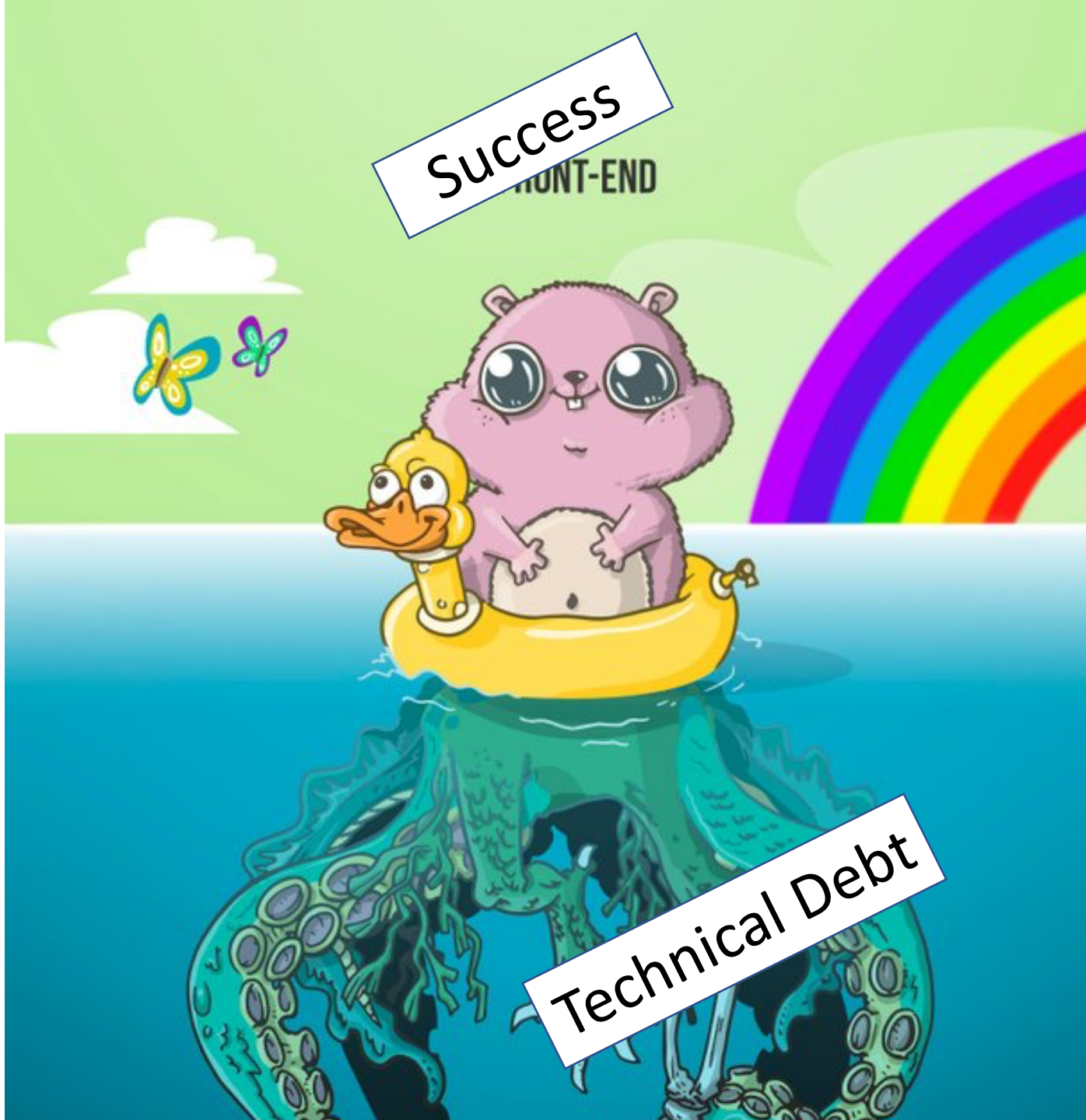
Best Practices introducing Kubernetes

Wojciech Barczyński [Head of Engineering]

SMACC.io | Hypatos.ai

Whoami?

- Developer
- System Engineer
- I am fixing things



Experience

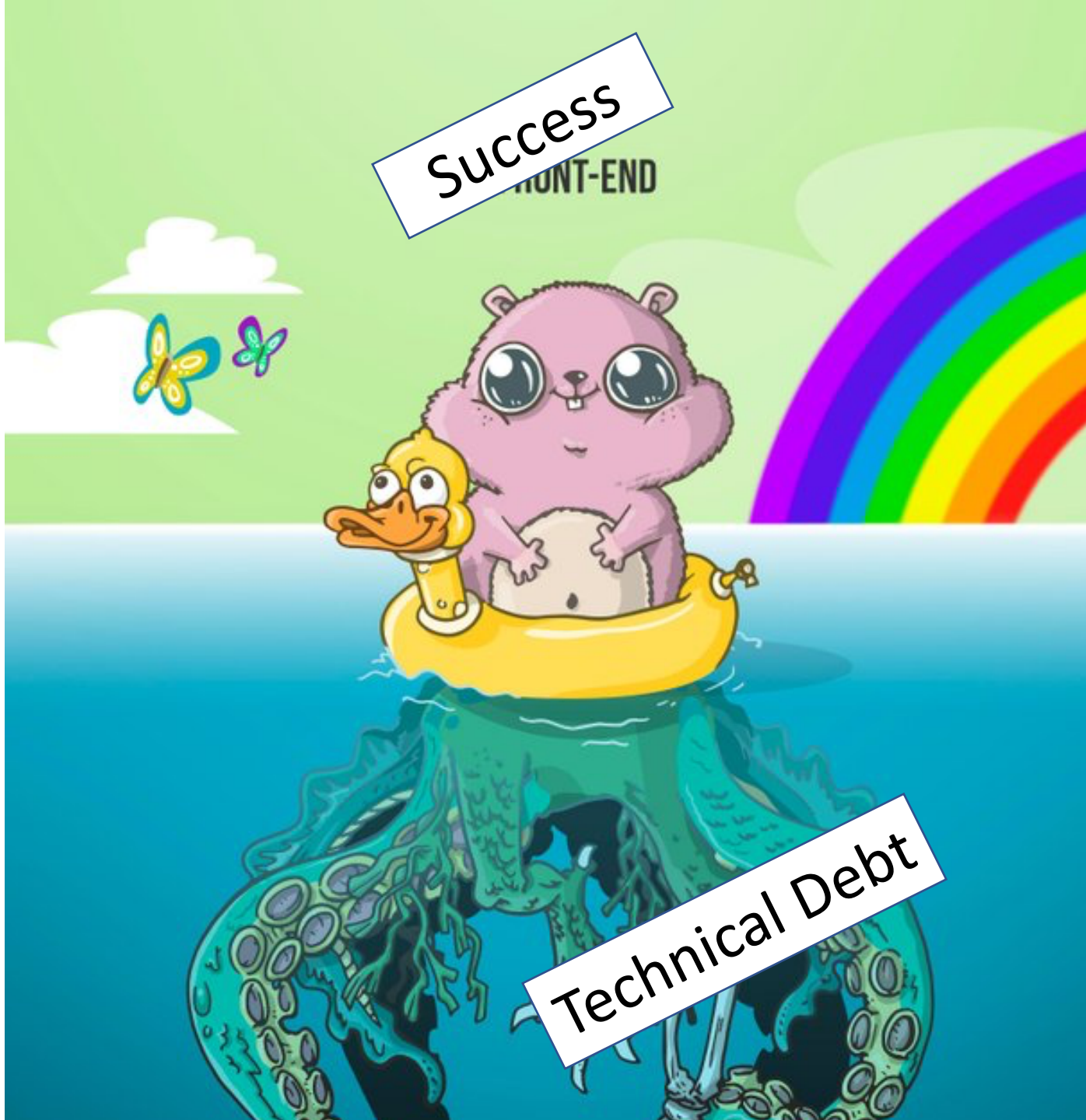
- Lykehq – [12.2016 - 07.2017]
- SMACC.io – [10.2017 - present]
- Hypatos.ai – [08.2018 - present]

Before:

Cloud&Heat (Openstack, green data centers)
and SAP R&D.

