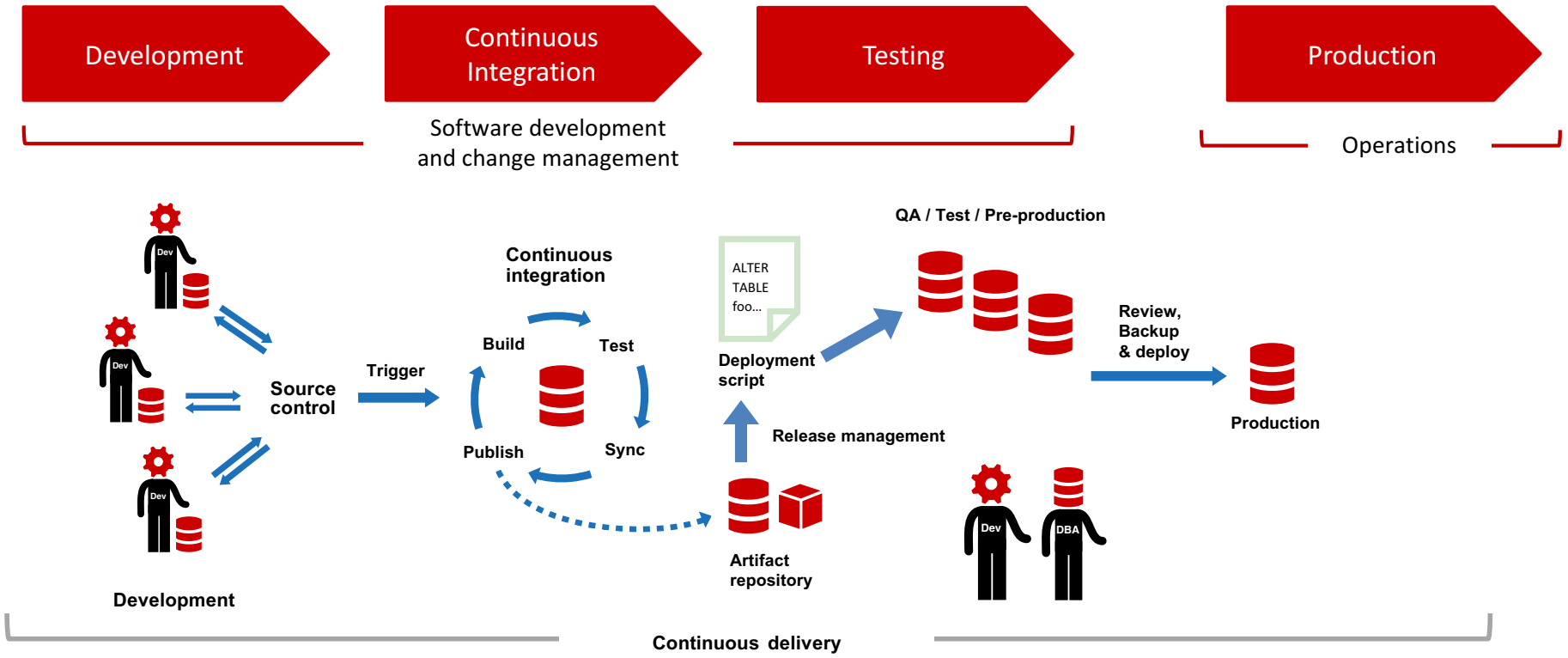
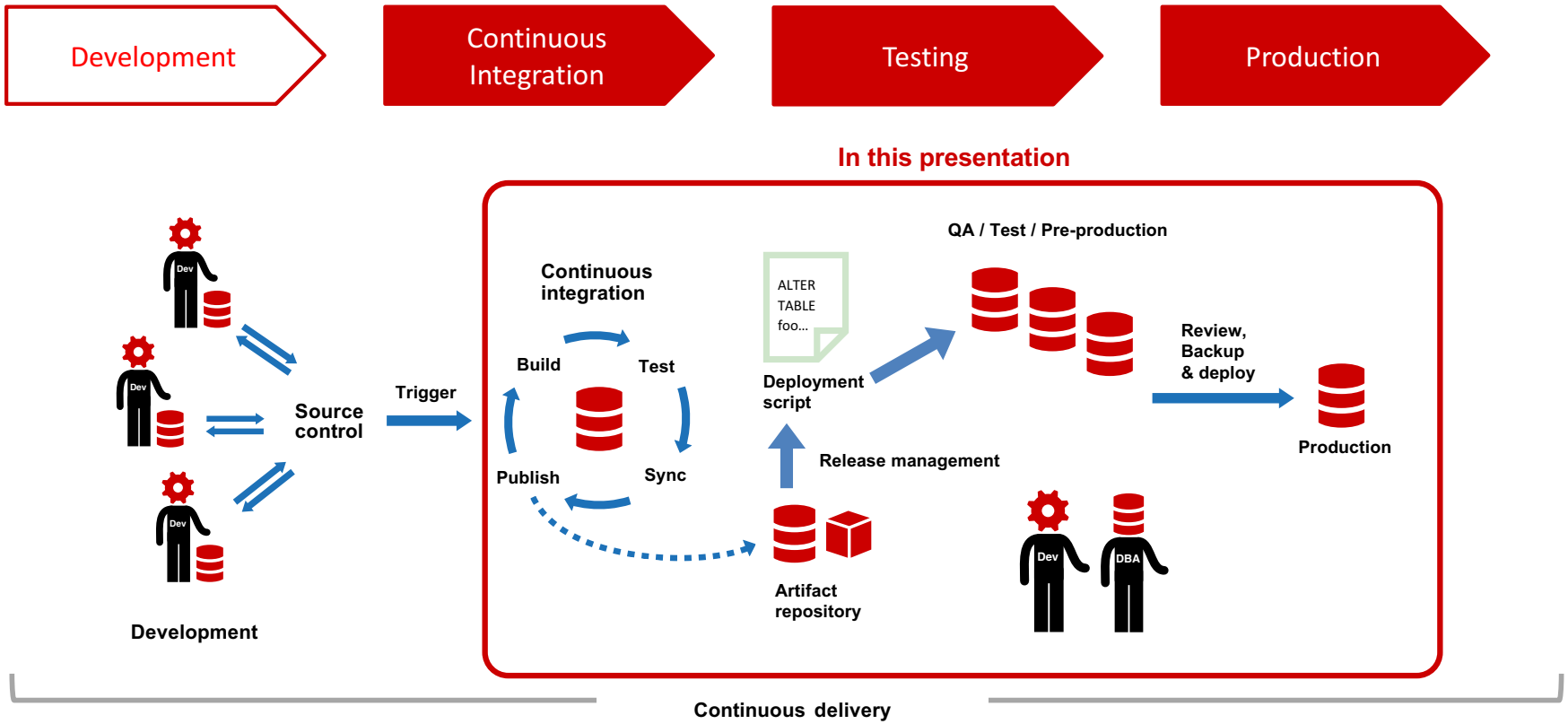
Agenda

No Agenda, we Just talk about
How software was built and
deploy safely… so code happines

