



**Empresa Brasil  
de Comunicação**

# de Gitlab a Kubernetes:

Serviços públicos com entrega ágil de sistemas de TIC sustentados por plataforma de orquestração contêineres

Adriano Vieira  
adriano.vieira@ebc.com.br



Empresa Brasil  
de Comunicação

# de Gitlab a Kubernetes

*Serviços públicos com entrega ágil de sistemas de TIC sustentados por  
plataforma de orquestração de contêineres*

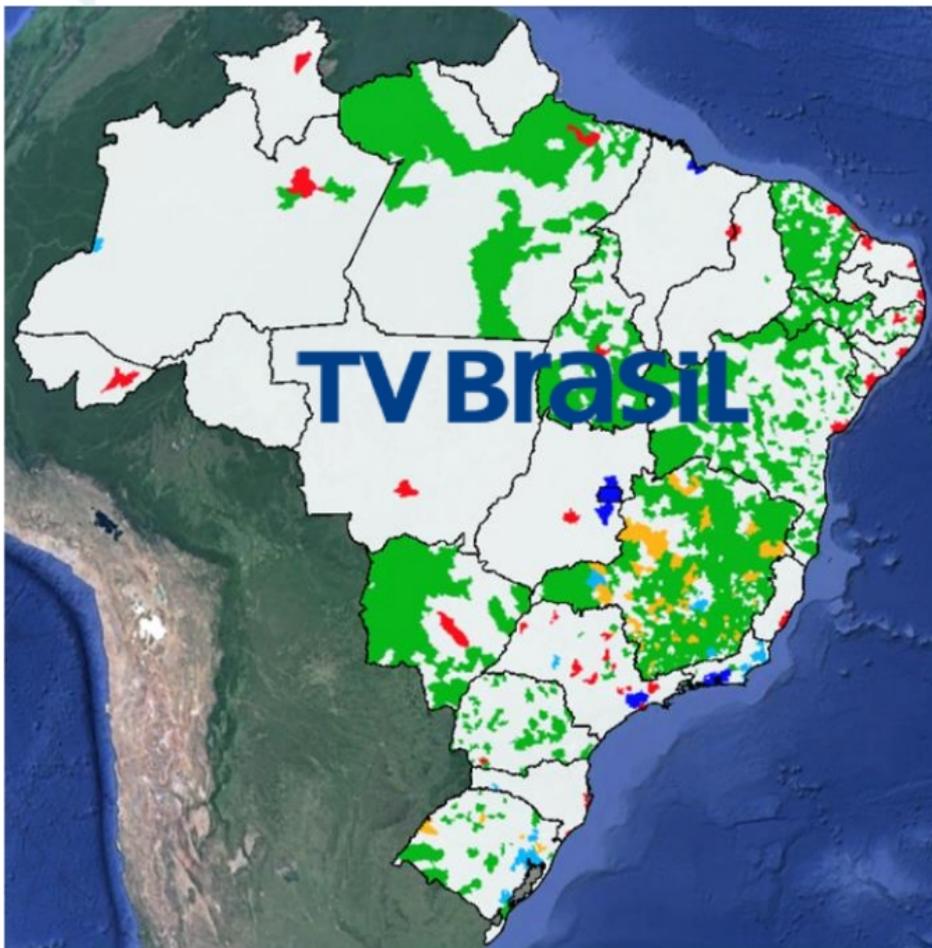
Eu na EBC  
Tecnologia  
DevOps  
Kubernetes  
Entrega Contínua  
Agilidade  
EBC  
Gitlab  
Processos  
Melhoria Contínua  
Docker  
Pessoas  
Integração Contínua  
Pipeline CI/CD  
Qualidade Contínua

# Empresa Brasil de Comunicação

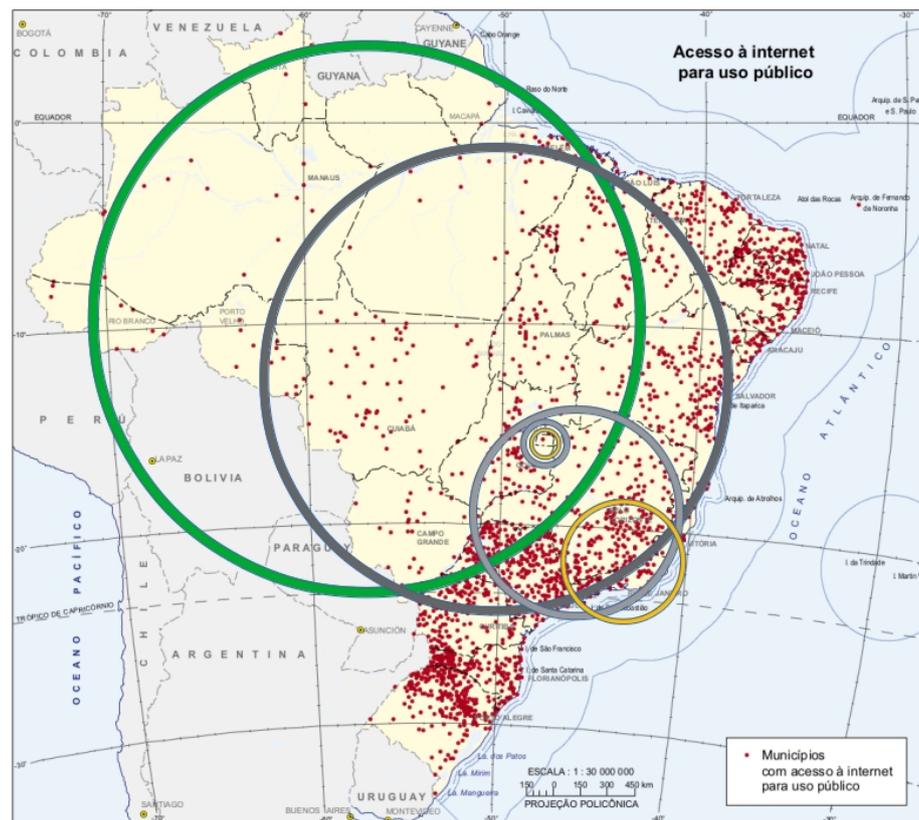


- **Agencia Brasil:** agência pública de notícias
- **NBR Governo Federal:** emissora estatal de televisão do Governo Federal
- **Portais:** Conteúdo público & Comunicação pública.
- **Rádioagência Nacional:** comunicação de conteúdos radiofônicos
- **Rádios:** Sistema EBC de Rádio que cobre todo o território nacional
- **Serviços:** diretamente prestados a clientes publico e privado
- **TV Brasil:** emissora de TV integrante da Rede Pública de Televisão

# Empresa Brasil de Comunicação



Emissoras e parcerias da TV Brasil  
(geradoras, retransmissoras, afiliadas)



Fonte: IBGE, Pesquisa de Informações Básicas Municipais 2009.

## Rádios

-  Nacional da Amazônia
-  Nacional de Brasília
-  Nacional do Rio de Janeiro
-  MEC
-  Internet: municípios com acesso para uso público  
(Fonte: IBGE/2009)

# Empresa Brasil de Comunicação



<http://www.ebc.com.br/institucional>

em TIC

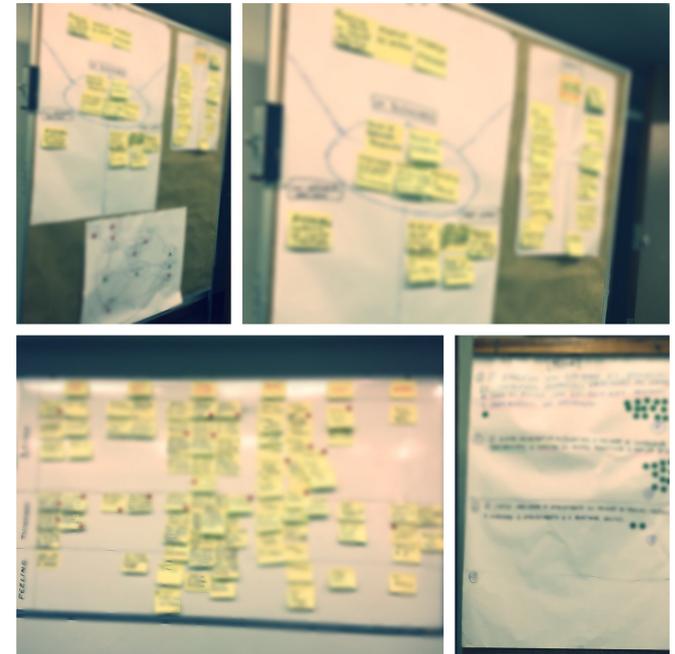
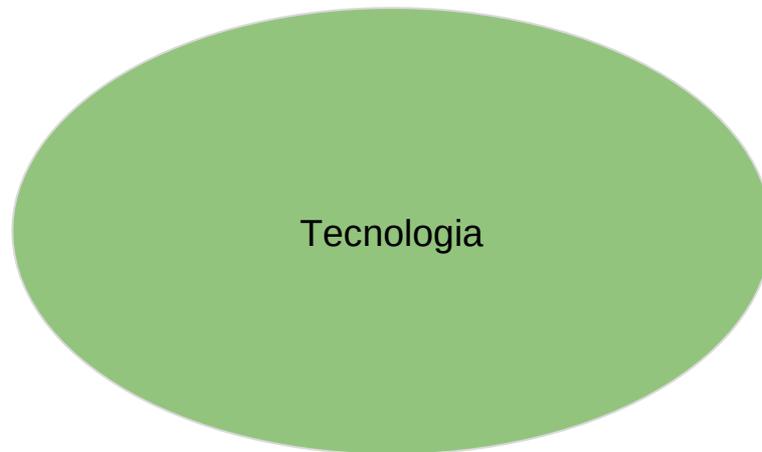
- Equipes: 4 (~30 Dev&Ops)
- Projetos: 300+
  - Java EE, Drupal, RoR, Python, NodeJS, C/C++, pyTorch
- Portais/Serviços/apps
  - Gestão de Ativos Digitais
  - Publicidade Legal
  - Portais EBC
  - EBC Play
  - EBC Rádios