Experience

- Continuous Deployment
- X to Kuberentes
- Microservices
- …



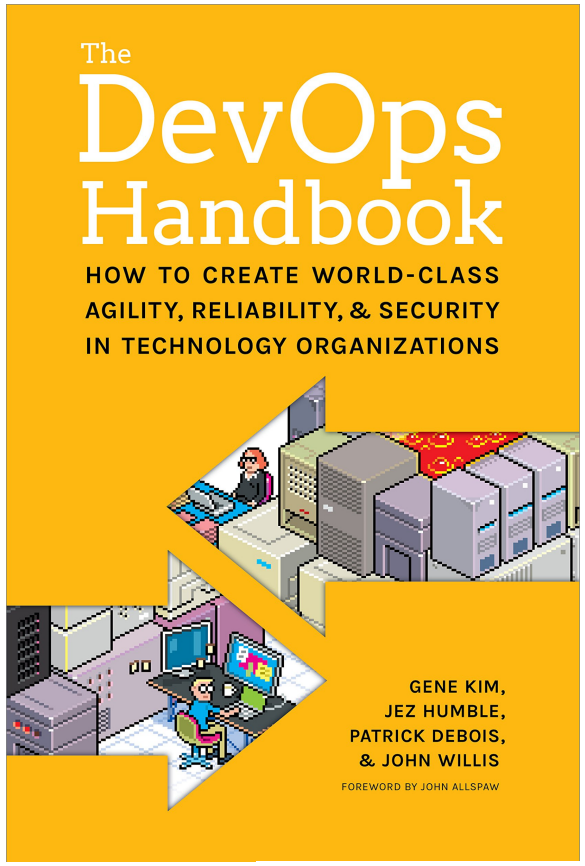
Hypatos / SMACC.io

Go

PYTORCH

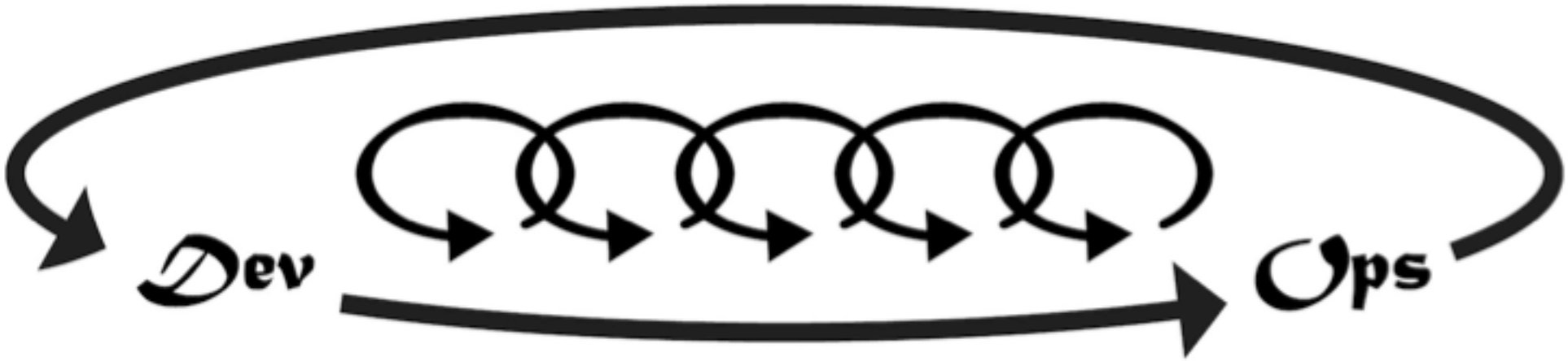
TensorFlow™

CloudNative: Prometheus, Grafana, Fluentd, Grafana Loki



Three ways:

- Flow
- Feedback
- Continuous Experimentation and Learning



Proces of small fixes.

Every sprint better then one before.

1 week sprints

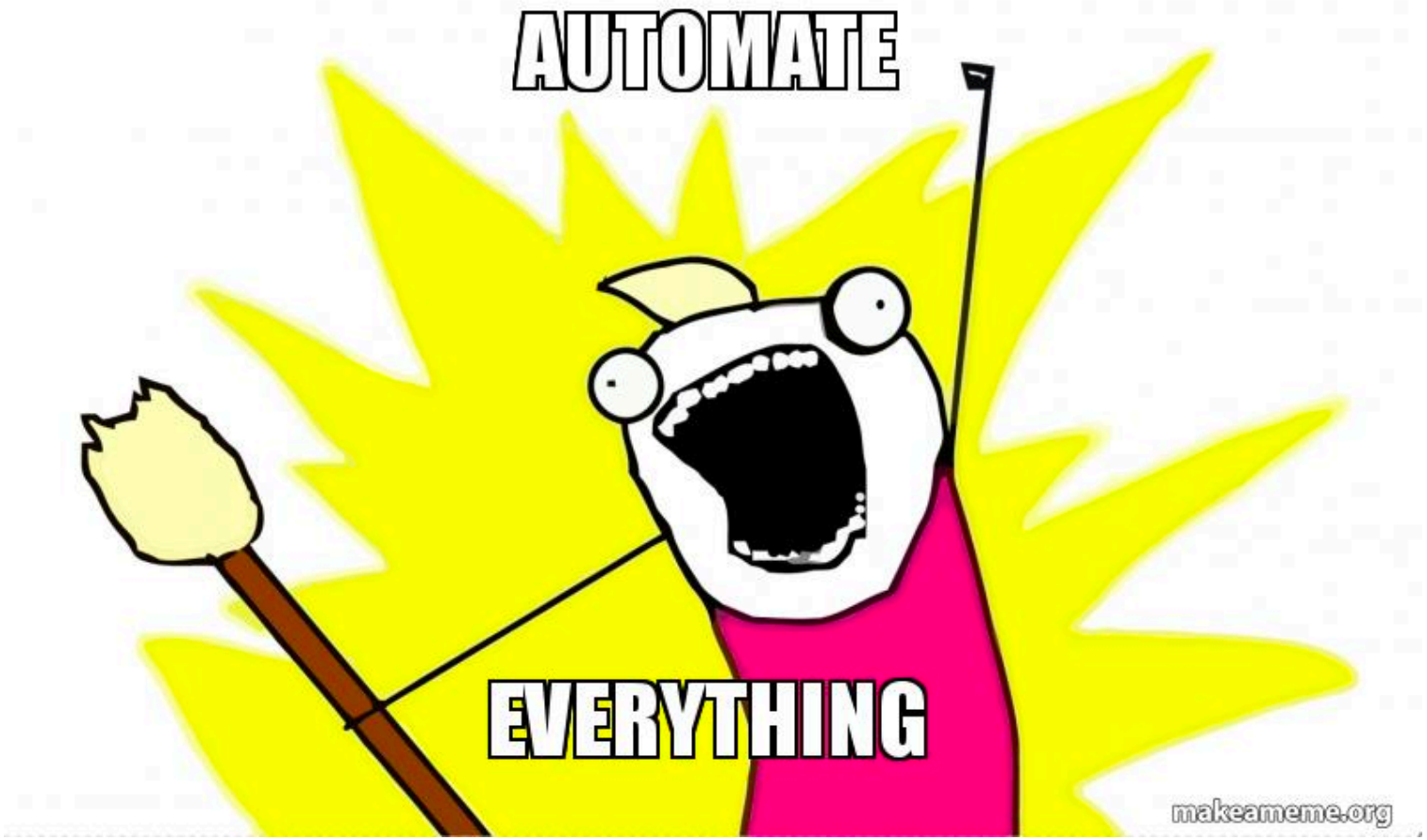
Max 2 weeks*

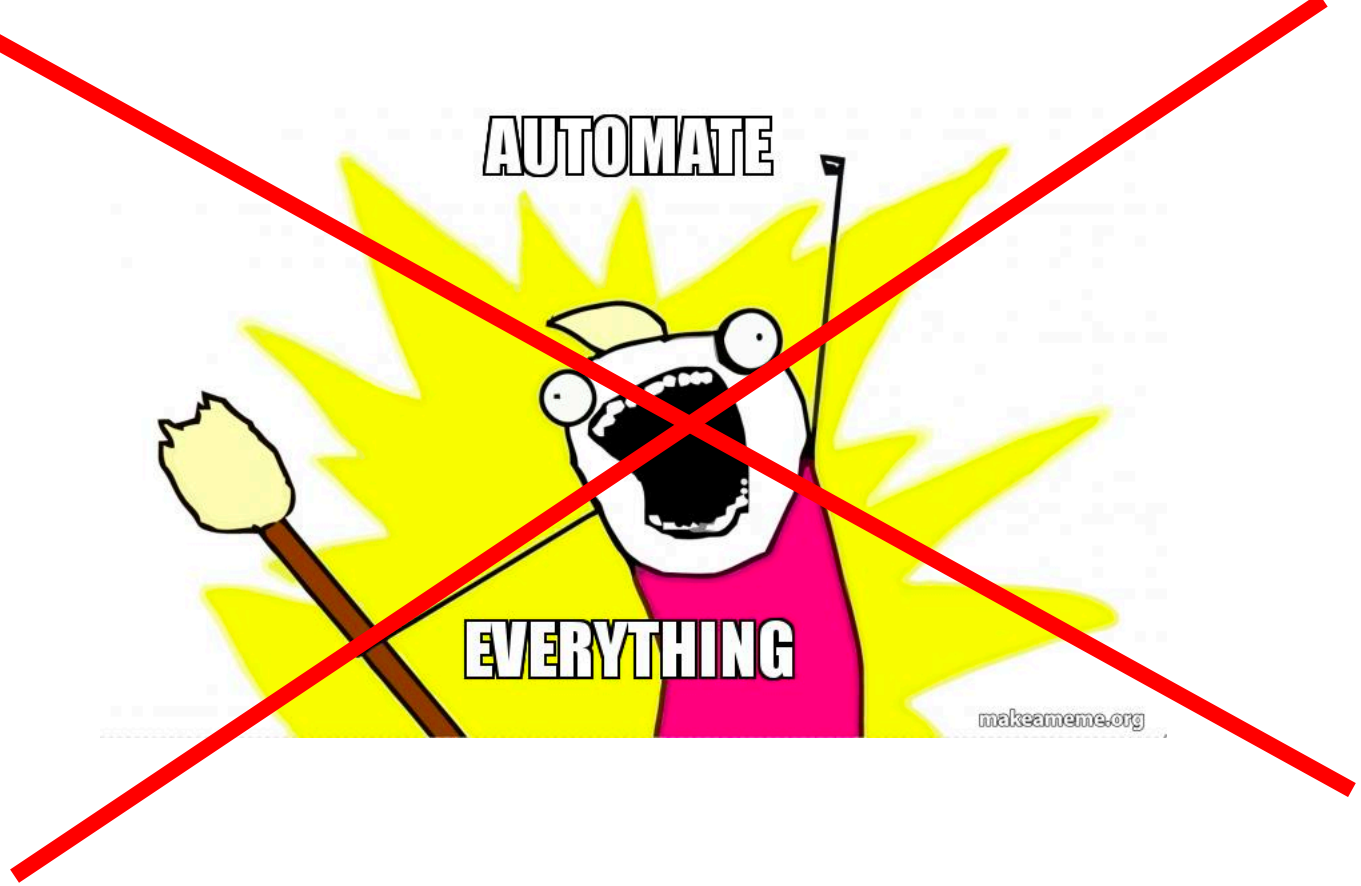
Changes

Productivity

Morale of each team members

Automate!





