
mundipagg

Thiago Barradas

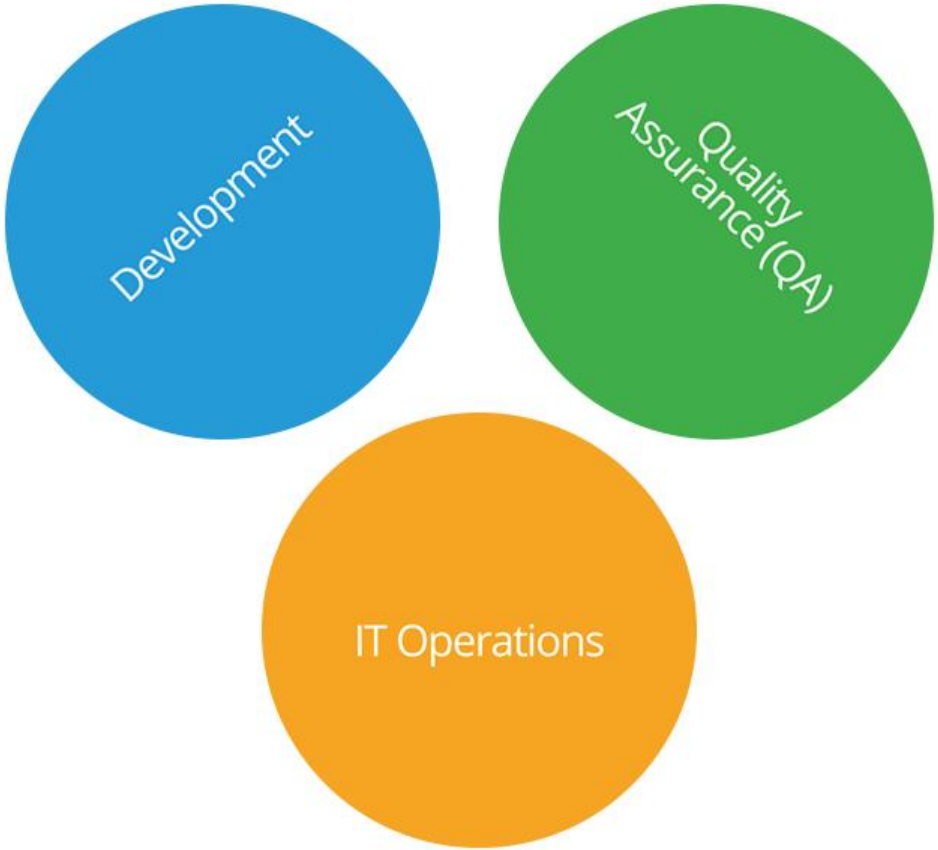
Software Enginner | Mundipagg

[Web Applications] [ASP .NET]
[API RESTful] [Microsoft ♥ Linux]
[Elasticsearch] [Docker]
[DevOps] [Agile]

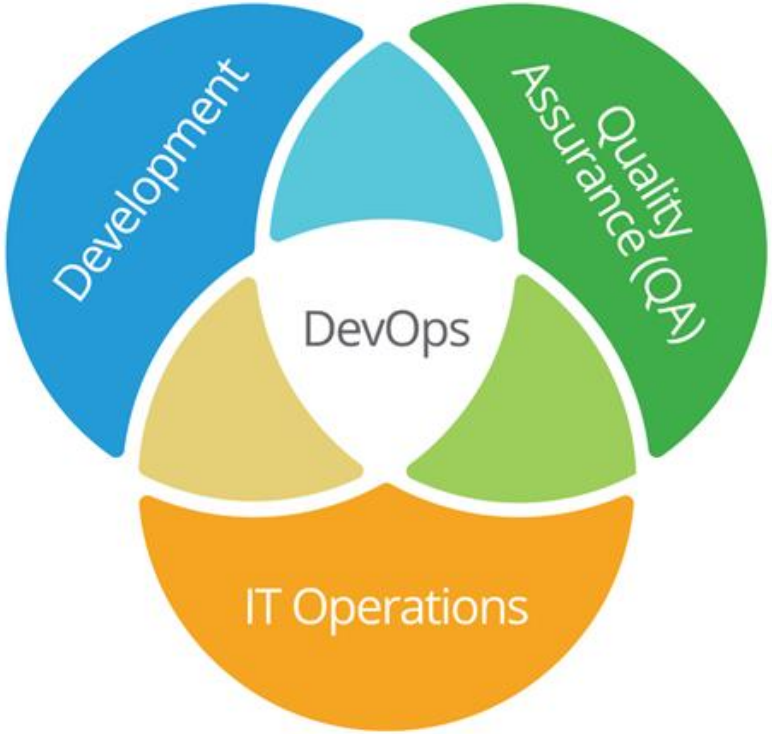
tbarradas@mundipagg.com
LinkedIn: thiagobarradas
(21) 99329-9143

DEVOPS DE UMA VEZ POR TODAS!

DevOps



DevOps



DevOps

"**DevOps** é um termo criado para definir o **conjunto de práticas** que **integram e automatizam** os **processos** entre as equipes de **desenvolvimento**, **operações** e de **apoio** (como QA) para a **produção rápida** e **confiável** de **software**."

DevOps

"O conceito do **DevOps** baseia-se em criar uma **cultura de colaboração entre as equipe** que sempre trabalharam separadas.

DevOps é uma **mudança de mentalidade, uma cultura, um movimento, uma filosofia**"

Framework CALMS

Culture

Automation

Lean

Measurement

Sharing

Framework CALMS

Todas **ferramentas e automações são inúteis** se não forem acompanhadas pela **verdadeira disposição** da área de **desenvolvimento e operações** em **trabalhar juntos.**

Framework CALMS

Porque **DevOps** não resolve problemas de ferramentas.

Resolve problemas humanos.

Framework CALMS

Automação **elimina o trabalho manual** repetitivo,

produz processos repetíveis e cria **sistemas confiáveis**.

Automatizar gera **velocidade na entrega**

e tornam os envolvidos **mais produtivos**.

Framework CALMS

Normalmente, **compilação, teste, implementação e provisionamento** automatizados são **o ponto de partida** típico para equipes que ainda não têm isso implantado.

Framework CALMS

Precisamos focar nas **entregas de valor ao cliente**. Precisamos ser **objetivos e enxutos**. Precisamos conhecer as nossas **limitações** e os **gargalos do processo**. Precisamos ser **Lean**.

Framework CALMS

A mentalidade **DevOps** vê oportunidades de **melhoria contínua** em toda parte. Identificando as **limitações**, podemos **otimizar o fluxo**, entregando **mais velocidade** e **maior eficiência**.

Framework CALMS

DevOps é **cíclico e infinito**. Mensurar e obter **métricas** é o ponto de partida para **novas melhorias**, seja para o **processo de desenvolvimento**, o **software produzido** ou as **regras de negócio**.

Framework CALMS

Além de gerar **conhecimento**, **métricas** criam **previsibilidade** sobre **possíveis incidentes** que possam vir a surgir.

Assim, temos **insumos** suficientes para **analisar falhas** e gerar **melhorias constantemente**.

Framework CALMS

O **compartilhamento de informações**, além de ser **saudável**, auxilia na **descentralização de conhecimento** em pessoas dos times, evitando que os **processos** se tornem **dependentes**.

Framework CALMS

Compartilhar conhecimento ajuda na criação de **times genéricos**, com conhecimentos básicos em diversos **assuntos do negócio** e **tecnologias**.

Assim, o time se torna **autossustentável**.

Framework CALMS

Culture

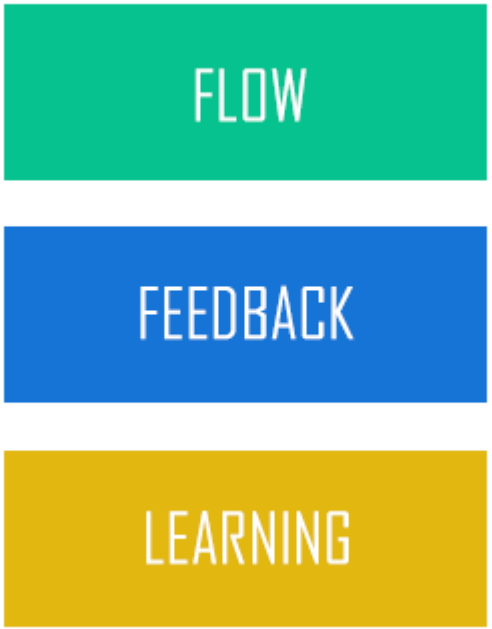
Automation

Lean

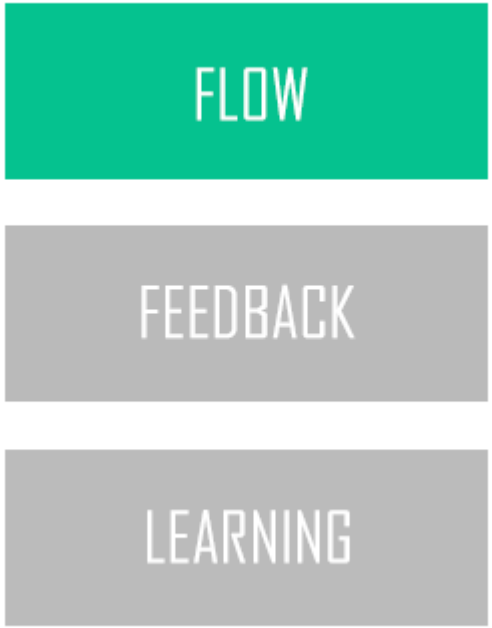
Measurement

Sharing

Os Três Caminhos

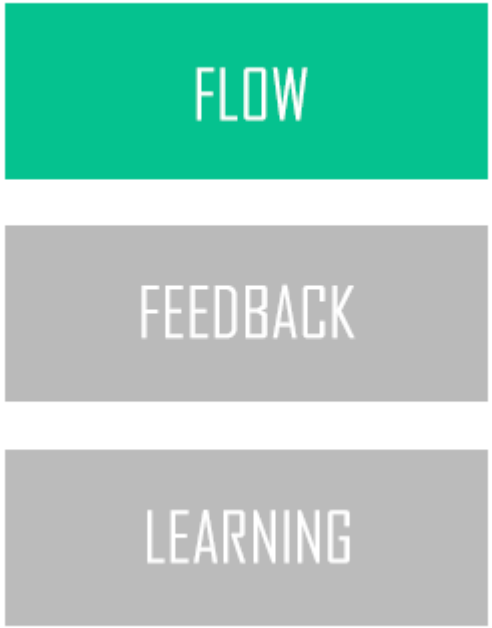


Os Três Caminhos



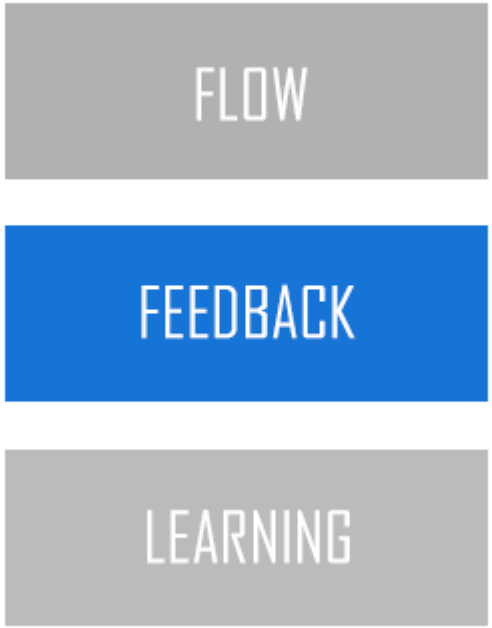
A **otimização do fluxo** visa **eliminar** desperdícios, **gargalos no processo**, transferência de responsabilidades e tempos de espera. Esse **caminho** é trilhado **entre a demanda e a entrega** em produção.

