transformation, Applying Agile and DevOps


The 10 best DevOps books you need to read
The Phoenix Project
REVISED WITH NEW RESOURCE GUIDE
A Novel About IT, DevOps, and Helping Your Business Win
Gene Kim, Kevin Behr, and George Spafford
is The Phoenix Project.

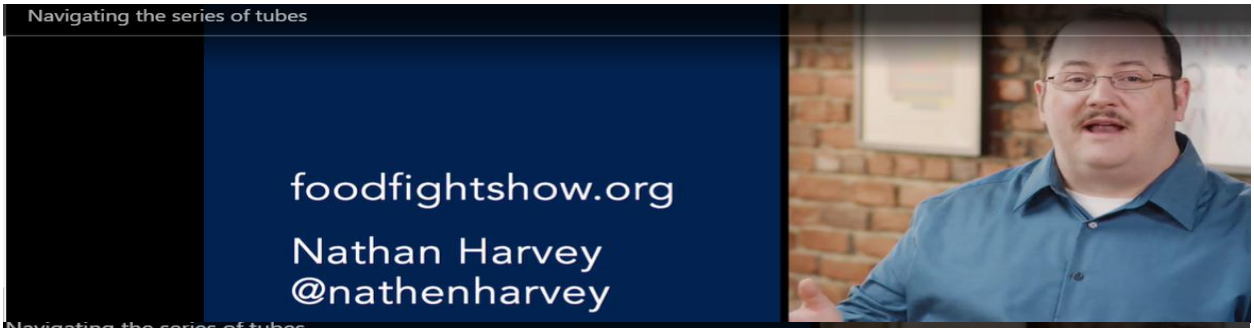

Navigating the series of tubes
devopsweekly.com
that rounds up the most notable items from the last week.

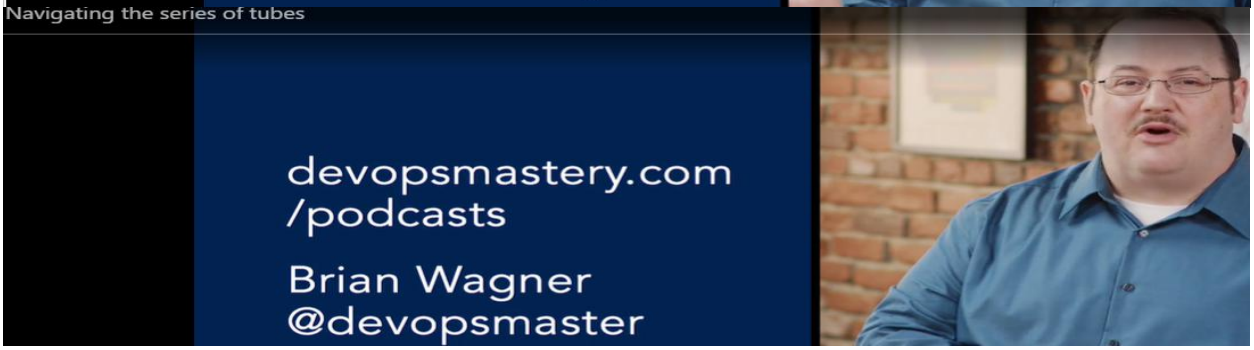

Navigating the series of tubes
dzone.com/
devops-tutorials-
tools-news
infoq.com/devops


devopscafe.org
John Willis
@botchgalupe
Damon Edwards
@damonedwards


Navigating the series of tubes
theshipshow.com


Navigating the series of tubes
arresteddevops.com


Navigating the series of tubes
foodfightshow.org
Nathan Harvey
@nathenharvey


Navigating the series of tubes
devopsmastery.com
/podcasts
Brian Wagner
@devopsmaster


Navigating the series of tubes
kitchensoap.com
itrevolution.com/
devops-blog
dev2ops.org
continuousdelivery.
com/blog


Navigating the series of tubes
Stochastic Resonance:
goo.gl/B6dN0s
jedi.be/blog
morethanseven.net


Navigating the series of tubes
theagileadmin.com
Ernest Mueller
@ernestmueller
James Wickett
@wickett
Karthik Gaekwad
@iteration1

Public cloud: Someone else's servers and data center

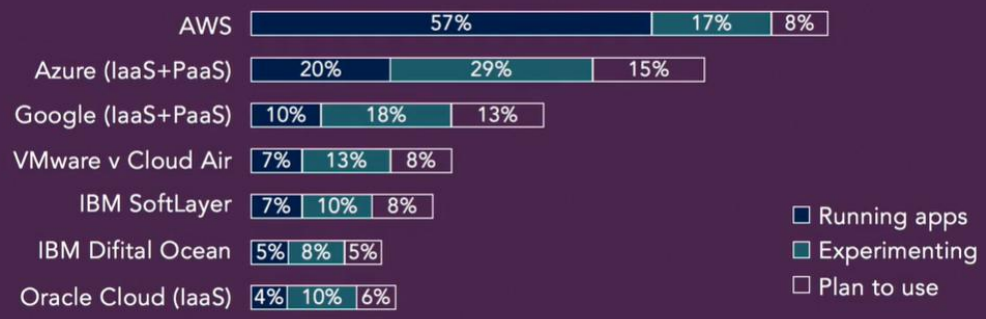
Private cloud: Your own servers and data center

Hybrid: A combination of public and private cl

technology made more profoundly self-service.

