Swimming upstream in the container revolution

Containerless Continuous Delivery

Bert Jan Schrijver

bertjan@jpoint.nl

@bjschrijver

Let's meet

Bert Jan Schrijver

● jpoint.

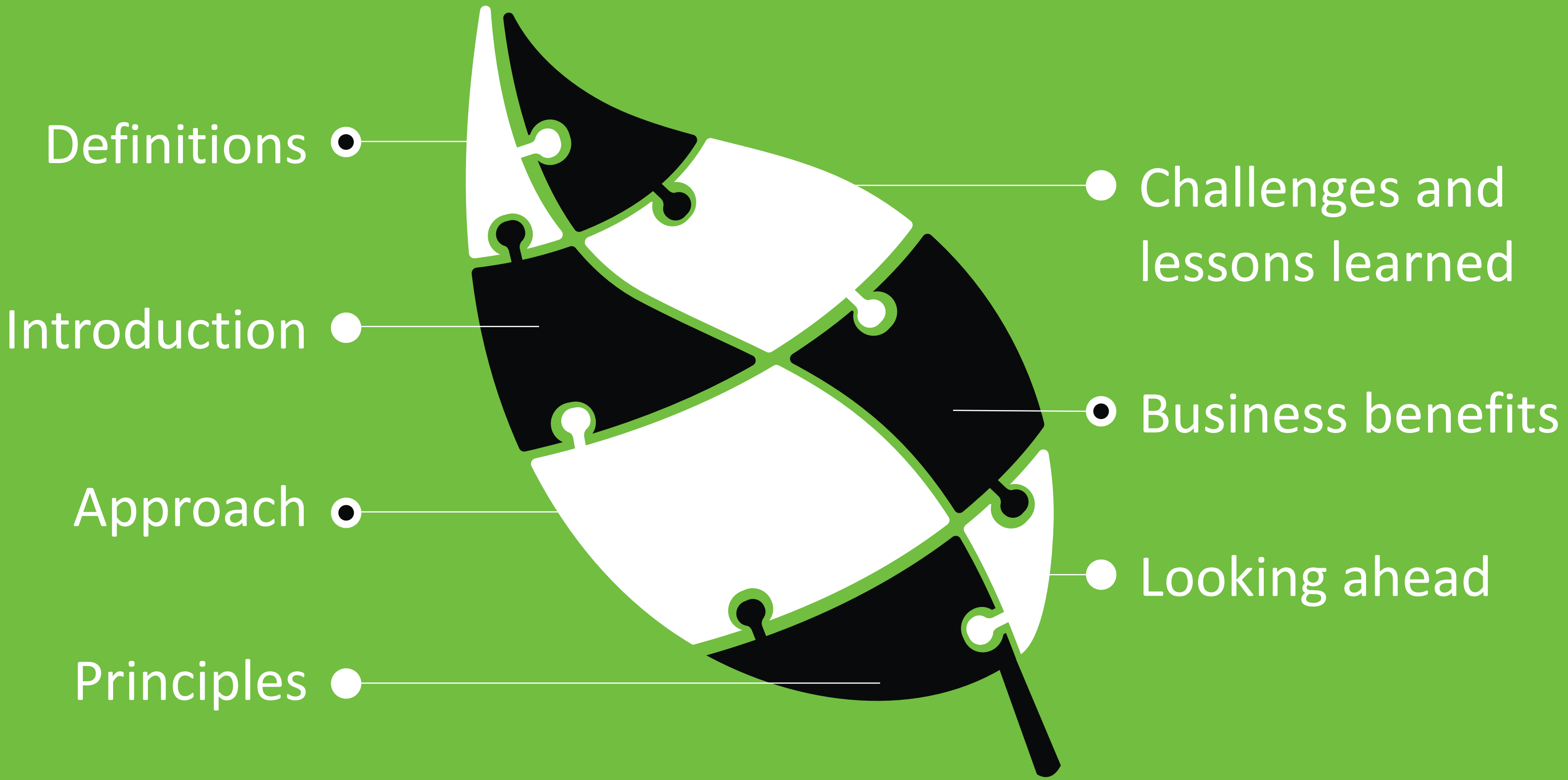
● MALMBERG a Sanoma company

● nljug

@bjschrijver

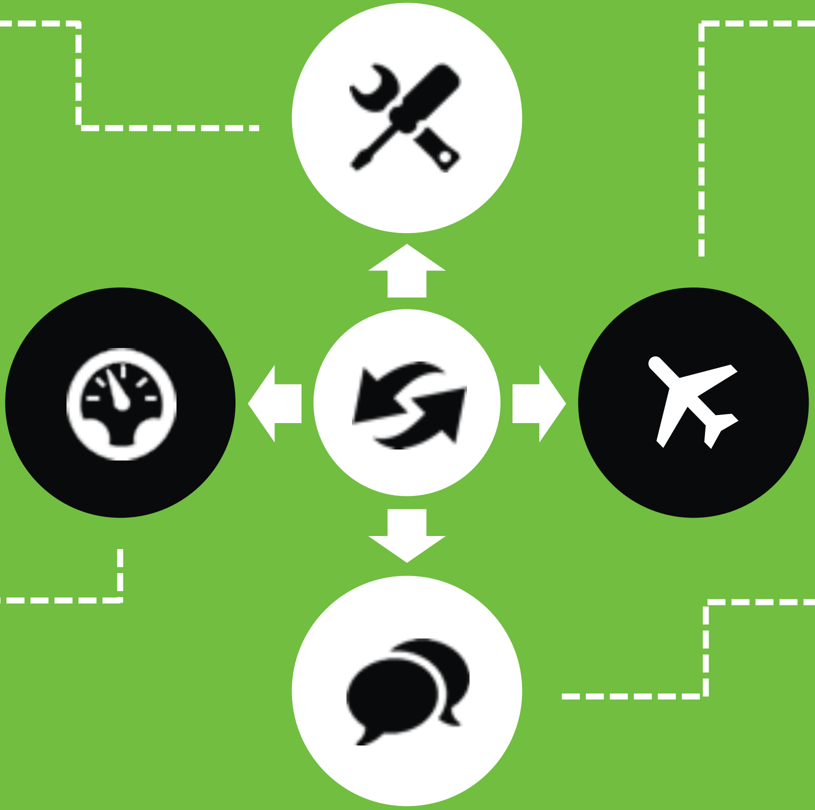
S o w h a t ' s n e x t ?

Outline



Definitions

Who's who in DevOps



Continuous Integration

Team members integrate their work frequently. Commits are verified by automated builds and tests.

Continuous Delivery

Building and testing software in such a way that the software can be released to production at any time.

