Continuous Deployments with CI\CD Pipelines & Kubernetes

DEVOPSCON Berlin ONLINE

14.06.2021

Oz Tiram

Workshop

/ˈwəːkʃɒp/

noun: workshop; plural noun: workshops

> a meeting at which a group of people engage in intensive discussion and activity on a particular subject or project.

Who am I?

Who are you?

- What is your role?
- Have you worked with k8s before?
- How are you using kubernetes?

Agenda

- Continuous integration and continuous deployment
- Deploying straight to production
- Building a pipeline using gitlab
- Deploying to Kubernetes

What is CI\CD?

- What is Continuous Integration
- What is Continuous Delivery
- Are you doing it? How are you doing it?
- What do you like about it?
- What do you not like about it?

Continuous Integration

Continuous integration (CI) is the practice, in software engineering, of merging all developer working copies with a shared mainline several times a day.

Continuous integration involves integrating early and often, so as to avoid the pitfalls of "integration hell". The practice aims to reduce rework and thus *reduce cost and time.*

Continuous Delivery

Continuous Delivery doesn't mean every change is deployed to production ASAP. It means every change is proven to be deployable at any time.

Where do you deploy to?

- Do you have Jenkins or Similar at work?
- Do you have multiple “environments” at work?
- If so, how do they defer?
- What possible problem could arise from such differences?

Manual Deployment and Testing

Manual Deployment and Testing

*Manual Testing is immoral. It's not just dumb - and it is dumb -
it is immoral, because you are taking people and you are asking them to act like machines.*

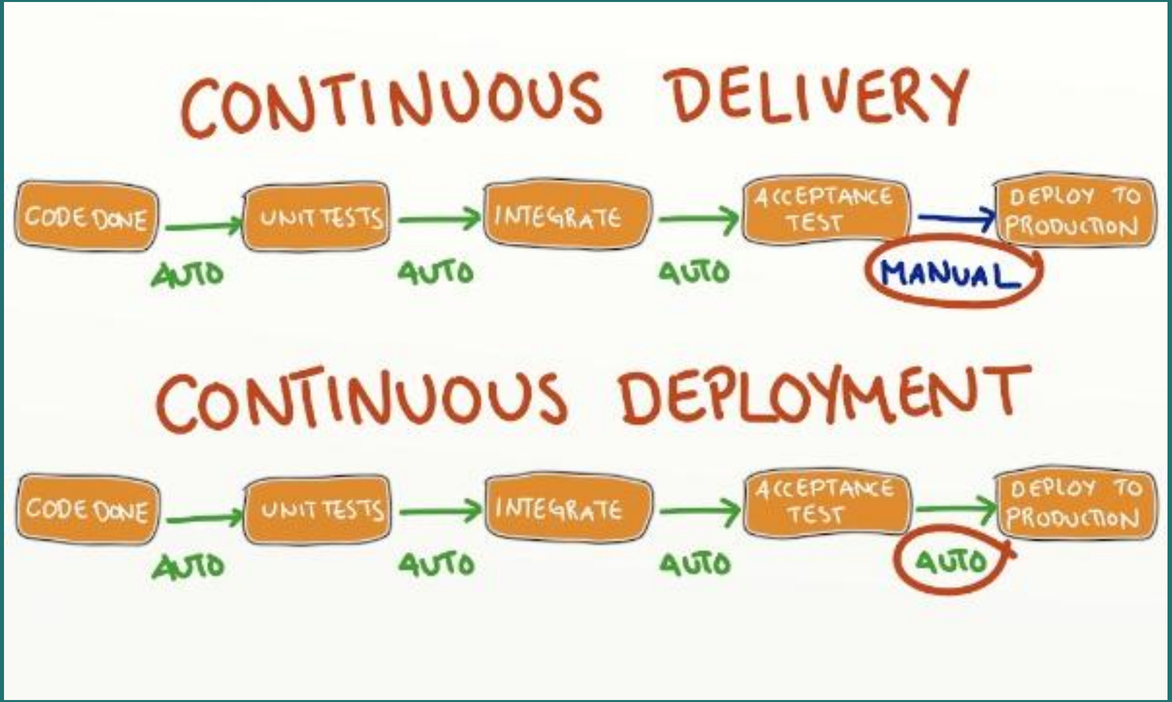
Bob Martin, The land that scrum forgot.

https://www.youtube.com/watch?v=hG4LH6P8Syk&t=2202s

Continuous

~~Delivery~~

Deployment!