Release pipeline



Release pipeline



Continuous delivery – what it is and isn't

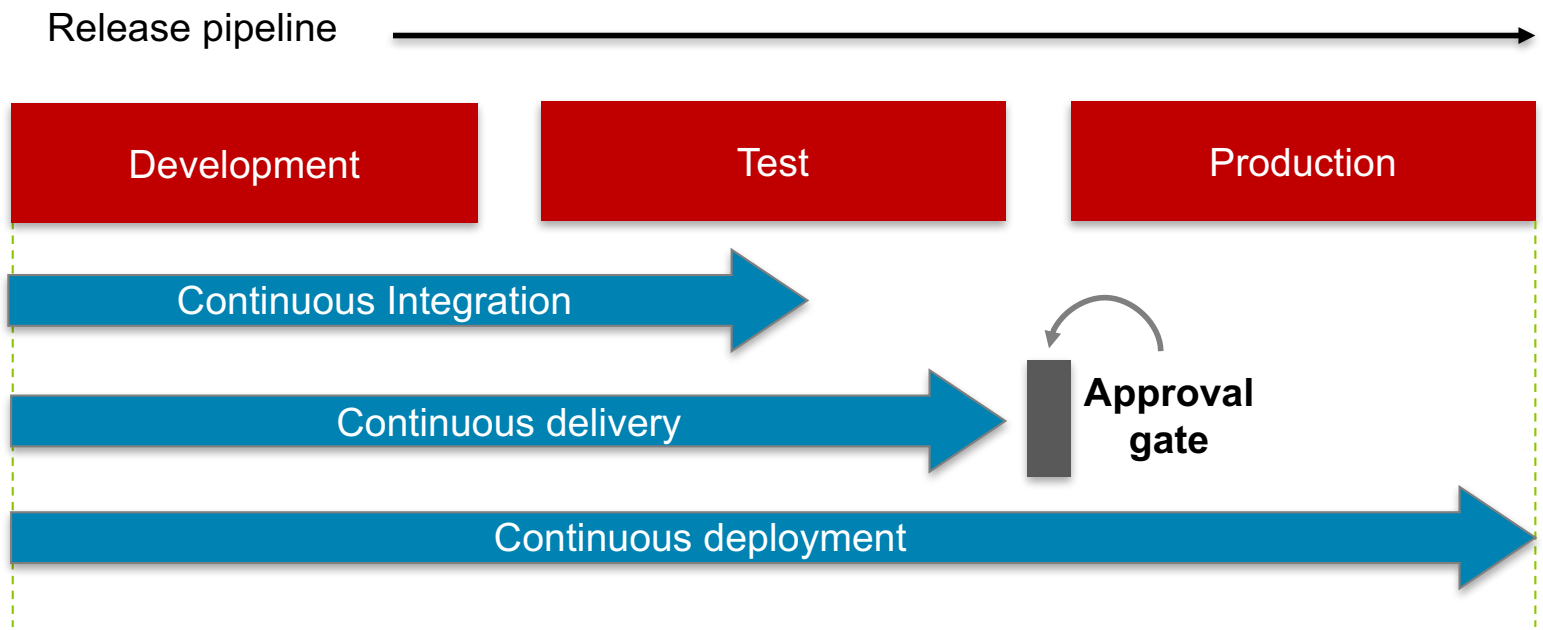
Common misconception:

Continuous delivery ≠ Continuous deployment

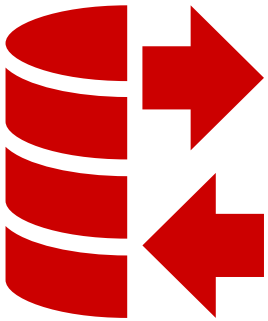
But in fact:

Continuous delivery means making sure your code changes are always production-**ready**.

Recap: Continuous Delivery



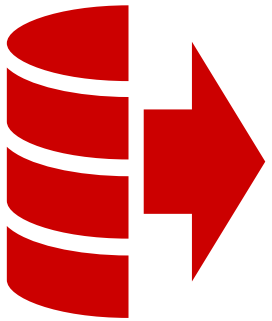
Four key stages of the deployment pipeline



SOURCE CONTROL

CONTINUOUS INTEGRATION

AUTOMATED TESTING



AUTOMATED DEPLOYMENT

Get your code under control

- Use version control for all code (including tests)
- Commit early, commit often
- Use tools
  - If it's hard, people don't do it
- Train people
- Build often