"Ship early, ship often, sacrificing features, never quality" - Kyle Neath

Continuous Deployment

Every change goes through the build/test pipeline and automatically gets put into production.

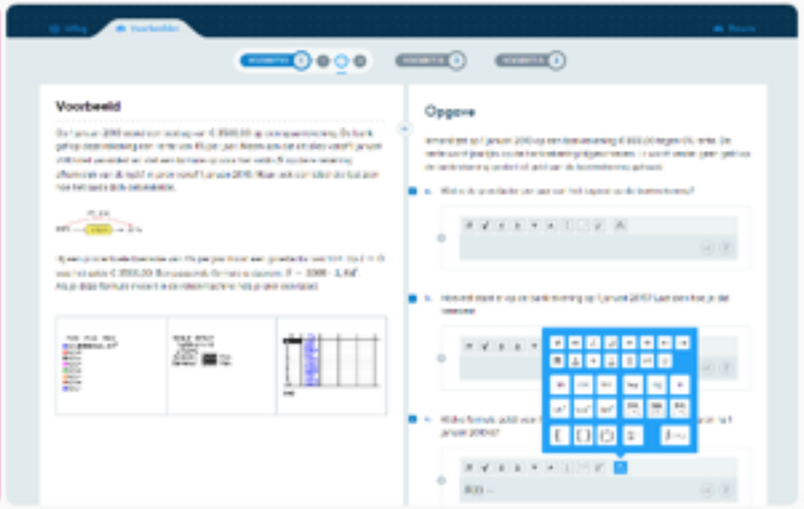
DevOps

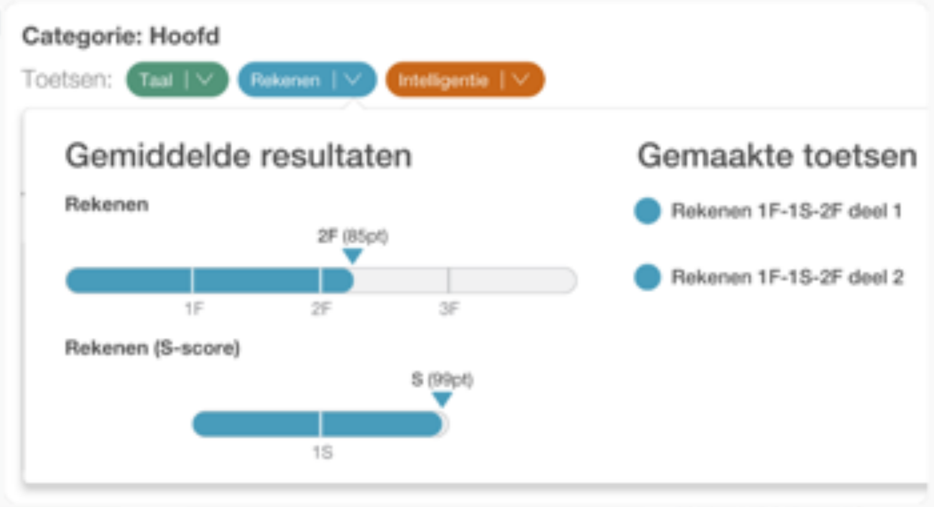
Development and operations engineers participate together in the entire product lifecycle - and are responsible together for the product.

About Malmberg

Malmberg is an educational publisher in the Netherlands. Malmberg is building modern, rich and scalable e-learning applications using Java 8, Vert.x, AngularJS and MongoDB, running on Amazon cloud services.