Finding golden mean

Risks of Automation

- Never-ending optimization
- Easy to frustrated
- Go in wrong direction
- As we go, it gets harder

Risks of Automation

- Never-ending optimization
- Easy to frustrated
- Go in wrong direction
- As we go, it gets harder

Time-boxing! V1/V2

Analogy: Quality of software

- Never-ending optimization
- Easy to frustrated
- Go in wrong direction
- As we go, it gets harder
- Rigid code

Analogy: Quality of software

- Never-ending optimization
- Easy to frustrated
- Go in wrong direction
- As we go, it gets harder
- Rigid code

1. **Ship it!**
2. **Make mistakes**
3. **Patching**
4. **Patching…**

Low Quality
Low Automation

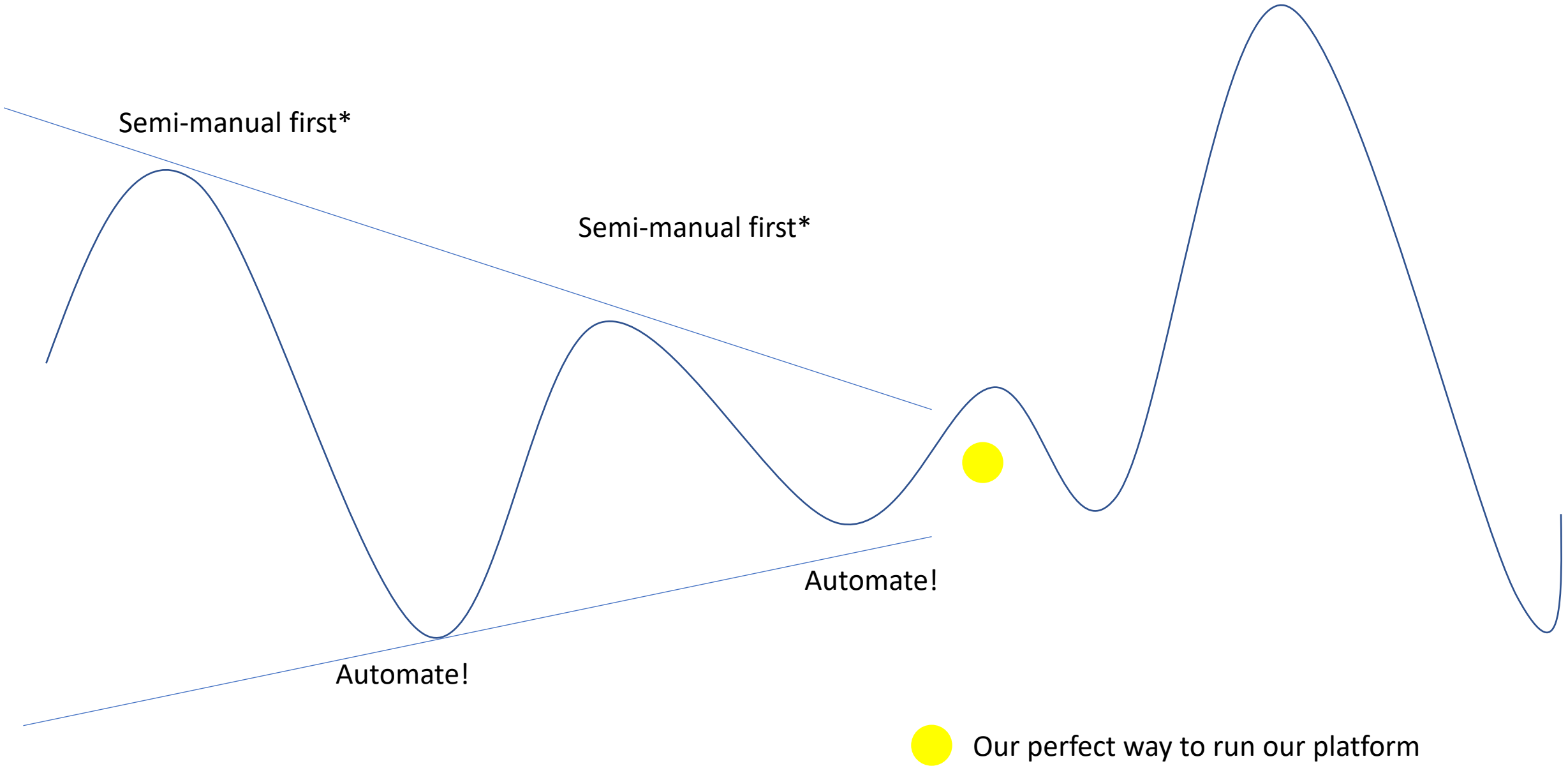
N-sprint:
We are working on Kubernetes…

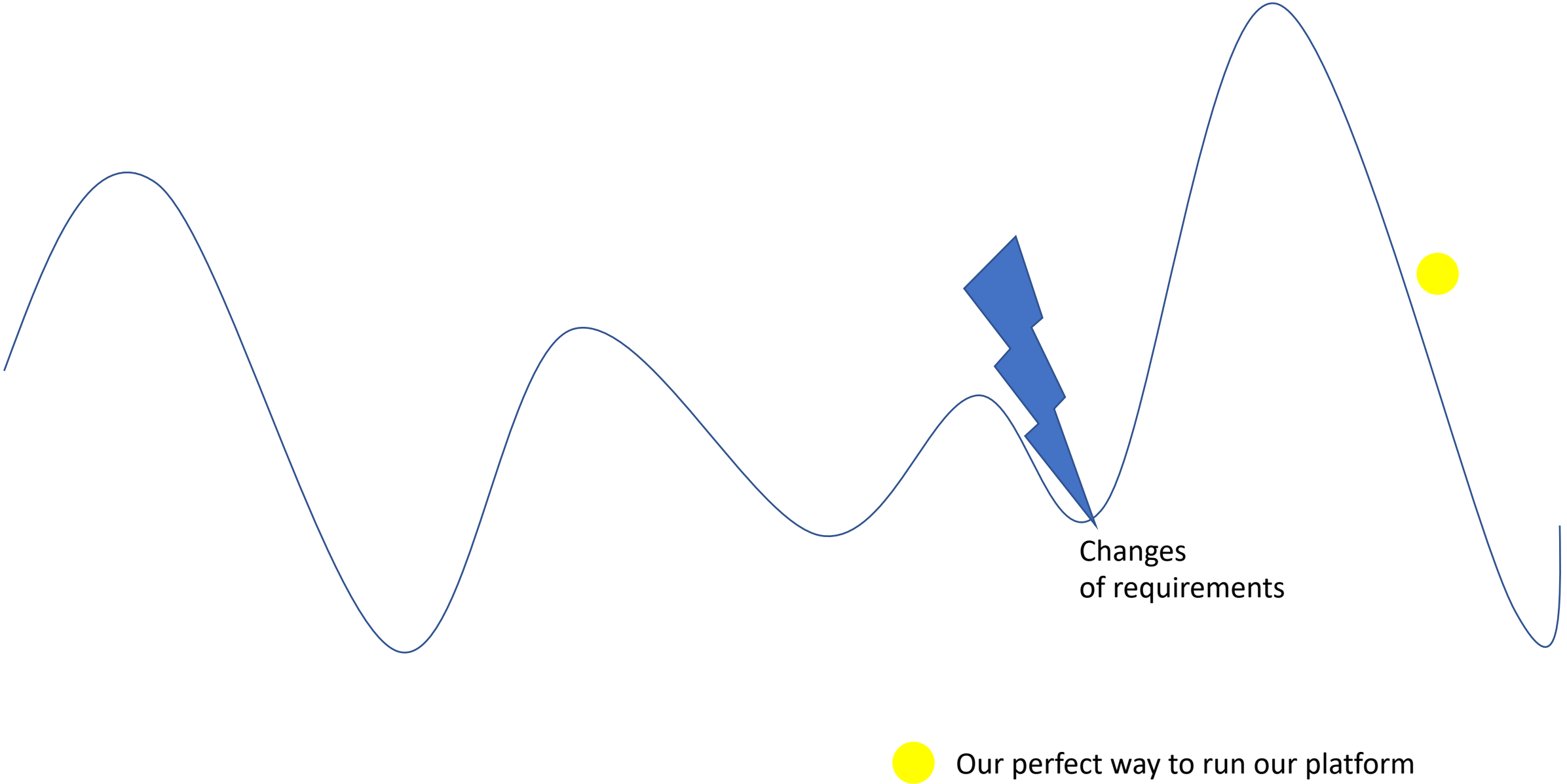
…

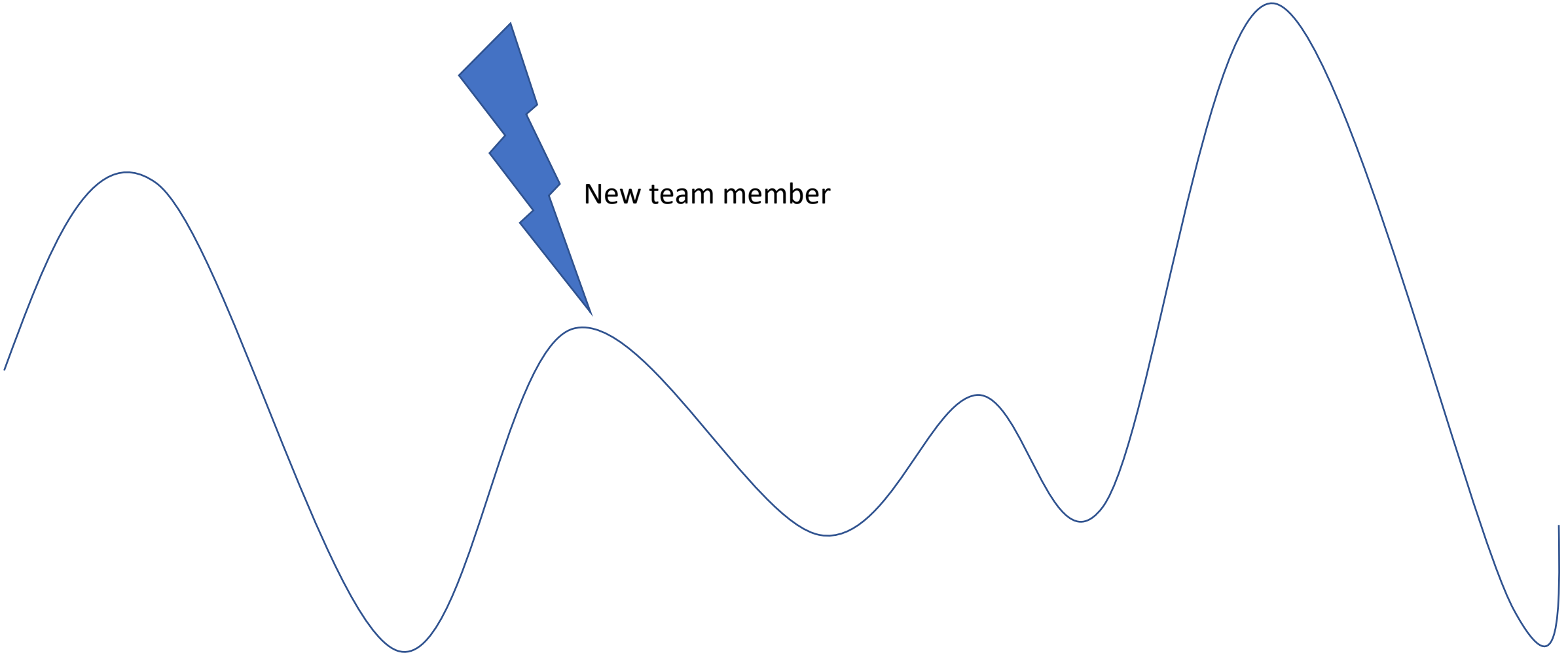