Continuous Integration

Requires

Code repository

Build automation

Delivers

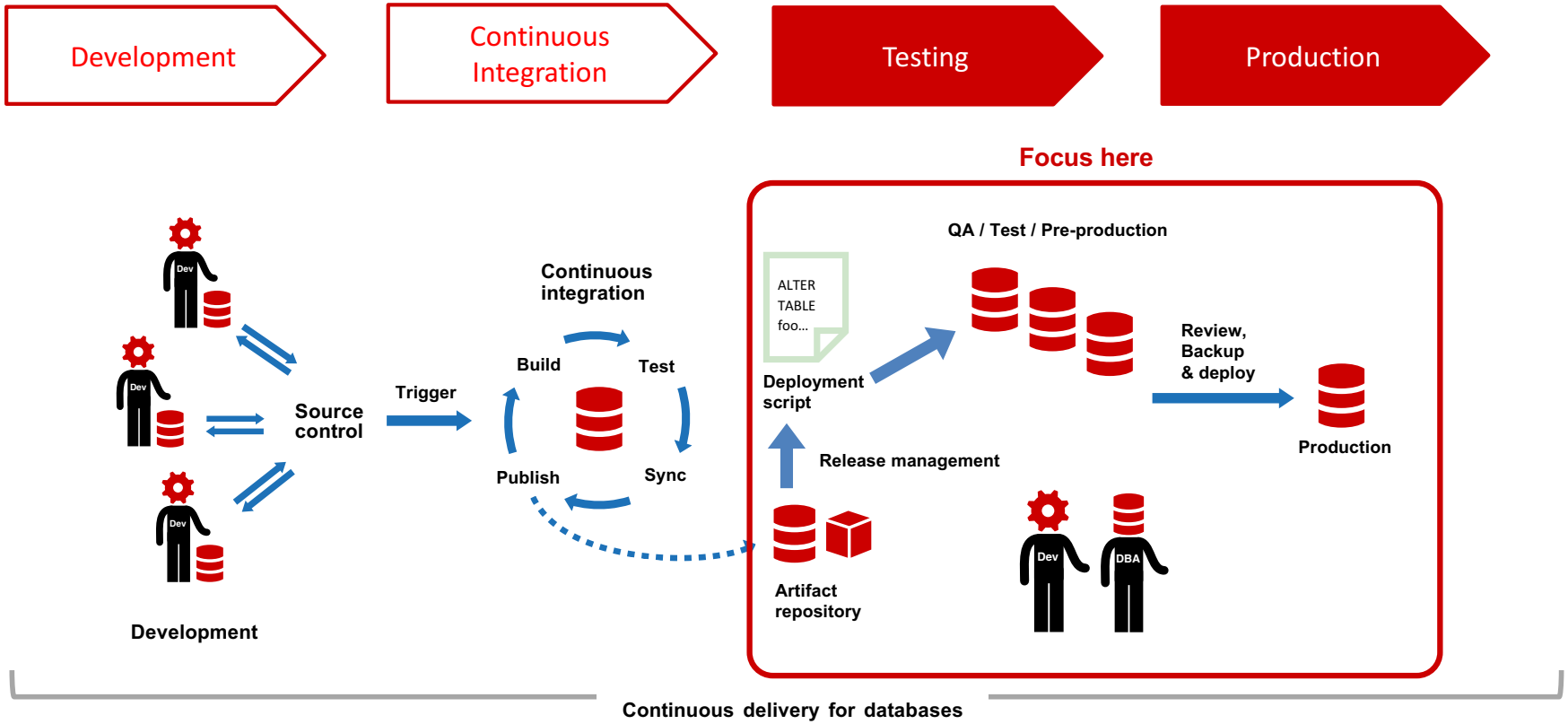
Functional testing

Delivery testing

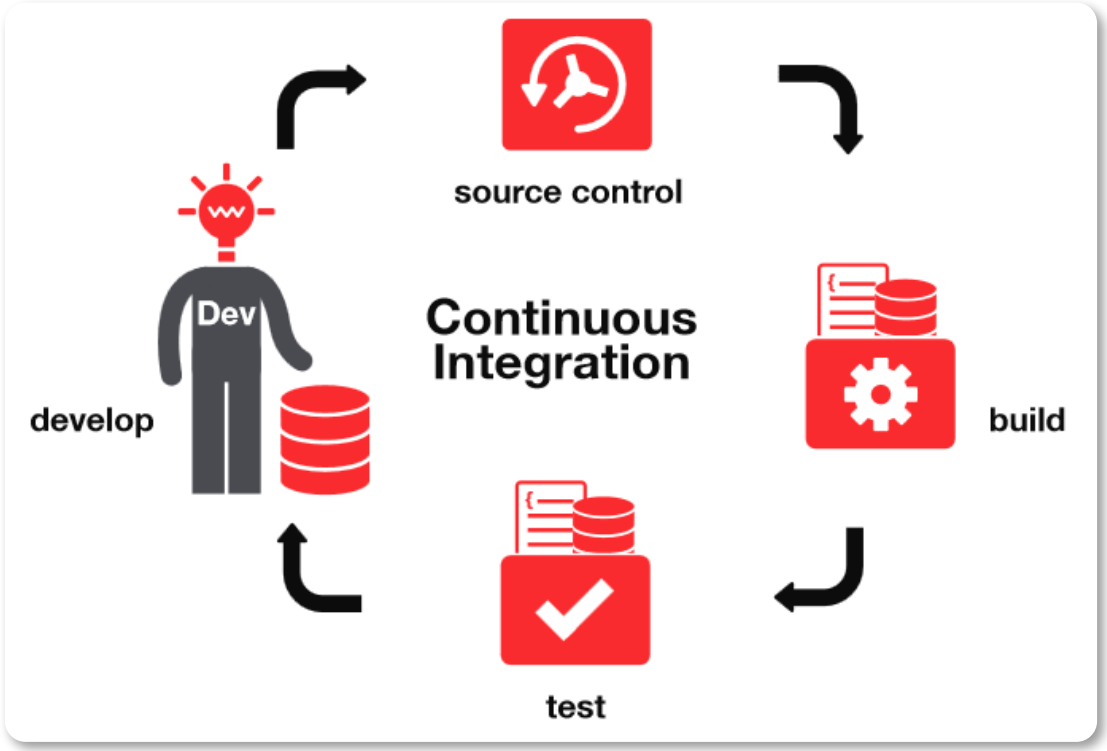
Artifact generation

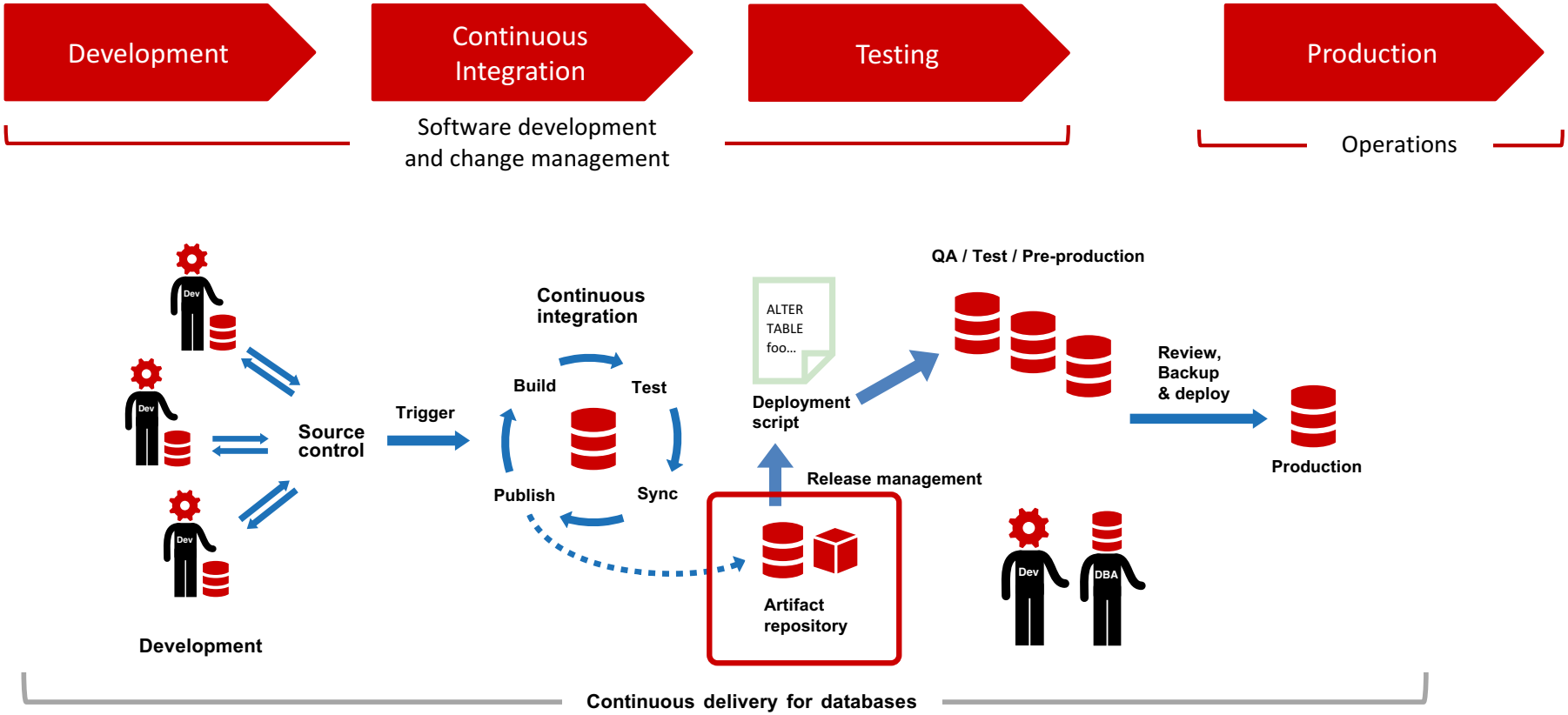
CONTINUOUS INTEGRATION: FUNDAMENTALS

Build out your release strategy



Automation


source control
Dev
develop
Continuous
Integration
build
test


Development
Continuous Integration
Testing
Production
Software development and change management
Operations
Dev
Dev
Dev
Source control
Development
Trigger
Continuous integration
Build
Test
Publish
Sync
ALTER TABLE foo...
Deployment script
QA / Test / Pre-production
Release management
Artifact repository
Review, Backup & deploy
Production
Dev
DBA
Continuous delivery for databases