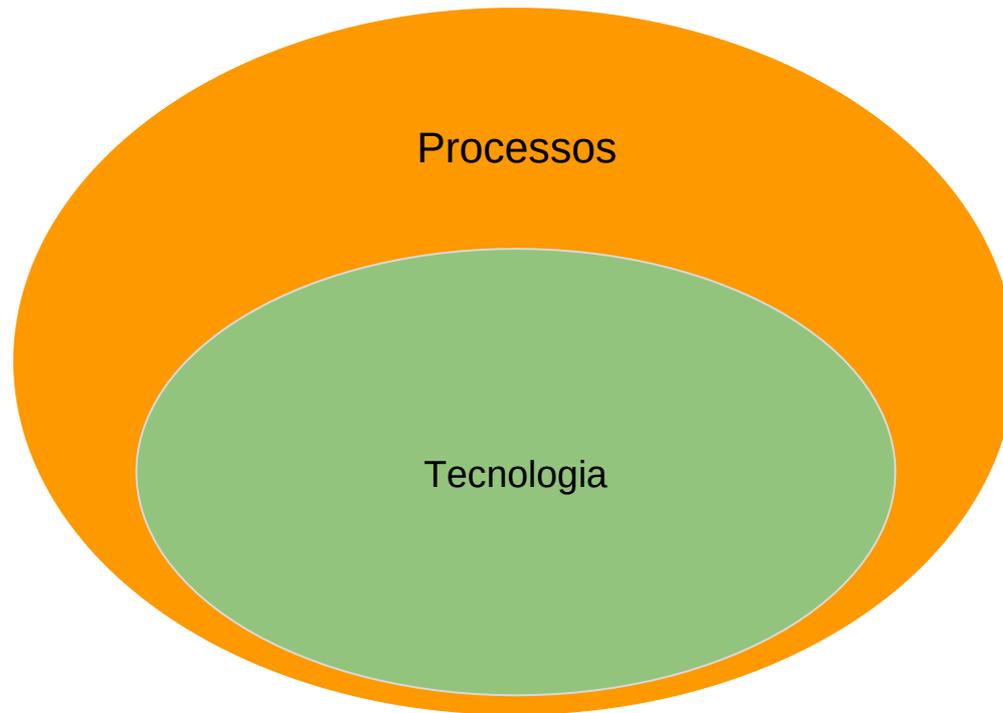
**Empresa Brasil  
de Comunicação**

# Melhorar e Adaptar

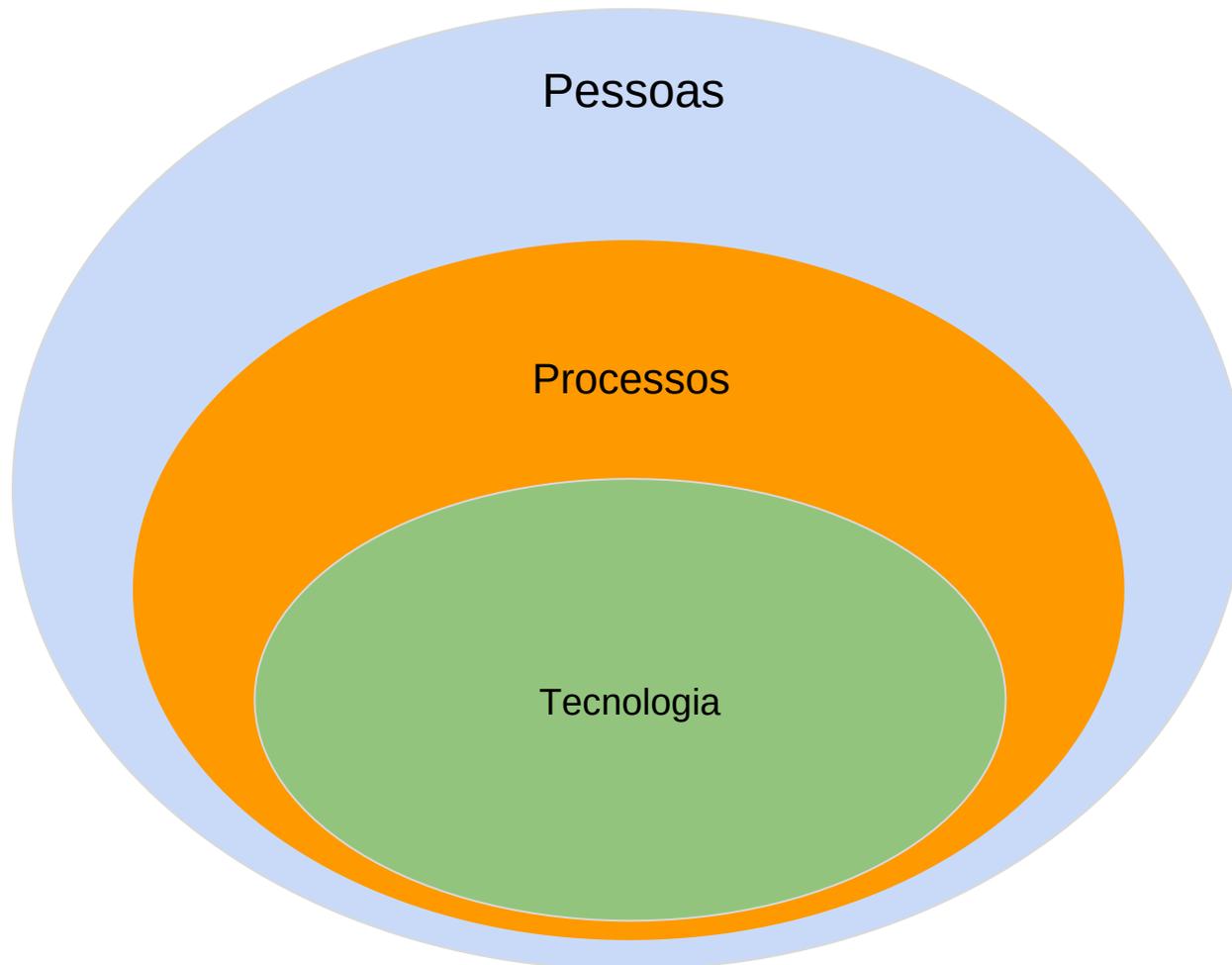
# Melhorar e Adaptar



# Melhorar e Adaptar

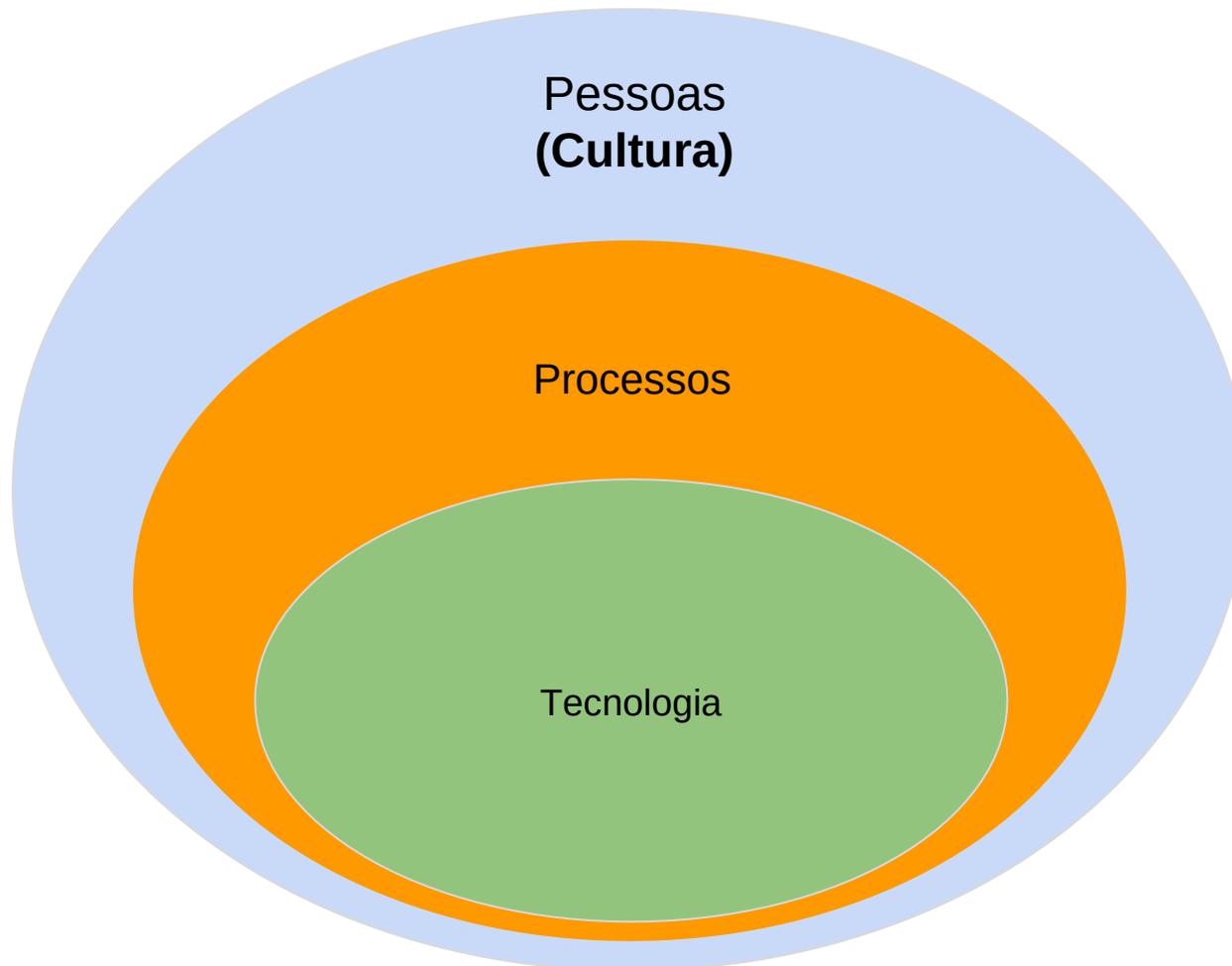


# Melhorar e Adaptar



Fonte: Making IT Lean

# Melhorar e Adaptar



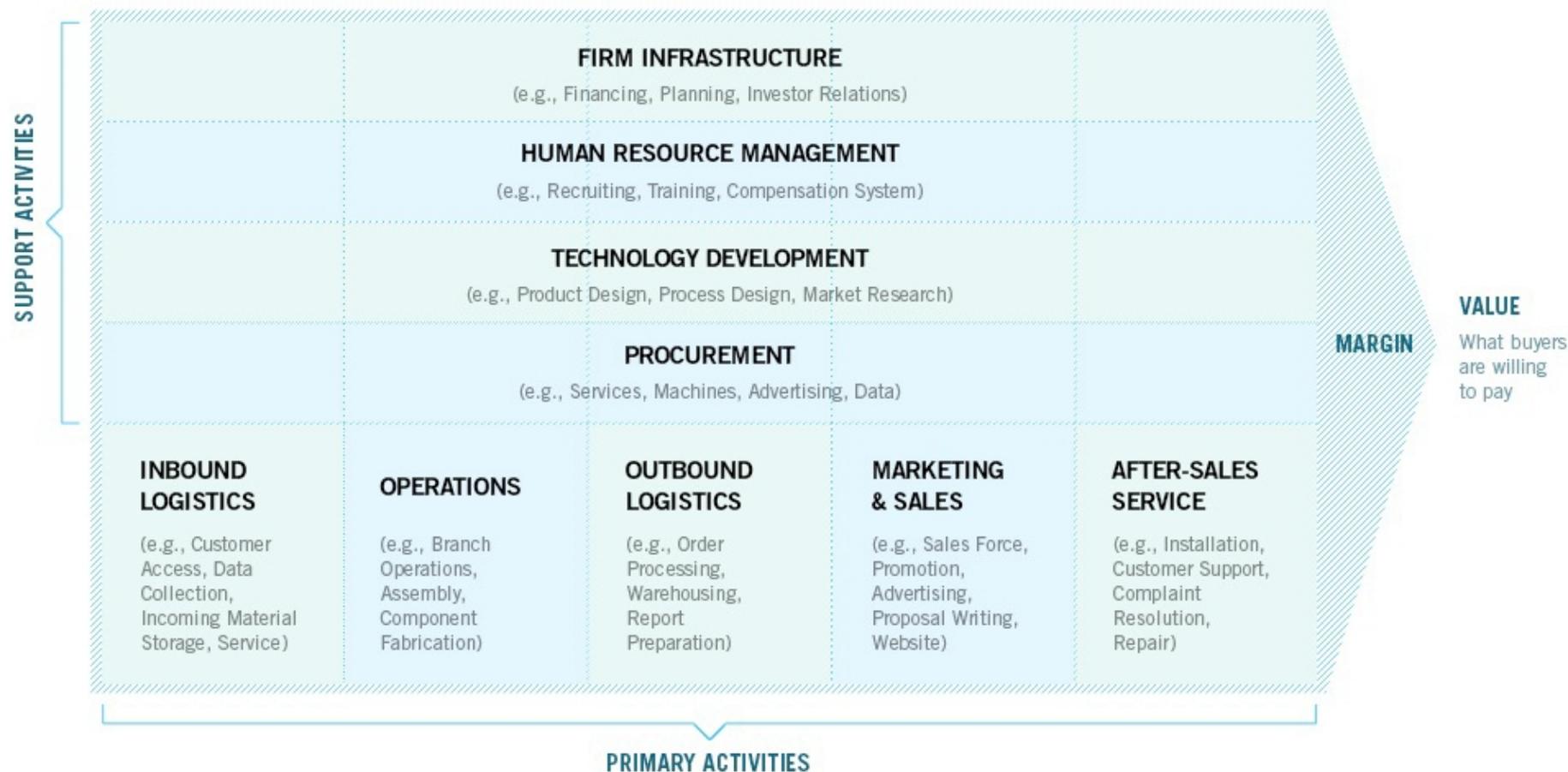
Fonte: Making IT Lean

# Melhorar e Adaptar



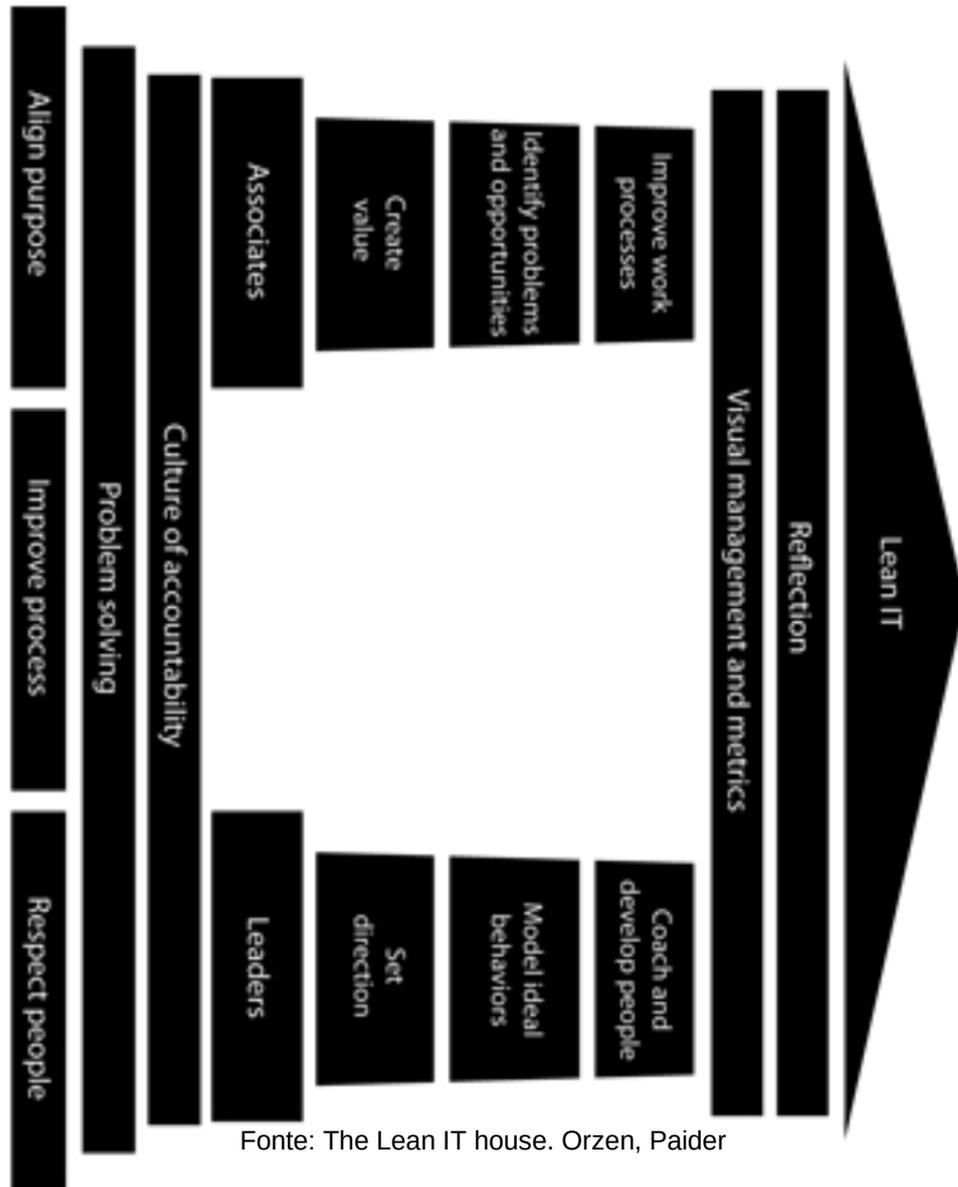
## Cadeia de Valor

Michael Porter



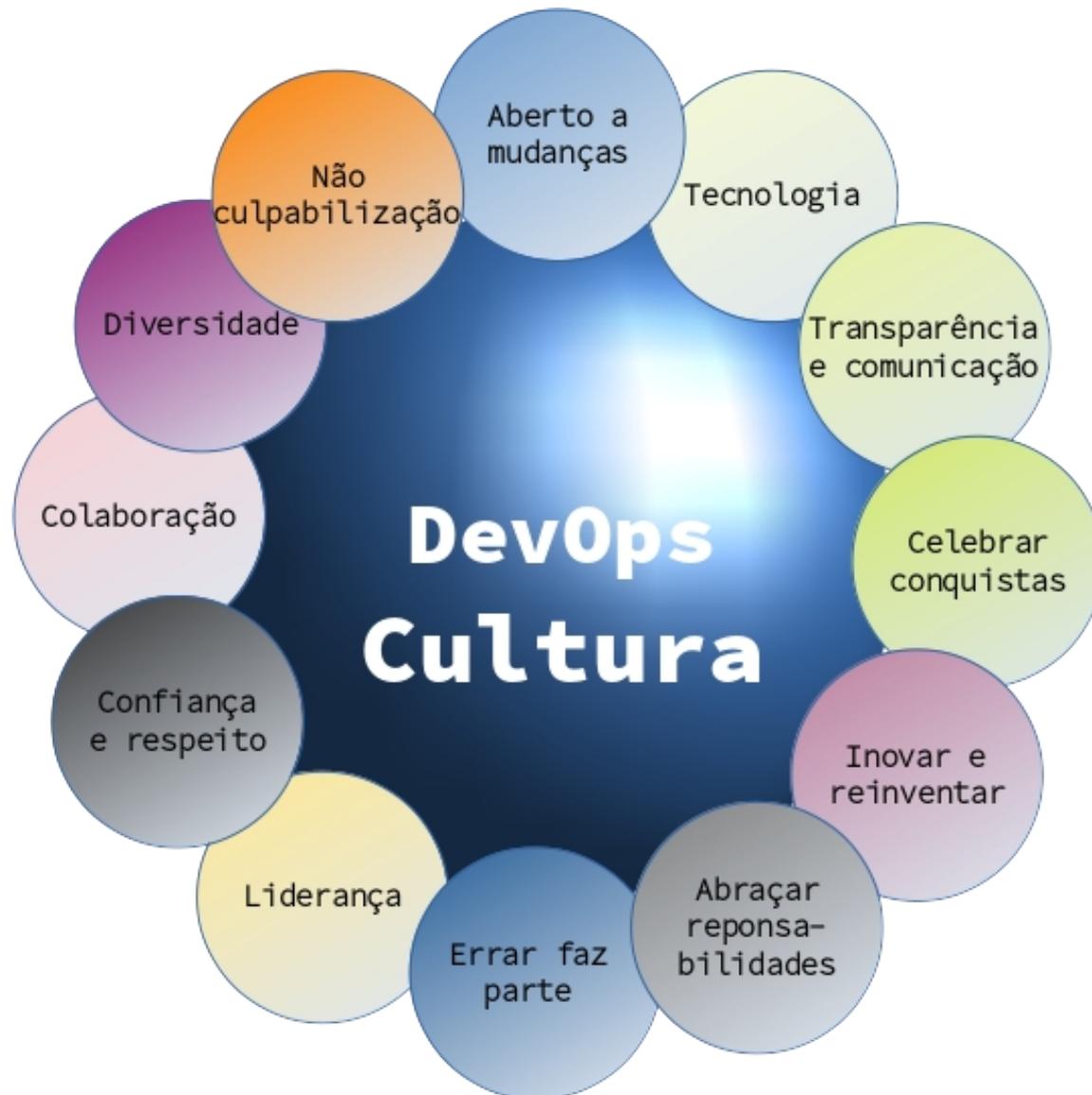
Fonte: HBS. <https://www.isc.hbs.edu/strategy/business-strategy/Pages/the-value-chain.aspx>

# Melhorar e Adaptar



Fonte: The Lean IT house. Orzen, Paider

# Melhorar e Adaptar



# Melhorar e Adaptar



## Cadeia de Valor em TI



**Eficiência  
Agilidade**

# Melhorar e Adaptar



Trabalhe em interações curtas  
ajustadas ao negócio

Cultura de experimentação

# Melhorar e Adaptar



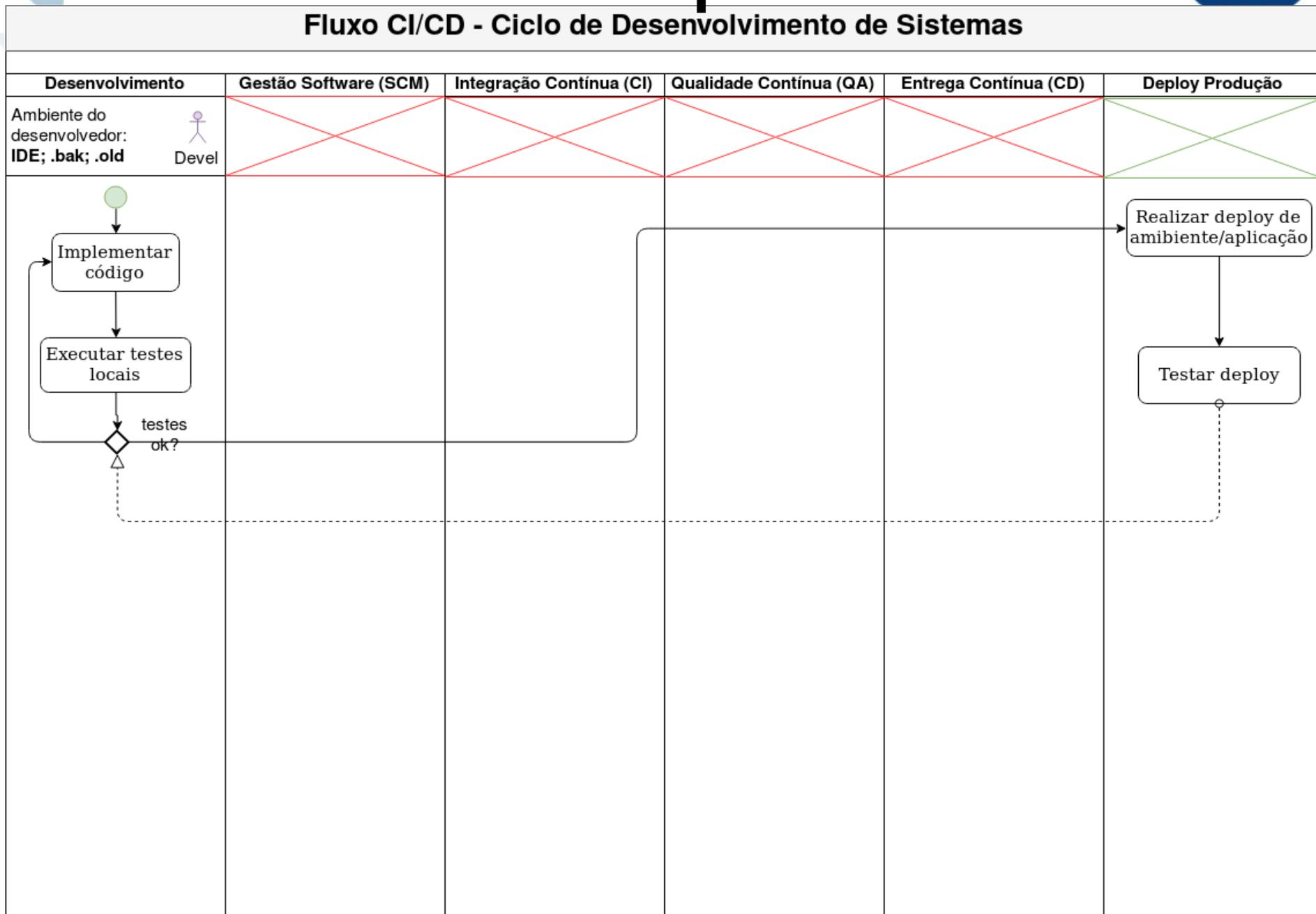
*Feedback* constante

Gestão apoiada em observação / monitoramento /  
KPI

# Melhorar e Adaptar



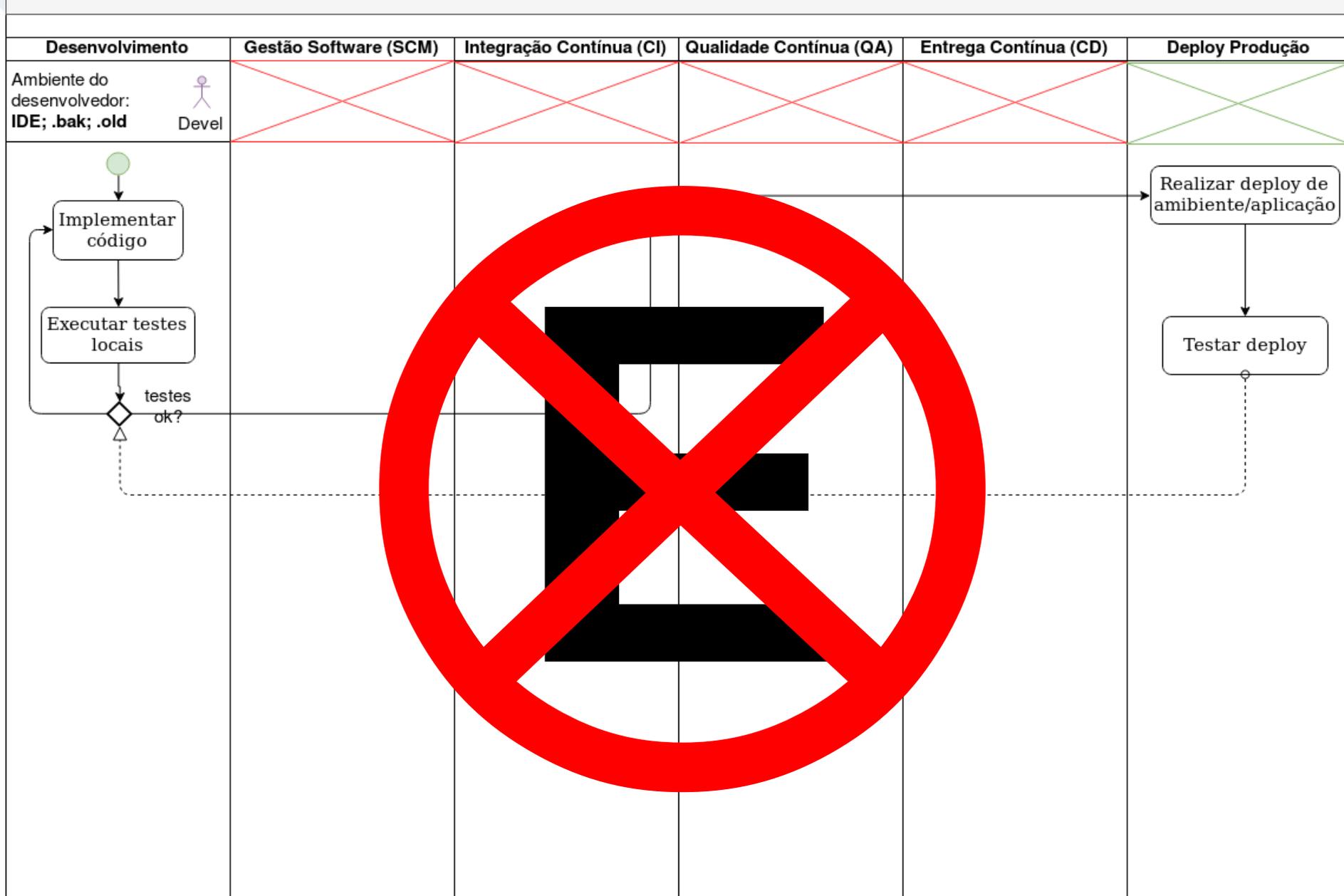
## Fluxo CI/CD - Ciclo de Desenvolvimento de Sistemas



# Melhorar e Adaptar



## Fluxo CI/CD - Ciclo de Desenvolvimento de Sistemas





**Empresa Brasil  
de Comunicação**

# Tecnologias

# Tecnologias



**PERIODIC TABLE OF DEVOPS TOOLS (V3)** EMBED [DOWNLOAD](#)

1 Os Gl GitLab																2 En Sp Splunk						
3 Fm Gh GitHub	4 En Dt Datical																5 En Xlr Xebialabs XL Release	6 Fm Aws AWS	7 Pd Az Azure	8 En Gc Google Cloud	9 En Op OpenShift	10 Fm Sl Sumo Logic
11 Os Sv Subversion	12 En Db DBMaestro																13 Os Dk Docker	14 En Ur UrbanCode Release	15 Pd Af Azure Functions	16 Pd Ld Lambda	17 En Ic IBM Cloud	18 Os Fd Fluentd
19 En Cw ISPW	20 En Dp Delphix	21 Os Jn Jenkins	22 Fm Cs Codeship	23 Os Fn FitNesse	24 Fr Ju JUnit	25 Fr Ka Karma	26 Os Su SoapUI	27 En Ch Chef	28 Fr Tf Terraform	29 En Xld Xebialabs XL Deploy	30 En Ud UrbanCode Deploy	31 Os Ku Kubernetes	32 Fm Cc CA CD Director	33 En Pr Plutora Release	34 Pd Al Alibaba Cloud	35 Os Os OpenStack	36 Os Ps Prometheus					
