



Open Talk 公开课

微服务架构下 CI/CD 如何落地

莫红波 又拍云平台开发部



大纲

- ◆ 从单体到微服务
- ◆ 微服务测试模型
- ◆ 持续集成环境
- ◆ 集成环境下的服务发现
- ◆ 持续交付
- ◆ 持续部署

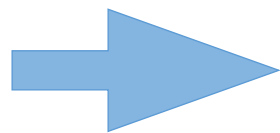
$$2020 = 1024 + 996$$

拥抱变化

- ◆ 互联网企业普遍面临的问题
- ◆ 业务上线、调整、下线经常发生

拥抱变化

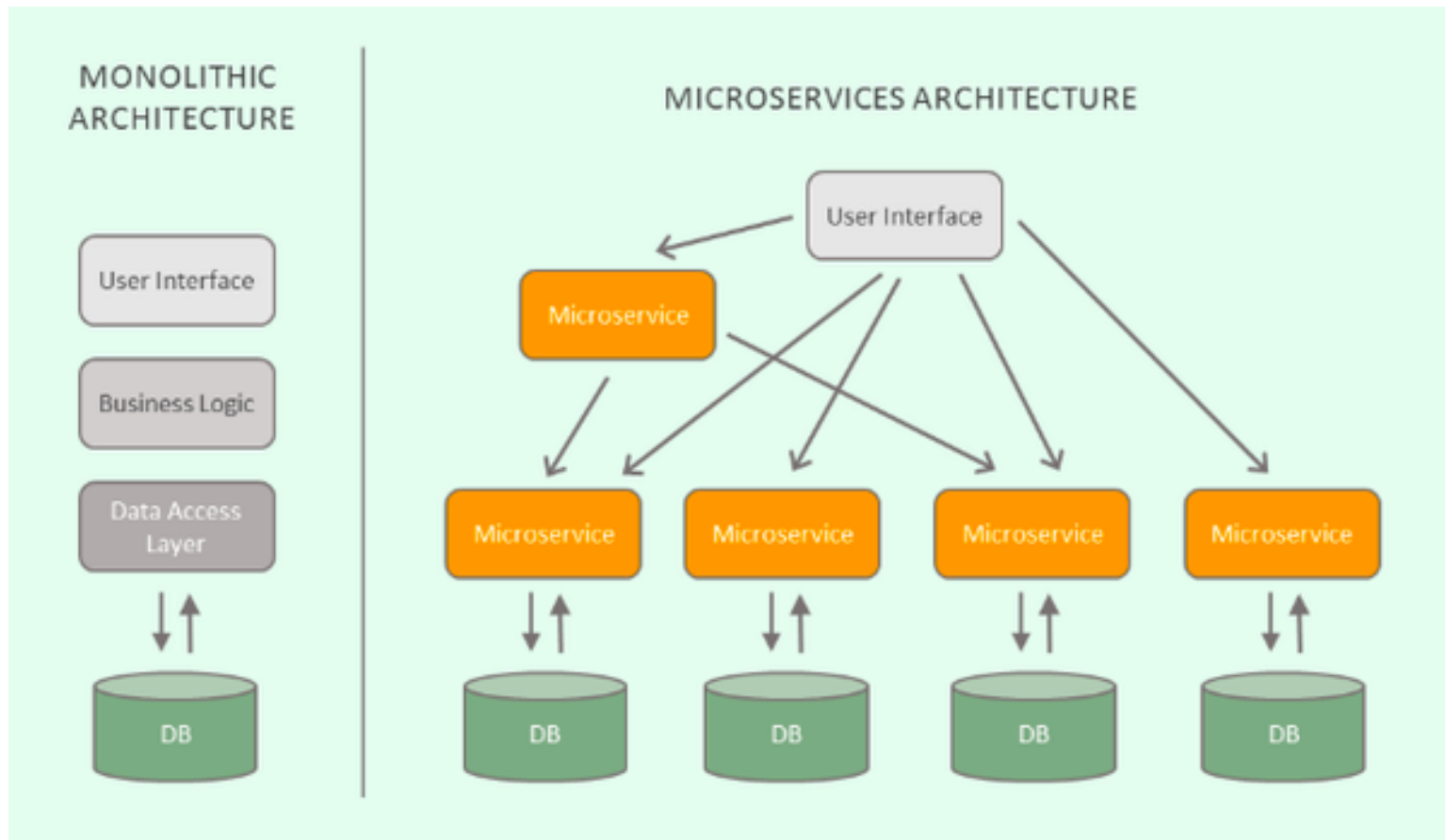
- ◆ 互联网企业普遍面临的问题
- ◆ 业务上线、调整、下线经常发生



松耦合架构

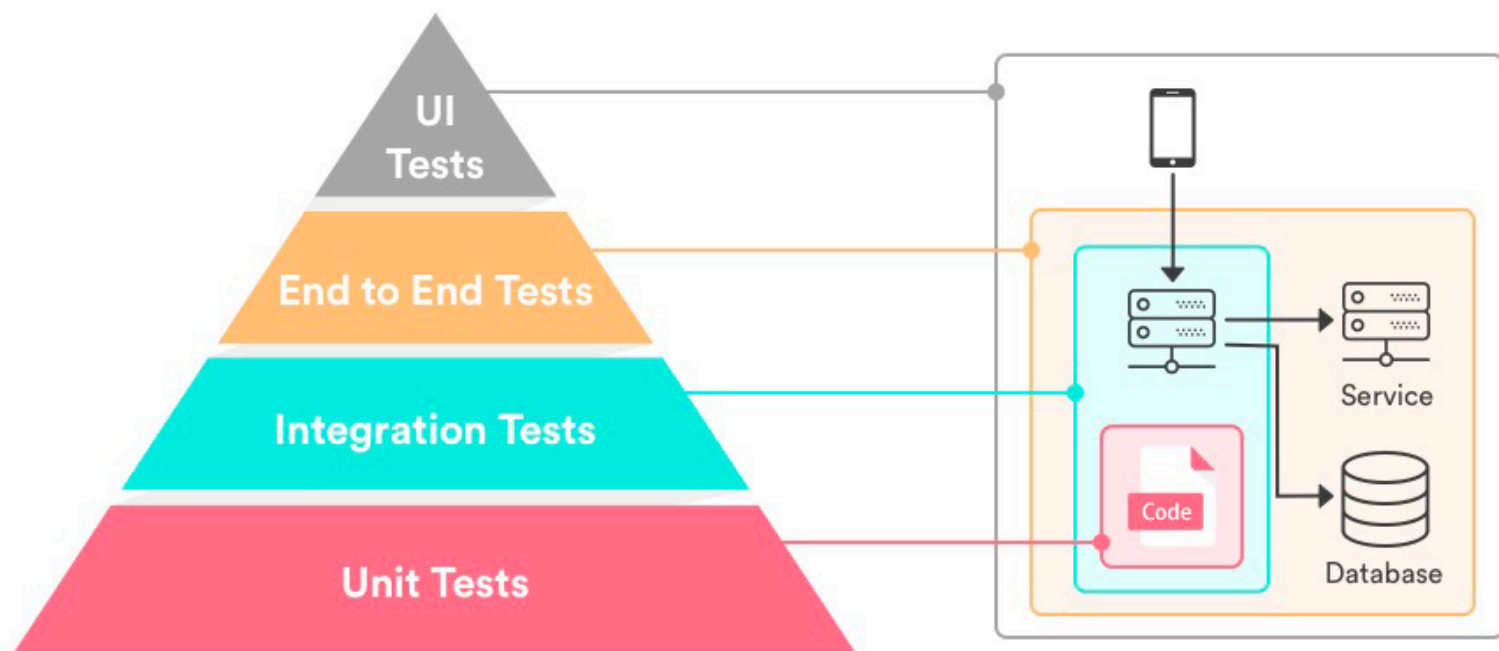
小团队适不适合上微服务？

从单体到微服务