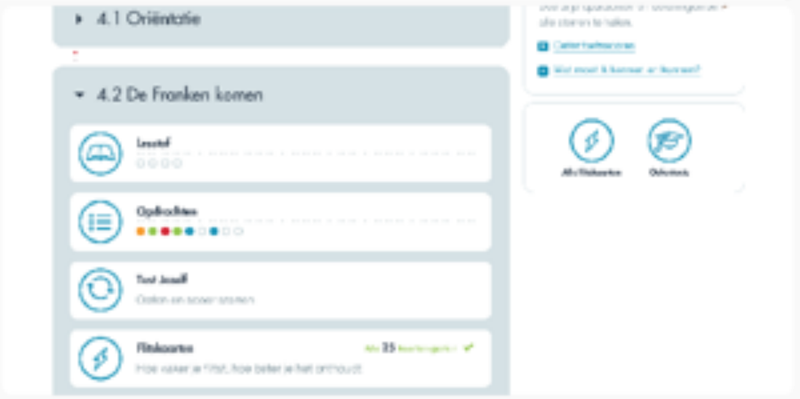
raspberry
Present simple in questions and negations
Congratulations!
You're a hero!
Collect flavour

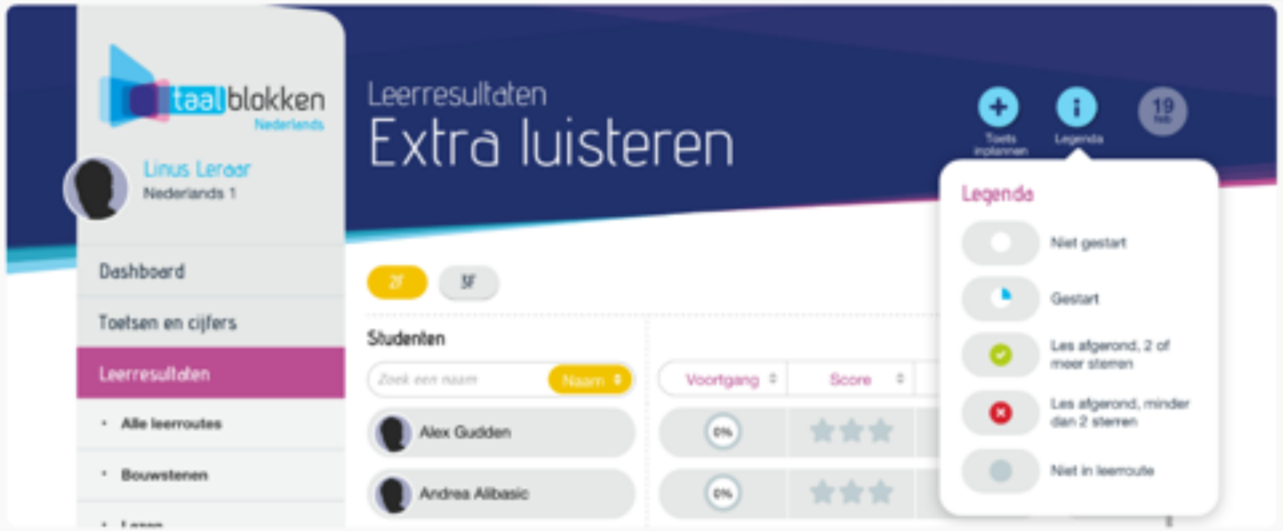

Voorbeeld
Opgave


Categorie: Hoofd
Toetsen:
Taal
Rekenen
Intelligentie
Gemiddelde resultaten
Rekenen
2F (85pt)
1F
2F
3F
Rekenen (S-score)
S (99pt)
1S
Gemaakte toetsen
Rekenen 1F-1S-2F deel 1
Rekenen 1F-1S-2F deel 2

Opgave
Verbind de juiste woorden met elkaar elkaar.
Boeren
Adel
Geestelijken
Bidden voor het volk
Land bewerken
Het volk beschermen
Bekijk jouw antwoord

bvj
Organen en cellen
1
2
3
4
5
6
7
THEMA 3
Organen en cellen


4.1 Oriëntatie
4.2 De Franken komen
Lesstof
Opdrachten
Test Jezelf
Oefen en scoor sterren
Flitskaarten
Alle 35
Hoe vaker je flitst, hoe beter je het onthoudt
Alle Flitskaarten
Oefentoets


taalblokken
Nederlands
Linus Leraar
Nederlands 1
Dashboard
Toetsen en cijfers
Leerresultaten
Alle leerroutes
Bouwstenen
Lezen
Leerresultaten
Extra luisteren
Toets inplannen
Legenda
19
2F
3F
Studenten
Zoek een naam
Naam
Voortgang
Score
Alex Gudden
Andrea Alibasic
0%
0%
Legenda
Niet gestart
Gestart
Les afgerond, 2 of meer sterren
Les afgerond, minder dan 2 sterren
Niet in leerroute

Collect these flavours
Before Monday, 3rd August 2015
Check out my icecream!
tropical sorbet
Yes/no questions requiring 'do'/'does'
Practise
questions
Collected
grapefruit
Present continuous: things that are happening now
Practise
the present continuous
mango
Present simple vs. present continuous (1)
Practise
the present simple

History

About a year ago

Modern development culture

Modern tools, lots of automation. Test environments are managed by developers.