37 Os At Artifactory	38 En Rg Redgate	39 Pd Ba Bamboo	40 Fm Vs VSTS	41 Fr Se Selenium	42 Fr Jm JMeter	43 Os Ja Jasmine	44 Pd Sl Sauce Labs	45 Os An Ansible	46 Os Ru Rudder	47 En Oc Octopus Deploy	48 Os Go GoCD	49 Os Ms Mesos	50 Pd Gke GKE	51 Fm Om OpenMake	52 Pd Cp AWS CodePipeline	53 Os Cy Cloud Foundry	54 En It ITRS					
55 Os Nx Nexus	56 Os Fw Flyway	57 Os Tr Travis CI	58 Fm Tc TeamCity	59 Os Ga Gatling	60 Fr Tn TestNG	61 Fm Tt Tricentis Tosca	62 Pd Pe Perfecto	63 En Pu Puppet	64 Os Pa Packer	65 Fm Cd AWS CodeDeploy	66 En Ec ElectricCloud	67 Os Ra Rancher	68 Pd Aks AKS	69 Os Rk Rkt	70 Os Sp Spinnaker	71 Pd Ir Ironio	72 Pd Mg Moogsoft					
73 Fm Bb BitBucket	74 En Pf Perforce HelixCore	75 Fm Cr Circle CI	76 Pd Cb AWS CodeBuild	77 Fr Cu Cucumber	78 Os Mc Mocha	79 Os Lo Looust.io	80 En Mf Micro Focus UFT	81 Os Sl Salt	82 Os Ce CFEngine	83 En Eb ElasticBox	84 En Ca CA Automatic	85 En De Docker Enterprise	86 Pd Ae AWS ECS	87 Fm Cf Codefresh	88 Os Hm Helm	89 Os Aw Apache OpenWhisk	90 Os Ls Logstash					
 <a href="#">Follow @xebialabs</a> <a href="#">Publication Guidelines</a> <a href="#">Download</a>																						
91 En Xli Xebialabs XL Impact	92 En Ki Kibana	93 Fm Nr New Relic	94 En Dt Dynatrace	95 En Dd Datadog	96 Fm Ad AppDynamics	97 Os El ElasticSearch	98 Os Ni Nagios	99 Os Zb Zabbix	100 En Zn Zenoss	101 En Cx Checkmarx SAST	102 En Sg Signal Sciences	103 En Bd BlackDuck	104 Os Sr SonarQube	105 Os Hv HashiCorp Vault								
106 En Sw ServiceNow	107 Pd Jr Jira	108 Fm Tl Trello	109 Fm Sl Slack	110 Fm St Stride	111 En Cn CollabNet VersionOne	112 En Ry Remedy	113 En Ac Agile Central	114 Pd Og OpsGenie	115 Pd Pd Pagerduty	116 Os Sn Snort	117 Os Tw Tripwire	118 En Ck CyberArk Conjur	119 En Vc Veracode	120 Os Ff Fortify SCA								

Periodic Table of DevOps Tools (v3) from @Xebialabs <https://xebialabs.com/periodic-table-of-devops-tools>

Keep C.A.L.M.S. and having fun!