Os Três Caminhos



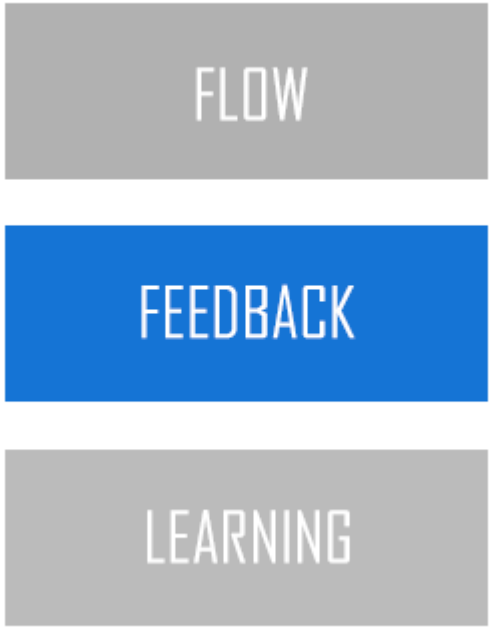
A **chave** para este caminho é a aplicação de **metodologias ágeis** e a **automatização** dos processos **do desenvolvimento à release**, como a **integração contínua** e/ou **entrega contínua**.

Os Três Caminhos



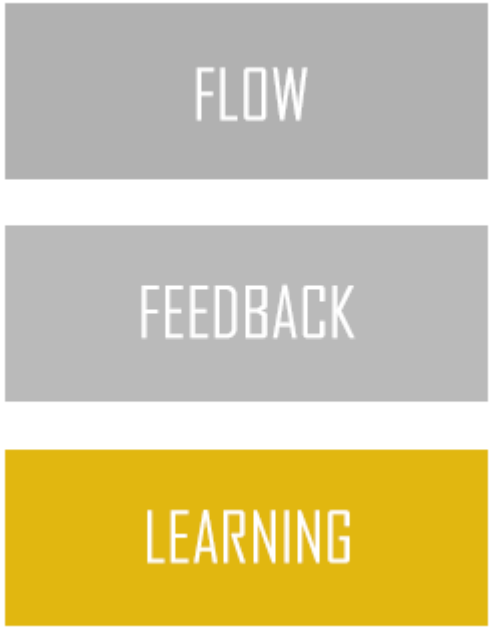
Ciclos rápidos de **feedbacks visam resolver problemas o quanto antes**, testando tudo, **alertando em qualquer falha**, considerando **todas as métricas** coletadas no ambiente produtivo sobre o **valor entregado**.

Os Três Caminhos



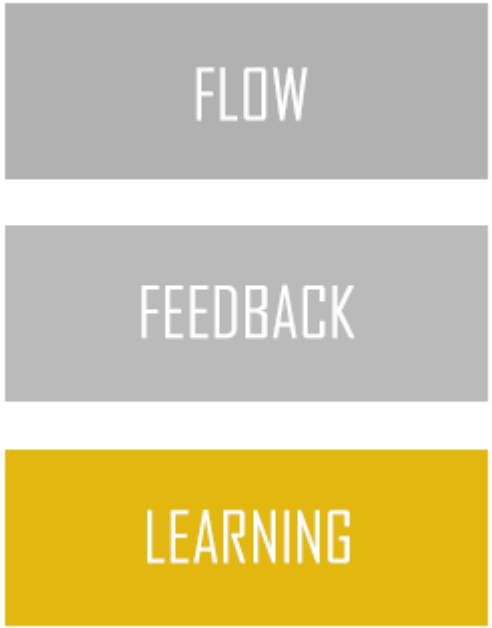
O **monitoramento** é a chave, ajudando a **gerar informações relevantes constantemente.** Com feedbacks rápidos, o **negócio consegue falhar rápido**, e **logo retomar o rumo**, caso necessário.

Os Três Caminhos



O **aprendizado contínuo** visa **gerar conhecimento** através da experimentação. **Hipóteses** são **melhores** do que uma **certeza** imediata. Este caminho é fruto do **processo científico** e produz **segurança psicológica**.

Os Três Caminhos



A chave é o **trabalho dinâmico**, com times realizando **experimentos** em seu trabalho diário para **gerar novas melhorias**.

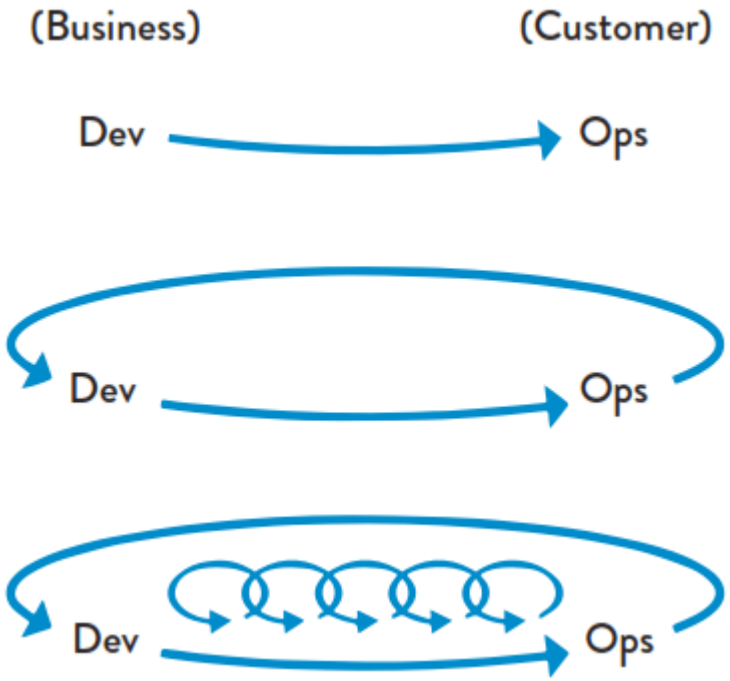
Elimine a **cultura da culpa** e a **aumente** a **colaboração** e o **compartilhamento de conhecimento**.