…

New tech

Early wins
Trust


Semi-manual first*
Semi-manual first*
Automate!
Automate!
Our perfect way to run our platform


Changes
of requirements
Our perfect way to run our platform


New team member

How to find it?

team

inclusion

diversity

engineering culture

Assumptions

- Basics covered: **Continuous Deployment**
- Without Continuous Delivery, hard to get started

Semi-manual first

- Literate Pull Request
- README
- Makefile

Literate Pull Request

- Comments / Description
- shall, salt,
  ansible, terraform, curl
- Screenshoty

README

Receipt: mix text and commends, avoid screenshots if we can:

- Description
- Run terraform
- Call curl, …
- Create user

README

Few sprints, we can live with commends C&P from README.

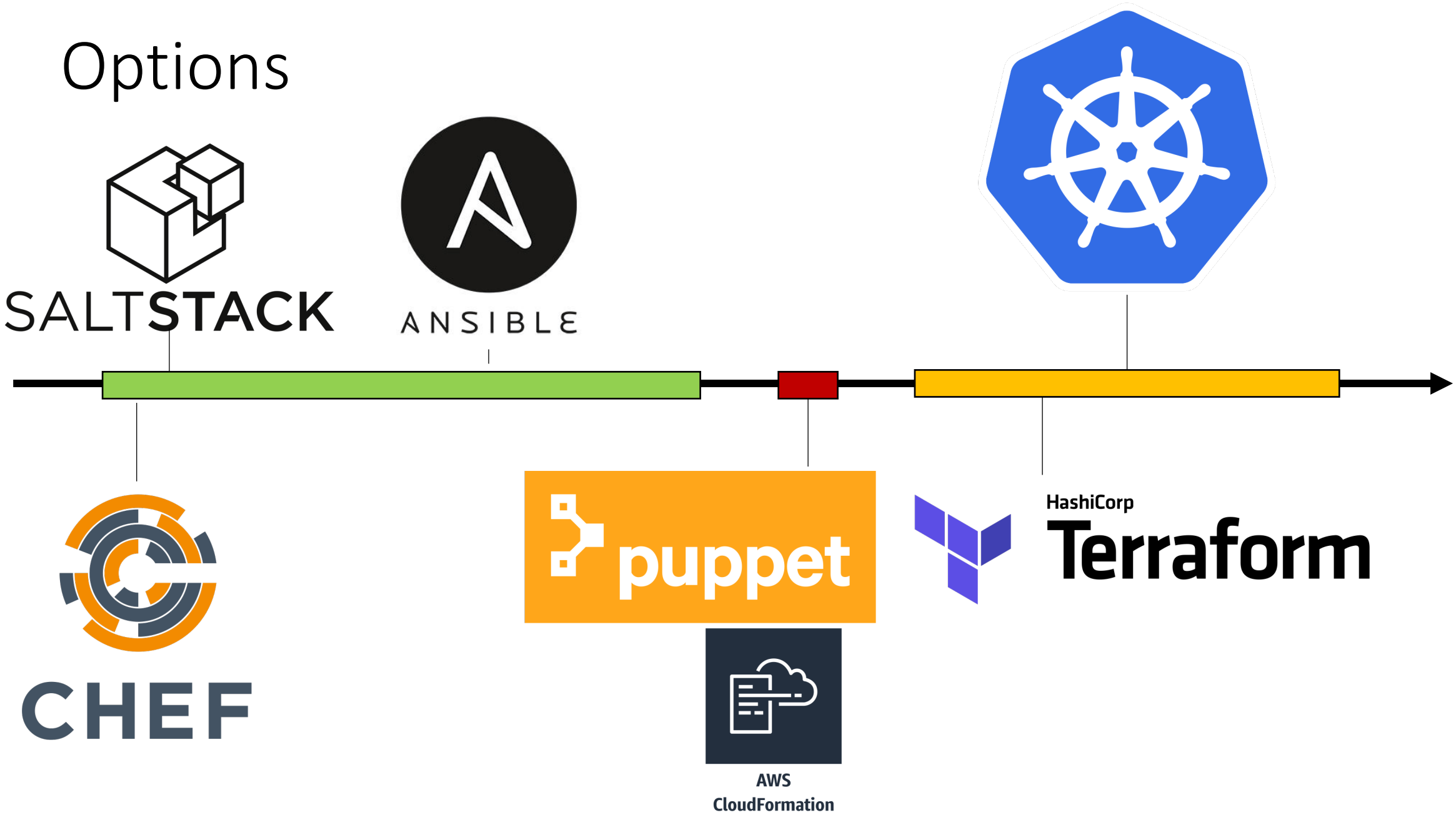
We treat it as a code.

