

DevOps

05.05.2020

**EL CAMINO DE LA ENTREGA DE
VALOR EN SOFTWARE**



Luis Fraile

CONSULTOR ALM-DEVOPS

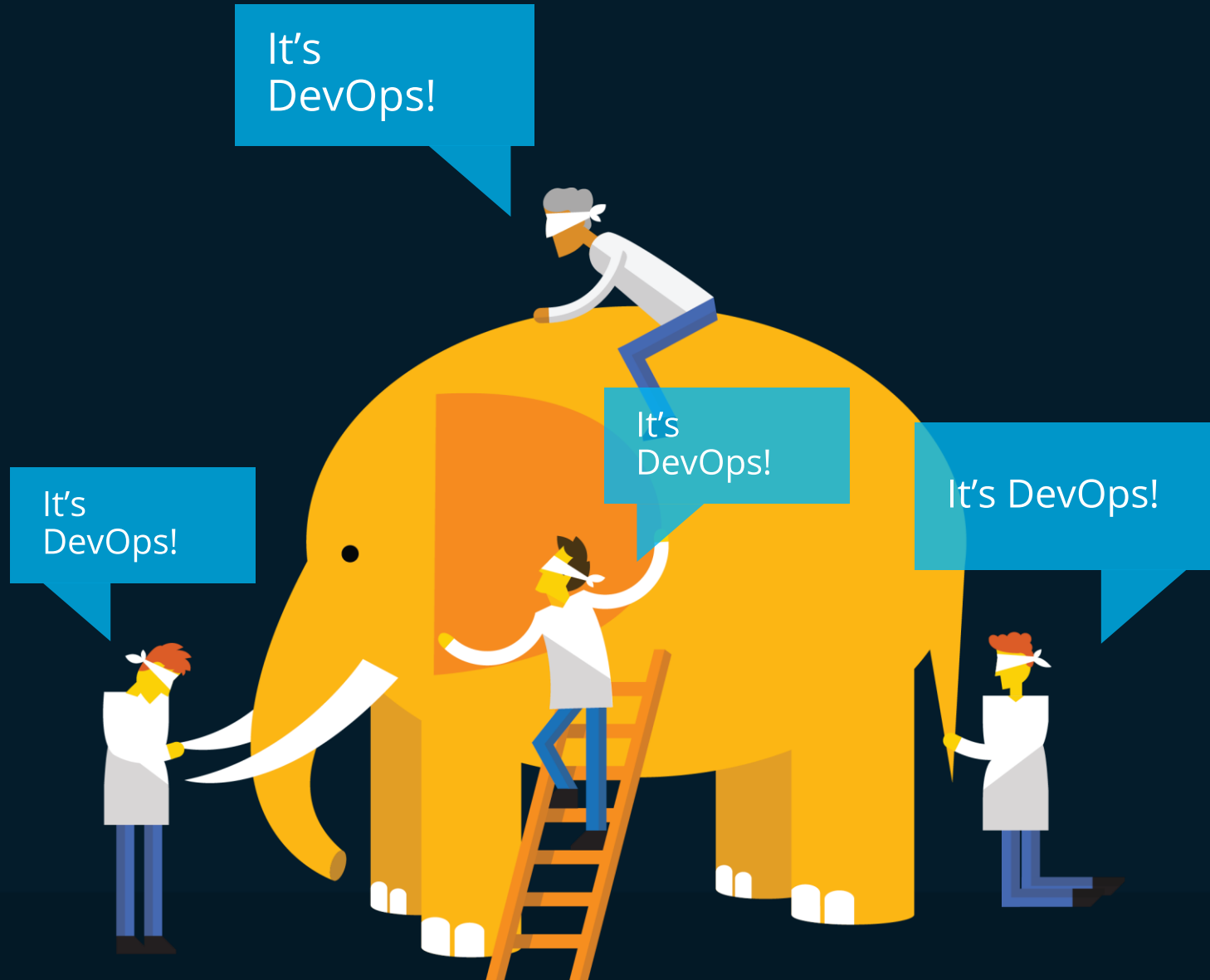
Digamos que ayudo a equipos de desarrollo de software a poder hacerlo mejor mediante DevOps y ... oh ... Ah ... *practices ágiles*