CONTINUOUS DELIVERY
CODE DONE
AUTO
UNIT TESTS
AUTO
INTEGRATE
AUTO
ACCEPTANCE TEST
MANUAL
DEPLOY TO PRODUCTION
CONTINUOUS DEPLOYMENT
CODE DONE
AUTO
UNIT TESTS
AUTO
INTEGRATE
AUTO
ACCEPTANCE TEST
AUTO
DEPLOY TO PRODUCTION

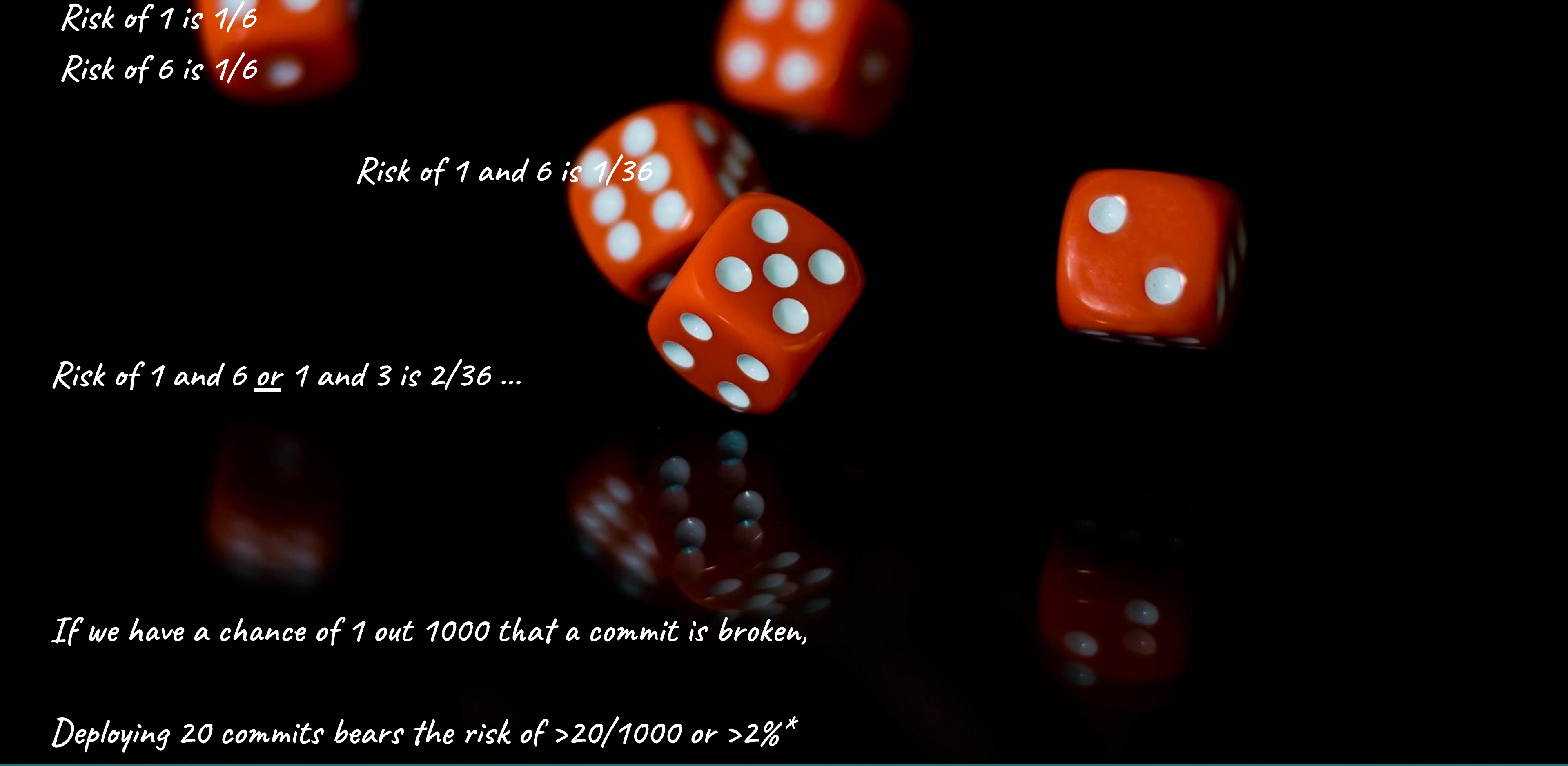
Continuous Deployment

- Reduces hardware and maintenance costs
- Reduces manual labor

Continuous Deployment

- Code Changes bare risk
- Delay → more change -⟩⟶ more risk

Continuous Deployment is reducing risk!



Risk of 1 is 1/6

Risk of 6 is 1/6

Risk of 1 and 6 is 1/36