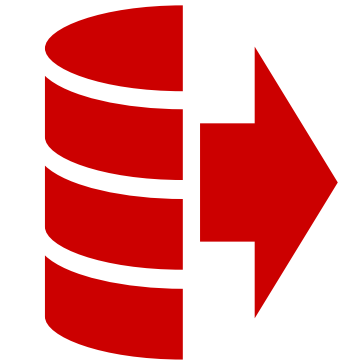
Pre-production is a dry-run

Staging matches production

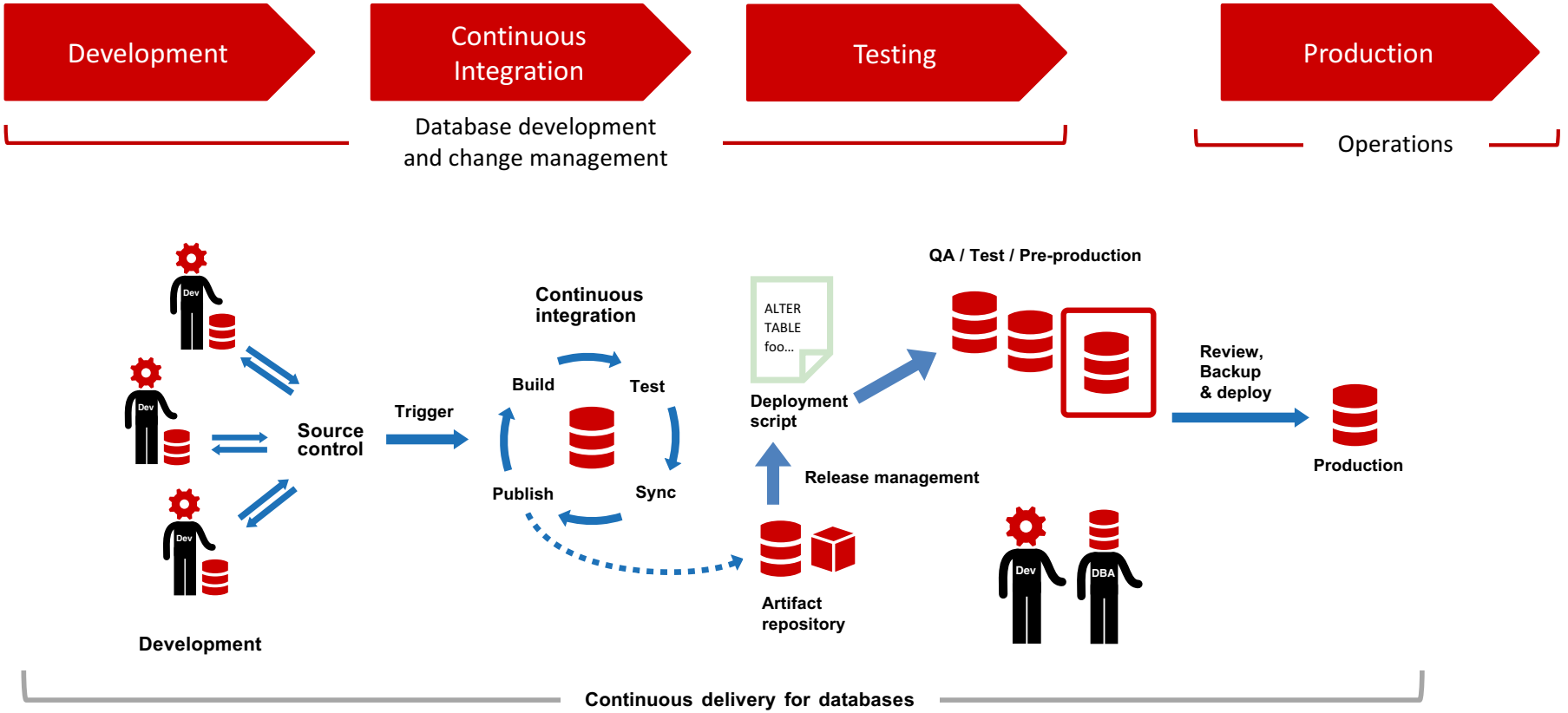
As close as possible

Can use diff tool

Or backup and restore



AUTOMATED
DEPLOYMENT


Development
Continuous Integration
Testing
Production
Database development and change management
Operations
Dev
Dev
Dev
Source control
Development
Trigger
Continuous integration
Build
Test
Publish
Sync
ALTER TABLE foo...
Deployment script
Release management
Artifact repository
QA / Test / Pre-production
Review, Backup & deploy
Production
Dev
DBA
Continuous delivery for databases