Differences lead to issues

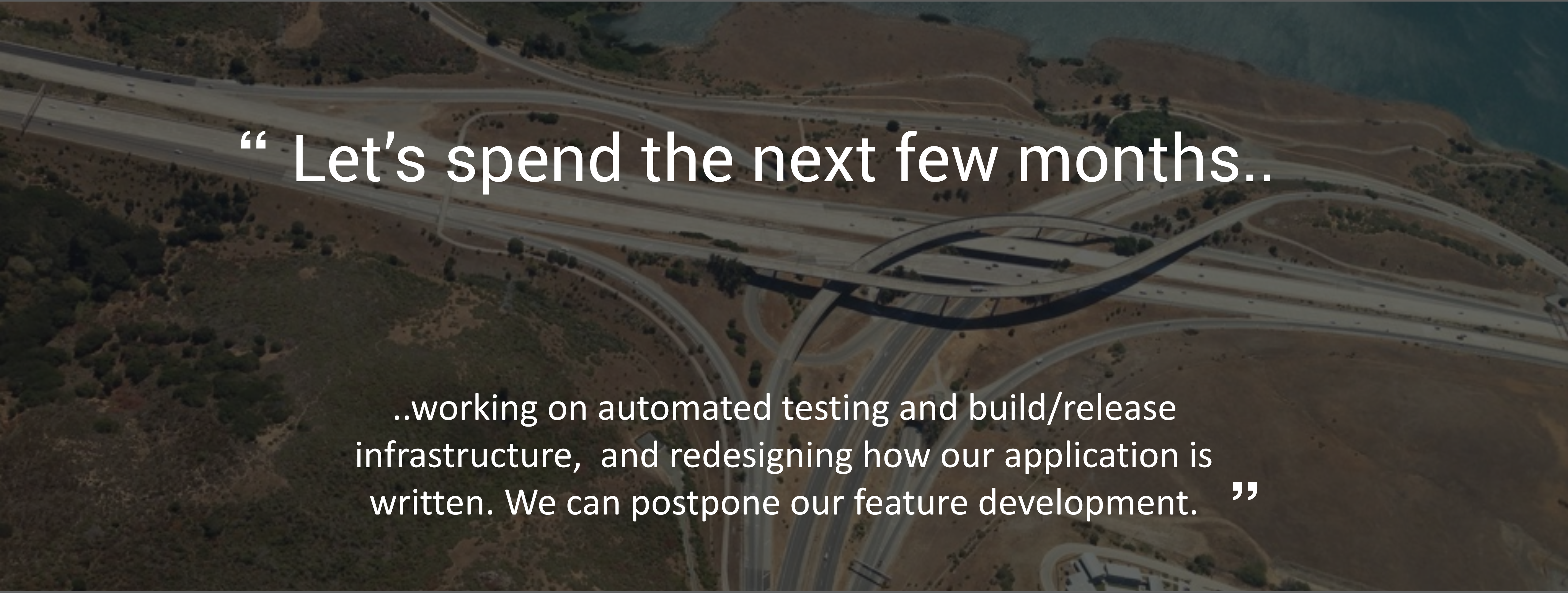
Communication between development and operations was slow, problem analysis in production was difficult and releases took a lot of time.

Traditional operations

Production environments managed by external operations partner. Differences in infrastructure between development and production.

Things needed to change

Issues and differences between development and operations were slowing us down. We needed to shift strategies to keep progressing.



“ Let’s spend the next few months..

..working on automated testing and build/release infrastructure, and redesigning how our application is written. We can postpone our feature development. ”

…said no product manager ever.

J. Paul Reed

Approach

How we initiated change.

Expert team

Build a dedicated team of Devs, Ops and Cloud experts.

Keep the shop open

Build a complete new setup to allow development teams to transform to the new situation at their own pace.

Define principles

Define key points that identify your approach and help you set goals.