Risk of 1 and 6 <u>or</u> 1 and 3 is 2/36 ...

If we have a chance of 1 out 1000 that a commit is broken,

Deploying 20 commits bears the risk of >20/1000 or >2%*

Continuous Deployment

- Code Changes bare risk
- Delay → more change -⟩⟶ more risk

Continuous Deployment is reducing risk!

Continuous Deployment Strategies

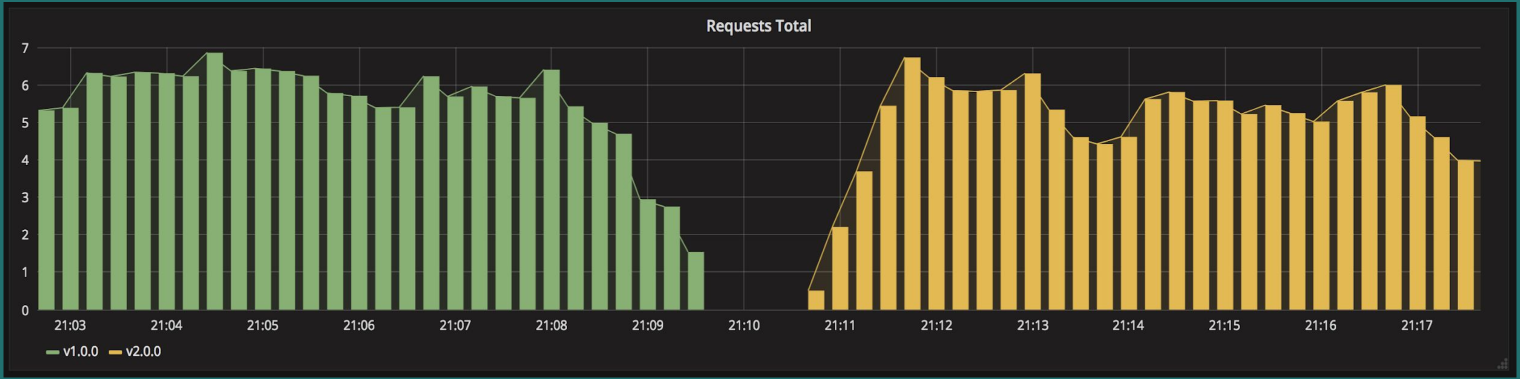
0. Application Recreation

- Strictly speaking this isn’t continuous deployment.
- We replace version 1 with version 2 with a “big bang”.
- Usually practiced with “Continuous Delivery”