Plan for things to go wrong

…you need a rollback and recovery strategy

AUTOMATED DEPLOYMENT

Protection

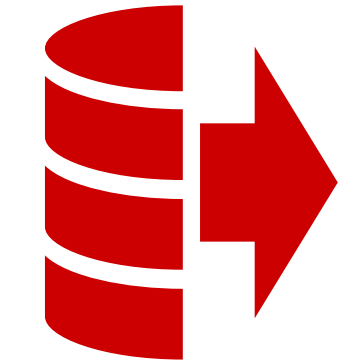
Backups

Snapshots

Rollback scripts

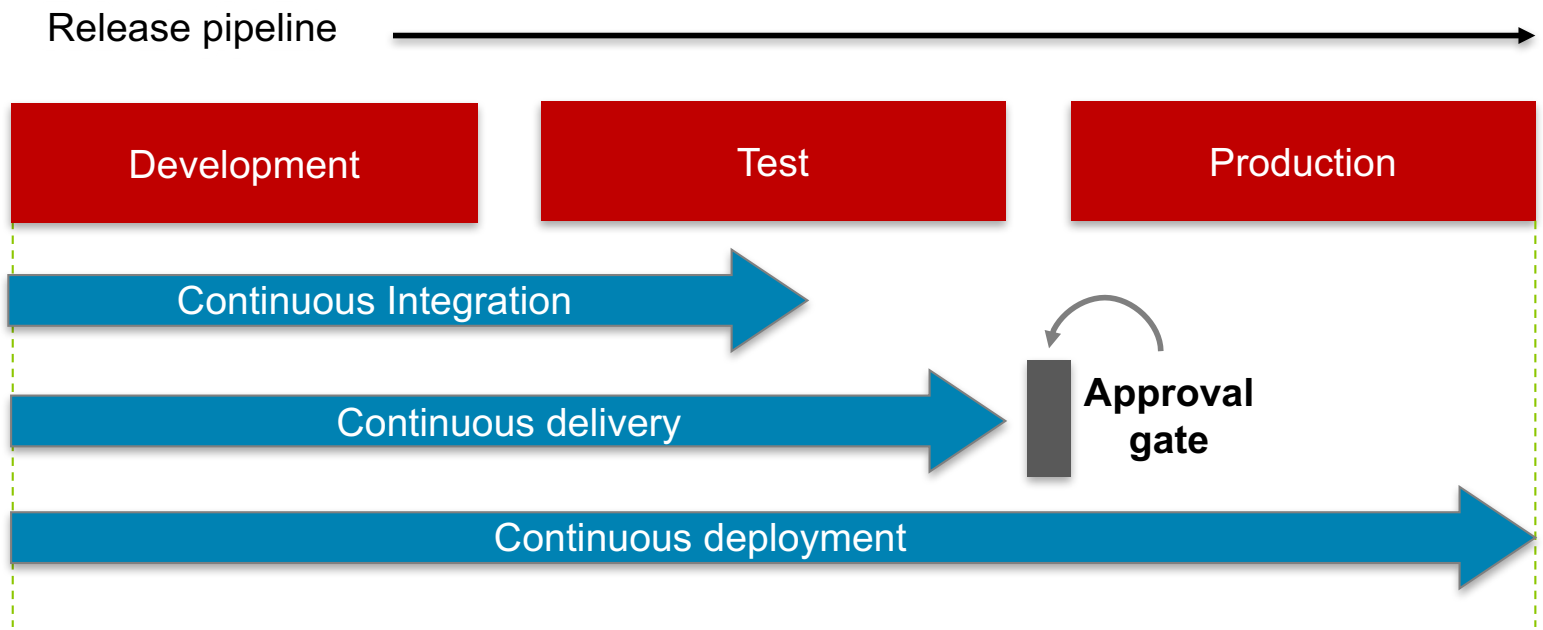
A/B or blue/green deployments

Roll forward scripts

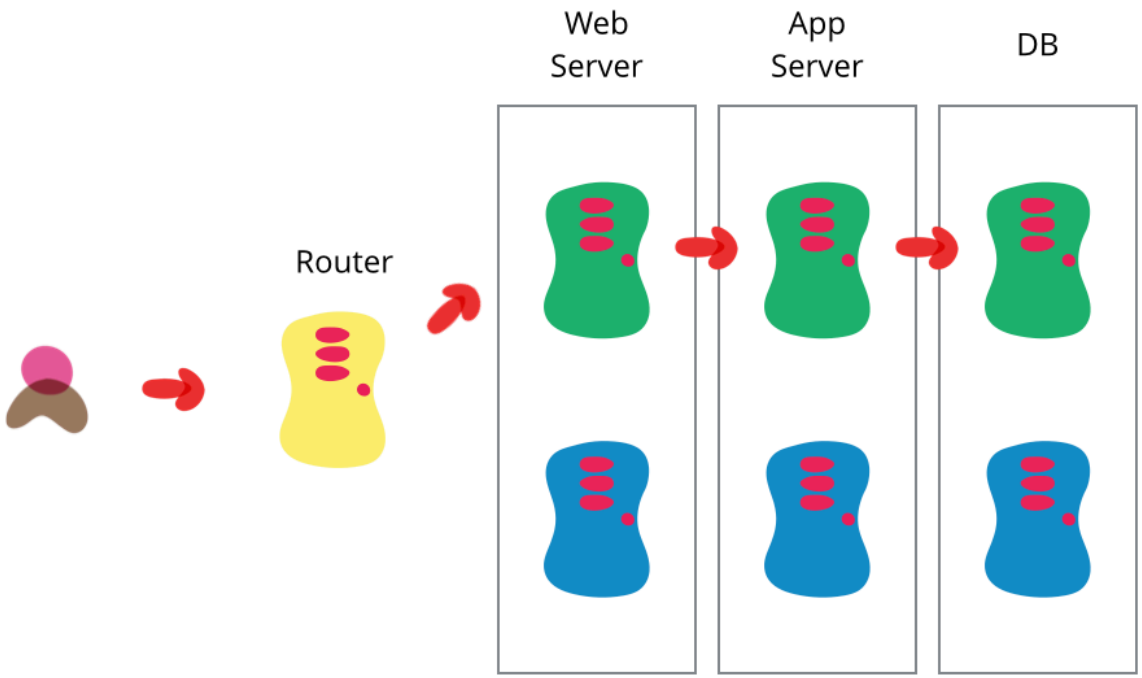


AUTOMATED
DEPLOYMENT

Recap: Continuous Delivery



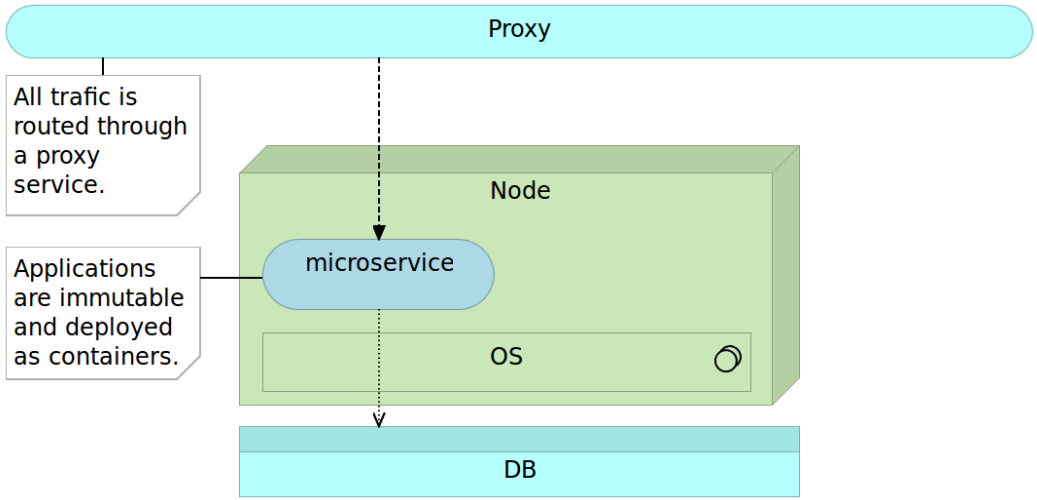
Blue / Green Deployments



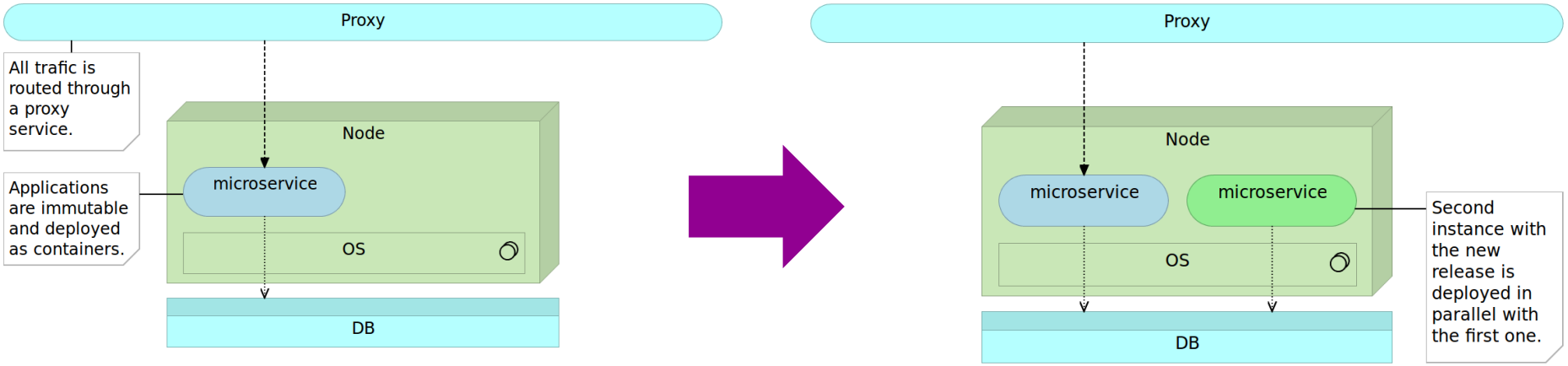
- You have an existing running version, say Blue
- You deploy a new version, say Green
- Once you are satisfied that green is healthy/tested, switch incoming requests at the router to green.