+ Architecture Decision Records

From semi- to automatic

- Better understanding of requirements
- New team member, new project, some kind of impulse

Options


SALTSTACK
ANSIBLE
CHEF
puppet
AWS
CloudFormation
HashiCorp
Terraform

Declarative

tf files easy to understand

HashiCorp Terraform

You know what it will do:
Dry run (plan)

One to rule them all

+ Terragrunt

Kubernetes

Hidding Infrastructure

Simple* Continuous Deployment

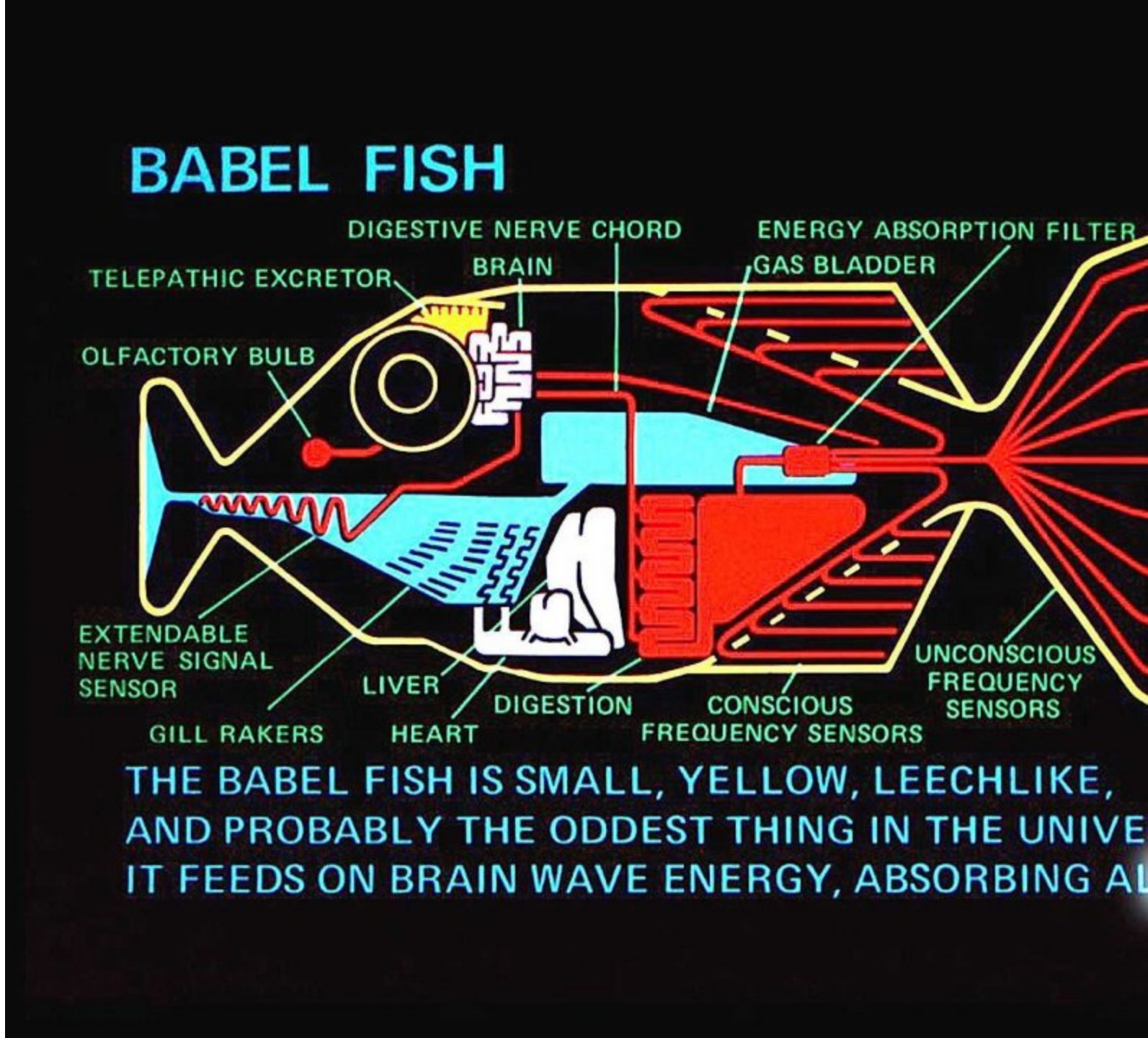
https://en.wikipedia.org/wiki/File:Dr_Who_(316350537).jpg

Common

Language

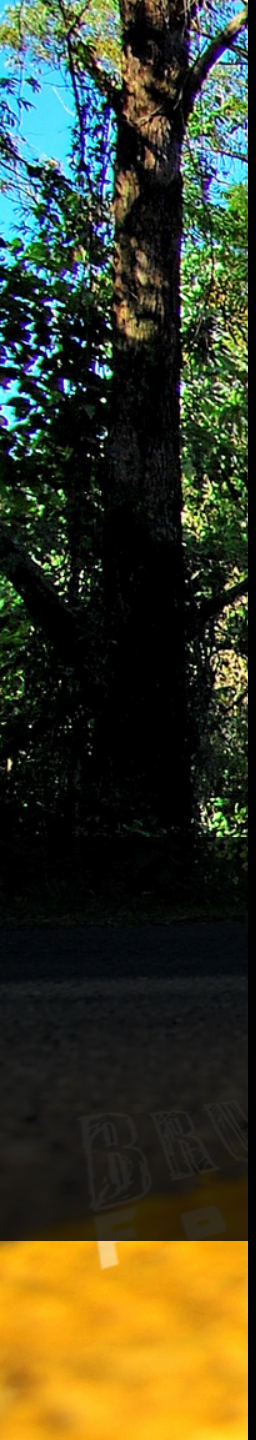
Artifacts

Platform



Learn-as-you-go

https://www.flickr.com/photos/bruno_brujah/



```
apiVersion: apps/v1
kind: Deployment
metadata:
  name: my-app
spec:
  replicas: 1
  template:
    metadata:
      labels:
        name:my
        app: ruletheworld
    spec:
      containers:
      - name: yourapp
        image: yourapp:1.0.0
        ports:
        - containerPort: 80
```

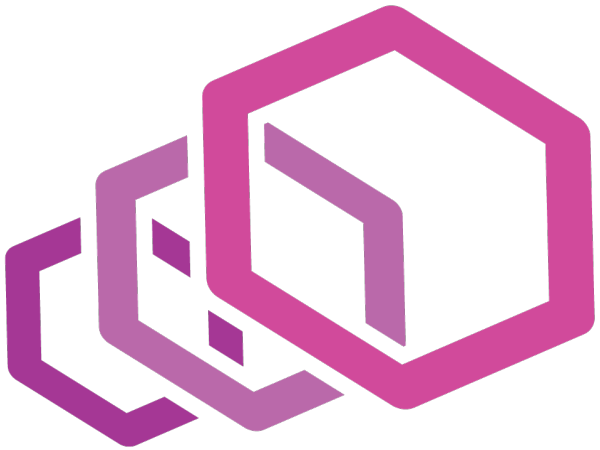
https://www.flickr.com/photos/bruno_brujah/

Experiment and Learn

1. Deploy Cloud-Native app
2. Make a Hell of Mistakes
3. Get it right or Postpone

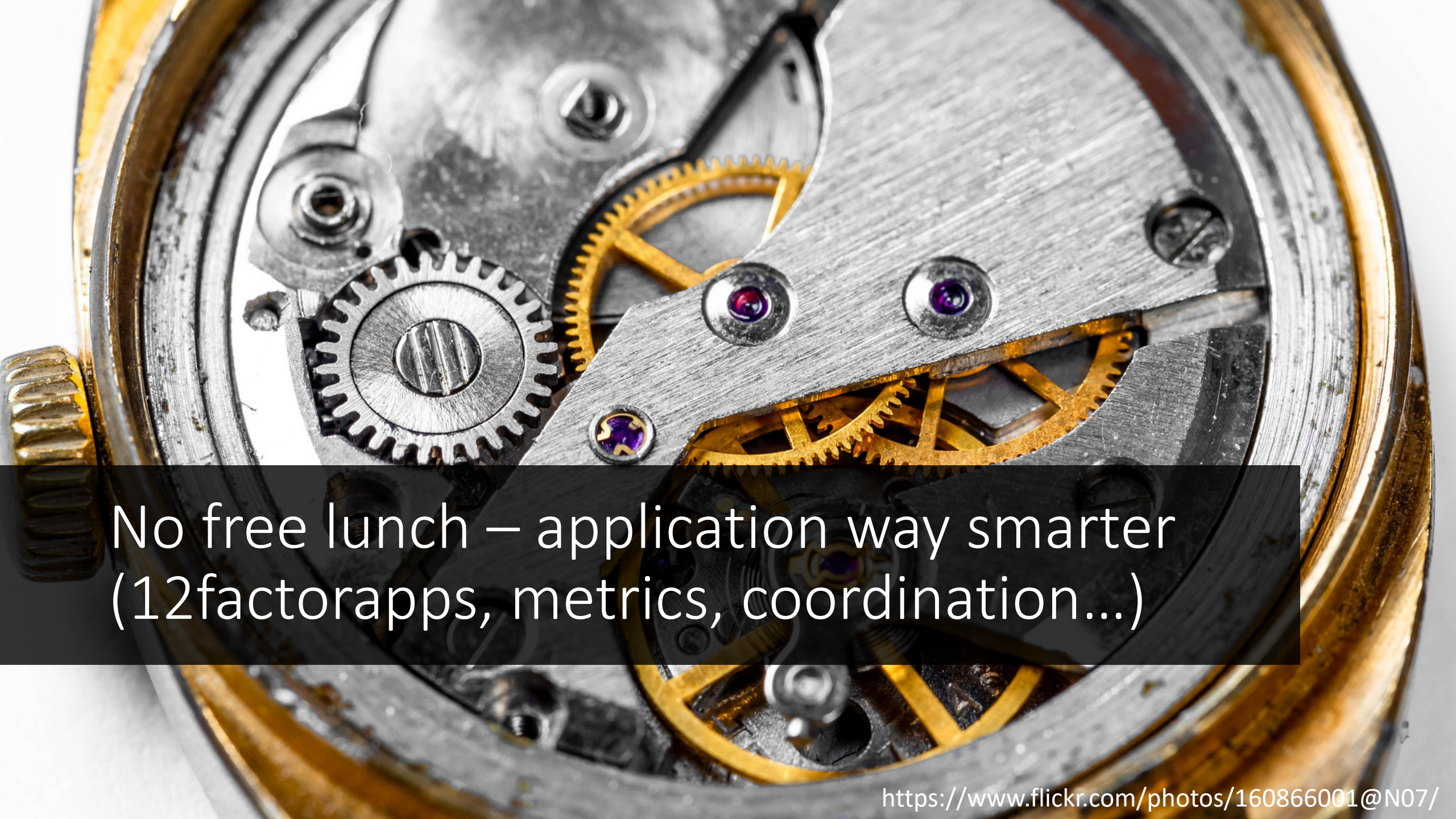
Experiment and Learn

1. Deploy Cloud-Native app
2. Make a Hell of Mistakes
3. Get it right or Postpone



envo

Deploy
and patch!