Public Cloud Adoption

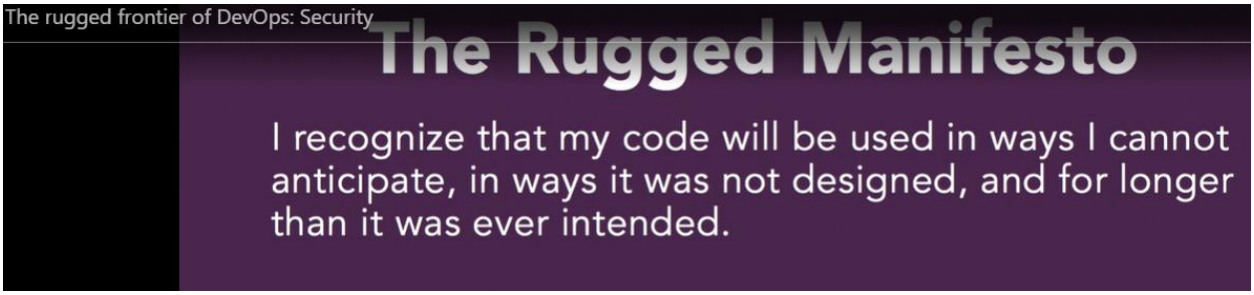
% of Respondents Running Applications

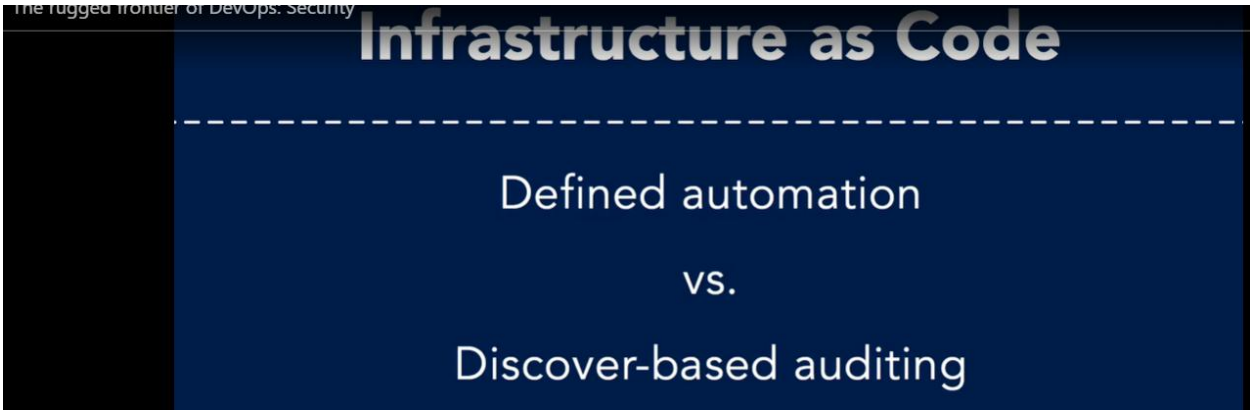


and 77% are using some kind of private cloud.

The Rugged Manifesto

I am rugged and, more importantly, my code is rugged.
I recognize that software has become a foundation of our modern world.
I recognize the awesome responsibility that comes with this foundational role.
I recognize that my code will be used in ways I cannot anticipate, in ways it was not designed, and for longer than it was ever intended.
I recognize that my code will be attacked by talented and persistent adversaries who threaten our physical, economic, and national security.
I recognize these things–and I choose to be rugged.
I am rugged because I refuse to be a source of vulnerability or weakness.
I am rugged because I assure my code will support its mission.
I am rugged because my code can face these challenges and persist in spite of them.
I am rugged not because it is easy, but because it is necessary and I am up for the challenge.


The rugged frontier of DevOps: Security
The Rugged Manifesto
I recognize that my code will be used in ways I cannot anticipate, in ways it was not designed, and for longer than it was ever intended.


Infrastructure as Code
Defined automation
vs.
Discover-based auditing


The rugged frontier of DevOps: Security
Software Supply Chain
Know your software bill of materials (BOM) and vulnerabilities therein
Materials
from the end of the cycle to the left.