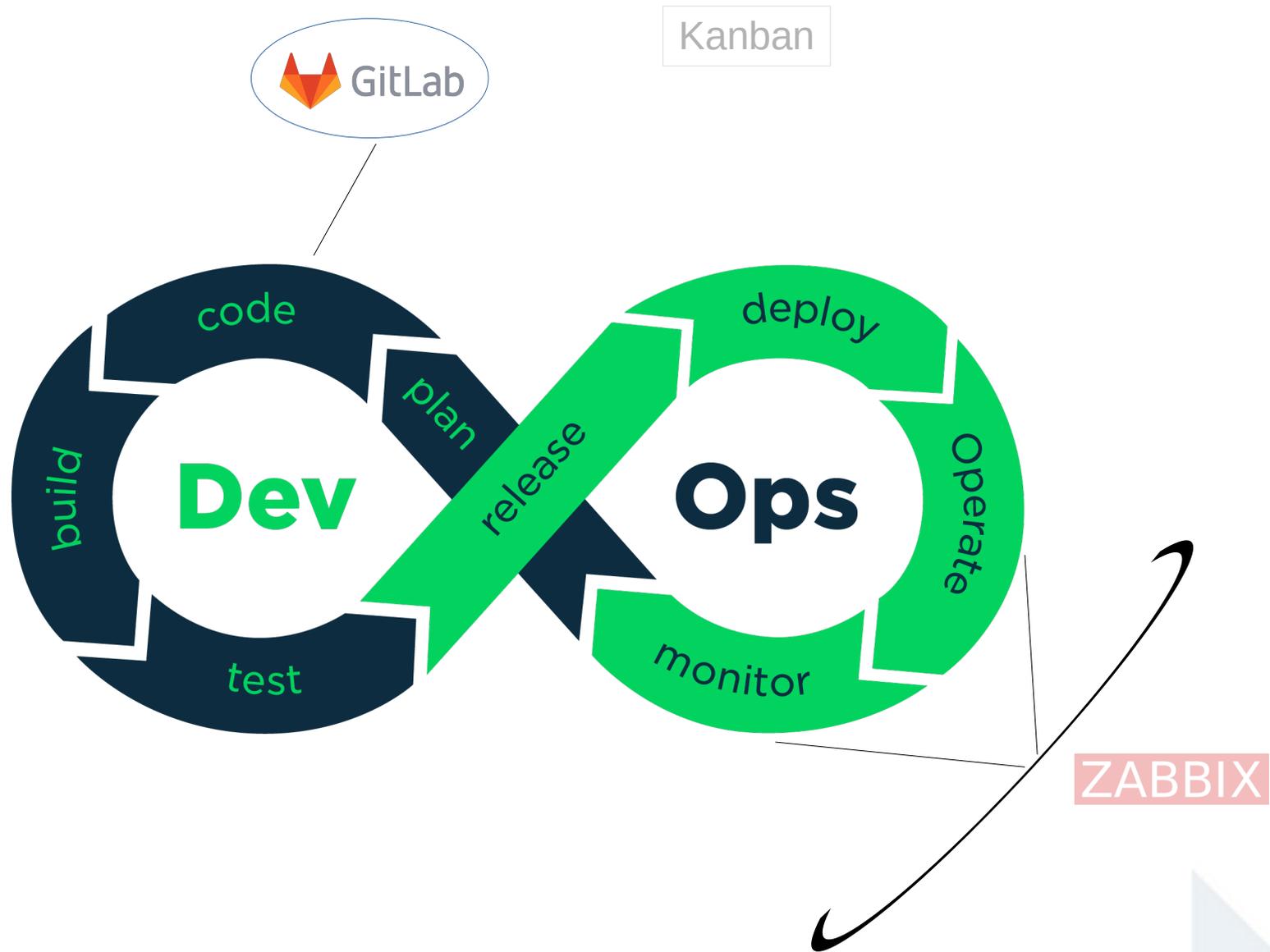
# Tecnologias



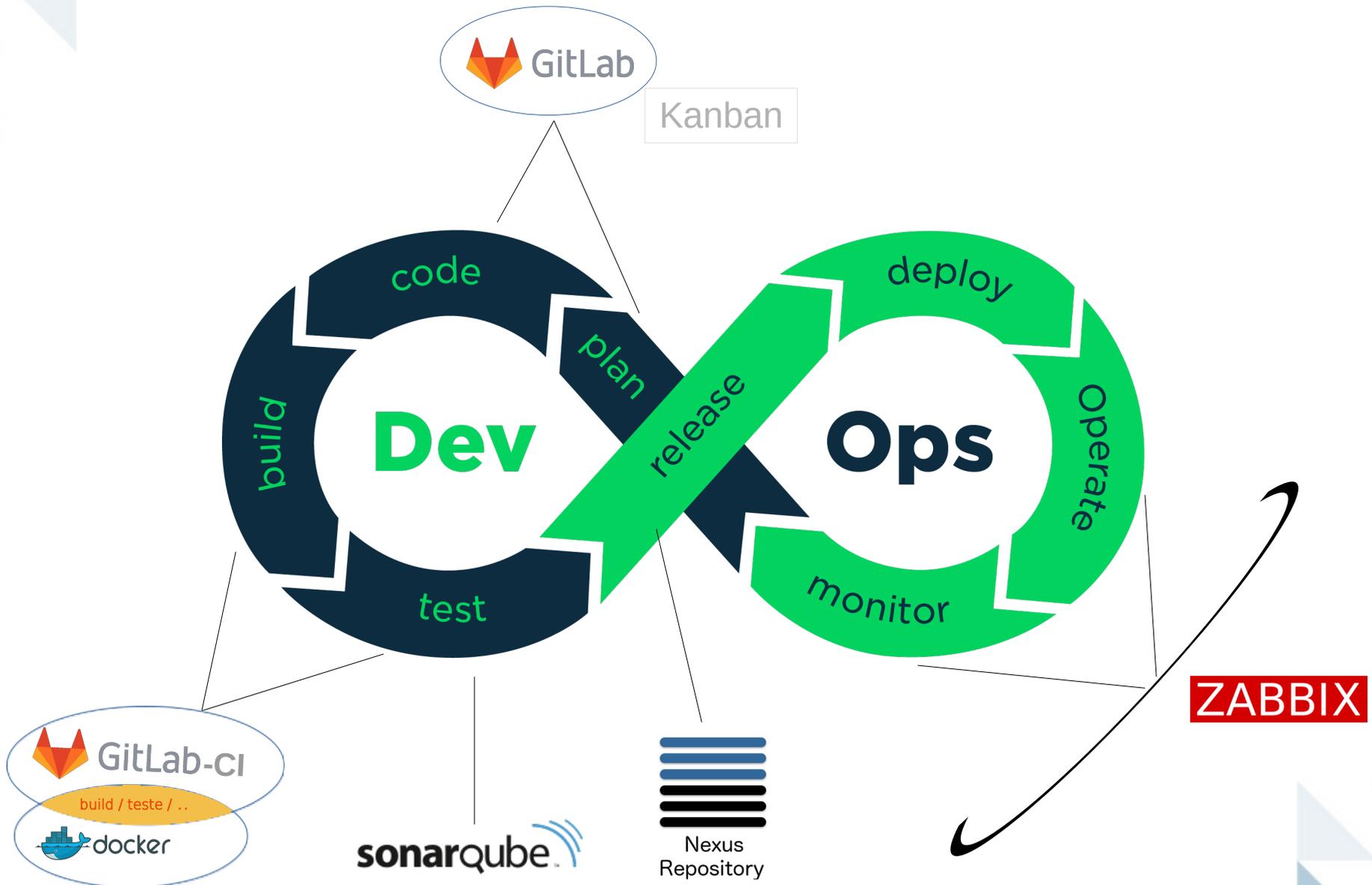
Adote a mais aderente a resolver o problema