DIGITAL INNOVATION ONE

FLOW

FEEDBACK

LEARNING



Entregando Software

Microsoft Teams

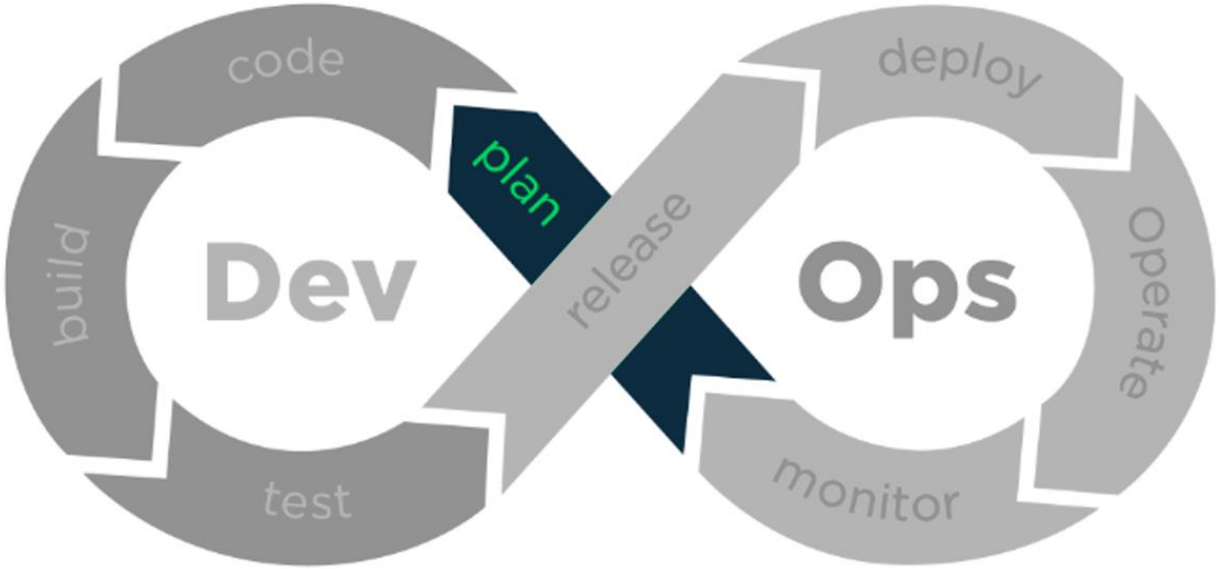
draw.io

Balsamiq

roadmunk

Confluence

Jira



Entregando Software

Visual Studio

GitHub

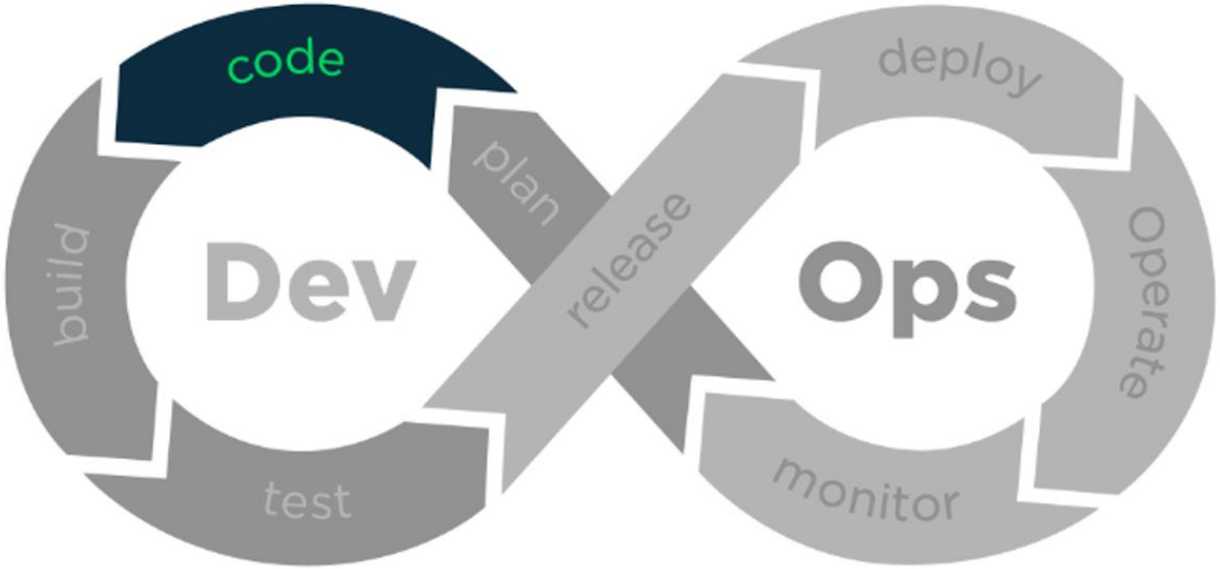
Sublime Text

cmder

git

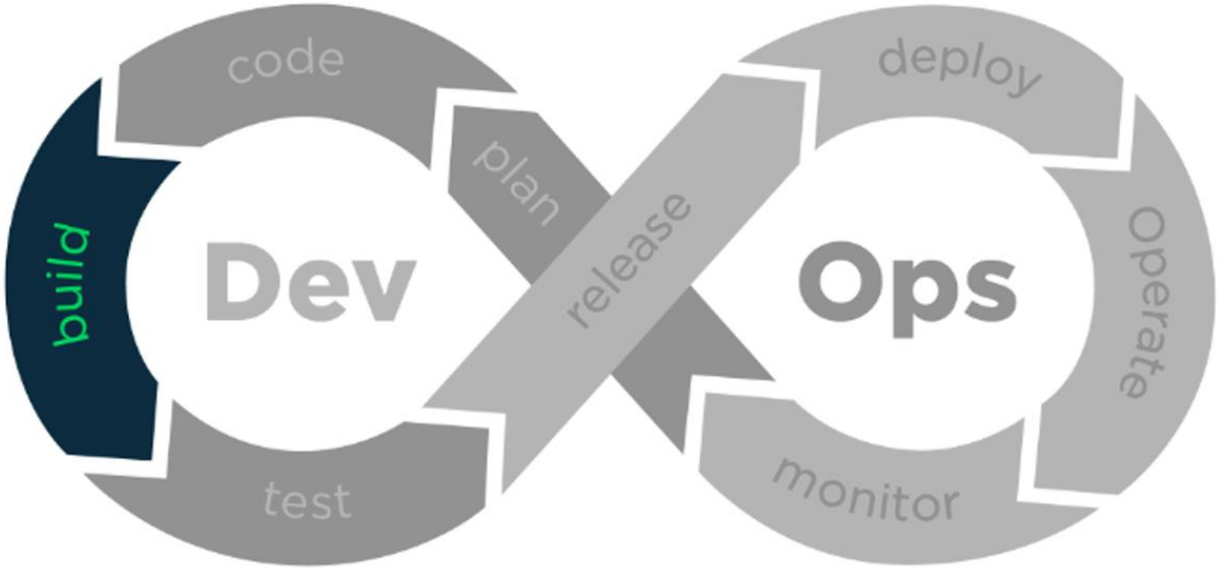
VS Code

Notepad++



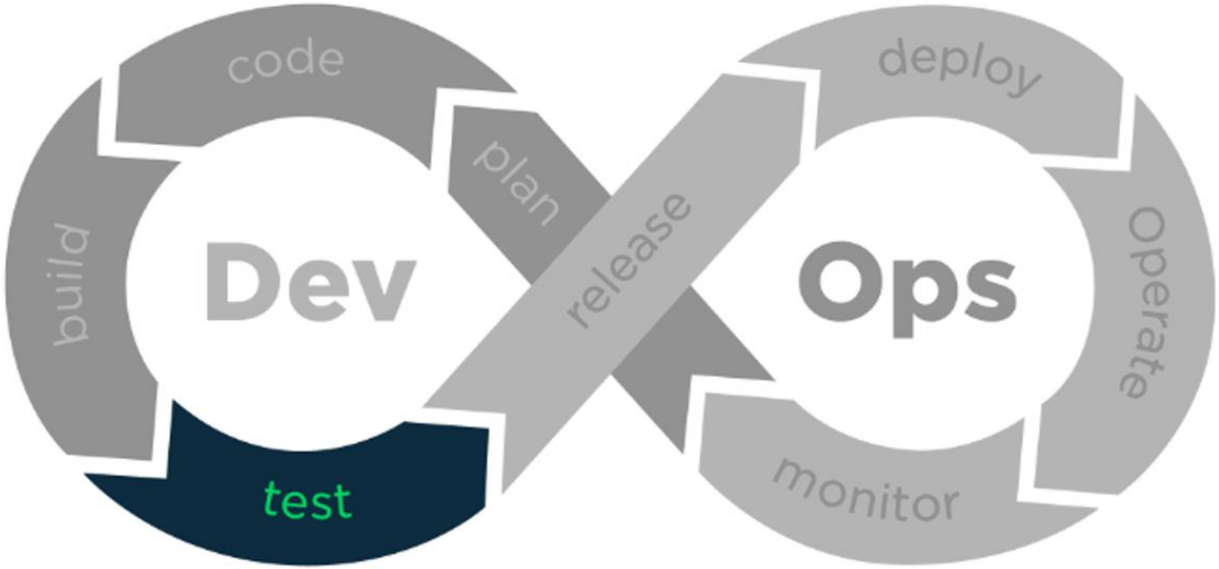
Entregando Software

containerd
docker
NuGet
npm
cri-o
.NET CLI
MSBuild

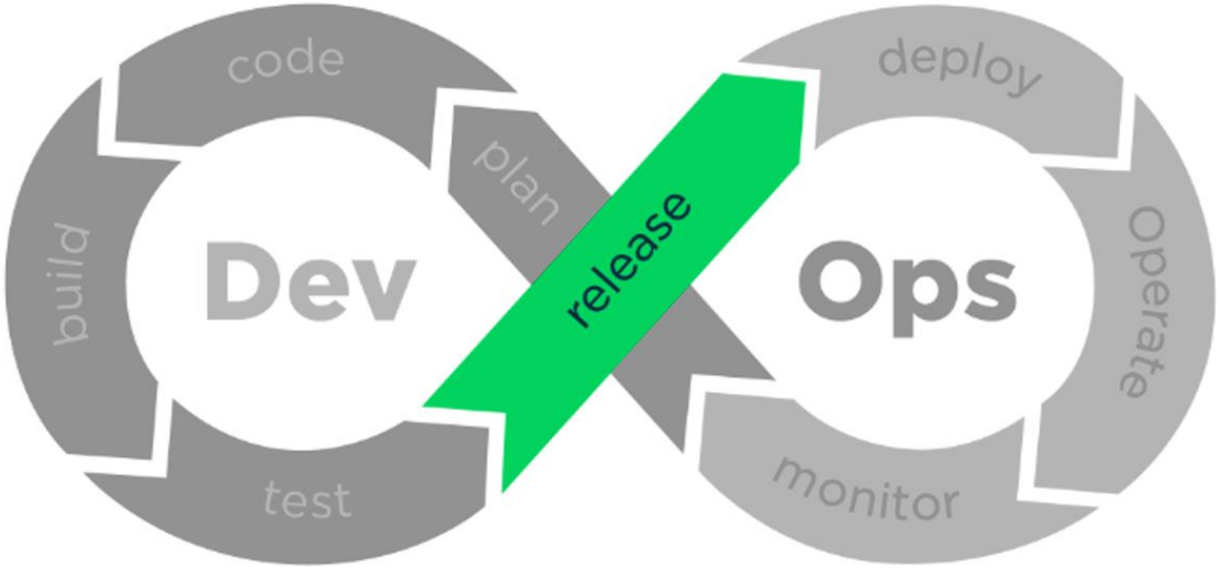


Entregando Software

xUnit.net
loader.io
APACHE JMeter™
SELENIUM
Runscope
POSTMAN
sonarqube
CODE CLIMATE
CODACY