https://martinfowler.com/bliki/BlueGreenDeployment.html

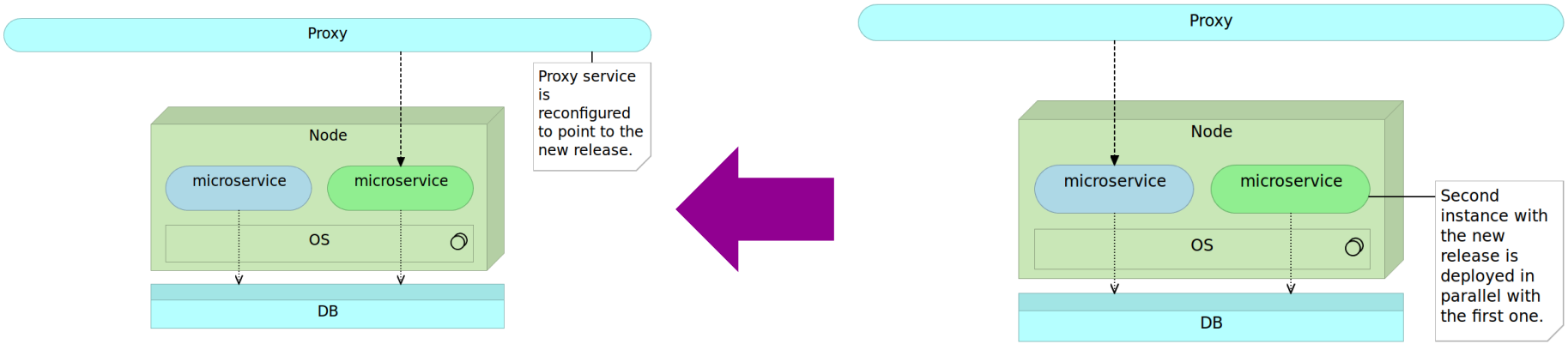
Blue / Green Deployments for m$



Blue / Green Deployments for m$

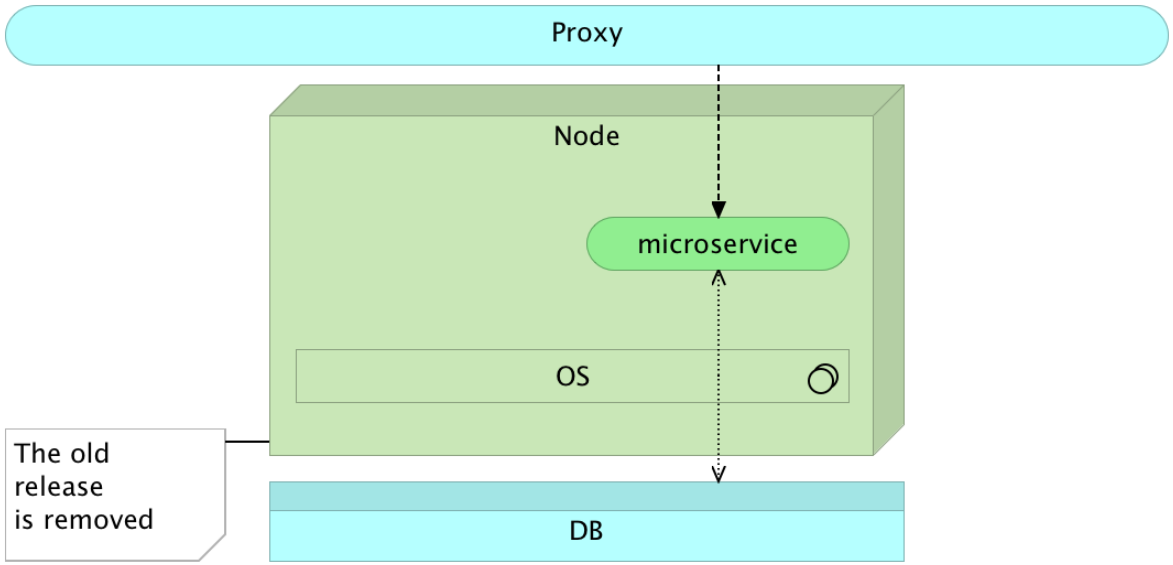


Blue / Green Deployments for m$



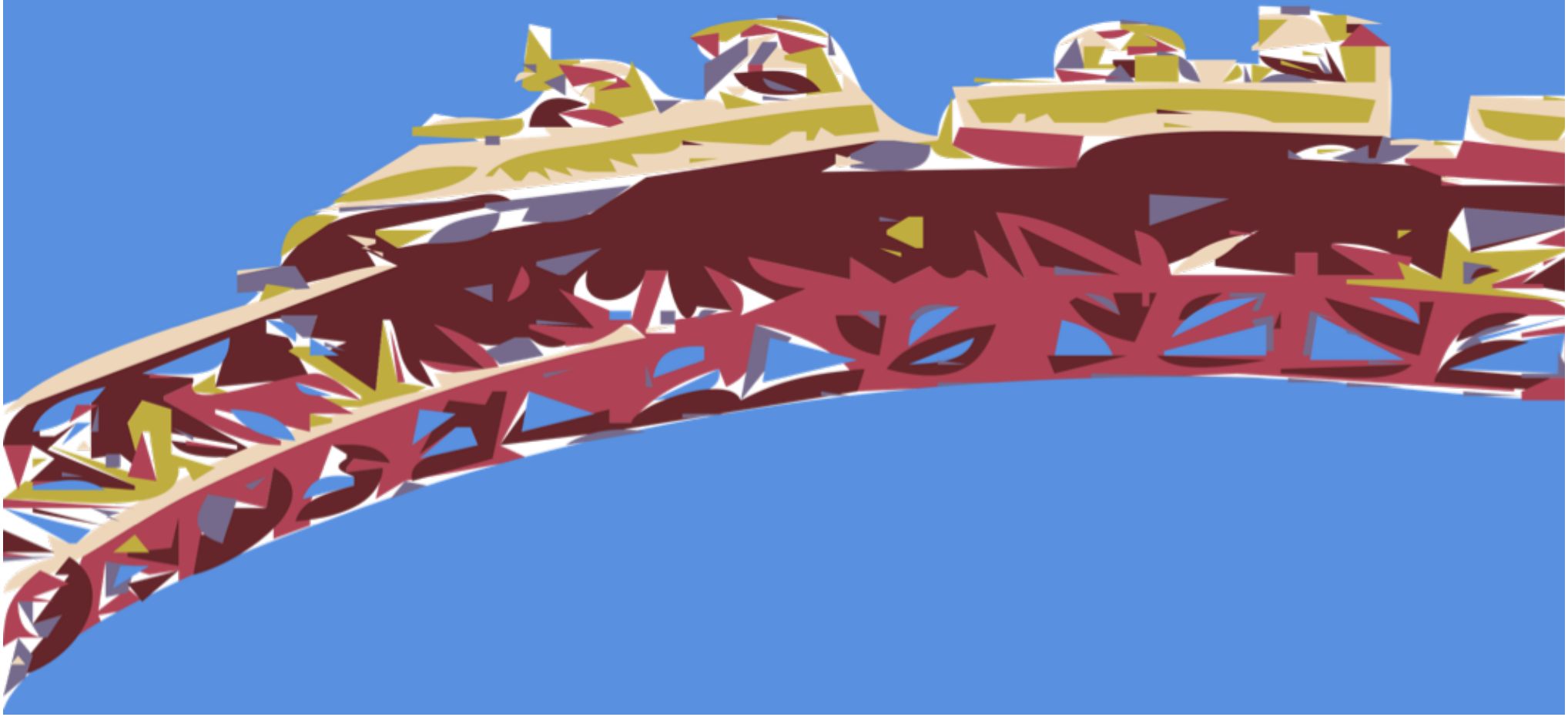
When a new release (for example green) is ready to be deployed, we run it in parallel with the current release

Blue / Green Deployments for m$

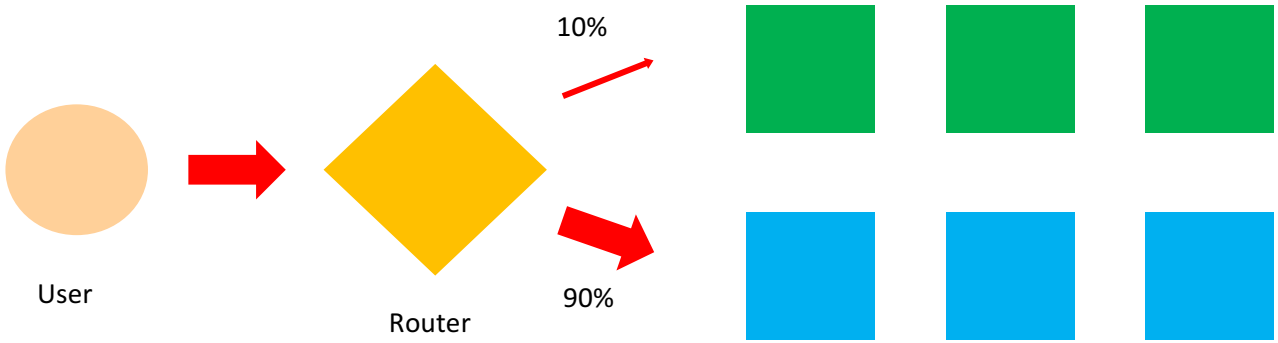


When a new release (for example green) is ready to be deployed, we run it in parallel with the current release

Canary / Rolling Updates



Canary Release / Rolling Updates



- Gradually roll out a change to a subset of users
- Existing version is blue, deploy a new version, Green
- Once you are satisfied that green is healthy/tested, switch *some %* of requests at the router to green.

https://martinfowler.com/bliki/CanaryRelease.html

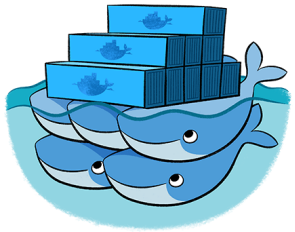
Canary, Rolling Updates

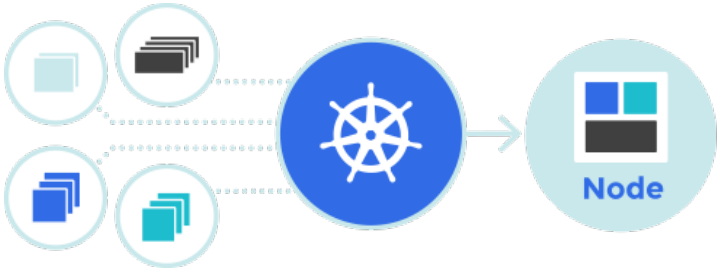
1. Trigger Deployment of the candidate via API
2. Wait for new deployment to be healthy
3. Store Service Discovery Key for the Candidate Version
4. Remove the Original (old candidate) Version
5. Gradually Increase Traffic to the candidate (from the stal version)
6. Promote the candidate to stable (gets all traffic)