Principles

Master branch is always releasable

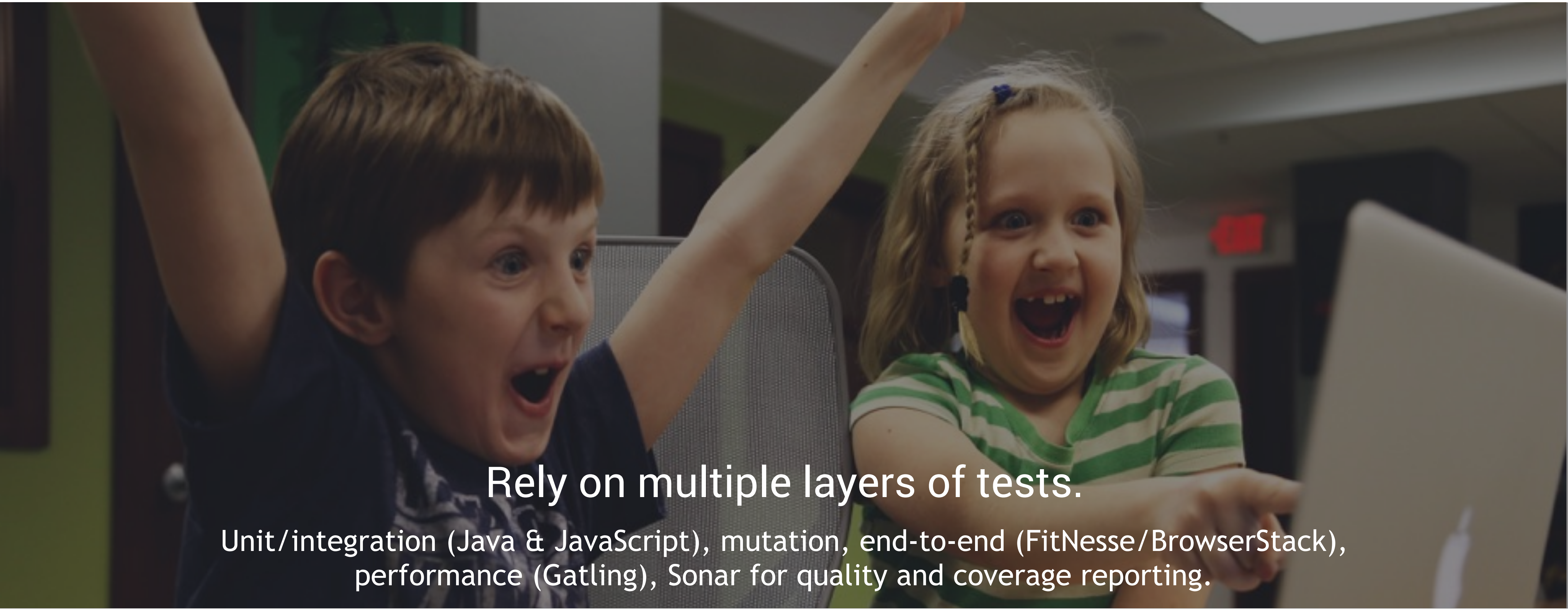
Principle 1

Don't merge it until it's *done.*

Every change is developed and tested in a feature branch.

Each commit is tested extensively

Principle 2

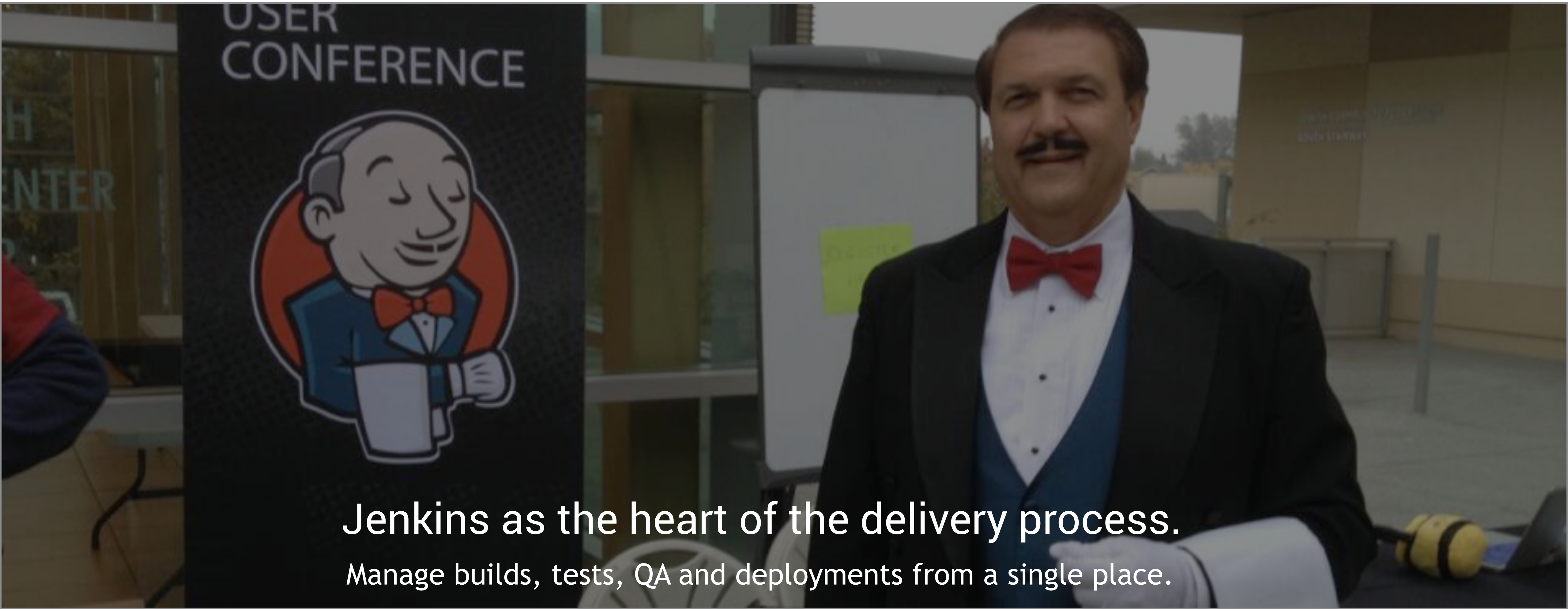


Rely on multiple layers of tests.