Cultura de experimentação & domínio da equipe

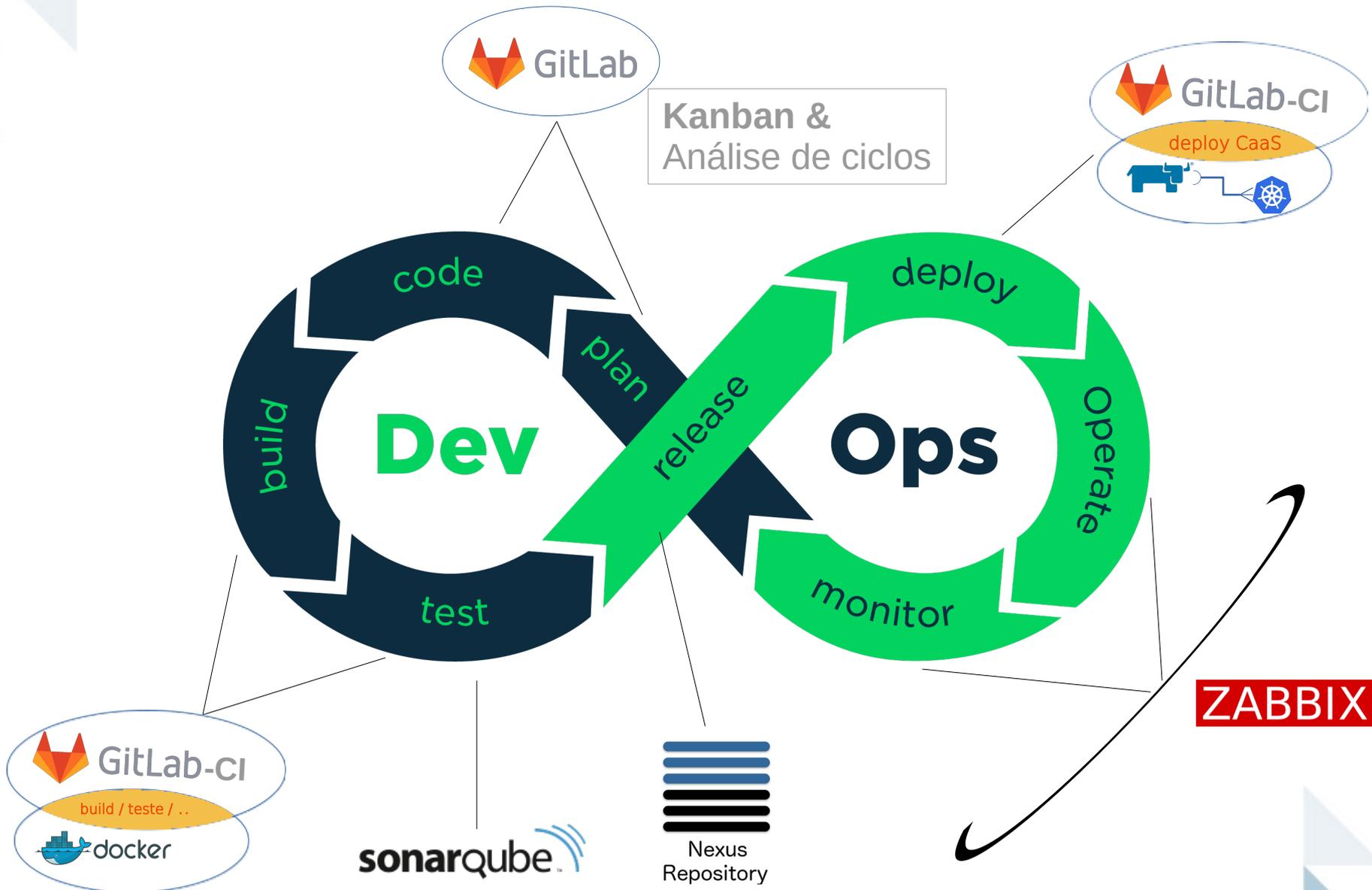
# Tecnologias



# Tecnologias



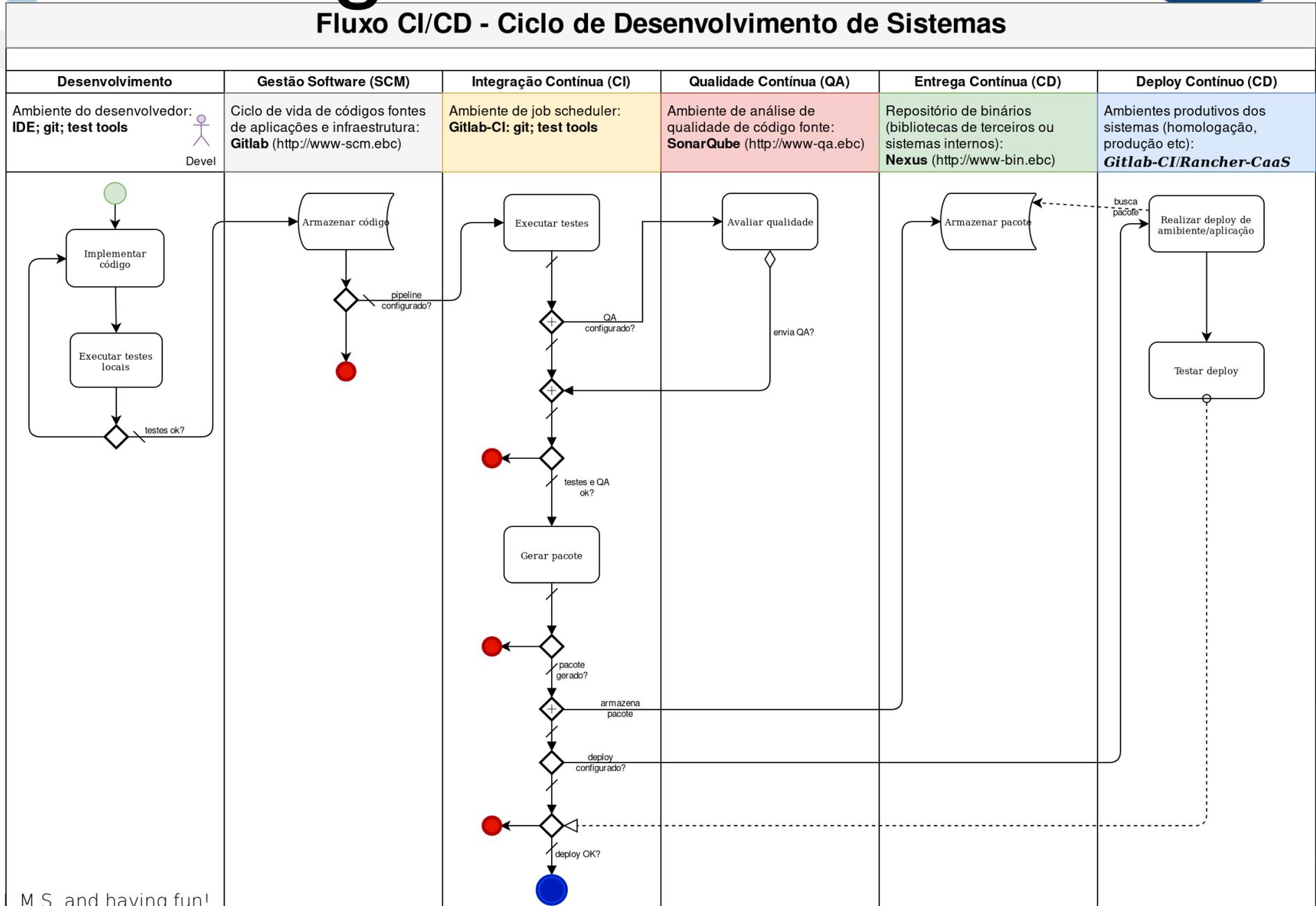
# Tecnologias



# Tecnologias



## Fluxo CI/CD - Ciclo de Desenvolvimento de Sistemas



# Tecnologias



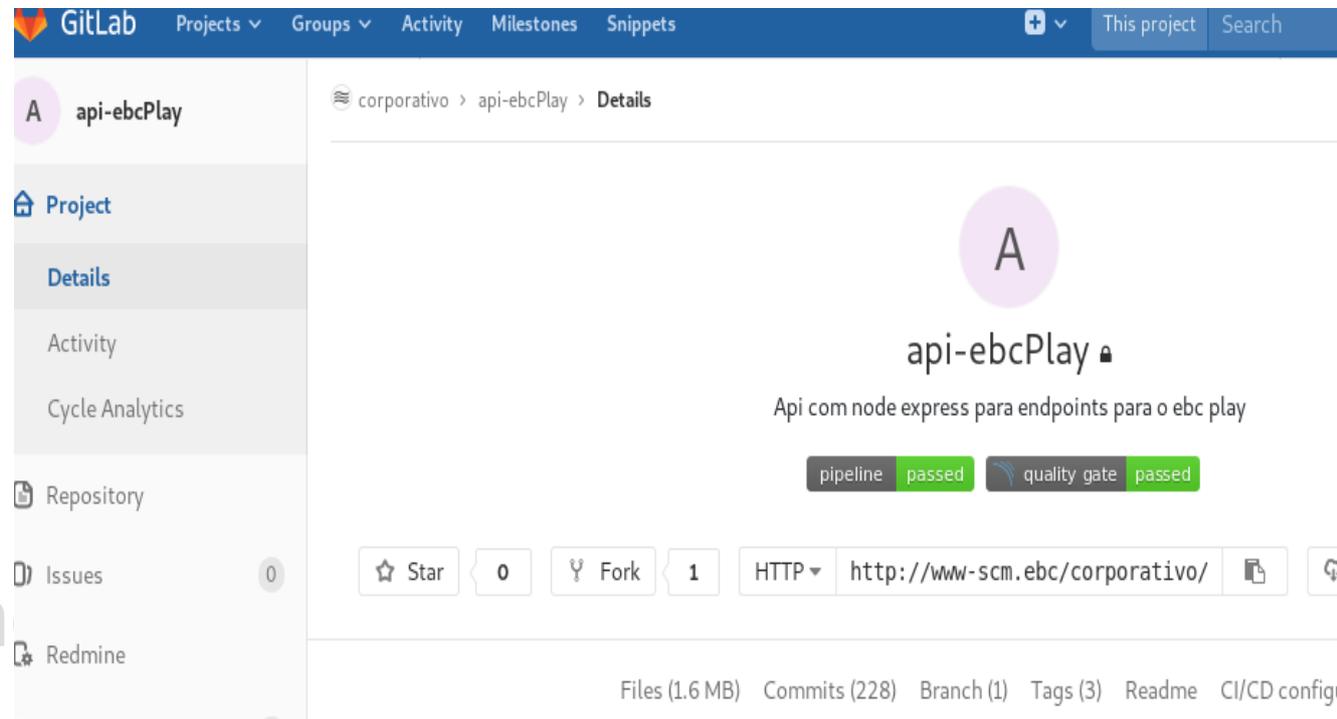
- **Gitlab**  
(<http://www-scm.ebc>)

- **SonarQube**  
(<http://www-qa.ebc>)

- **Nexus**  
(<http://www-bin.ebc>)

- **Gitlab-CI Runn**

- Docker
- PostgreSQL
- Rancher



# Tecnologias



- Gitlab  
(<http://www-scm.ebc>)
- SonarQube  
(<http://www-qa.ebc>)
- Nexus  
(<http://www-bin.ebc>)
- Gitlab-CI Runner
  - Docker
  - PostgreSQL
  - Rancher

The screenshot displays the SonarQube interface for the project 'api-ebcPlay' on the 'master' branch. The top navigation bar includes 'sonarqube', 'Projects', 'Issues', 'Rules', 'Quality Profiles', and 'Quality Gates'. Below the navigation, the project name 'api-ebcPlay' and branch 'master' are shown. The main content area features a 'Quality Gate' section with a green 'Passed' status. Below this, there are two summary cards: 'Bugs' showing 5 (with a 'D' indicator) and 'Vulnerabilities' showing 0 (with an 'A' indicator). Further down, there are two more summary cards: 'Code Smells' showing 4h (with an 'A' indicator) and 'Debt' showing 23. The 'Debt' card also includes the text 'started last month'. At the bottom, there is a 'Coverage' section with a link icon.

# Tecnologias



- Gitlab  
(<http://www-scm.ebc>)
- SonarQube  
(<http://www-qa.ebc>)
- Nexus  
(<http://www-bin.ebc>)
- Gitlab-CI Runner
  - Docker
  - PostgreSQL
  - Rancher

The screenshot displays the Sonatype Nexus Repository Manager interface. The main view shows a tree structure of repositories under 'docker-public', with 'ebcplay-api' selected. The right-hand panel provides a detailed view of the manifest for 'v2/ebcplay-api/manifests/2.0.0-alpha6'.

Summary	
Repository	docker-public
Format	docker
Component Name	ebcplay-api
Component Version	2.0.0-alpha6
Path	<a href="#">v2/ebcplay-api/manifests/2.0.0-alpha6</a>
Content type	application/vnd.docker.distribution.manifest.v2+json
File size	2.4 KB
Blob created	Tue Apr 09 2019 14:51:18 GMT-0300 (Horário Padrão de Brasília)
Blob updated	Tue Apr 09 2019 14:51:18 GMT-0300 (Horário Padrão de Brasília)
Last downloaded	Thu Jun 06 2019
Locally cached	true
Blob reference	docker-private-ebc@1f9d7a9f-6e34227a-3b2b45da-3e1da600-3e7b7d45:360c6c12-aba2-49c4-9362-c30f31da1011
Containing repo	docker-private-ebc
Uploader	gitlab-ci-runner@docker
Uploader's IP Address	10.10.10.10

Usage	
Docker	<code>docker pull ebcplay-api:2.0.0-alpha6</code>

Attributes	
<b>Checksum</b>	
sha1	a5cbf99e9330e44db79df674222bf765fa0861d7
sha256	65cblbdaa5694b7ee7ca0b0dba791bf8d54ce6fc2bd582037077934affb15f2
<b>Content</b>	
last_modified	2019-04-09T14:51:18.344-03:00
<b>Docker</b>	
asset_kind	MANIFEST

# Tecnologias



- Gitlab  
(<http://www-scm.ebc>)
- SonarQube  
(<http://www-qa.ebc>)
- Nexus  
(<http://www-bin.ebc>)
- Gitlab-CI Runner
  - Docker
  - PostgreSQL
  - Rancher

The screenshot displays the GitLab Pipelines page for the 'api-ebcPlay' project. The interface includes a navigation sidebar on the left with options like Project, Repository, Issues, Redmine, Merge Requests, CI/CD, Pipelines, Jobs, Schedules, Charts, Operations, Wiki, and Settings. The main content area shows a summary of pipeline statuses: All 131, Pending 0, Running 0, and Finished 131. Below this is a table of pipeline runs, each with a 'passed' status, a pipeline ID, the user who triggered it, and the commit SHA. The table is as follows:

Status	Pipeline	Commit	Stages
passed	#2964 by [user]	development -> 226b766a	Adiciona exemplo de j... ✓
passed	#2963 by [user]	development -> 2f0240cc	Atualiza documentaçã... ✓
passed	#2962 by [user]	development -> 55823ee5	Adiciona tópico Deplo... ✓
passed	#2961 by [user]	development -> 0cd8a7a7	Merge branch '11-corri... ✓
passed	#2960 by [user]	11-corriger... -> 12552385	Atualiza formatação d... ✓
passed	#2959 by [user]	1.1.0-alpha3 -> b87ba6b0	Corrige valor para titul... ✓✓✓✓

# Tecnologias



- Gitlab  
(<http://www-scm.ebc>)
- SonarQube  
(<http://www-qa.ebc>)
- Nexus  
(<http://www-bin.ebc>)
- **Gitlab-CI Runner**
  - Docker
  - PostgreSQL
  - **Rancher**

The screenshot shows the Rancher management interface. At the top, there's a navigation bar with a blue bull icon, a dropdown menu for 'rke-devel-cl01 Corporativo', and links for 'Workloads', 'Apps', 'Resources', 'Namespaces', and 'Members'. Below this, there's a sub-navigation bar with 'Workloads', 'Load Balancing', 'Service Discovery', 'Volumes', and ' Pipelines'. The main content area shows a list of workloads. The first section is for the 'ebcplay-api' namespace, containing one workload named 'ebcplay-api /v2' which is in an 'Active' state. The second section is for the 'financeiro' namespace, containing four workloads: 'financeiro 80/http', 'financeiro-review-11-manual-hjxaqb 80/http', 'financeiro-review-12-numero-57iak 80/http', and 'financeiro-review-13-cria-jo-8vx65u 80/http', all of which are in an 'Active' state.

# Tecnologias



- Gitlab  
(<http://www-scm.ebc>)

- SonarQube  
(<http://www-qa.ebc>)

- Nexus  
(<http://www-bin.ebc>)

- Gitlab-CI Runner

- Docker
- PostgreSQL
- **Rancher**

The screenshot displays the Rancher management interface for a workload named 'ebcplay-api'. At the top, the navigation bar shows the current namespace as 'rke-devel-cl01 Corporativo' and various menu options like 'Workloads', 'Apps', 'Resources', 'Namespaces', 'Members', and 'Tools'. The workload details section includes:

- Namespace: ebcplay-api
- Image: registry-bin.ebc/ebcplay-api:2.0.0-alpha6
- Workload Type: Daemon Set
- Endpoints: /v2
- Config Scale: 1 per node (with expand/collapse icons)
- Ready Scale: 3
- Created: 05/13/2019

Below the details, there is a 'Pods' section and a 'Workload Metrics' section. The 'Workload Metrics' section is currently expanded to show live metrics from Grafana. The metrics are displayed in four panels:

- CPU Utilization:** A line chart showing CPU usage in mCPU over time. The y-axis ranges from 0 to 2.1 mCPU. The x-axis shows timestamps from 15:41:07 to 15:45:04.
- Memory Utilization:** A line chart showing memory usage in MiB over time. The y-axis ranges from 0 to 80 MiB. The x-axis shows the same timestamps.
- Network Packets:** A line chart showing network traffic in Pps (Packets per second) over time. The y-axis ranges from 0 to 2.5 Pps. The x-axis shows the same timestamps.
- Network I/O and Disk I/O:** These sections are partially visible at the bottom of the screenshot.

Overlaid on the metrics section is the text 'Day 2 DevOps' in large orange font. A refresh button and a time range selector set to '5 minutes' are also visible.