@lfraile

“DevOps is development and operations **collaboration**”

“DevOps is using **automation**”

“DevOps is **small** deployments”



“DevOps is treating your infrastructure as code”

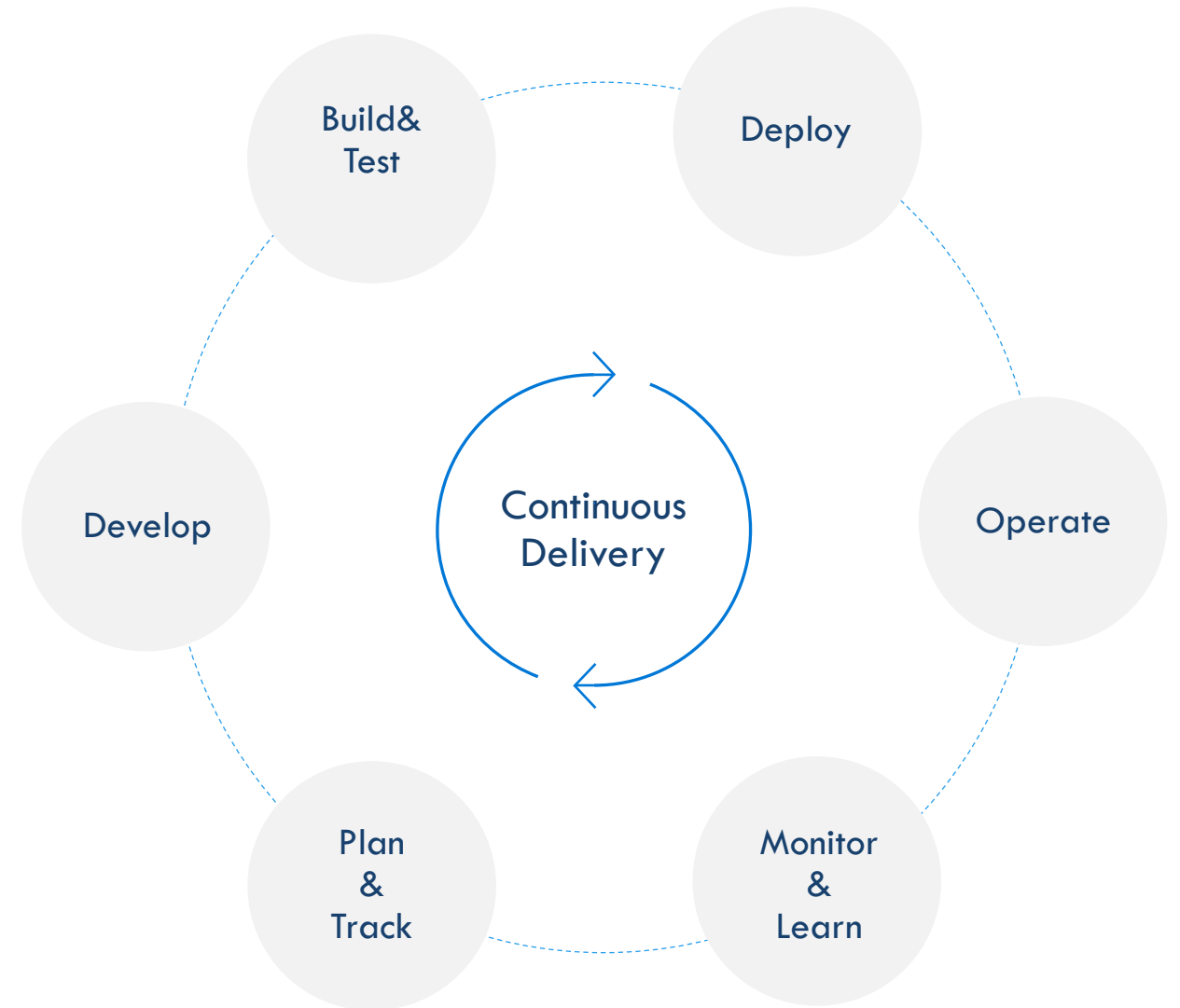
“DevOps is feature switches”

“Scrum for Ops?”