Entregando Software



Entregando Software

Azure Pipelines

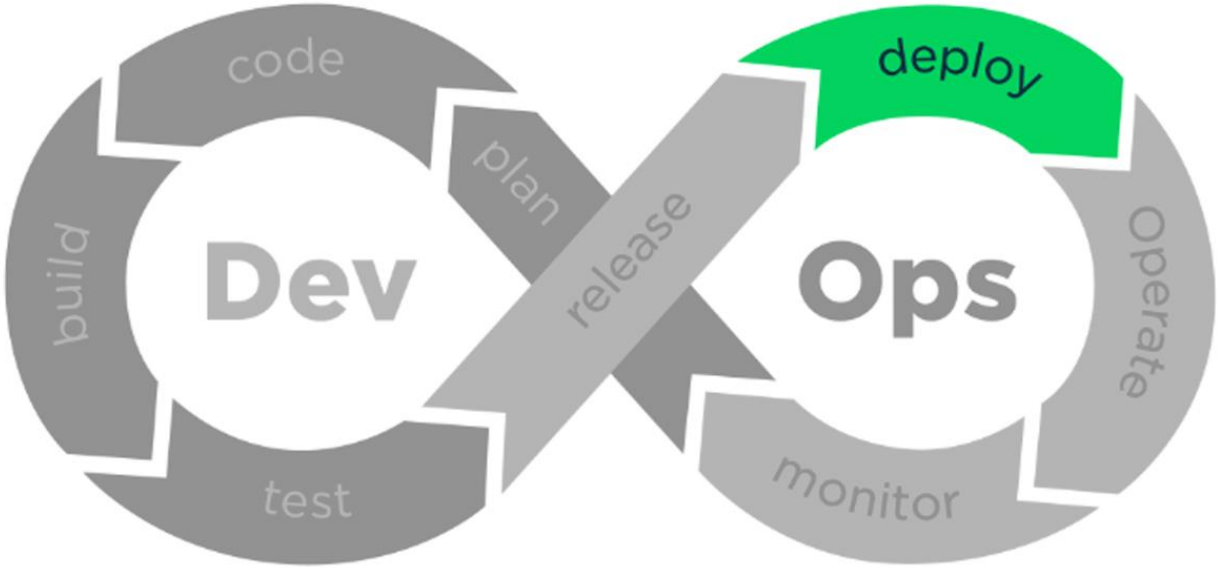
circleci

AppVeyor

GitLab CI

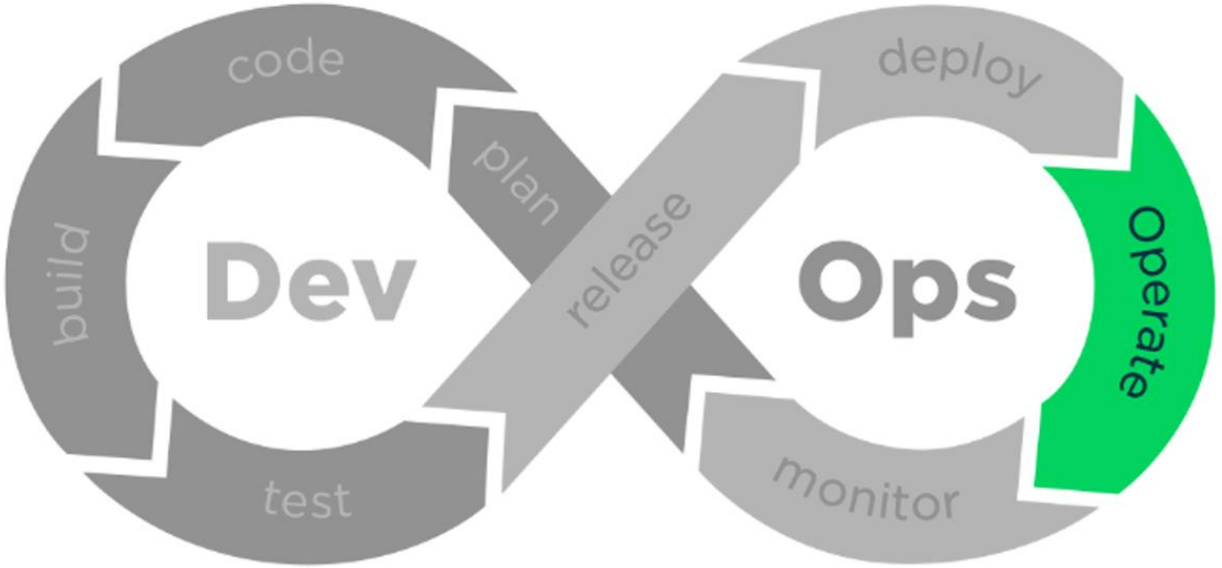
Travis CI

Jenkins



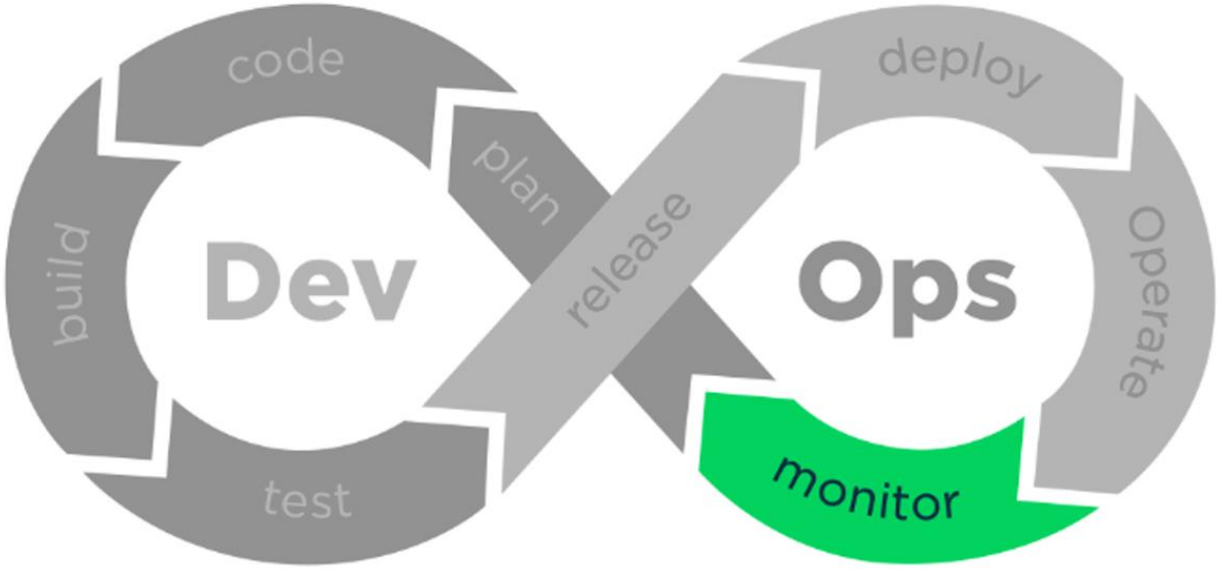
Entregando Software

kubernetes
RANCHER
Microsoft Azure
puppet
Terraform
CHEF
OPENSHIFT
aws

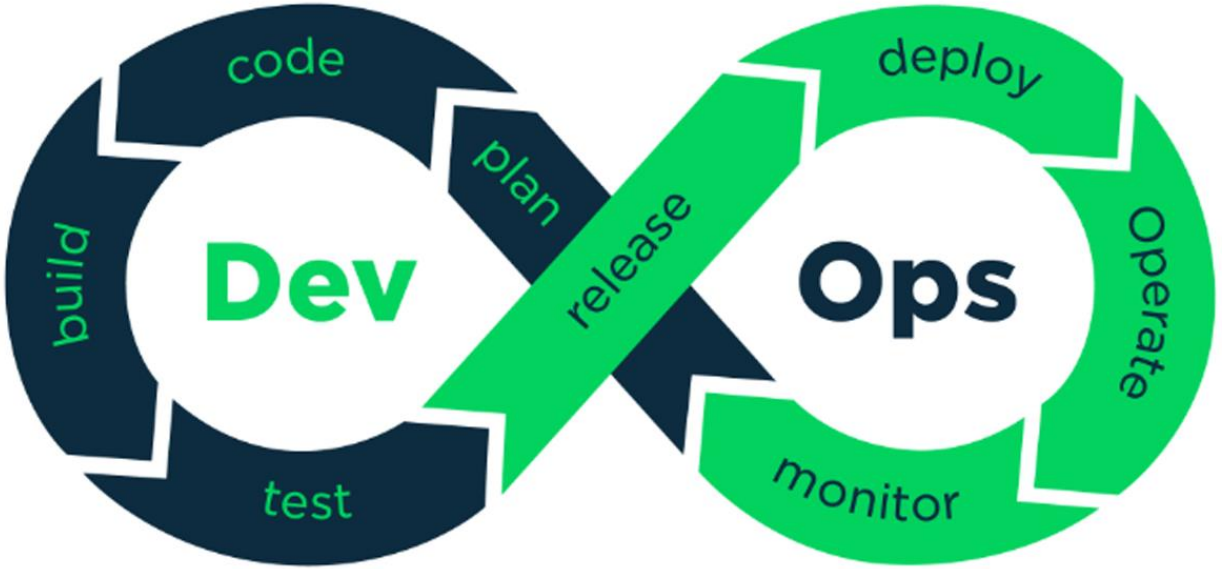


Entregando Software

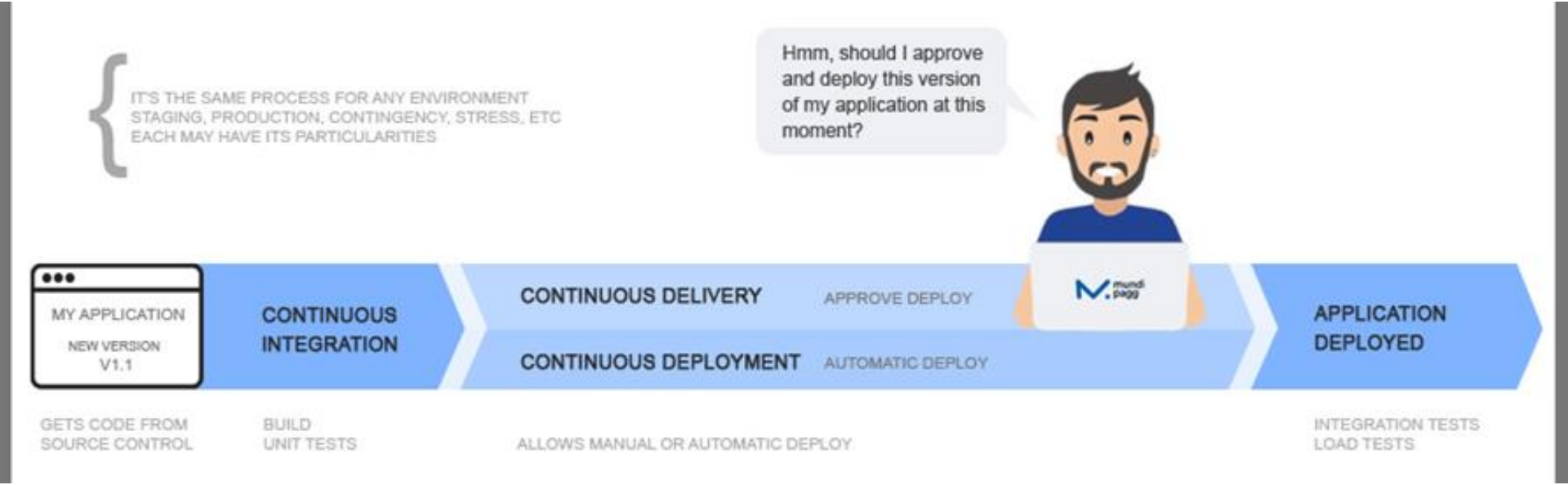
DATADOG
Prometheus
APPMETRICS
Rollbar
ZABBIX
New Relic.
Pushover
Seq
monitis