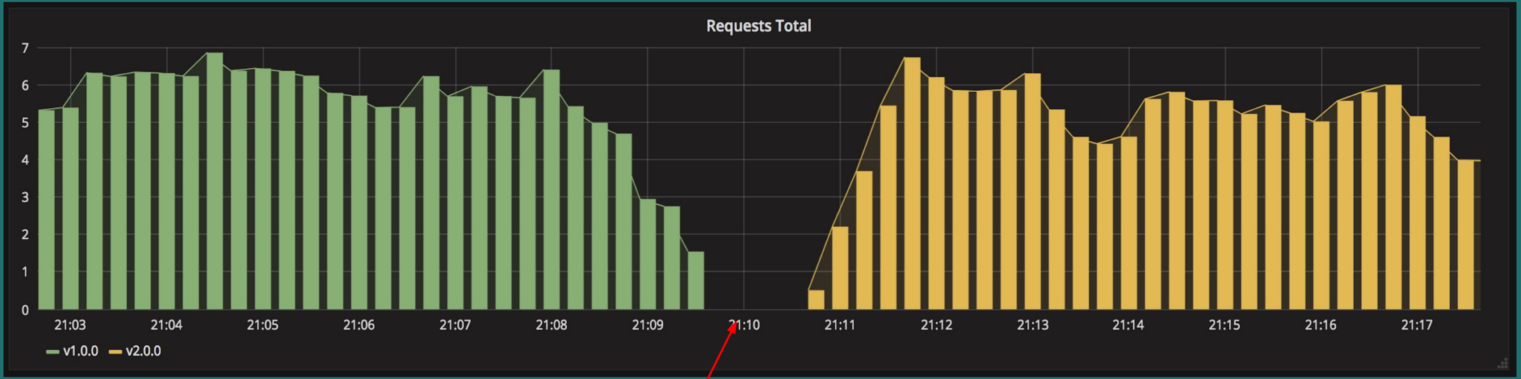
0. Application Recreation

- The most common strategy
- Companies that have this may have one or more of:
  - Separate environments
  - Separate deployment pipeline
  - Release ceremonies and or release manager

0. Application Recreation

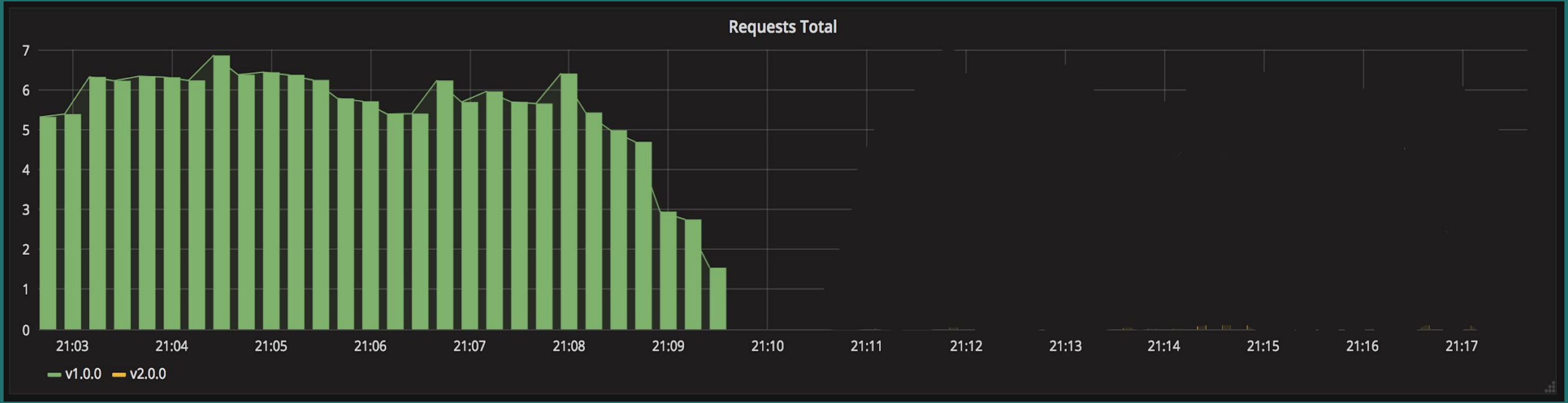


0. Application Recreation



Downtime!