Technology behind

Containers as a Service

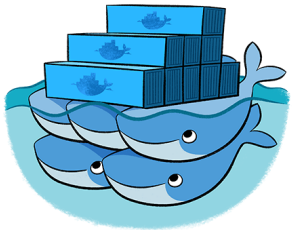
- We need a way (management system) to deploy containers
  - Could be DIY, or leverage a container service
- Stop, start containers, remove deployments
- Integrate via API





Rolling Update Support in Orchestrators (K8S and Swarm)

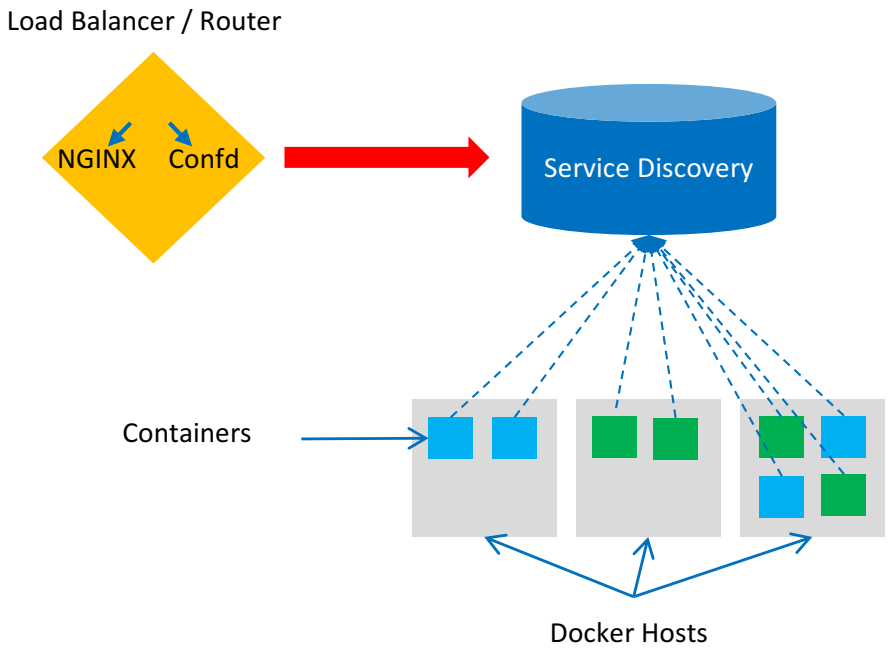
- Kubernetes supports rolling updates (via deployments in Kubernetes 1.2+)
- Part of deployments, control through “Max Unavailable” and “Max Surge” parameters
- https://kubernetes.io/docs/user-guide/deployments/#rolling-update-deployment



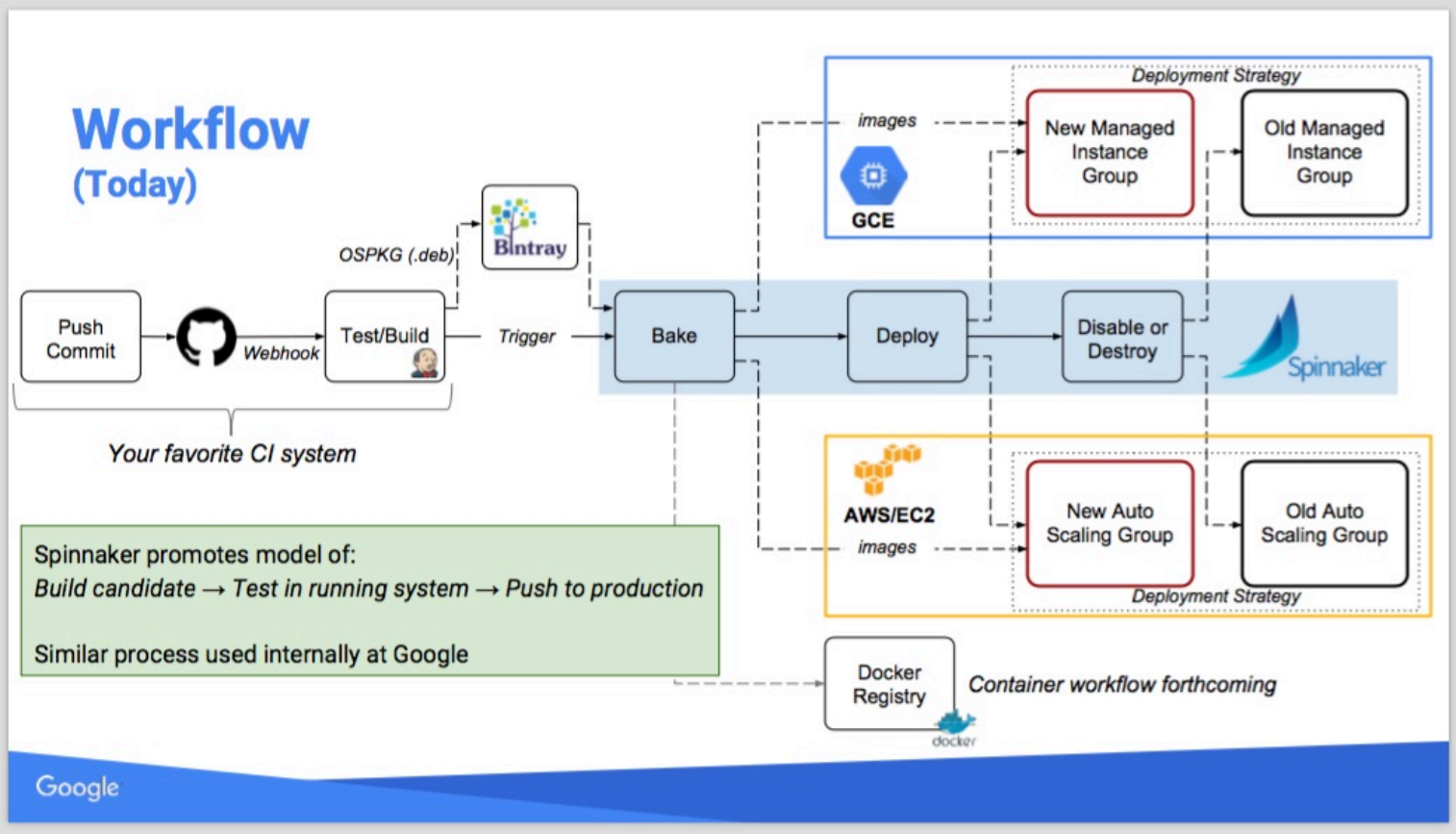
- Docker Swarm rolling updates in 1.12+
- Can define the #containers to update at a time and the delay
- Stops existing containers before starting new ones
- https://docs.docker.com/engine/swarm/swarm-tutorial/rolling-update/