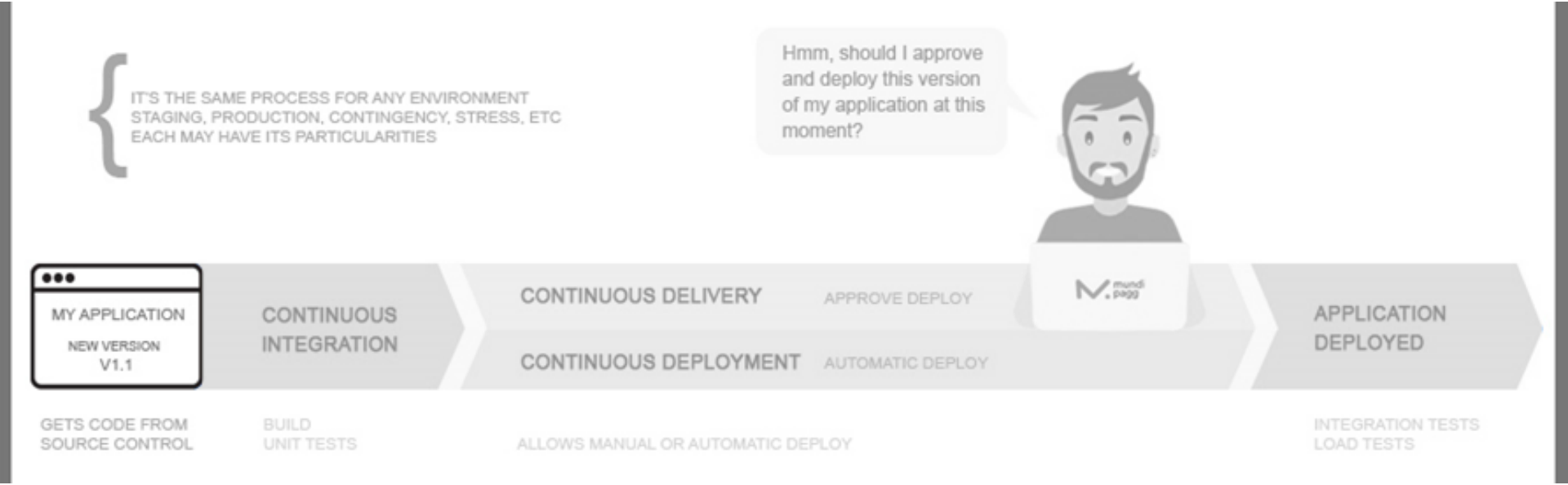
Entregando Software



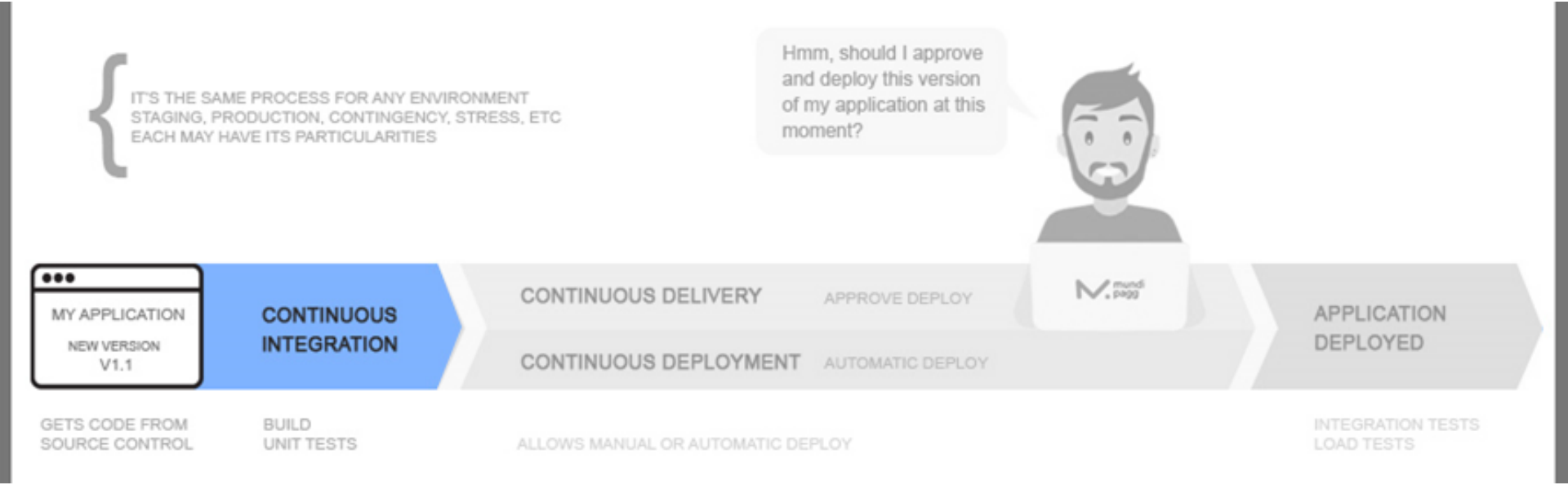
CI vs CD


IT'S THE SAME PROCESS FOR ANY ENVIRONMENT
STAGING, PRODUCTION, CONTINGENCY, STRESS, ETC
EACH MAY HAVE ITS PARTICULARITIES
Hmm, should I approve and deploy this version of my application at this moment?
MY APPLICATION
NEW VERSION
V1.1
CONTINUOUS INTEGRATION
CONTINUOUS DELIVERY
APPROVE DEPLOY
CONTINUOUS DEPLOYMENT
AUTOMATIC DEPLOY
APPLICATION DEPLOYED
GETS CODE FROM SOURCE CONTROL
BUILD
UNIT TESTS
ALLOWS MANUAL OR AUTOMATIC DEPLOY
INTEGRATION TESTS
LOAD TESTS

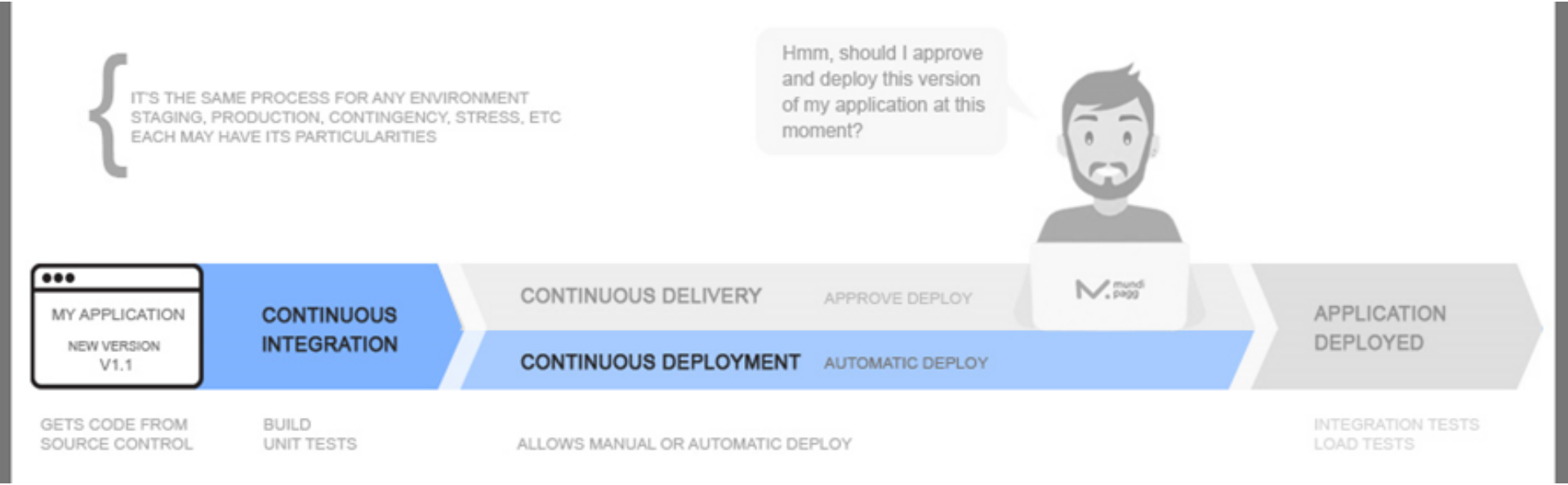
CI vs CD


IT'S THE SAME PROCESS FOR ANY ENVIRONMENT
STAGING, PRODUCTION, CONTINGENCY, STRESS, ETC
EACH MAY HAVE ITS PARTICULARITIES
Hmm, should I approve and deploy this version of my application at this moment?
MY APPLICATION
NEW VERSION
V1.1
CONTINUOUS INTEGRATION
CONTINUOUS DELIVERY
APPROVE DEPLOY
CONTINUOUS DEPLOYMENT
AUTOMATIC DEPLOY
APPLICATION DEPLOYED
GETS CODE FROM SOURCE CONTROL
BUILD
UNIT TESTS
ALLOWS MANUAL OR AUTOMATIC DEPLOY
INTEGRATION TESTS
LOAD TESTS

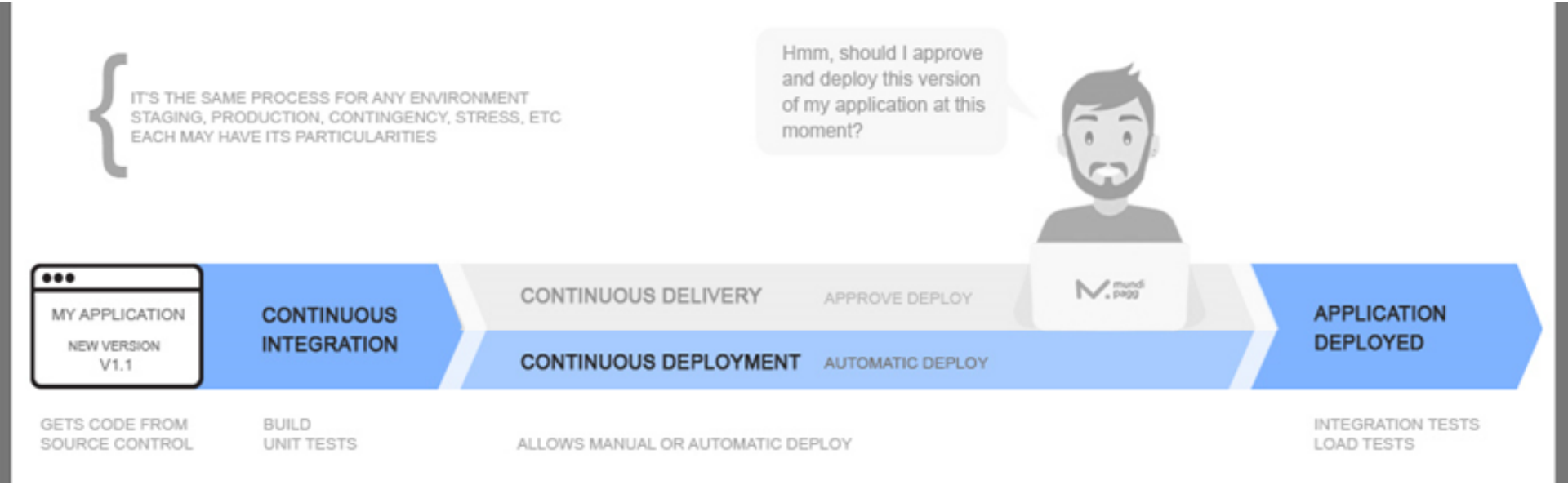
CI vs CD


IT'S THE SAME PROCESS FOR ANY ENVIRONMENT
STAGING, PRODUCTION, CONTINGENCY, STRESS, ETC
EACH MAY HAVE ITS PARTICULARITIES
Hmm, should I approve and deploy this version of my application at this moment?
MY APPLICATION
NEW VERSION
V1.1
CONTINUOUS INTEGRATION
CONTINUOUS DELIVERY
APPROVE DEPLOY
CONTINUOUS DEPLOYMENT
AUTOMATIC DEPLOY
APPLICATION DEPLOYED
GETS CODE FROM SOURCE CONTROL
BUILD
UNIT TESTS
ALLOWS MANUAL OR AUTOMATIC DEPLOY
INTEGRATION TESTS
LOAD TESTS