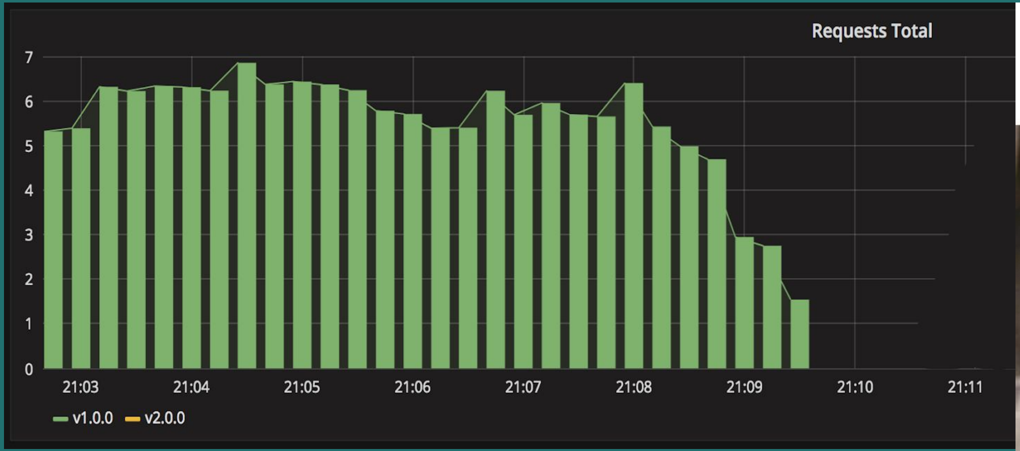
0. Application Recreation



Downtime with release gone wrong!

DevOpsCon THE ONLINE CONFERENCE

0. Application Recreation



Downtime with release gone wrong!

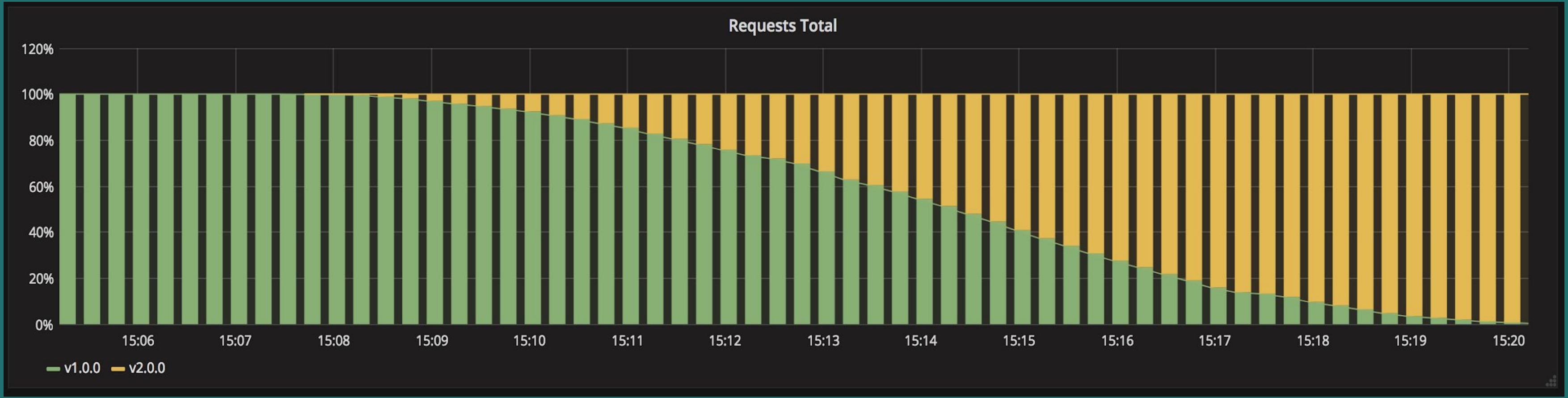
Downtime, how long ?!

Uptime SLA	Monthly downtime
99.99	4 m 22s
99	7h 18m 17s
96	1d 5h 13m 9s

1. Rolling release

- Easy if you automate all your tests.
- Can be rolled back automatically, if you have monitoring in place!
- <u>Usually, it is the norm *outside* the software industry</u>!