# Tecnologias



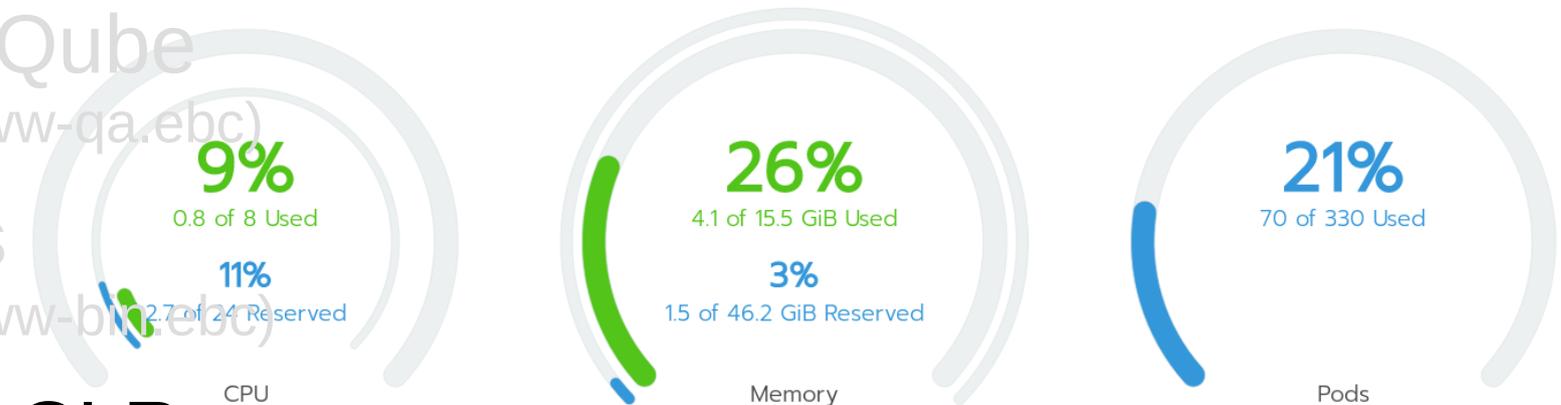
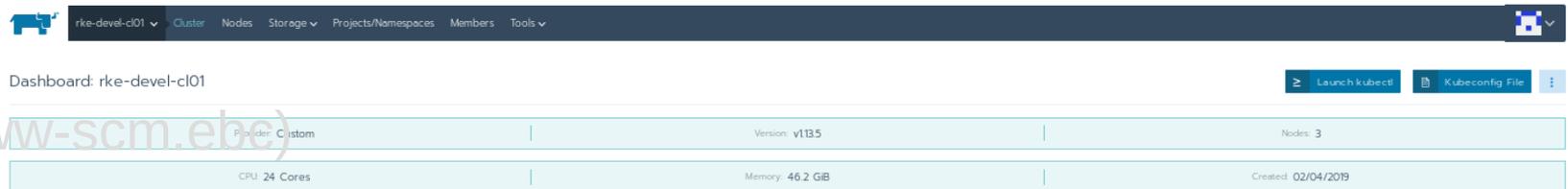
- Gitlab  
(<http://www-scm.ebc>)

- SonarQube  
(<http://www-qa.ebc>)

- Nexus  
(<http://www-bin.ebc>)

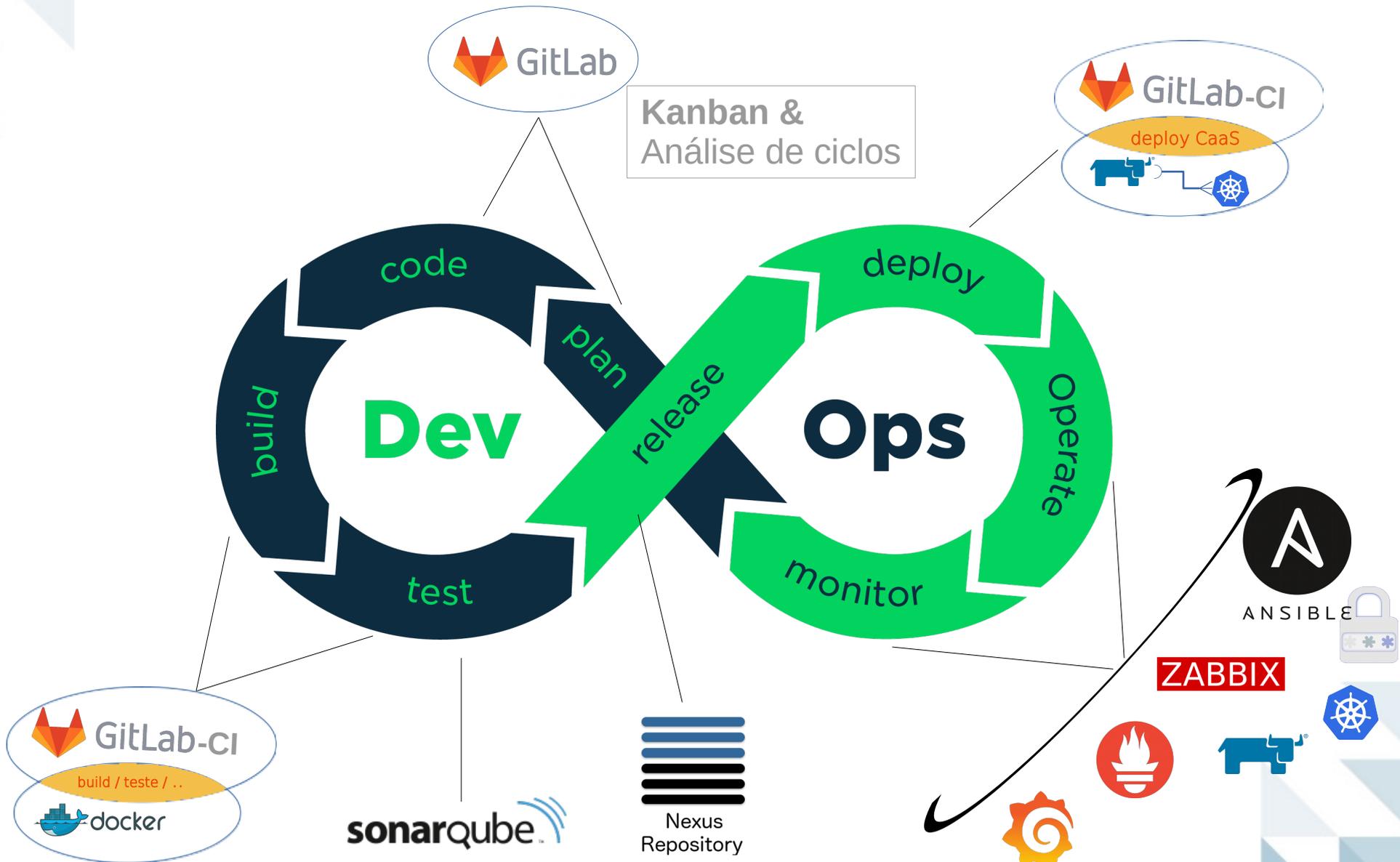
- Gitlab-CI Runner

- Docker
- PostgreSQL
- Rancher



Day 2 DevOps

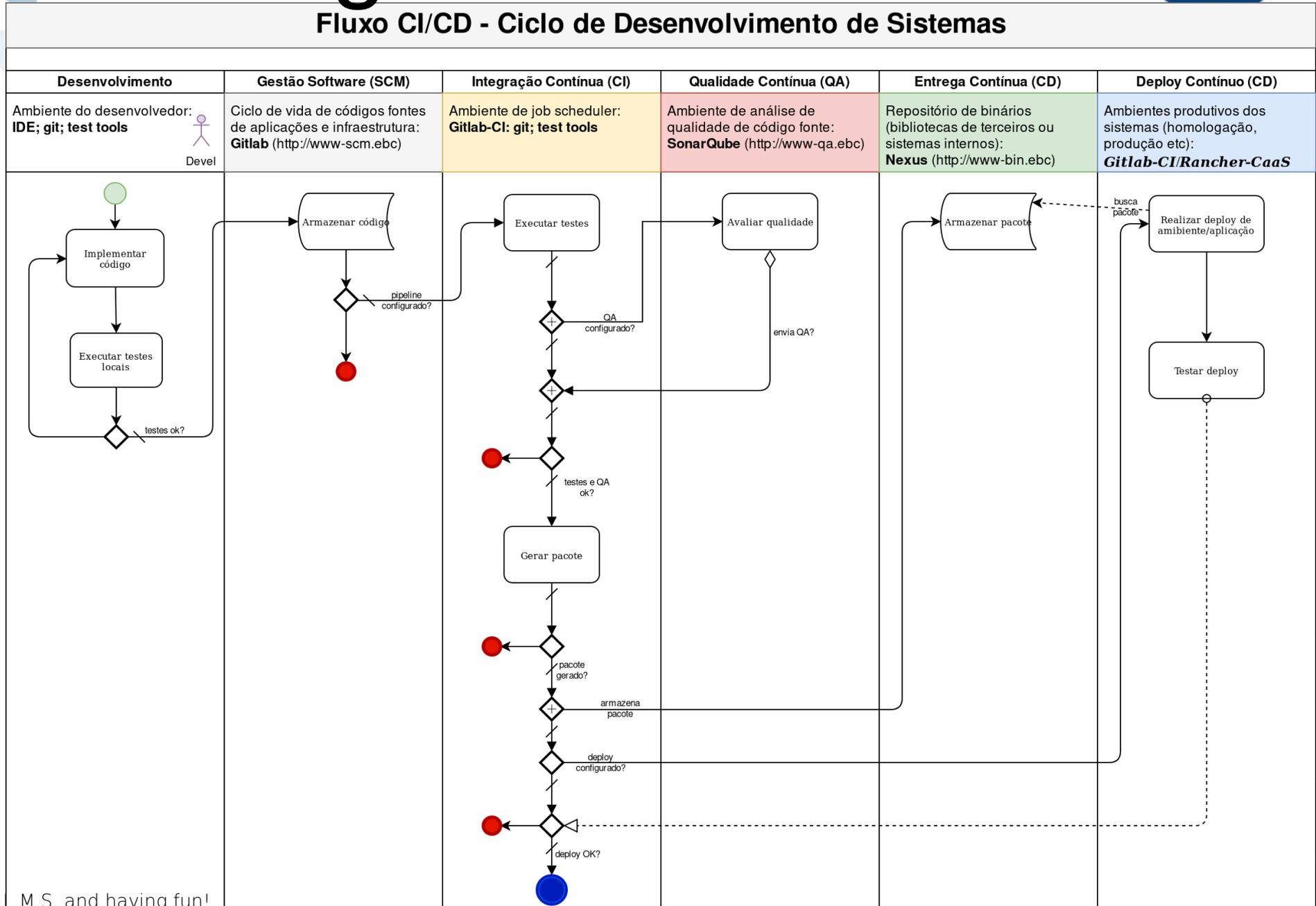
# Tecnologias



# Tecnologias



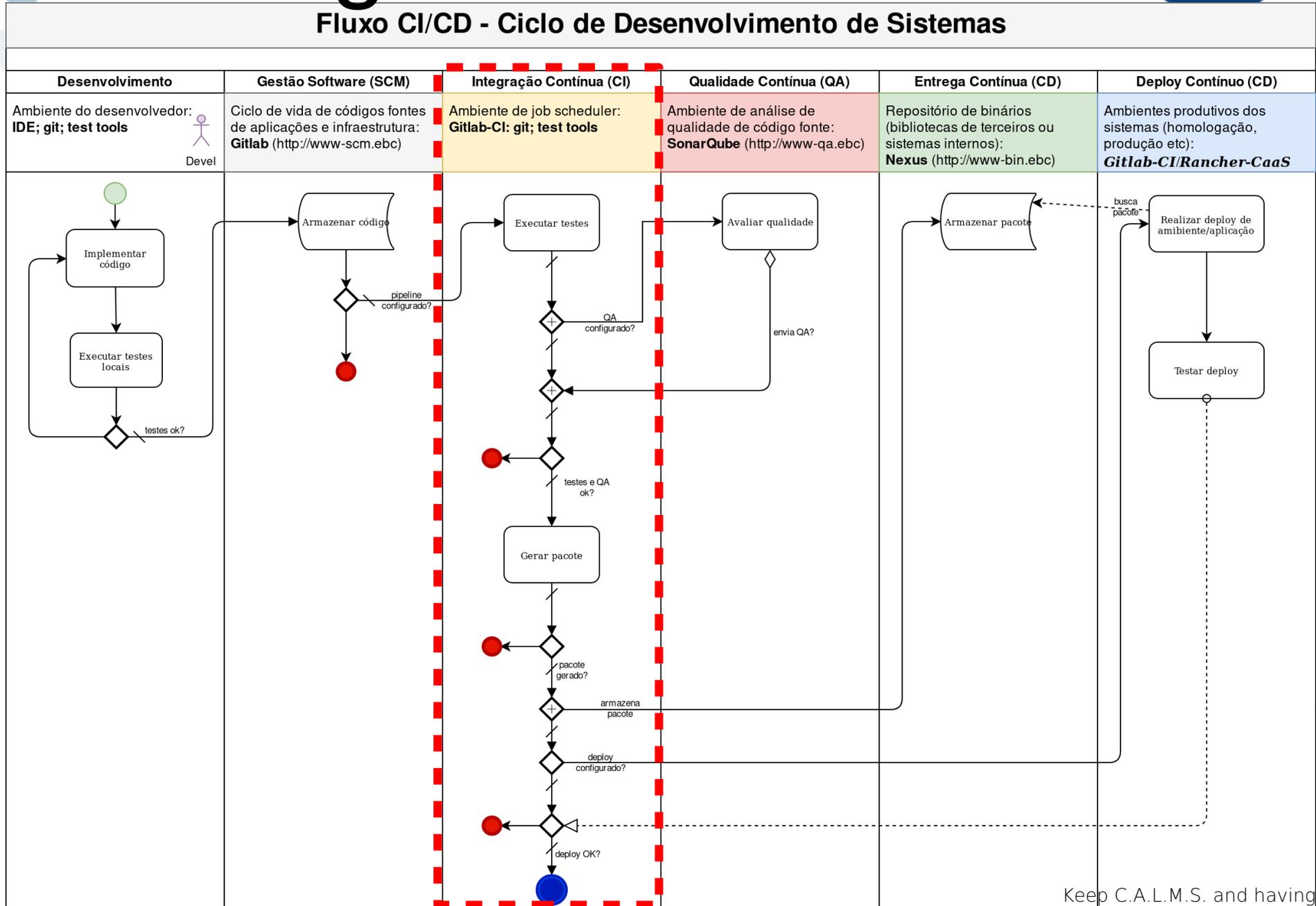
## Fluxo CI/CD - Ciclo de Desenvolvimento de Sistemas



# Tecnologias



## Fluxo CI/CD - Ciclo de Desenvolvimento de Sistemas



# Fluxo CI/CD aplicado



- DSL do pipeline como código

→ .gitlab-ci.yml + docker

- DSL deploy (rancher + k8s)