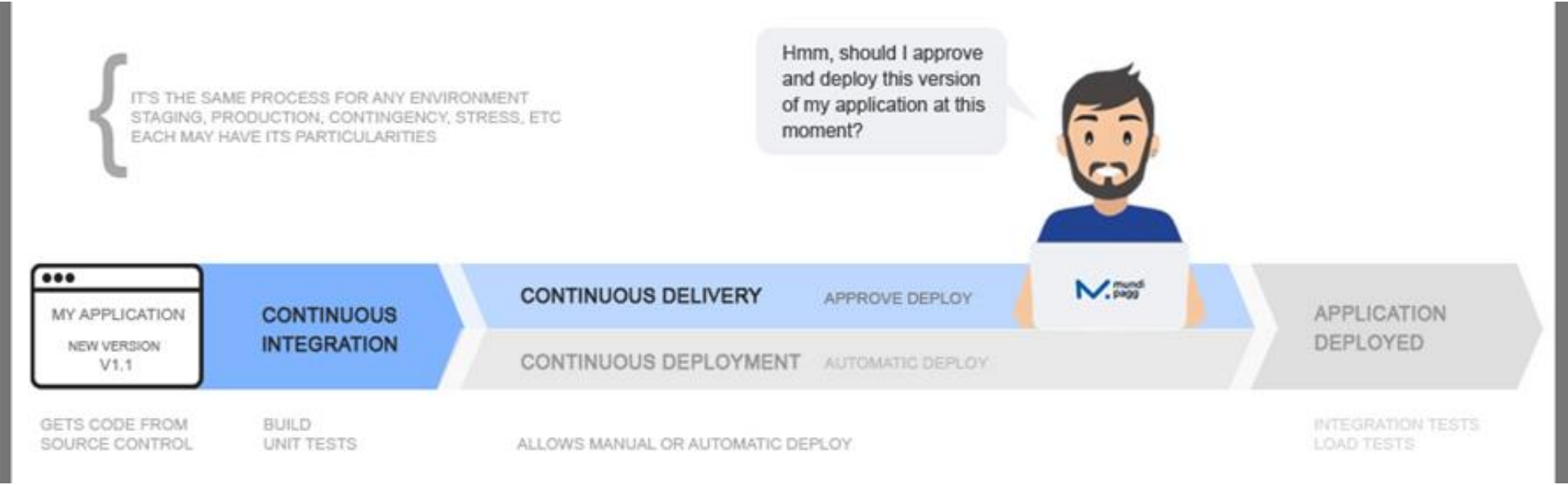
CI vs CD


IT'S THE SAME PROCESS FOR ANY ENVIRONMENT
STAGING, PRODUCTION, CONTINGENCY, STRESS, ETC
EACH MAY HAVE ITS PARTICULARITIES
Hmm, should I approve and deploy this version of my application at this moment?
MY APPLICATION
NEW VERSION
V1.1
CONTINUOUS INTEGRATION
CONTINUOUS DELIVERY
APPROVE DEPLOY
CONTINUOUS DEPLOYMENT
AUTOMATIC DEPLOY
APPLICATION DEPLOYED
GETS CODE FROM SOURCE CONTROL
BUILD
UNIT TESTS
ALLOWS MANUAL OR AUTOMATIC DEPLOY
INTEGRATION TESTS
LOAD TESTS

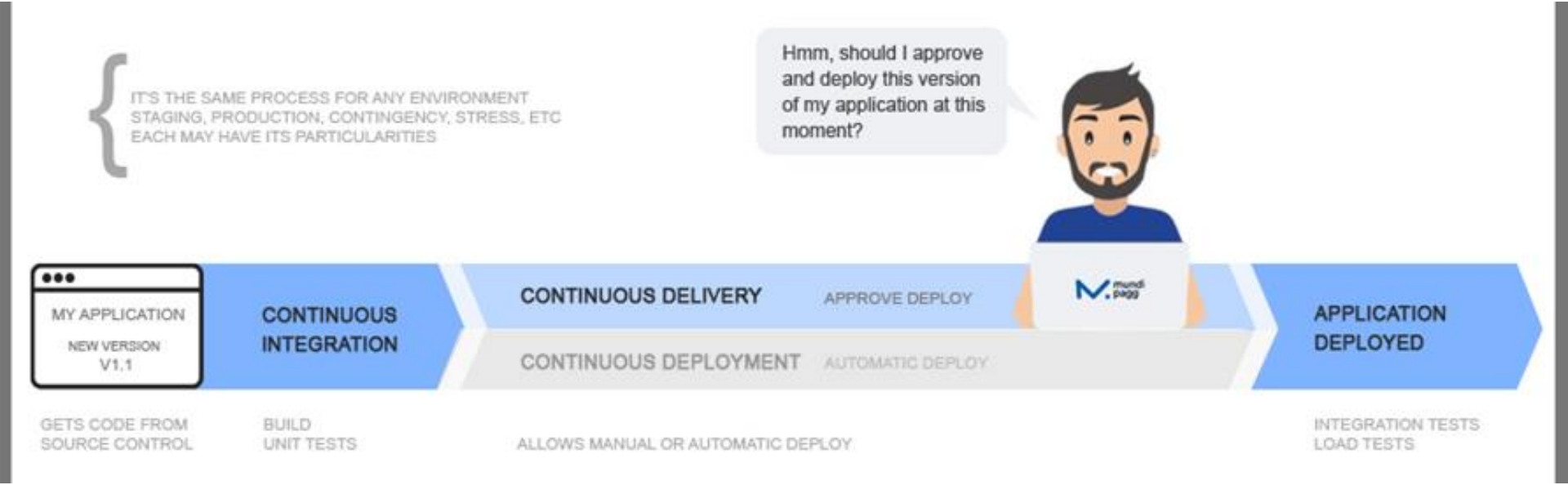
CI vs CD


IT'S THE SAME PROCESS FOR ANY ENVIRONMENT
STAGING, PRODUCTION, CONTINGENCY, STRESS, ETC
EACH MAY HAVE ITS PARTICULARITIES
Hmm, should I approve and deploy this version of my application at this moment?
MY APPLICATION
NEW VERSION
V1.1
CONTINUOUS INTEGRATION
CONTINUOUS DELIVERY
APPROVE DEPLOY
CONTINUOUS DEPLOYMENT
AUTOMATIC DEPLOY
APPLICATION DEPLOYED
GETS CODE FROM SOURCE CONTROL
BUILD
UNIT TESTS
ALLOWS MANUAL OR AUTOMATIC DEPLOY
INTEGRATION TESTS
LOAD TESTS

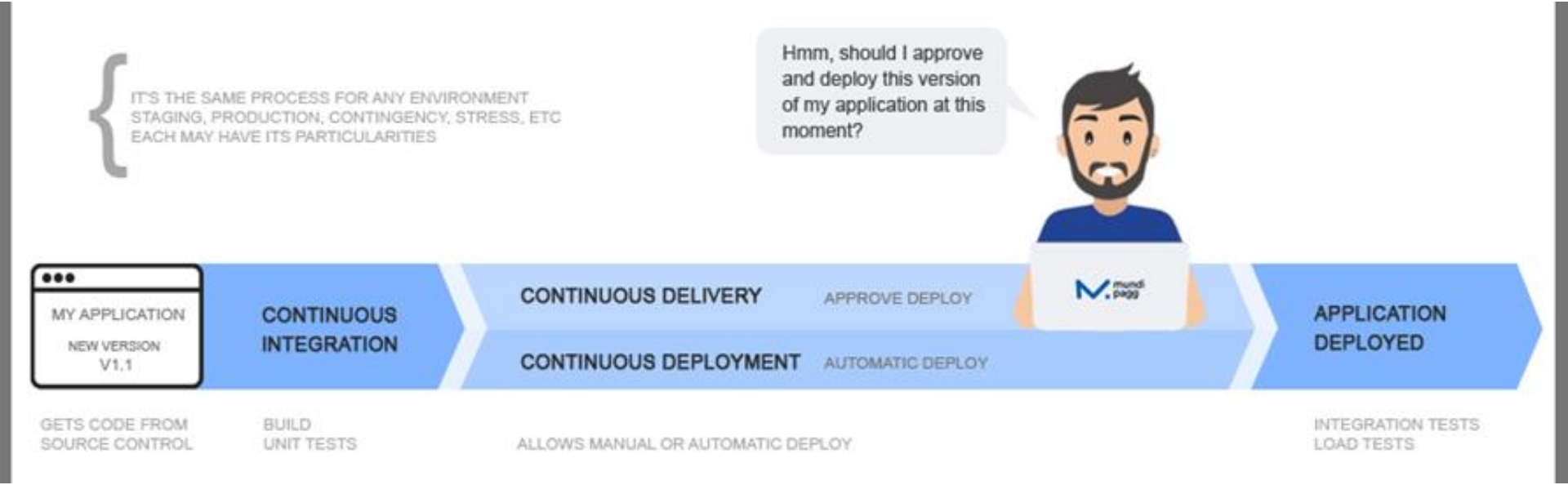
CI vs CD


IT'S THE SAME PROCESS FOR ANY ENVIRONMENT
STAGING, PRODUCTION, CONTINGENCY, STRESS, ETC
EACH MAY HAVE ITS PARTICULARITIES
Hmm, should I approve and deploy this version of my application at this moment?
MY APPLICATION
NEW VERSION
V1.1
CONTINUOUS INTEGRATION
CONTINUOUS DELIVERY
APPROVE DEPLOY
CONTINUOUS DEPLOYMENT
AUTOMATIC DEPLOY
APPLICATION DEPLOYED
GETS CODE FROM SOURCE CONTROL
BUILD
UNIT TESTS
ALLOWS MANUAL OR AUTOMATIC DEPLOY
INTEGRATION TESTS
LOAD TESTS