Unit/integration (Java & JavaScript), mutation, end-to-end (FitNesse/BrowserStack), performance (Gatling), Sonar for quality and coverage reporting.

Every delivery step is a Jenkins job

Principle 3



Jenkins as the heart of the delivery process.

Manage builds, tests, QA and deployments from a single place.

Deployments are roll-forward only

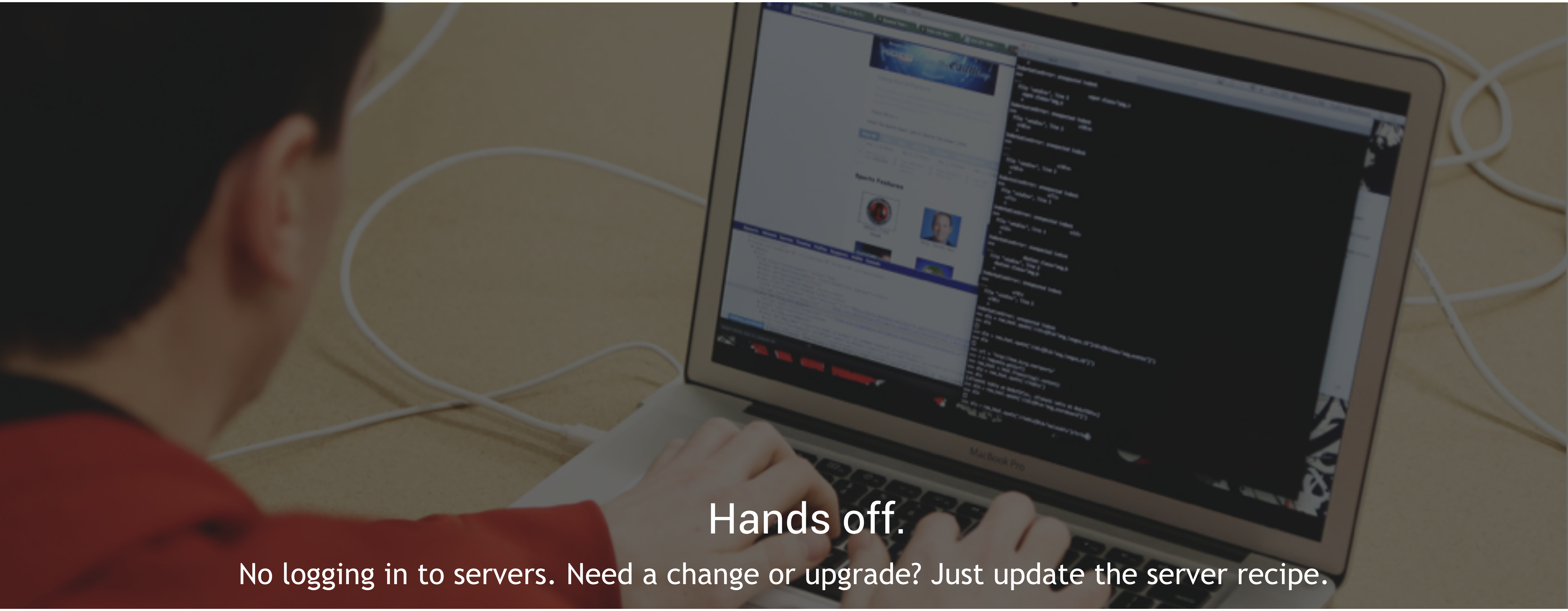
Principle 4

Keep moving ahead.

After deploying 6 new features, when one has an issue, why roll back 5 good features?
Don’t. Just roll out a fix quickly.