No free lunch – application way smarter (12factorapps, metrics, coordination…)

https://www.flickr.com/photos/160866001@N07/

Application

- Work with devs side by side
- Help to create Dockerfile or optimize it
- Adding metrics, structured logs
- Preparing copy&paste templates

Application

- Get more sophisticated
- More operation responsibility, e.g., data migrations
- See: 12factorapps

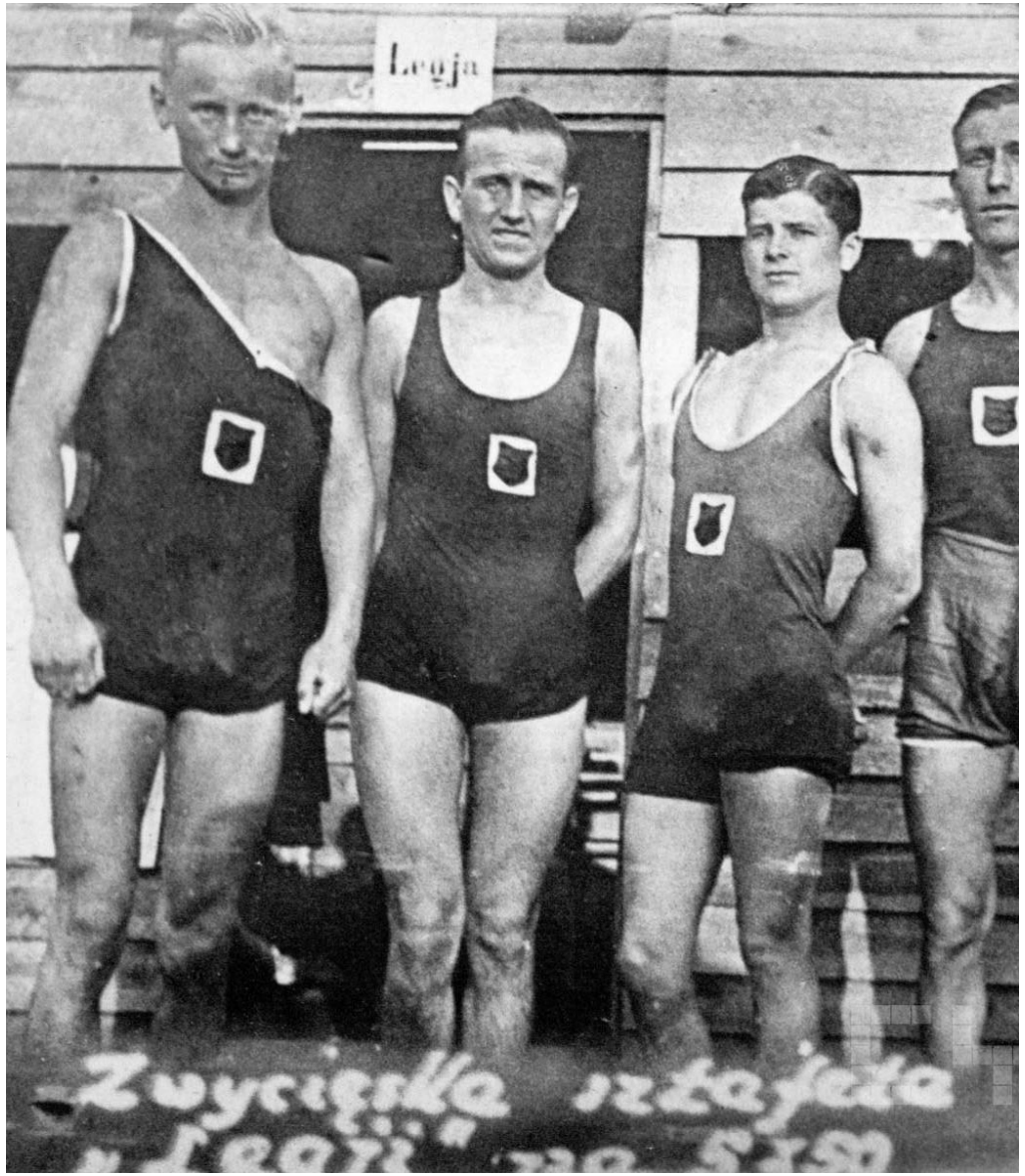
Keep it simple

https://www.flickr.com/photos/bruno_brujah/

Keep everybody in the process

1. Teach the team Kubernetes definitions

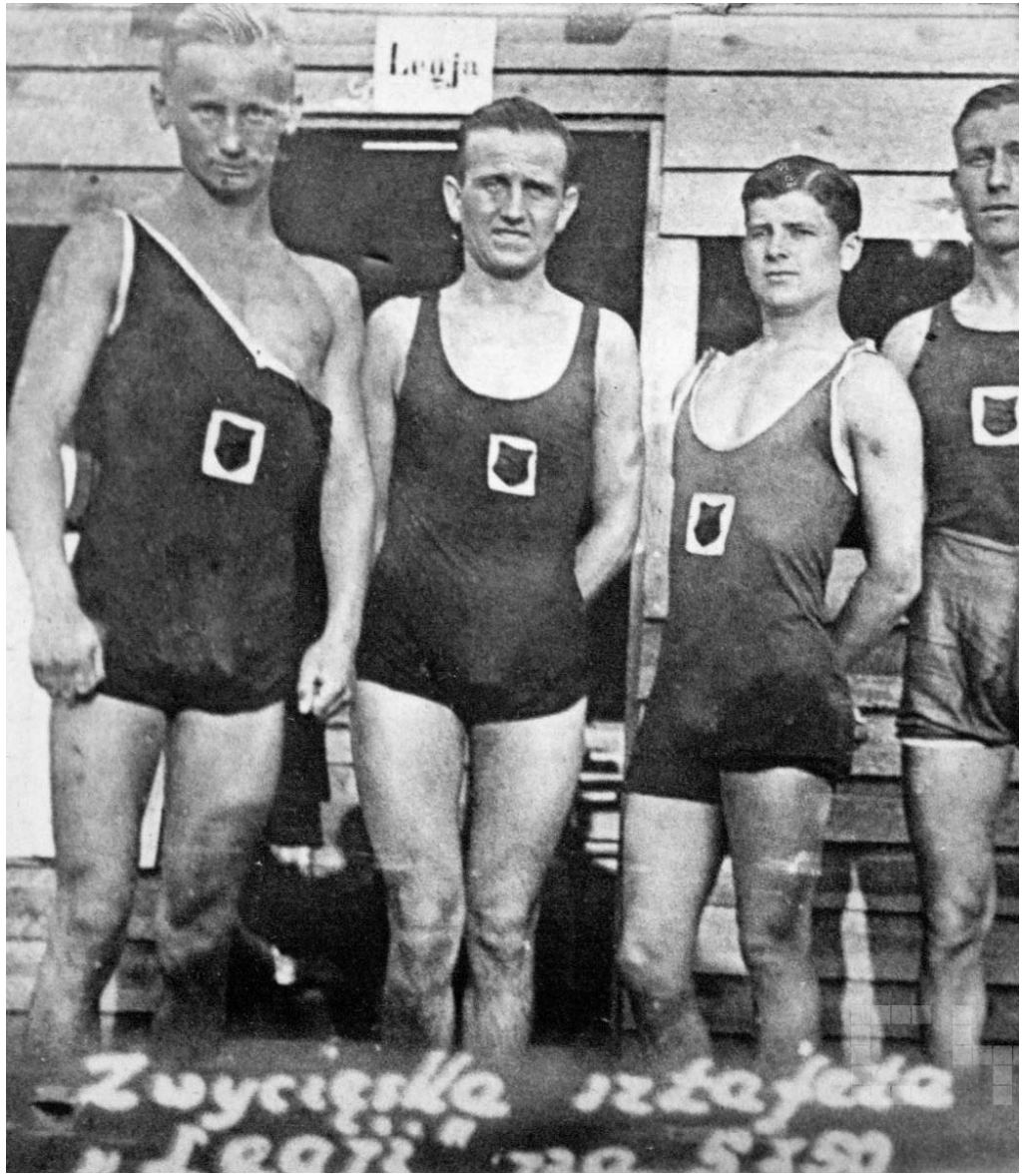
2. Keep the process understandable and simple