DevOps is the union of **people**, **process**, and **products** to enable continuous delivery of value to your end users. ”

Donovan Brown



Pero ¿por qué hacemos DevOps?

Agilidad (¿ágile?)

Validar pronto y continuadamente

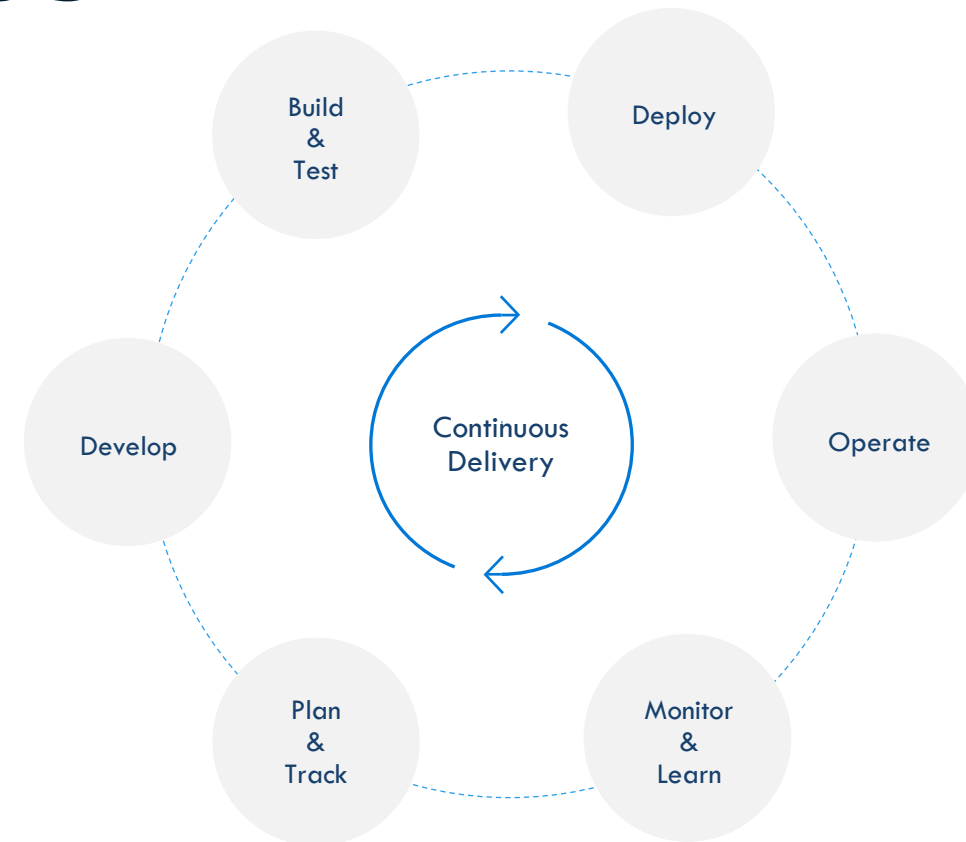
Rapidez en la entrega

Y en la respuesta a desastres

Aseguramiento de calidad

Entregar más no significa entregar cualquier cosa

... simplemente vivir mejor desarrollando software.



Estrategias de despliegue entrega

Unas cuantas

Blue/Green

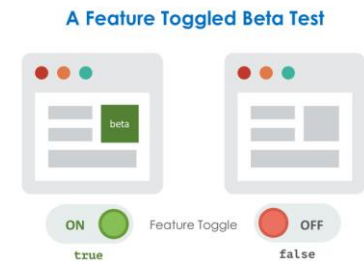
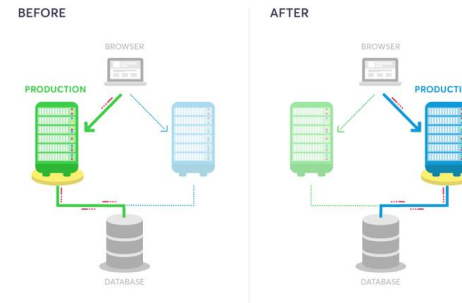
Toggles

Rolling updates

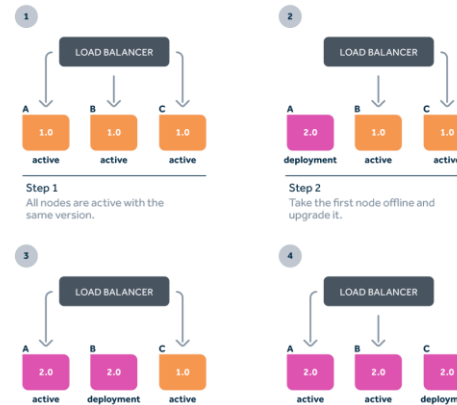
Por entornos ...