→ deployment.yml

+ service.yml

+ loadbalancing.yml

```
1 image: registry.bin.ebc/ebc-nodejs:8.10-slim-ebc-3.1
2
3 services:
4   - name: registry.bin.ebc/ebc-postgres:9.6-alpine-ebc-1.0
5     alias: dind-pgsrvr-runner
6
7 variables:
8   # Configura postgres service (https://hub.docker.com/ /postgres/)
9   # Estes valores usados somente para o aprovisionar o serviço aqui na esteira
10  POSTGRES_DB: db_prateleira
11  POSTGRES_USER: prateleira
12  POSTGRES_PASSWORD: prateleira
13  PGPASSWORD: $(POSTGRES_PASSWORD)
14  PORT: 5555
15  URL: http://localhost
16  DATABASE_HOST: dind-pgsrvr-runner
17  DATABASE_NAME: $(POSTGRES_DB)
18  DATABASE_USERNAME: $(POSTGRES_USER)
19  DATABASE_PASSWORD: $(POSTGRES_PASSWORD)
20  DATABASE_URL: postgres://$(POSTGRES_USER):$(POSTGRES_PASSWORD)@$(DATABASE_HOST)
21
22 stages:
23   - build test
24   - code QA
25   - release
26   - deploy
27
28 build project test:
29   stage: build test
30   before_script:
31     - mkdir -p logs
32     - npm install && logs/npm-install.log
33     - psql -h "dind-pgsrvr-runner" -U "POSTGRES_USER" -d "POSTGRES_DB" < test/sql
34     - ./node_modules/forever/bin/forever start -p ./logs -d -v ./bin/server.js
35   script:
36     - ./node_modules/forever/bin/forever list
37     - npm test
38     - ./node_modules/forever/bin/forever stop ./bin/server.js
39   artifacts:
40     when: always # on_failure | on_success
41     paths:
42     - logs/*.log
43     expire_in: 1 day
44   tags:
45     - nodejs
46
47 SonarQube Continuous Code Quality:
48   stage: code QA
49   before_script:
50     - sed -e "/0.0.0-VERSION/${CI_COMMIT_REF_NAME}/" -i package.json
51   script:
52     - sonar-scanner
53     -sonar.host.url=${EBC_SONARQUBE_QA_URL}
54     -sonar.projectKey=${PROJECT_SONAR_KEY}
55     -sonar.login=${PROJECT_SONAR_TOKEN}
56     -sonar.sources=.
57   only:
58     - /^(?:[v\d+].)?(?:[v\d+].)?(?:[v\d+].)-alpha\d+$/
59     - /^(?:[v\d+].)?(?:[v\d+].)?(?:[v\d+].)-beta\d+$/
60     - /^(?:[v\d+].)?(?:[v\d+].)?(?:[v\d+].)-RC\d+$/
61     - /^(?:[v\d+].)?(?:[v\d+].)?(?:[v\d+].)$/
62   tags:
63     - nodejs
64
65 npm publish release:
66   stage: release
67   before_script:
68     - sed -e "/0.0.0-VERSION/${CI_COMMIT_REF_NAME}/" -i package.json
69     - npm config set registry $(EBC_NEXUS_BIN_NPM_REGISTRY)
70     - npm config set init.author.name $(GITLAB_USER_LOGIN)
71     - npm config set init.author.email $(GITLAB_USER_EMAIL)
72     - npm config set email $(GITLAB_USER_EMAIL)
73     - npm config set always-auth=true
74     - npm config set_auth $(echo -n "${EBC_NEXUS_BIN_DEPLOYMENT_USERNAME}:${EBC_NEXUS_BIN_DEPLOYMENT_PASSWORD}@bin.ebc.com")
75   script:
76     - npm publish
77   only:
78     - /^(?:[v\d+].)?(?:[v\d+].)?(?:[v\d+].)-alpha\d+$/
79     - /^(?:[v\d+].)?(?:[v\d+].)?(?:[v\d+].)-beta\d+$/
80     - /^(?:[v\d+].)?(?:[v\d+].)?(?:[v\d+].)-RC\d+$/
81     - /^(?:[v\d+].)?(?:[v\d+].)?(?:[v\d+].)$/
82   except:
83     - master
84
85 ansible deploy desenvolvimento:
86   image: registry.bin.ebc/ebc-ansible:2.7-ebc-1.0
87   stage: deploy
88   variables:
89     ANSIBLE_HOST_KEY_CHECKING: "False"
90   before_script:
91     - sed -e "/0.0.0-VERSION/${CI_COMMIT_REF_NAME}/" -i package.json
92     - ansible-playbook -syntax-check deploy/ansible/playbook.yml
93     - ansible all -m ping -i deploy/ansible/inventory/desenvolvimento.yml --become --extra-vars "${PROJECT_DEVELOP_ANSIBLE_EXTRAVARS}"
94   script:
95     - ansible-playbook deploy/ansible/playbook.yml
96     -i deploy/ansible/inventory/desenvolvimento.yml
97     --become
98     --extra-vars "${PROJECT_DEVELOP_ANSIBLE_EXTRAVARS}"
99   environment:
100    name: desenvolvimento
101   only:
102     # somente branches ou tags (ex: 1.0-0-alpha)
103     - /^(?:[v\d+].)?(?:[v\d+].)?(?:[v\d+].)-alpha\d+$/
```

# Fluxo CI/CD aplicado



- DSL do pipeline como código  
  `.gitlab-ci.yml` + docker

```
image: registry-bin.ebc/ebc-nodejs:8.10-slim-ebc-3.1

services:
  - name: registry-bin.ebc/ebc-postgres:9.6-alpine-ebc-1.0
    alias: dind-pgserver.runner

variables:
  # Configura postgres service (https://hub.docker.com/_/postgres/)
  # Estes valores usados somente para o aprovisionar o serviço aqui na
  POSTGRES_DB: db_prateleira
  POSTGRES_USER: prateleira
  POSTGRES_PASSWORD: prateleira
  PGPASSWORD: ${POSTGRES_PASSWORD}
  PORT: 5555
  URL: http://localhost
  DATABASE_HOST: dind-pgserver.runner
  DATABASE_NAME: ${POSTGRES_DB}
  DATABASE_USERNAME: ${POSTGRES_USER}
  DATABASE_PASSWORD: ${POSTGRES_PASSWORD}
  DATABASE_URL: postgresql://${POSTGRES_USER}:${POSTGRES_PASSWORD}@${

stages:
  - build_test
  - code QA
  - release
  - deploy
```

```
build_project_test:
  stage: build_test
  before_script:
    - mkdir -p logs
    - npm install &> logs/npm-install.log
    - psql -h "dind-pgserver.runner" -U "$POSTGRES_USER" -d "$POSTGRES_DB" < test/sql/massadetest
    - ./node_modules/forever/bin/forever start -p ./logs -d -v ./bin/server.js
  script:
    - ./node_modules/forever/bin/forever list
    - npm test
    - ./node_modules/forever/bin/forever stop ./bin/server.js
  artifacts:
    when: always # on_failure | on_success
    paths:
      - logs/*.log
    expire_in: 1 day
  tags:
    - nodejs

SonarQube Continuous Code Quality:
  stage: code QA
  before_script:
    - sed -e "s/0.0.0-VERSION/${CI_COMMIT_REF_NAME}/" -i package.json
  script:
    - sonar-scanner
      -Dsonar.host.url=${EBC_SONARQUBE_QA_URL}
      -Dsonar.projectKey=${PROJECT_SONAR_KEY}
      -Dsonar.login=${PROJECT_SONAR_TOKEN}
      -Dsonar.sources=.
  only:
    - /^(?:\d+)\.?(?:\d+)\.?(.*)-alpha\d+$/
    - /^(?:\d+)\.?(?:\d+)\.?(.*)-beta\d+$/
    - /^(?:\d+)\.?(?:\d+)\.?(.*)-RC\d+$/
    - /^(?:\d+)\.?(?:\d+)\.?(.*)$/
  tags:
    - nodejs
```



# Fluxo CI/CD aplicado



- DSL do pipeline como código  
    `.gitlab-ci.yml + docker`

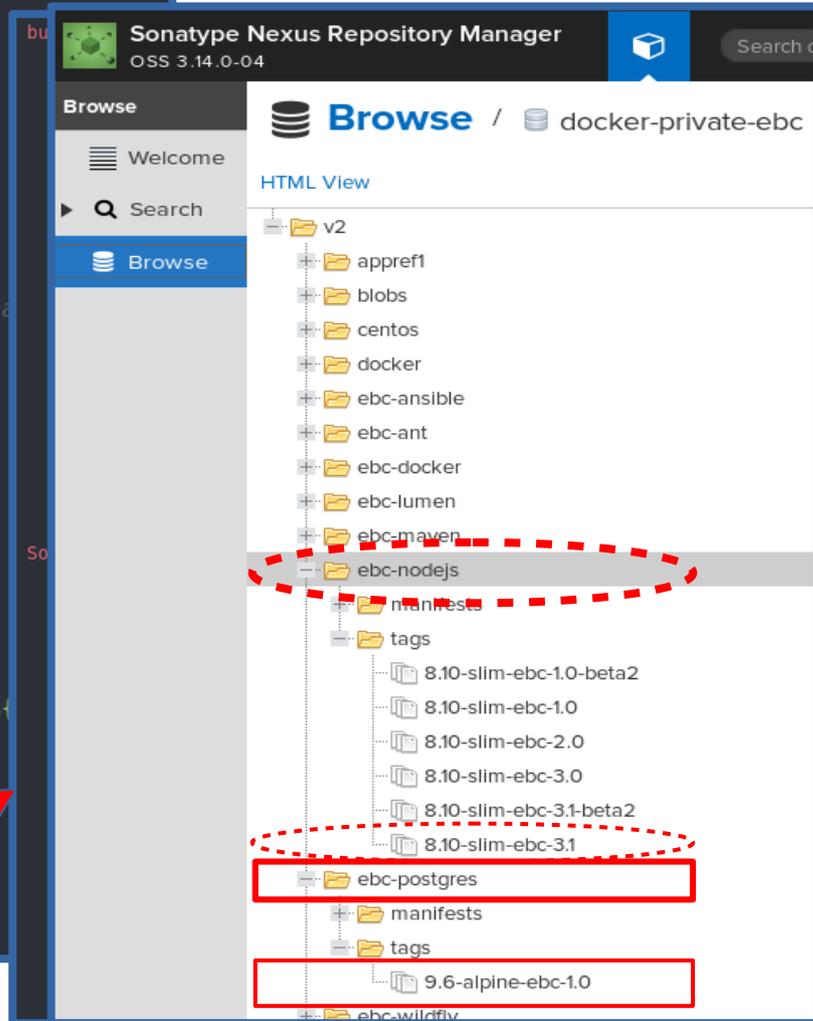
```
image: registry-bin.ebc/ebc-nodejs:8.10-slim-ebc-3.1

services:
- name: registry-bin.ebc/ebc-postgres:9.6-alpine-ebc-1.0
  alias: dind-pgserver.runner

variables:
# Configura postgres service (https://hub.docker.com/_/postgres/)
# Estes valores usados somente para o aprovisionar o serviço aqui na
POSTGRES_DB: db_prateleira
POSTGRES_USER: prateleira
POSTGRES_PASSWORD: prateleira
PGPASSWORD: ${POSTGRES_PASSWORD}
PORT: 5555
URL: http://localhost
DATABASE_HOST: dind-pgserver.runner
DATABASE_NAME: ${POSTGRES_DB}
DATABASE_USERNAME: ${POSTGRES_USER}
DATABASE_PASSWORD: ${POSTGRES_PASSWORD}
DATABASE_URL: postgresql://${POSTGRES_USER}:${POSTGRES_PASSWORD}@${

stages:
- build_test
- code QA
- release
- deploy
```

em tempo de CI,  
instancia serviço de  
BD em docker



```
3" < test/sql/massadetest
ver.js
```