Keep everybody in the process

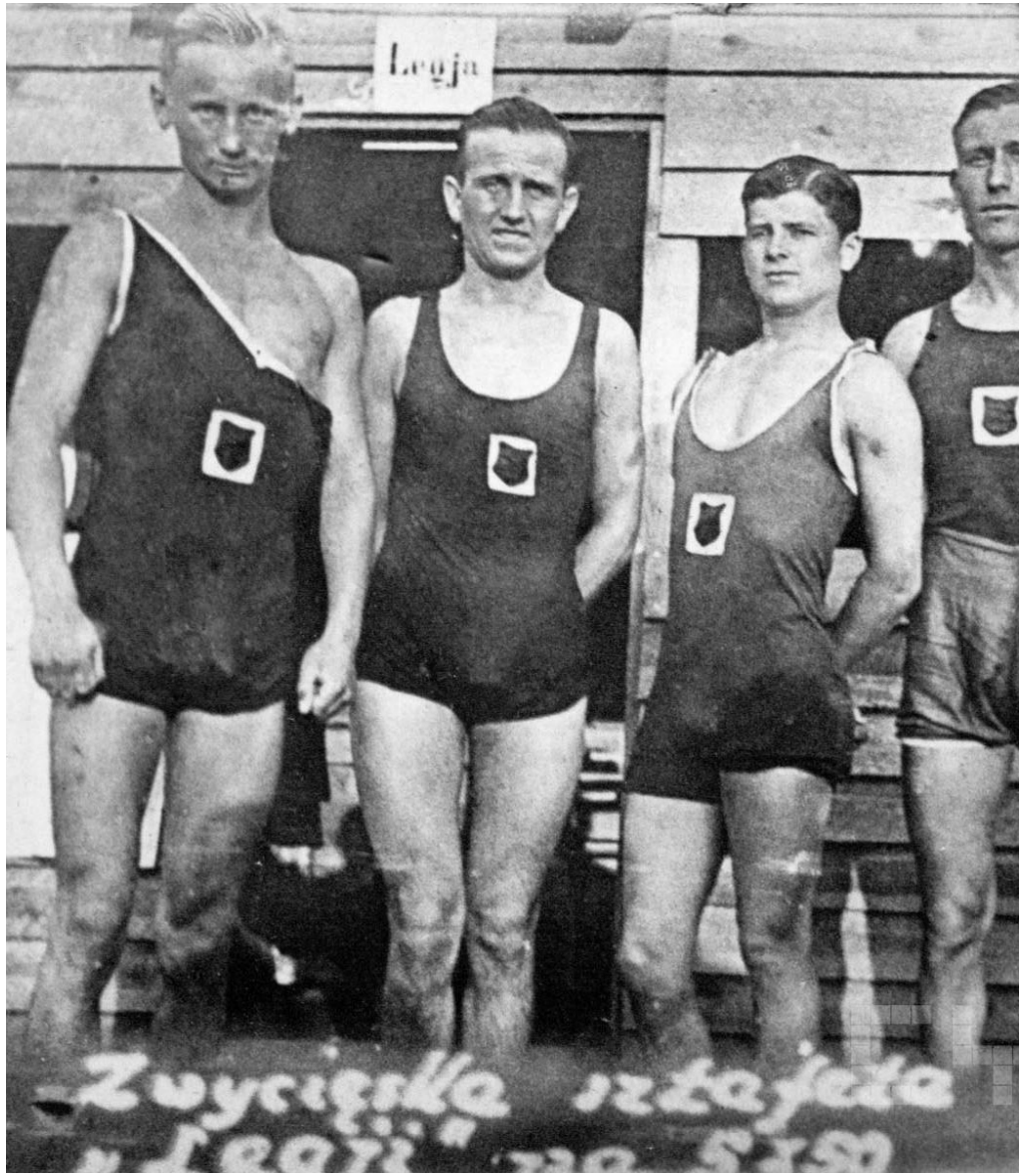
3. Do not terrorize with how-amazing-Kubernetes-is :D

4. Hide complexity



Keep everybody in the process

5. Amplify the feedback



Conventions over tools!

- Common conventions for repos
- No a single deploying tool
- No encrypted data in repo

ps. Only when you are really really ready.

Keep everybody in the process

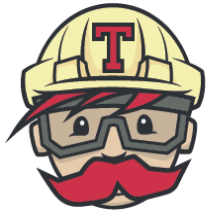
Copy & Paste:

1. Makefile
2. Kubernetes files
3. TravisCI

```
curl https://github.com/smacc-ci/deploy.sh | bash
```

Friendly CD

1. make run_local
2. Code on master → develop env

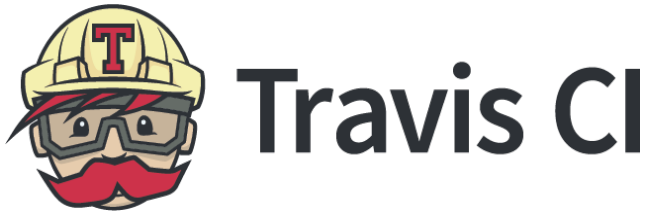


Travis CI

Development

Friendly CD

3. Git tag → staging env
4. PR accepted on hy-platform.git
   → production env




Staging


Production

How we got there?

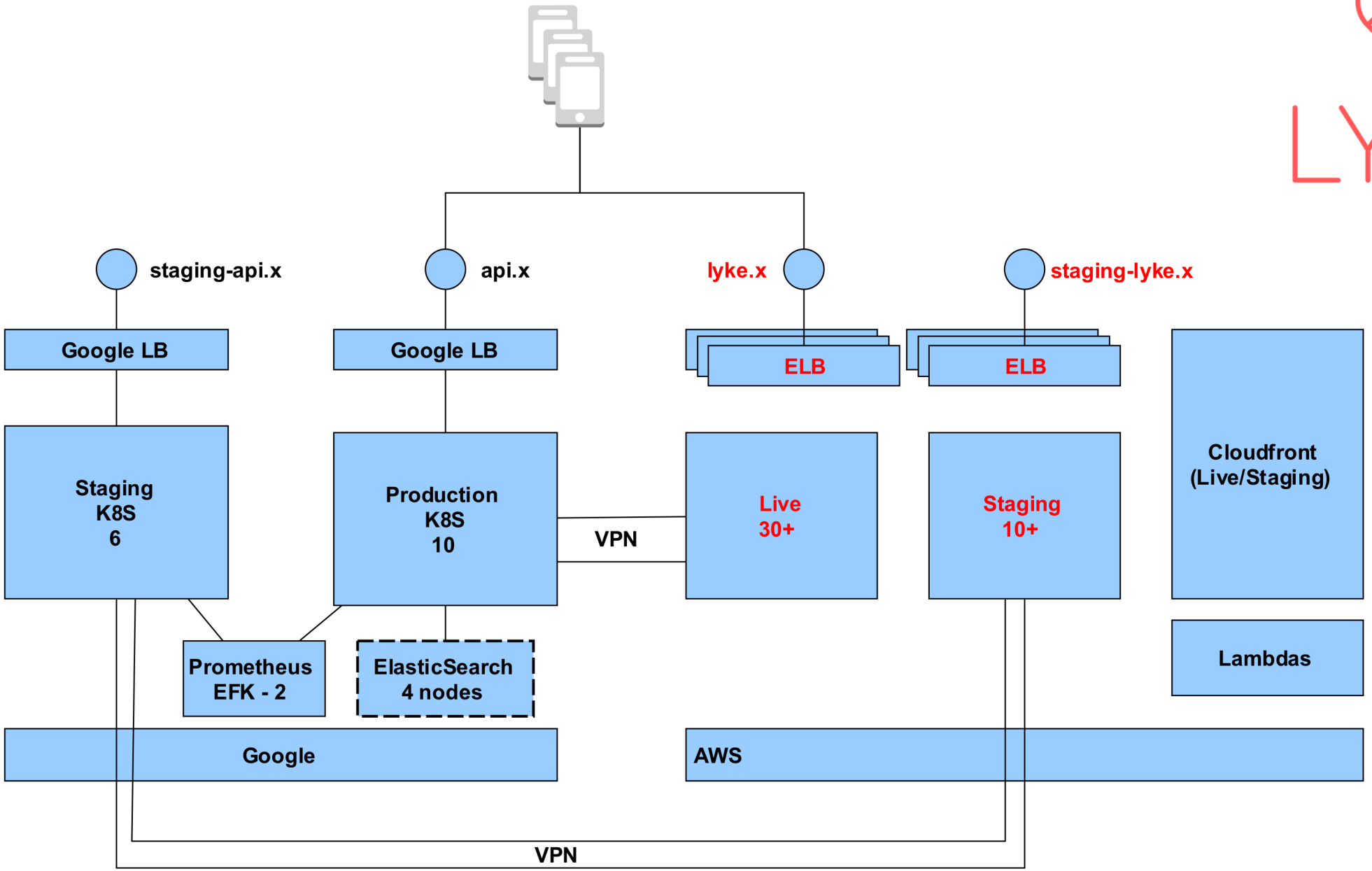
https://www.flickr.com/photos/bruno_brujah/

- Every component deployed independently to k8s
- No platform.git
- C&P Makefile, TravisCI

- **Kubernetes training**
- System Engineer (me) next to devs
- Makefile with a lot of k8s commands

GIT REPO

```
|- tools
|     |- kube-service.yaml
|     \- kube-deployment.yaml
|
|- Dockerfile
|- VERSION
\- Makefile
```


staging-api.x
api.x
lyke.x
staging-lyke.x
Google LB
Google LB
ELB
ELB
Staging
K8S
6
Production
K8S
10
VPN
Live
30+
Staging
10+
Cloudfront
(Live/Staging)
Prometheus
EFK - 2
ElasticSearch
4 nodes
Lambdas
Google
AWS
VPN

- Two teams
  in Warsaw and Berlin
- Machine Learning components

Hypatos

SMACC

- Two teams
  in Warsaw and Berlin
- Machine Learning components

Hypatos

SMACC

Keep it
simple!