Load Balancer and Service Discovery

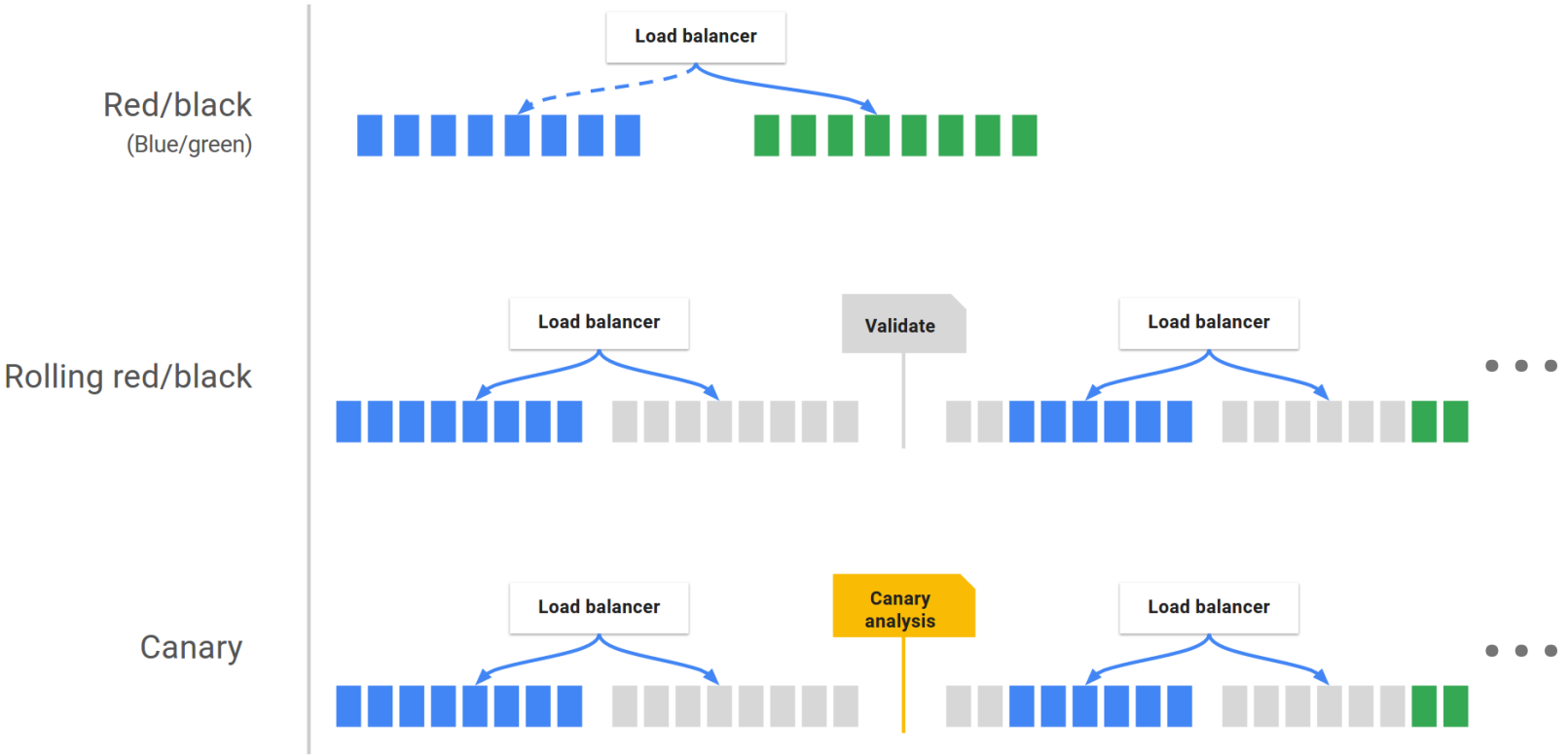
- Load Balancer Runs **NGINX**
- Each container registers with service discovery
- Service discovery for container placement
  - Reloads NGINX config i.e. picks up container adds and removes, config changes

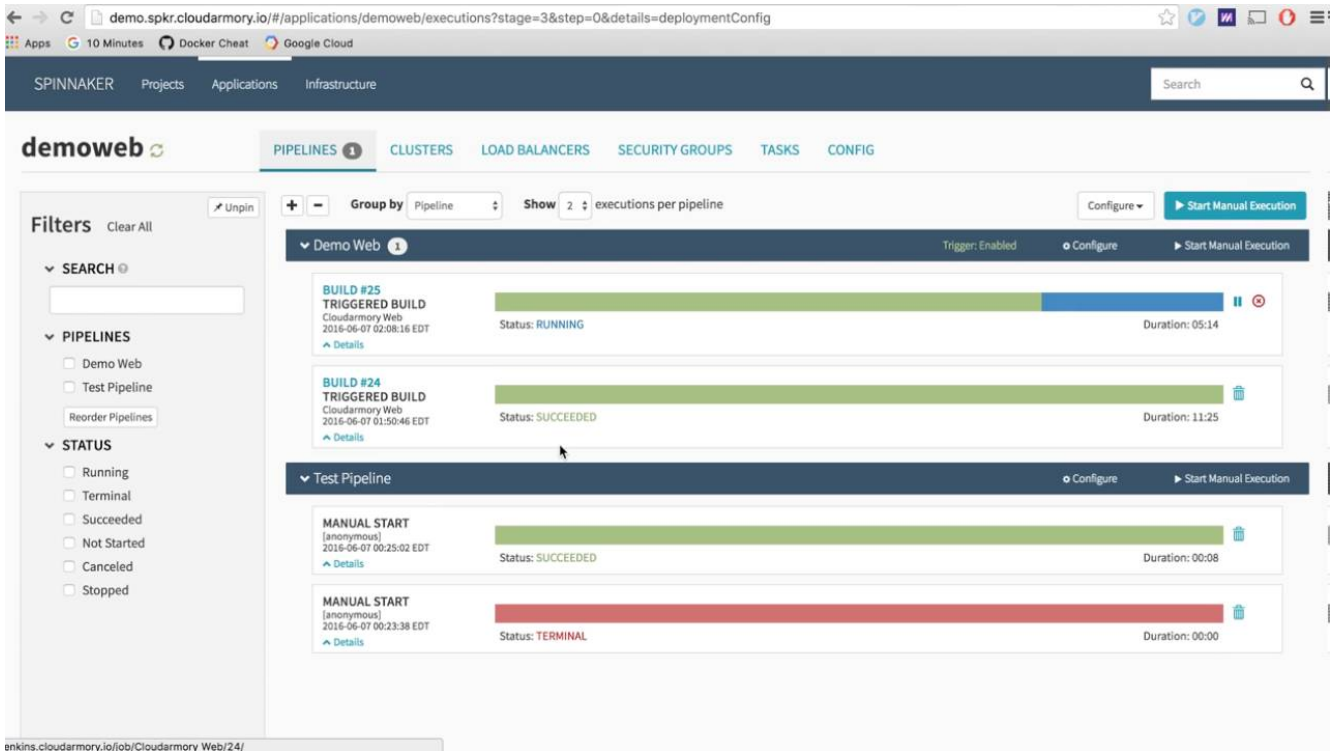


Putting it Together



Blue / Green Demo


Red/black
(Blue/green)
Load balancer
Rolling red/black
Load balancer
Validate
Load balancer
Canary
Load balancer
Canary analysis
Load balancer


demo.spkr.cloudarmory.io/#/applications/demoweb/executions?stage=3&step=0&details=deploymentConfig
Apps
10 Minutes
Docker Cheat
Google Cloud
SPINNAKER
Projects
Applications
Infrastructure
Search
demoweb
PIPELINES 1
CLUSTERS
LOAD BALANCERS
SECURITY GROUPS
TASKS
CONFIG
Unpin
Filters
Clear All
SEARCH
PIPELINES
Demo Web
Test Pipeline
Reorder Pipelines
STATUS
Running
Terminal
Succeeded
Not Started
Canceled
Stopped
Group by
Pipeline
Show
2
executions per pipeline
Configure
Start Manual Execution
Demo Web 1
Trigger: Enabled
Configure
Start Manual Execution
BUILD #25
TRIGGERED BUILD
Cloudarmory Web
2016-06-07 02:08:16 EDT
Details
Status: RUNNING
Duration: 05:14
BUILD #24
TRIGGERED BUILD
Cloudarmory Web
2016-06-07 01:50:46 EDT
Details
Status: SUCCEEDED
Duration: 11:25
Test Pipeline
Configure
Start Manual Execution
MANUAL START
[anonymous]
2016-06-07 00:25:02 EDT
Details
Status: SUCCEEDED
Duration: 00:08
MANUAL START
[anonymous]
2016-06-07 00:23:38 EDT
Details
Status: TERMINAL
Duration: 00:00
enkins.cloudarmory.io/job/Cloudarmory Web/24/

https://www.youtube.com/watch?v=UOkZJazycQs&list=PLx2By31njbhrs8caX08BEusyD_fBDu-XG&index=28

https://www.spinnaker.io/guides/tutorials/codelabs/hello-deployment/

http://blog.armory.io/complex-pipelines-with-spinnaker/