Infrastructure as code - for everything

Principle 5

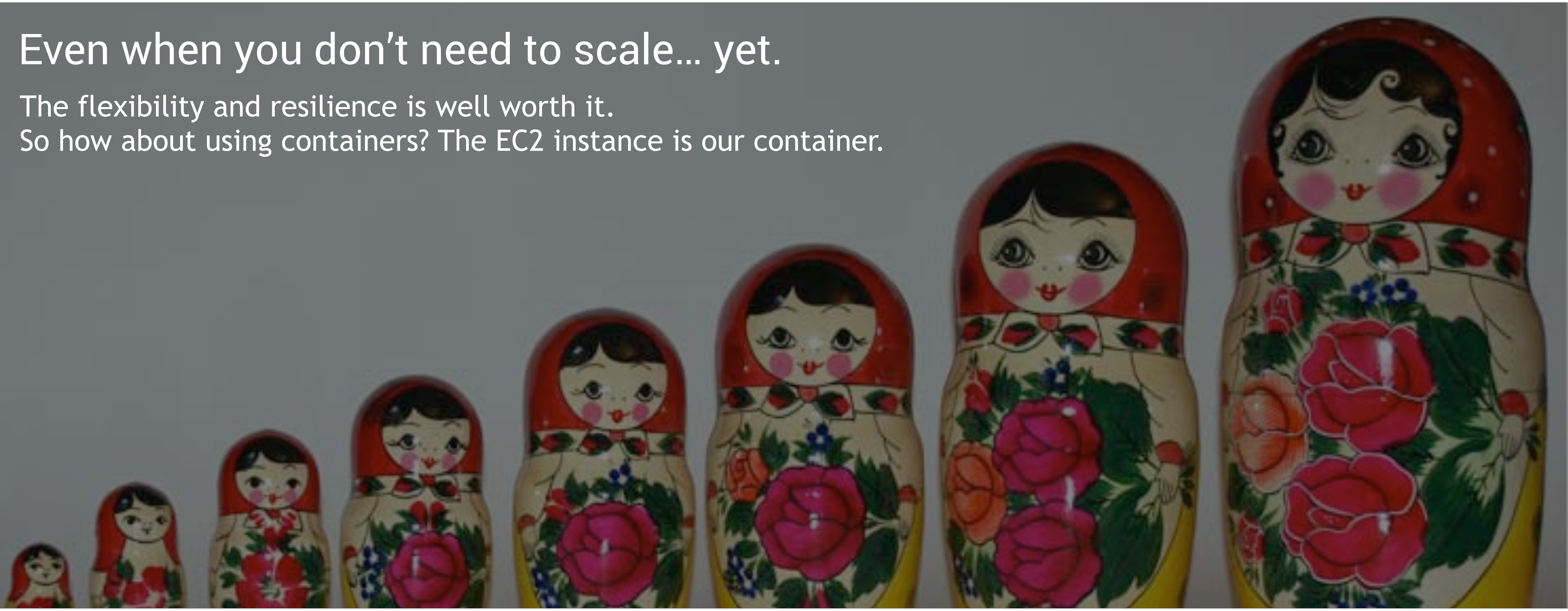


Hands off.

No logging in to servers. Need a change or upgrade? Just update the server recipe.

Put everything in auto scaling groups

Principle 6



Even when you don't need to scale… yet.

The flexibility and resilience is well worth it.
So how about using containers? The EC2 instance is our container.

No downtime in production

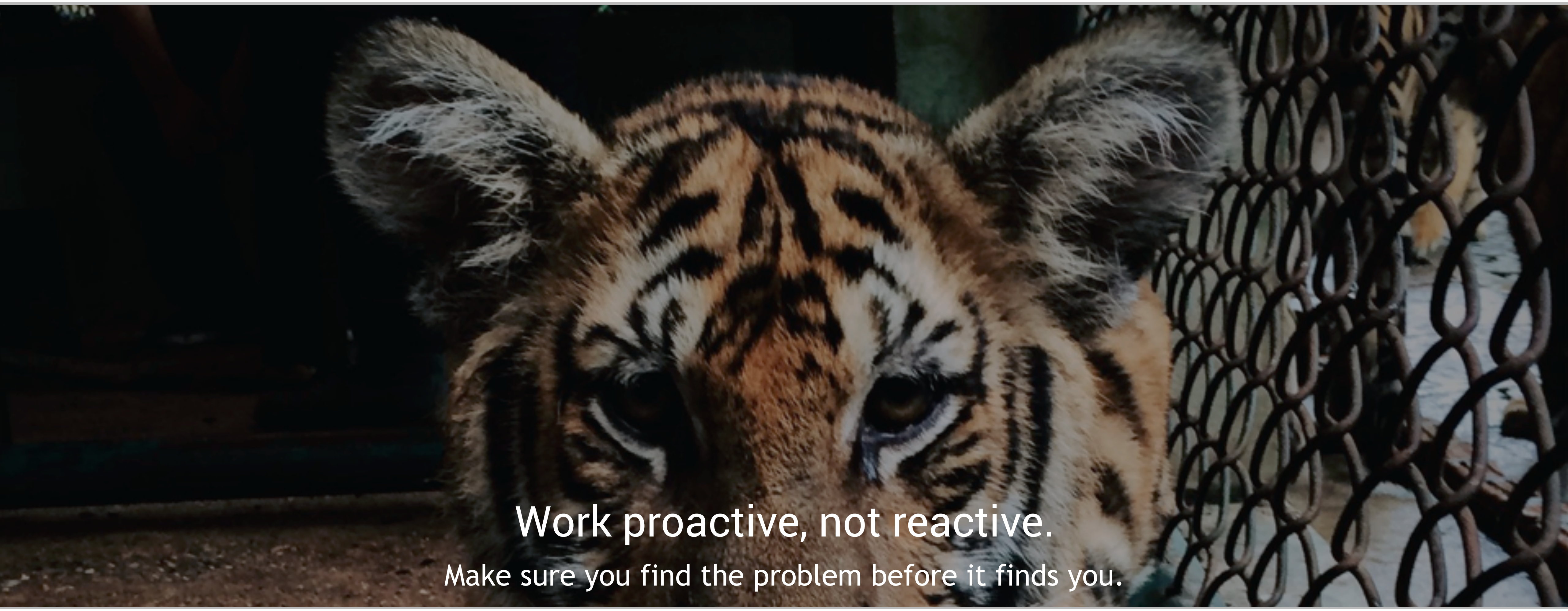
Principle 7

Our end users are the Facebook generation.