O todas a la vez

Pero todas comparten algo



Rolling Updates



Empieza todo en el desarrollo

Hay que cambiar la mentalidad de desplegar a **entregar**

Necesitamos cambiar el modo de pensar en el desarrollo

No es sólo escribir *código limpio*

Definir estrategia de entrega, junto a negocio, junto a ops, junto a sec

Aspectos a tener en cuenta

Calidad

Criterios de
entrega

Sostenibilidad

Monitorización

Disaster
recovery

Service
Reliability
Engineering

Monitorización

¿Cómo saber cuándo hacemos rollback, desactivamos toggle, etc?

Volvemos a Dev: estrategias de monitorización en el Código

- Herramientas de monitorización (no solo logs)

- “Tracking” de eventos

Y vamos a Ops

- Alertas

- Prevención temprana

Y sobre todo: incluirlas en nuestros flujos de entrega automatizados

Criterios de entrega

¿Sabes cuando y dónde entregar?

No es entregar software sin más

Establecer nuestros criterios de entrega

- Calidad

- Rendimiento

- Aceptación de los usuarios

- Integración en el producto

Reglas de calidad mínima de paso entre entornos

Infraestructura

Forma parte de la *entrega*

Hay que asegurar la consistencia y estabilidad

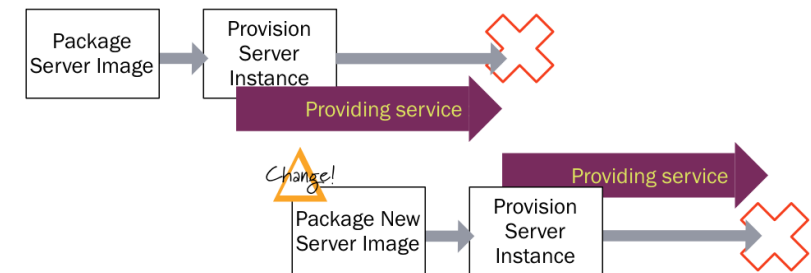
Infraestructura inmutable <https://martinfowler.com/bliki/ImmutableServer.html>

Siempre se destruye lo antiguo y se crea nueva infraestructura

Infrastructure as Code

Terraform, ansible, puppet, ARM, etc.

Se desarrolla y entrega con las versiones



Lo que vamos a ver



1. Flujo completo cd CI/CD (YAML) Azure DevOps
2. Despliegue mediante slots
3. Uso de *canaries* con Azure Web App Slots
4. Feature Flags (Esquío)
5. Flujos de aprobación automática y manual en YAML
6. Flujo de rollout mediante YAML



Talk is cheap
~~show me the~~
~~code~~
~~boxes~~
~~yaml~~

delivery

#20200505.5 Merged PR 52: Flags enabled
on CD-DemosPlainConceptsAzOps

Summary

Manually run by Luis Fraile

Repository and version
DemosPlainConceptsAzOps
master 1a855c5

Time started and elapsed
Just now
47s

Stages Jobs

```
graph LR; B1[Build 0/1 completed 45s] --> DE[DeployEarly Not started]; B1 --> DM[DarkModeForAll Not started]; B1 --> IA[IncrementProduction... Not started]; DE --> GA[GeneralAvailability Not started]; DM --> GA; IA --> GA;
```

Build 0/1 completed 45s
Cancel

DeployEarly Not started

DarkModeForAll Not started

GeneralAvailability Not started

IncrementProduction... Not started



Thank you

[@lfraile](#)

[@plainconcepts](#)

www.plainconcepts.com