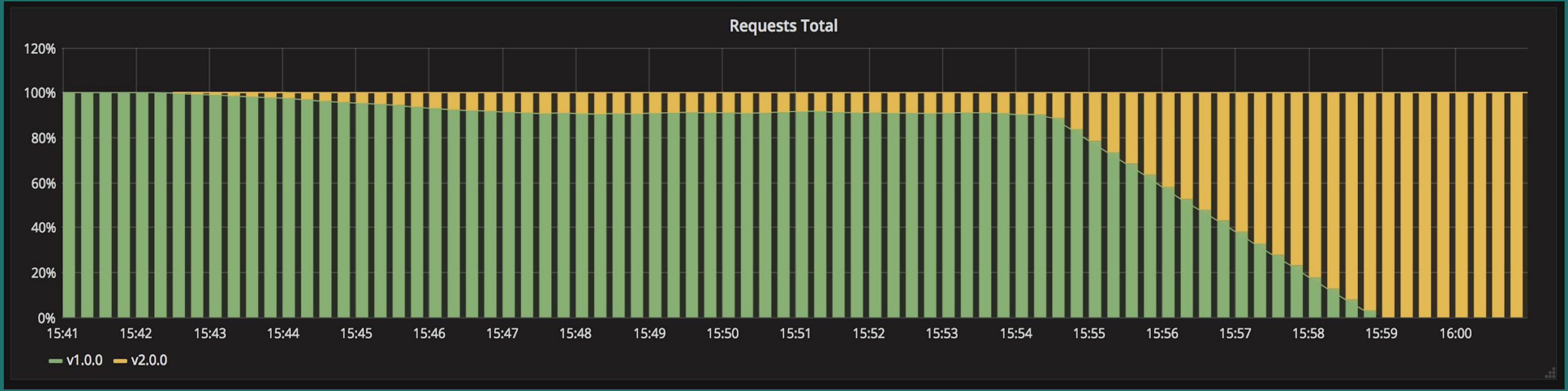
1. Rolling release



Risk control and management

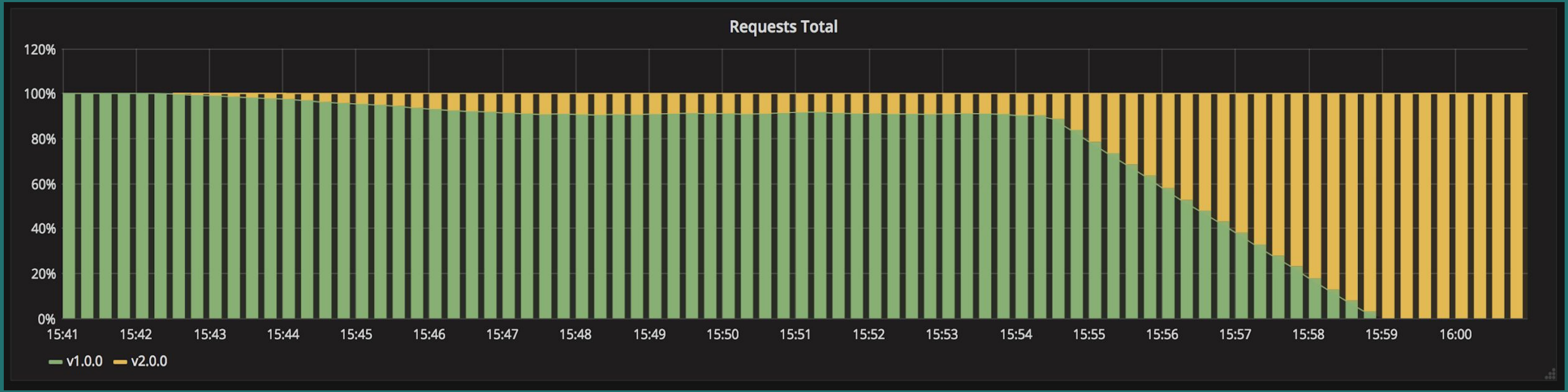
DevOpsCon THE ONLINE CONFERENCE

2. Canary releases



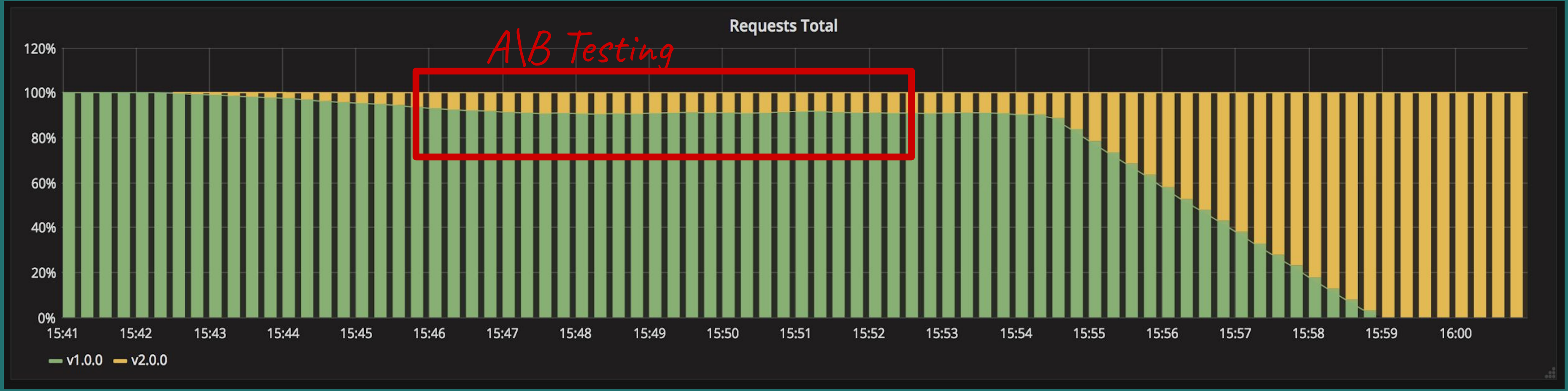
The Scientific method!

2. Canary releases



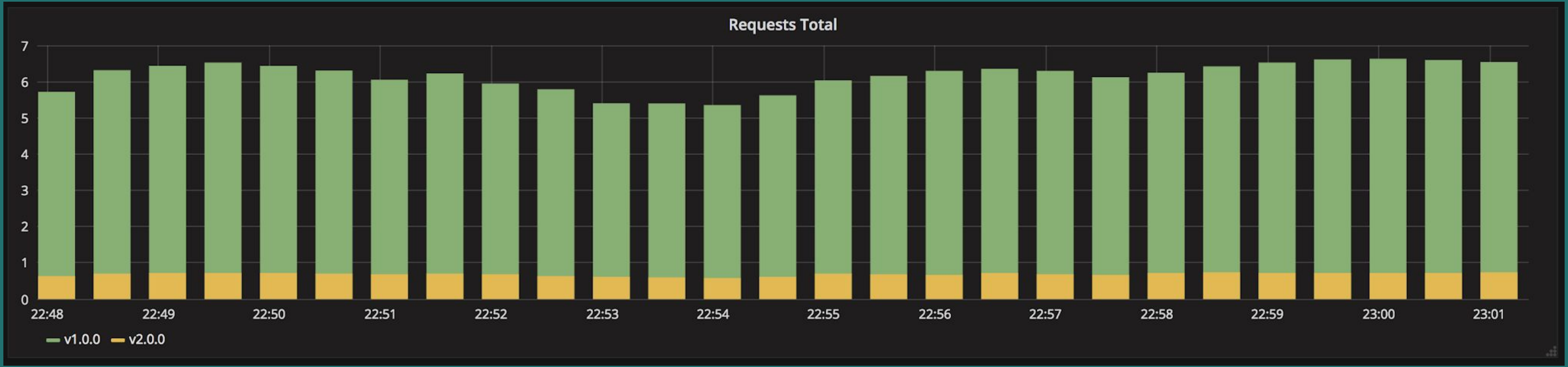
The Scientific method!

2. Canary releases



The Scientific method!

3. A\B, Blue\Green deployment



The Scientific method with user specific targeting

Canary release, A\B Testing

- Can’t be avoided in large organizations
- Allow organizations to carefully test in production

Strategy Comparison

Strategy	Downtime	Real traffic testing	Targeting users	Rollback duration	Impact on users	Complexity
Recrate	Yes	No	No	Long	High	Low
Rolling	No	No	No	Long	Medium	Medium
Canary	No	Yes	Yes	Short	Low	High

DevOpsCon
THE ONLINE CONFERENCE