# Fluxo CI/CD aplicado



## Testes: Execução

```
$ npm test

> api-ebcPlay@0.0.0-VERSION test /builds/corporativo/api-ebcPlay
> ./node_modules/mocha/bin/mocha

Teste API play.etc.com.br
  ✓ Deve retornar ao menos 10 conteudos (542ms)
  ✓ Deve retornar pelo menos 10 conteudos usando o termo (1685ms)
  ✓ Retornar os conteudos do tema de ID 3(Entretenimento) - Deve receber ao menos 3 conteudo
  ✓ consultar um conteudo especifico
  ✓ endpoint /populares deve retornar pelo menos 10 conteudos (436ms)
  ✓ endpoint /priorizados deve retornar pelo menos 1 conteudo (93ms)
  ✓ endpoint /conteudos/id/episodios deve retornar pelo menos 3 episódios
nr_segundos_exibidos 6331
st_exibicao 2
  ✓ fazer um post de conteudos em exibicao (39ms)

Teste API play.etc.com.br /streaming
  ✓ Endpoint streaming deve responder status 200
  ✓ Endpoint streaming deve retornar pelo menos um conteudo
  ✓ Streaming sort nm_plataforma e order desc nome da plataforma e TV Brasil

11 passing (3s)

$ ./node_modules/forever/bin/forever stop ./bin/server.js
info: Forever stopped process:
  uid  command  script  forever pid id logfile  uptime
[0] sNX5 /usr/local/bin/node bin/server.js 56 68 logs/sNX5.log 0:0:0:5.284

Uploading artifacts...
logs/*.log: found 3 matching files
Uploading artifacts to coordinator... ok id=3707 responseStatus=201 Created token=piksFzKy
Job succeeded
```

```
1 image: registry-bin.etc/ebc-nodejs:0.10-slim-etc-3.1
2
3 services:
4 - name: registry-bin.etc/ebc-postgres:9.6-alpine-etc-1.0
5 alias: dind-pgsrvar.runner
6
7 variables:
8 # Configura postgres service (https://hub.docker.com/postgres/)
9 # Estes valores usados somente para o aprovisionar o serviço aqui na estera
10 POSTGRES_DB: db_prateleira
11 POSTGRES_USER: prateleira
12 POSTGRES_PASSWORD: prateleira
13 PGPASSWORD: $(POSTGRES_PASSWORD)
14 PORT: 5555
15 URL: http://localhost
16 DATABASE_HOST: dind-pgsrvar.runner
17 DATABASE_NAME: $(POSTGRES_DB)
18 DATABASE_USERNAME: $(POSTGRES_USER)
19 DATABASE_PASSWORD: $(POSTGRES_PASSWORD)
20 DATABASE_URL: postgres://$(POSTGRES_USER):$(POSTGRES_PASSWORD)@$(DATABASE_HOST)
21
22 stages:
23 - build test
24 - code QA
25 - release
26 - deploy
27
28 build project test:
29 stage: build test
30 before_script:
31 - mkdir -p logs
32 - npm install --log-level=warn
33 - psql -h "dind-pgsrvar.runner" -U "$POSTGRES_USER" -d "$POSTGRES_DB" < test/sql
34 - ./node_modules/forever/bin/forever start -p ./logs -d -v ./bin/server.js
35 script:
36 - ./node_modules/forever/bin/forever list
37 - npm test
38 - ./node_modules/forever/bin/forever stop ./bin/server.js
39 artifacts:
40 when: always # on_failure | on_success
41 paths:
42 - logs/*.log
43 expire_in: 1 day
44 tags:
45 - nodejs
46
47 SonarQube Continuous Code Quality:
48 stage: code QA
49 before_script:
50 - sed -e "/0.0.0-VERSION/s(CI_COMMIT_REF_NAME)/" -i package.json
51 script:
52 - sonar-scanner
53 -Dsonar.host.url=${EBC_SONARQUBE_QA_URL}
54 -Dsonar.projectKey=${PROJECT_SONAR_KEY}
55 -Dsonar.login=${PROJECT_SONAR_TOKEN}
56 -Dsonar.sources=
57 only:
58 - /*!(vde)\.?(?!(vde)\.?(?!vde)-alpha)/vde/
59 - /*!(vde)\.?(?!(vde)\.?(?!vde)-beta)/vde/
60 - /*!(vde)\.?(?!(vde)\.?(?!vde)-RC)/vde/
61 - /*!(vde)\.?(?!(vde)\.?(?!vde)s/
62 tags:
63 - nodejs
64
65 npm publish release:
66 stage: release
67 before_script:
68 - sed -e "/0.0.0-VERSION/s(CI_COMMIT_REF_NAME)/" -i package.json
69 - npm config set registry $(EBC_NEXUS_BIN_NPM_REGISTRY)
70 - npm config set init.author.name $(GITLAB_USER_LOGIN)
71 - npm config set init.author.email $(GITLAB_USER_EMAIL)
72 - npm config set email $(GITLAB_USER_EMAIL)
73 - npm config set always-auth=true
74 - npm config set_auth $(echo -n "${EBC_NEXUS_BIN_DEPLOYMENT_USERNAME}:${EBC_NEXUS_BIN_DEPLOYMENT_PASSWORD}")
75 script:
76 - npm publish
77 only:
78 - /*!(vde)\.?(?!(vde)\.?(?!vde)-alpha)/vde/
79 - /*!(vde)\.?(?!(vde)\.?(?!vde)-beta)/vde/
80 - /*!(vde)\.?(?!(vde)\.?(?!vde)-RC)/vde/
81 - /*!(vde)\.?(?!(vde)\.?(?!vde)s/
82 except:
83 - master
84
85 ansible deploy desenvolvimento:
86 image: registry-bin.etc/ebc-ansible:2.7-etc-1.0
87 stage: deploy
88 variables:
89 ANSIBLE_HOST_KEY_CHECKING: "False"
90 before_script:
91 - sed -e "/0.0.0-VERSION/s(CI_COMMIT_REF_NAME)/" -i package.json
92 - ansible-playbook --syntax-check deploy/ansible/playbook.yml
93 - ansible all -m ping -i deploy/ansible/inventory/desenvolvimento.yml --become --extra-vars "${PROJECT_DEVELOP_ANSIBLE_EXTRAVARS}"
94 script:
95 - ansible-playbook deploy/ansible/playbook.yml
96 -i deploy/ansible/inventory/desenvolvimento.yml
97 --become
98 --extra-vars "${PROJECT_DEVELOP_ANSIBLE_EXTRAVARS}"
99 environment:
100 name: desenvolvimento
101 only:
102 # somente branches ou tags (ex: 1.0.0-alpha)
103 - /*!(vde)\.?(?!(vde)\.?(?!vde)-alpha)/vde/
```

# Fluxo CI/CD aplicado



- DSL deploy (`.gitlab-ci.yml`)

(Rancher+K8s) `deployment.yml` + `service.yml` + `ingress-lb.yml`

```
deploy::review da aplicação no caas:
  stage: deploy
  image: registry-bin.ebc/ebc-rkecli:3.0
  before_script:
    - rancher login --skip-verify
      --token ${EBC_CAAS_TOKEN}
      --context ${PROJECT_CAAS_DEVEL_CONTEXT}
      ${EBC_CAAS_URL}

    - rancher kubectl delete --ignore-not-found=true --namespace ebcplay-api
      deployment ebcplay-api-${CI_ENVIRONMENT_SLUG}
  script:
    - rancher kubectl apply --namespace ebcplay-api --filename deploy/k8s/deployment.yml
    - rancher kubectl apply --namespace ebcplay-api --filename deploy/k8s/service.yml
    - rancher kubectl apply --namespace ebcplay-api --filename deploy/k8s/loadbalancing.yml
  environment:
    name: review/${CI_COMMIT_REF_NAME}
    url: http://ebcplay-api-${CI_ENVIRONMENT_SLUG}.local
  only:
    - branches
  except:
    - master
    - tags
```

```
deploy::staging da api no caas:
  stage: deploy
  image: registry-bin.ebc/ebc-rkecli:1.0.0
  environment:
    name: staging
    url: http://api.local/v2
  before_script:
    - rancher login --skip-verify --token ${EBC_CAAS_TOKEN}
      --context ${PROJECT_CAAS_CONTEXT} ${EBC_CAAS_URL}
  script:
    - rancher kubectl apply --namespace ebcplay-api
      --filename deploy/k8s/deployment.yml
  only:
    # somente branches ou tags (ex: 1.0.0-alpha1)
    - /^(?:(\d+)\.)*?(?:(\d+)\.)*?(\*|\d+)-alpha\d+$/
```

# Fluxo CI/CD aplicado



- DSL deploy (`.gitlab-ci.yml`)

(Rancher+K8s) `deployment.yml` + `service.yml` + `ingress-lb.yml`

```
---
apiVersion: apps/v1
kind: Deployment
metadata:
  name: "ebcplay-api-{{CI_ENVIRONMENT_SLUG}}"
  labels:
    app: "ebcplay-api-{{CI_ENVIRONMENT_SLUG}}"
spec:
  replicas: 1
  selector:
    matchLabels:
      app: "ebcplay-api-{{CI_ENVIRONMENT_SLUG}}"
  template:
    metadata:
      labels:
        app: "ebcplay-api-{{CI_ENVIRONMENT_SLUG}}"
    spec:
      containers:
        - name: "ebcplay-api-{{CI_ENVIRONMENT_SLUG}}"
          image: registry-bin.ebc/ebcplay-api:{{CI_COMMIT_REF_NAME}}
          imagePullPolicy: Always
          env:
            - name: APP_DEBUG
              value: "true"
            - name: APP_ENV
              value: "local"
          envFrom:
            - secretRef:
                name: ebcplay-api
                optional: false
          imagePullSecrets:
            - name: registry-bin-ebc
```

```
---
apiVersion: v1
kind: Service
metadata:
  name: "ebcplay-api-{{CI_ENVIRONMENT_SLUG}}"
  labels:
    run: "ebcplay-api-{{CI_ENVIRONMENT_SLUG}}"
spec:
  type: ClusterIP
  ports:
    - port: 80
      protocol: TCP
      targetPort: 80
  selector:
    app: "ebcplay-api-{{CI_ENVIRONMENT_SLUG}}"
```

```
apiVersion: extensions/v1beta1
kind: Ingress
metadata:
  name: "ebcplay-api-{{CI_ENVIRONMENT_SLUG}}"
  annotations:
    nginx.ingress.kubernetes.io/rewrite-target: /
spec:
  rules:
    - host: "ebcplay-api-{{CI_ENVIRONMENT_SLUG}}-{{DOMAIN_NAME}}"
      http:
        paths:
          - backend:
              serviceName: "ebcplay-api-{{CI_ENVIRONMENT_SLUG}}"
              servicePort: 80
      tls:
        - secretName: ebcplay-api-cert-tls
```

# Desafios



Identificar cadeia de valor

Testes incipientes e manuais

Domínio da infraestrutura

Deploy dia(s)

Monitoramento de serviços baixa maturidade

# Desafios



Entender a cadeia de valor

Testes automatizados dev\*. \*ops

Domínio da infraestrutura

n Deploy/dia

Monitoramento de serviços alta maturidade

# Resultados



- agilidade na entrega: ↑
  - de: dia(s) ↑ para: < 1h ↓
- maturidade de testes ↑
  - de: manual ↓ para: automatizado ↑
- entregas com mais qualidade ↑
  - Débito técnico: < 5%

# Oportunidades



- Aprimorar cobertura de testes
- Ampliar uso da esteira CI/CD para mais projetos
- Aumentar a qualidade do código
- Amplificar monitoramento de comportamento de aplicações e ambientes

*“Eficiência & Agilidade” (~5min),*  
<https://www.youtube.com/watch?v=u00S-hCnmFY>

*Fome de poder / The Founder*



Empresa Brasil  
de Comunicação

Smart Risk  
 Cadeia de valor Burnout syndrome  
 Débito técnico Automação Pipeline CI/CD  
 Testes APM Melhoria Contínua Docker  
 Agilidade DevSecOps KPI PDCA

# Perguntas?

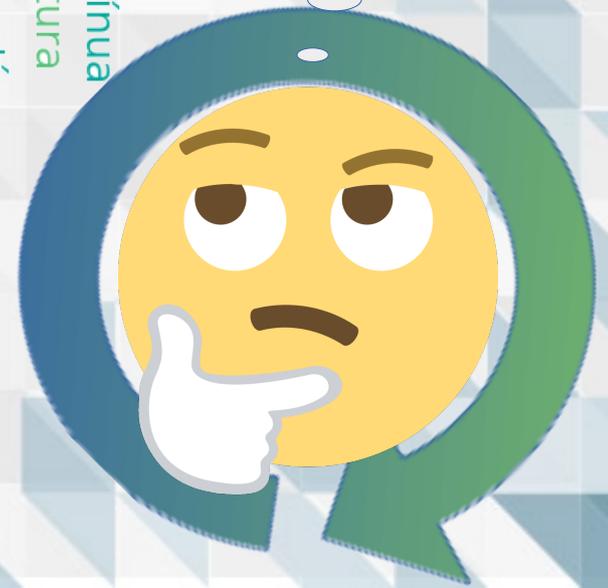
Processos  
 Colaboração  
 DevOps  
 Pessoas  
 Confiança  
 Kubernetes  
 Kata  
 Gitlab  
 Métricas  
 Integração Contínua  
 Day 2 DevOps  
 Liderança  
 Feedback  
 Lean  
 Kaban  
 Ansible  
 Security  
 Lean IT

Observability  
 Experimentação  
 Cultura organizacional  
 Tecnologia

Continuous learning

Qualidade Contínua  
 Infraestrutura  
 Entrega Contínua

Estou *Preparado* para o  
*Futuro do Trabalho?*

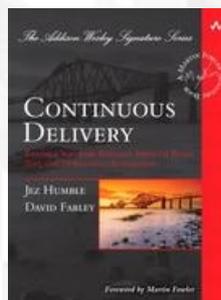
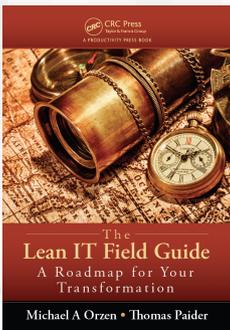
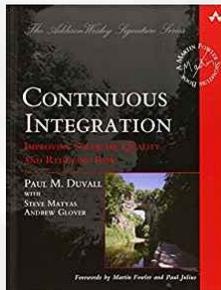
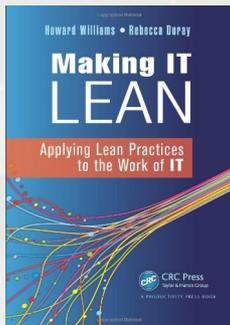




Empresa Brasil  
de Comunicação

# Referências

- Accelerate. Nicole Forsgren, Gene Kim, Jez Humble. 2018
- Continuous Delivery: Reliable Software Releases through Build, Test, and Deployment Automation. Jez Humble, David Farley. 2010
- Continuous Integration: Improving Software Quality and Reducing Risk. Andrew Glover, Steve Matyas, Paul M. Duvall. 2007
- The Lean IT Field Guide. Thomas A. Paider, Michael A. Orzen. 2017
- Making IT Lean. Rebecca Duray, Howard Williams. 2012
- GitLab Continuous Integration (GitLab CI/CD), <https://docs.gitlab.com/ce/ci>
- Continuous integration, [https://en.wikipedia.org/wiki/Continuous\\_integration](https://en.wikipedia.org/wiki/Continuous_integration)
- Ansible IT automation tool documentation, <https://docs.ansible.com/>
- Docker Documentation, <https://docs.docker.com>
- SonarQube documentation, <https://docs.sonarqube.org/>





[@adriano\\_vieira](https://twitter.com/adriano_vieira)  
[speakerdeck.com/adrianovieira](https://speakerdeck.com/adrianovieira)  
[gitlab.com/adrianovieira](https://gitlab.com/adrianovieira)

Keep C.A.L.M.S. and having fun!

## Adriano Vieira

Entusiasta em integração de equipes e agilidade em entrega de resultados, atua como agente influenciador na implementação de infraestruturas ágeis e na internalização de práticas Lean & DevOps.

Profissional de TI com mais 20 anos de experiência trabalhando e gerenciando equipes em ambientes de Datacenter, tendo atuado em desenho, análise e desenvolvimento de sistemas de baixa a alta complexidade como ERP. Também atuou como líder do desenvolvimento de projeto *open-source* e contribui para projetos desta natureza.

Atualmente trabalha na Empresa Brasil de Comunicação (EBC) como Coordenador de Desenvolvimento e Distribuição de Conteúdos. É formado em engenharia mecânica e em MBA.