You can’t explain maintenance windows to modern end users anymore.

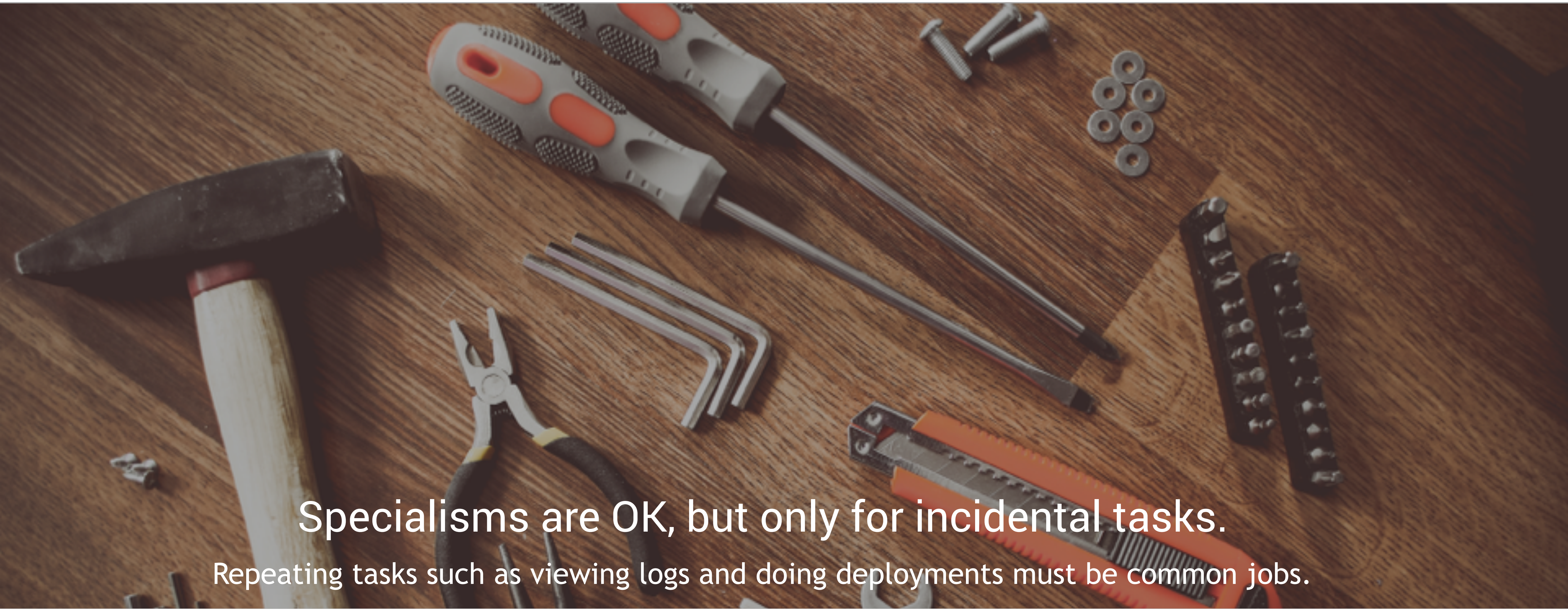
Eyes and ears in production

Principle 8


Work proactive, not reactive.
Make sure you find the problem before it finds you.

Repeating tasks are executable for all team members

Principle 9



Specialisms are OK, but only for incidental tasks.

Repeating tasks such as viewing logs and doing deployments must be common jobs.

DevOps teams work on a self service basis

Principle 10

Give teams the freedom to work in a way that works for them.

Differences between teams are OK. A team that's dependent on external help is not.

Challenges

and lessons learned

01
02
03
04
05
06

Amazon has limits

When you automate everything and keep growing, chances are you're going to hit limits.

Resistance

Don't assume that cultural change won't be an issue. It will.

Communication is key

When transforming an organisation, you need to be really clear about *where* things are going, *why* things are happening and *when* this will impact teams.

How to test Puppet changes

All environments are provisioned automatically. Challenge: how to prevent testing directly in production.

Don't depend on availability of Ops experts

This kills team progress.

Devs need to step up their game

Not all developers are comfortable with managing infrastructure and middleware.

Business benefits

How to sell this to your boss.

Availability
Auto scaling and pro-active monitoring boost availability. A lot.

Continuity
Automated provisioning makes sure that every environment can be re-built from scratch in minutes.

Agility
High level of automation results in shorter release cycles and faster time to market.

Cost reduction
Lower operations costs due to scheduling and scaling. Lower maintenance costs due to high degree of automation.

Better reaction speed
Faster problem analysis and solution.

Looking ahead

Stuff we're still working on

01 Better monitoring and dashboards

Get the teams the information they need, readily available on a dashboard visible from their desks.

02 Continuous performance testing

Daily performance runs on test environments and continuous end-user performance monitoring in production.

03 Continuous security testing

There is no silver bullet here, but useful tools and practices do exist.

04 Automated resilience testing

The only way to be really prepared for failure is to make sure that things will fail by making it fail yourself.

Questions?

@bjschrijver

Thanks for your time.

Liked it? Tweet it!

@bjschrijver