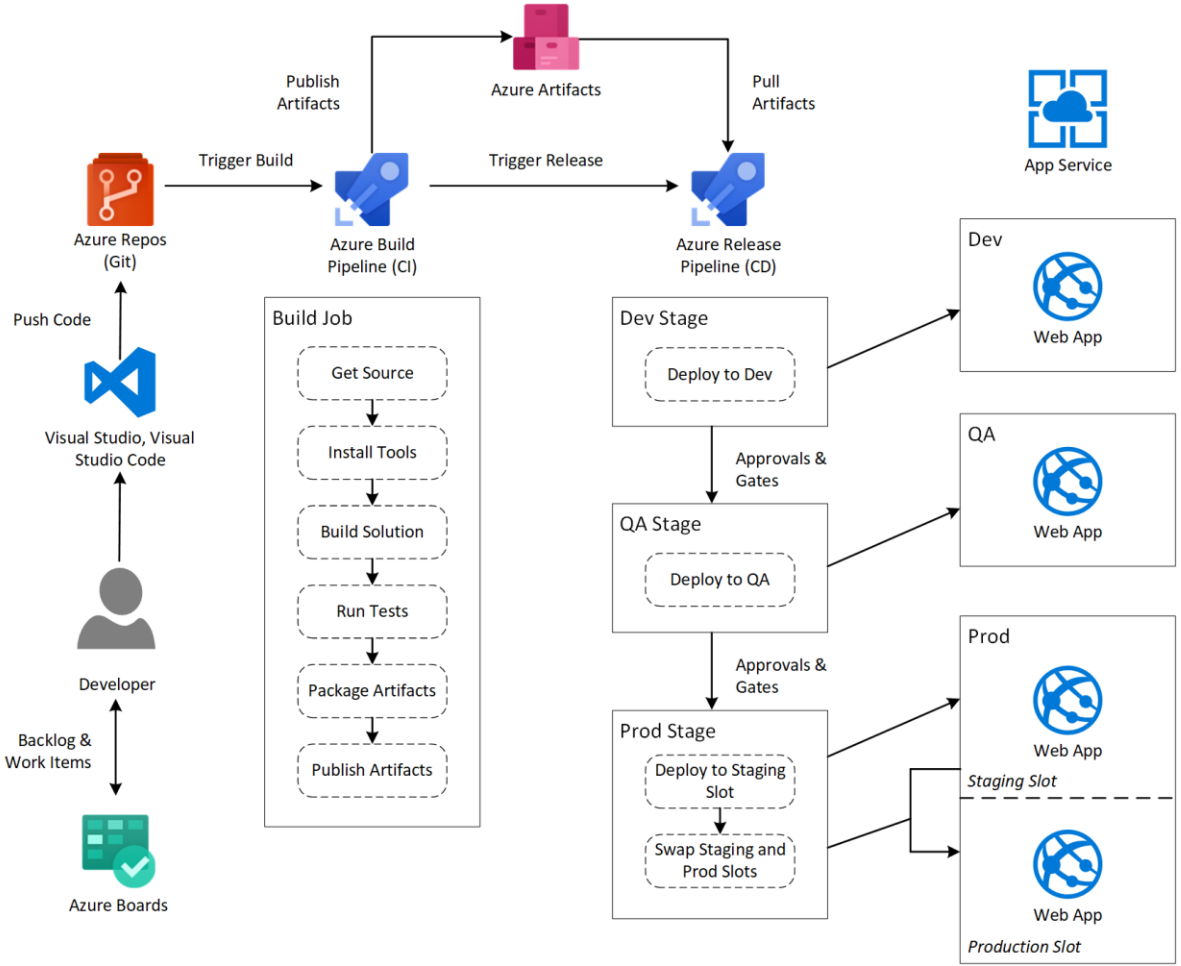
CI vs CD


IT'S THE SAME PROCESS FOR ANY ENVIRONMENT
STAGING, PRODUCTION, CONTINGENCY, STRESS, ETC
EACH MAY HAVE ITS PARTICULARITIES
Hmm, should I approve and deploy this version of my application at this moment?
MY APPLICATION
NEW VERSION
V1.1
CONTINUOUS INTEGRATION
CONTINUOUS DELIVERY
APPROVE DEPLOY
CONTINUOUS DEPLOYMENT
AUTOMATIC DEPLOY
APPLICATION DEPLOYED
GETS CODE FROM SOURCE CONTROL
BUILD
UNIT TESTS
ALLOWS MANUAL OR AUTOMATIC DEPLOY
INTEGRATION TESTS
LOAD TESTS

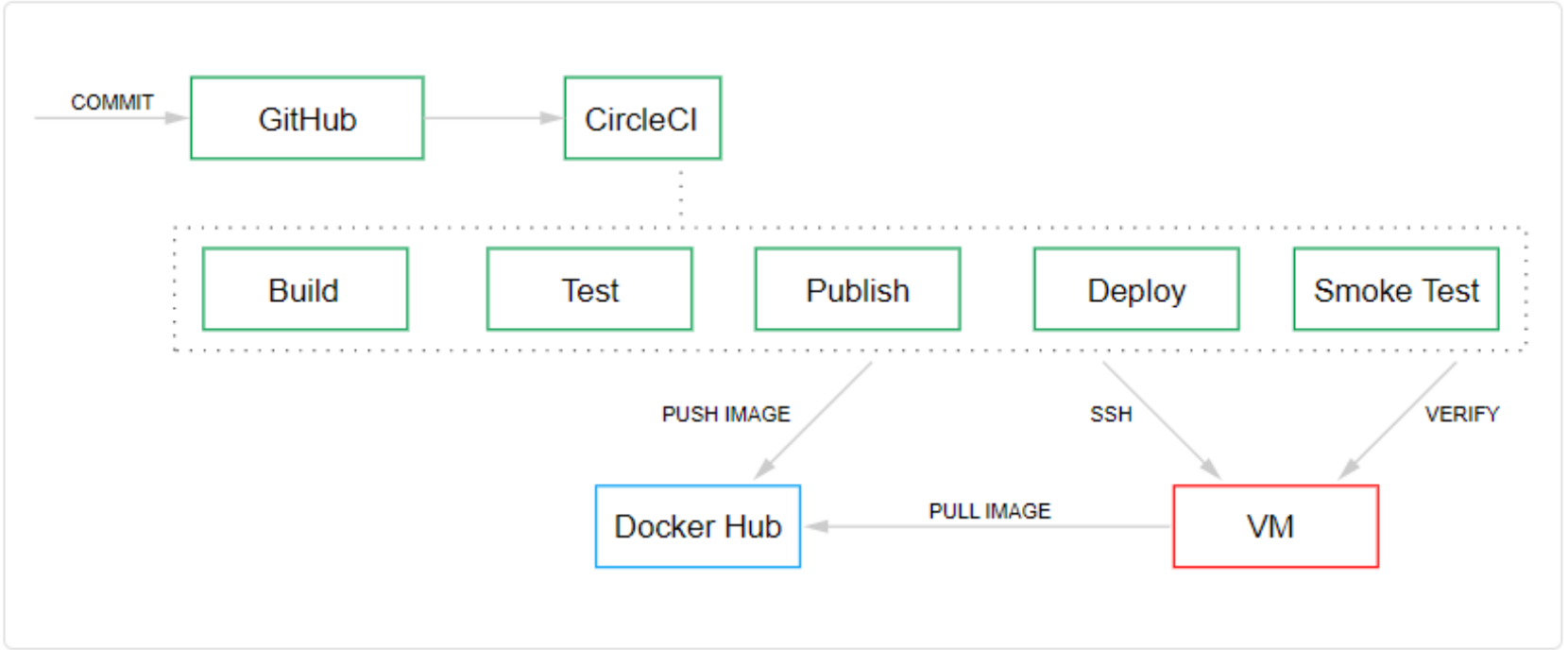
CI vs CD


IT'S THE SAME PROCESS FOR ANY ENVIRONMENT
STAGING, PRODUCTION, CONTINGENCY, STRESS, ETC
EACH MAY HAVE ITS PARTICULARITIES
Hmm, should I approve and deploy this version of my application at this moment?
MY APPLICATION
NEW VERSION
V1.1
CONTINUOUS INTEGRATION
CONTINUOUS DELIVERY
APPROVE DEPLOY
CONTINUOUS DEPLOYMENT
AUTOMATIC DEPLOY
APPLICATION DEPLOYED
GETS CODE FROM SOURCE CONTROL
BUILD
UNIT TESTS
ALLOWS MANUAL OR AUTOMATIC DEPLOY
INTEGRATION TESTS
LOAD TESTS

Pipelines


Publish Artifacts
Azure Artifacts
Pull Artifacts
App Service
Trigger Build
Trigger Release
Azure Repos (Git)
Azure Build Pipeline (CI)
Azure Release Pipeline (CD)
Push Code
Build Job
Get Source
Install Tools
Build Solution
Run Tests
Package Artifacts
Publish Artifacts
Dev Stage
Deploy to Dev
Approvals & Gates
QA Stage
Deploy to QA
Approvals & Gates
Prod Stage
Deploy to Staging Slot
Swap Staging and Prod Slots
Dev
Web App
QA
Web App
Prod
Web App
Staging Slot
Web App
Production Slot
Visual Studio, Visual Studio Code
Developer
Backlog & Work Items
Azure Boards

Pipelines


COMMIT
GitHub
CircleCI
Build
Test
Publish
Deploy
Smoke Test
PUSH IMAGE
SSH
VERIFY
Docker Hub
PULL IMAGE
VM

Framework FLAIR

Framework FLAIR

Um **pipeline** deve ser **rápido** e permitir sua execução **diversas vezes**. O fluxo de CD deve ser ainda mais otimizado que o fluxo de CI, possibilitando **mudança de versões em tempo recorde**.

Framework FLAIR

A **interação humana** deve ser a **mínima** possível. No caso de **falhas**, o time deve ser **notificado**. No caso de **sucesso**, **pode-se ter uma interação** para que o **fluxo continue**, sendo tolerada mais de um nível de aprovação.