Makefile

```
|- tools
|     |- Makefile
|     |- kube-service.yaml
|     \- kube-deployment.yaml
|
|- Dockerfile
\- Makefile
```

Makefile only tasks for dev

Makefile

```
SERVICE_NAME=v-connector
DOCKER_REGISTRY=eu.gcr.io

test: test_short test_integration

run_local:

release: docker_build docker_push

kube_create_config:

deploy:
```

Copy&Paste from the project to project

Repo .travis.yml

```
language: go
go:
- '1.16'
services:
  - docker
install:
  - curl -sL https://${GITHUB_TOKEN}@raw.githubusercontent.com
  - if [ -f "tools/travis/install.sh" ]; then bash tools/travi
script:
  - dep ensure
  - make lint
  - make test
  - if [ -z "${TRAVIS_TAG}" ]; then make snapshot; fi;
deploy:
```

Keep it simple

- Plain kubernetes files
- Next step: kustomize
- Do not use Helm
- We avoid operators

```
apiVersion: v1
kind: Service
metadata:
  name: api-status
spec:
  ports:
  - port: 80
    protocol: TCP
  selector:
    name: api-status-nginx
---
apiVersion: apps/v1beta1
kind: Deployment
metadata:
  name: api-status-nginx
spec:
  replicas: 1
  template:
```

Our stack

On Prem

Observability

Ingress

- Simple to configure
- Integrations out-of-the-box, e.g., let’s encrypt

Keeping an eye on

Kubeflow

Workflows

Summary

https://www.flickr.com/photos/bruno_brujah/

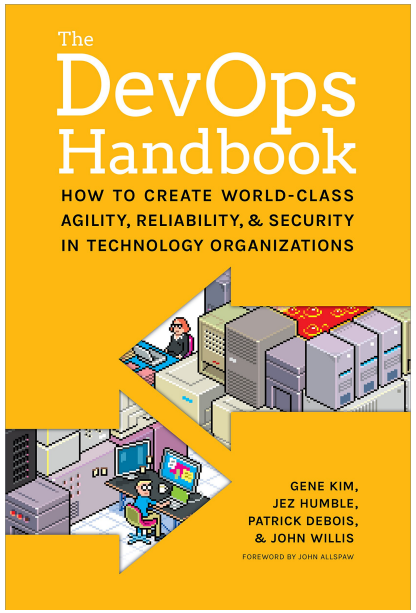
- Early wins
- Train your team members
- Keep it simple
- Deliver
  and patch, patch!

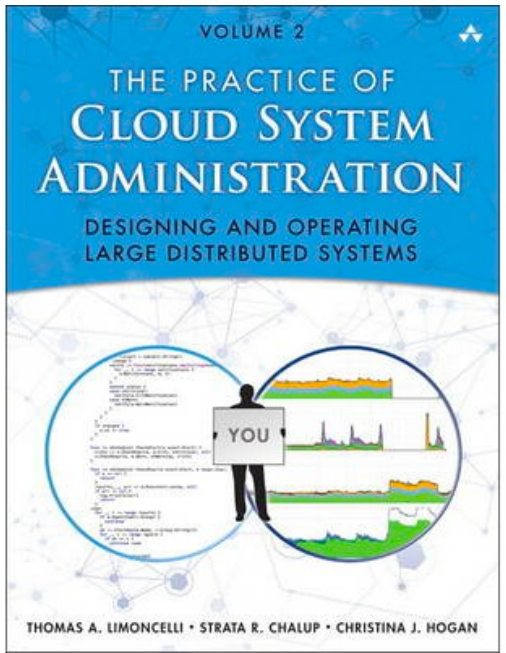
Thank you. Questions?

https://www.flickr.com/photos/bruno_brujah/

Inspiratoins

- Heroku Values (https://gist.github.com/adamwiggins/5687294)
- 12factorapps (https://12factor.net/)





Backup slides

https://www.flickr.com/photos/bruno_brujah/

Zmiana

- Twoja praca jest 150% gotowa
- Ship it!
- Powoli rozszerzaj wsparcie, np., pomagaj
- Relacje
- Wybierz proces przynoszący największą wartość w firmie

https://gist.github.com/adamwiggins/5687294

Zmiana

- Wszyscy mają pomysły, więc Make it real*
- Ship it
- Szukaj **Early wins**

Wsparcie szefa często skraca albo wydłuża drogę.

https://gist.github.com/adamwiggins/5687294

One more thing

- Nie narzekaj, że X nie działa
- Przyjdź z pomysłem rozwiązania. Make it Real*.
- Let’s together ship it!

One more thing

https://www.flickr.com/photos/bruno_brujah/

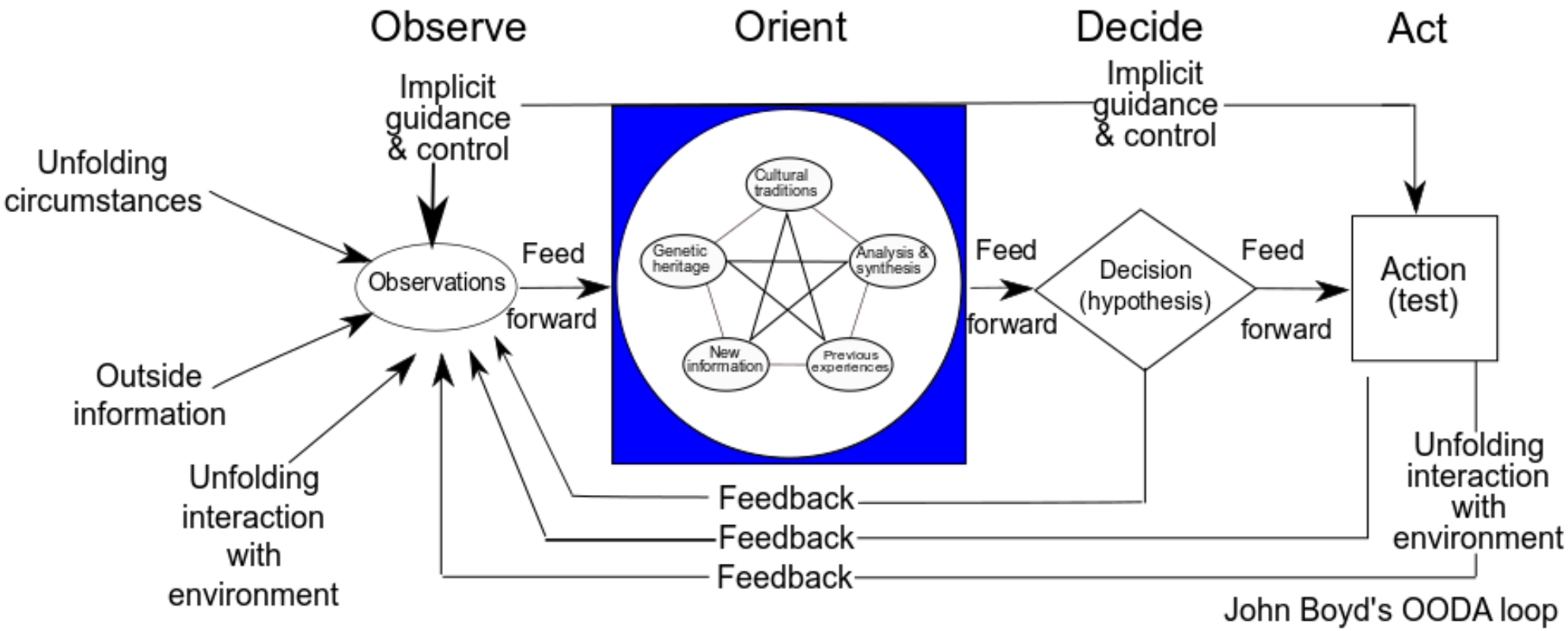
- Strong opinions weekly hold
- Zakładaj najlepsze intencje
- Nie zakładaj, że ktoś sobie nie poradzi

One more thing

https://www.flickr.com/photos/bruno_brujah/

- Murdering the unchosen alternative

OODA



https://upload.wikimedia.org/wikipedia/commons/3/3a/OODA.Boyd.svg

Cooper color code

- White
- Yellow
- Red

https://en.wikipedia.org/wiki/Jeff_Cooper