Framework FLAIR

Qualquer versão deve ser passível de ser **distribuída em qualquer momento**. Para a redistribuição, a construção deve ser dispensada, utilizando o artefato previamente criado. Para releases "antigas" é tolerável refazer o processo de CI.

Framework FLAIR

Execute sempre as **etapas da mesma forma**. O código que foi "construído" e "testado", deve ser o código do artefato. O **artefato usado** em **homologação/QA** deve **ser o mesmo** artefato de **produção**.

Framework FLAIR

Um **deploy** que **apenas altera configurações** do seu projeto **não deve gerar uma nova versão**, dado que o software é o mesmo.

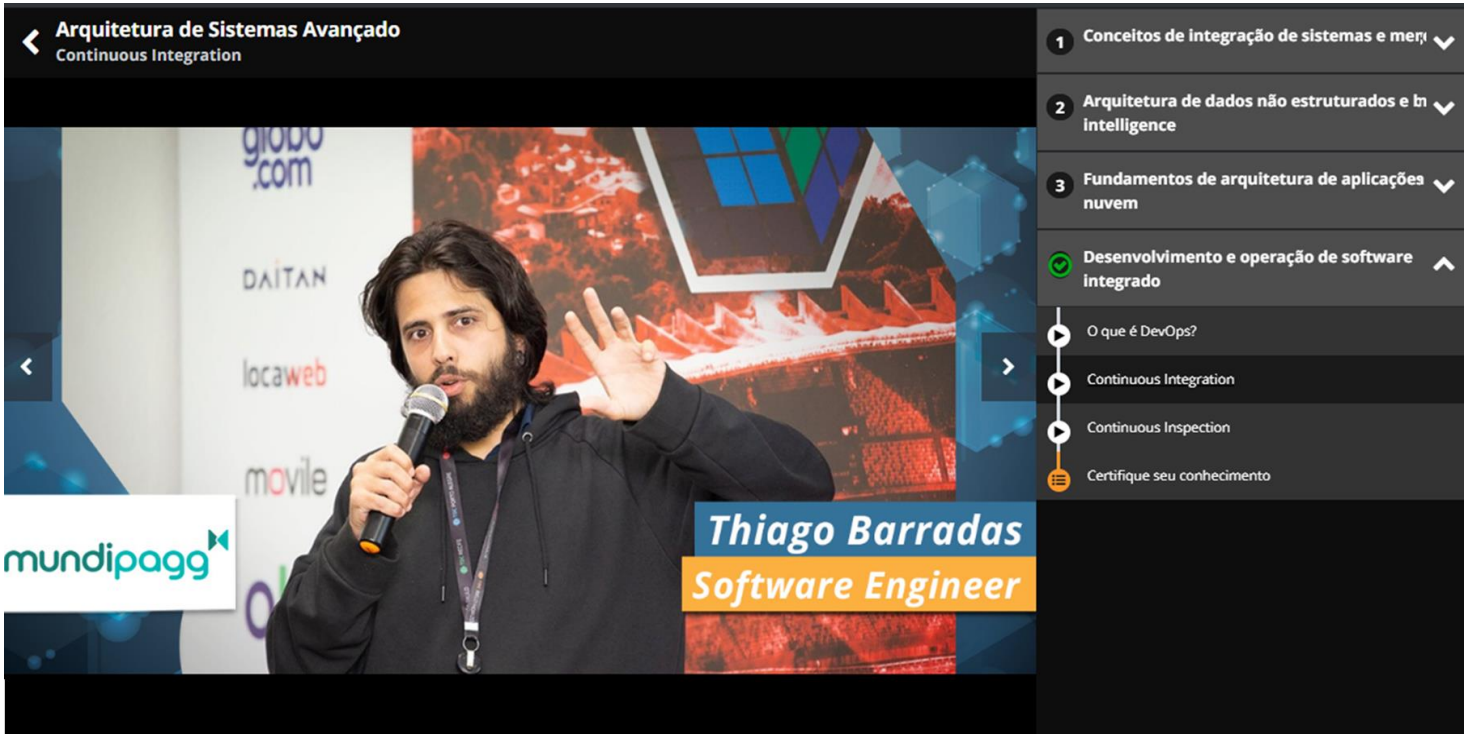
Framework FLAIR

Um **bom pipeline** deve ser **facilmente replicável** para projetos similares. Devem ser **definidos** por um **arquivo de configuração** ao invés de criados via interface. Se possível **utilize** uma ferramenta de **gerenciamento de configurações**.

Framework FLAIR

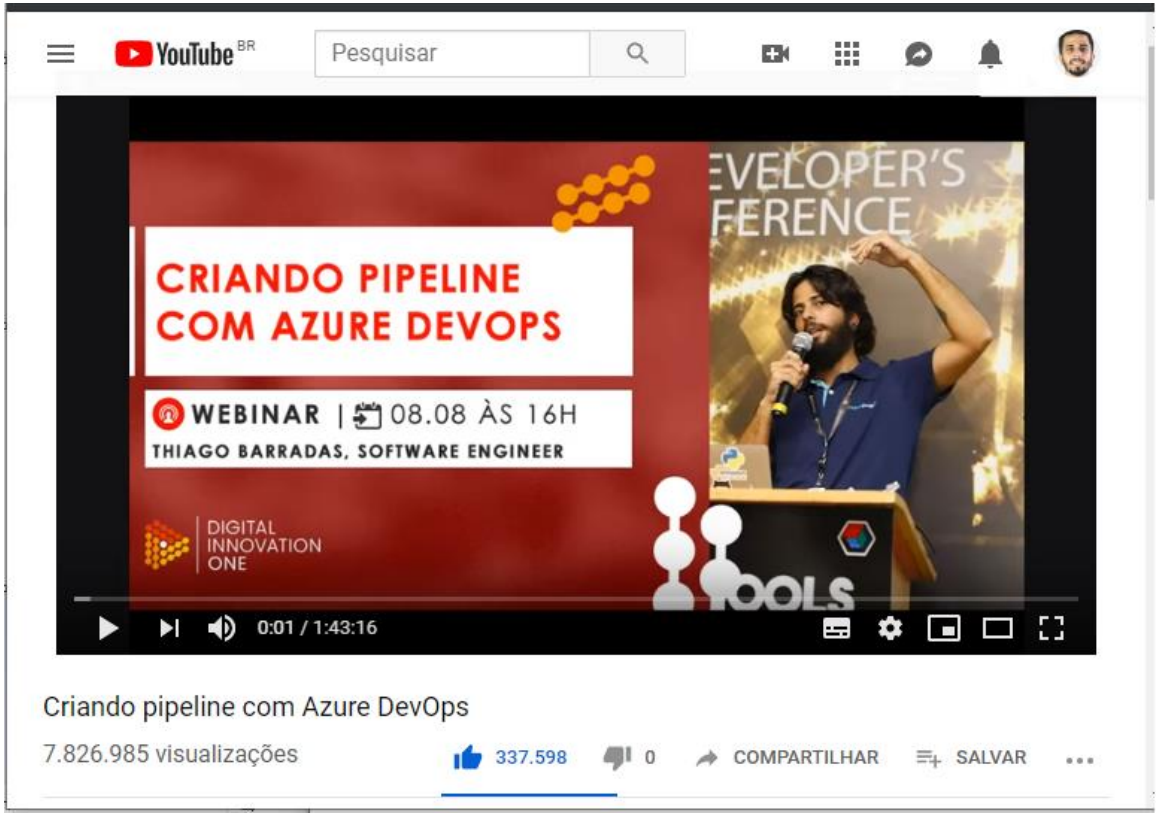
Um **bom pipeline** deve ser **facilmente replicável** para projetos similares. Devem ser **definidos** por um **arquivo de configuração** ao invés de criados via interface. Se possível **utilize** uma ferramenta de **gerenciamento de configurações**.

Curso DevOps



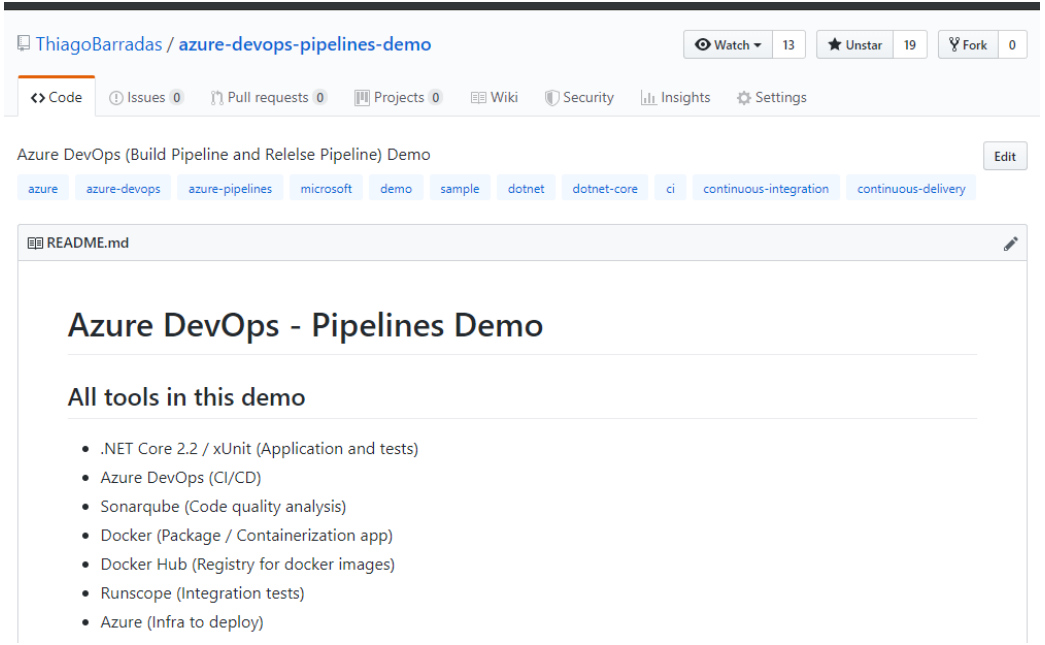
https://digitalinnovation.one

Hands-on



https://tinyurl.com/azure-pipelines

Step by step



https://tinyurl.com/demo-azure-devops

FLAIR Framework

FLAIR

Flair é um framework que define os 5 pontos principais que todo fluxo de CI deveria seguir.

https://tinyurl.com/flair-framework

Thiago Barradas

tbarradas@mundipagg.com

+55 (21) 99329-9143

Linkedin: **thiagobarradas**

Obrigado!