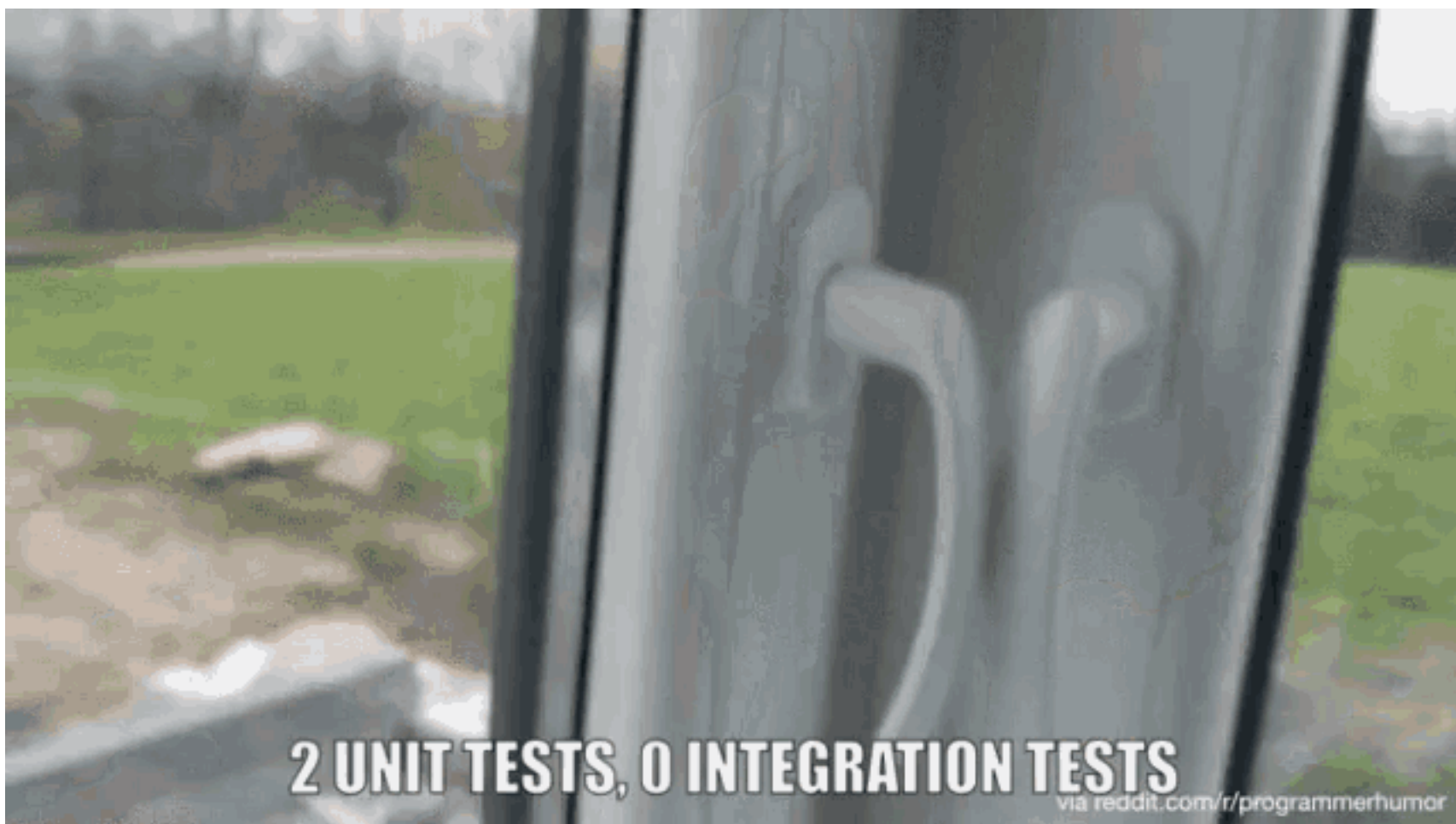
- ◆ 根据不同领域拆分
- ◆ 服务之间通过网络协议通信
- ◆ 拥有独立的数据库
- ◆ 特定的对外开放的接口

微服务测试模型



- ◆ 越底层成本越低
- ◆ 越底层效率越高
- ◆ 越底层自动化越简单

只做单元测试?!



实践？两步走！

第一步，对微服务进行单元测试集成测试用例的补充。如果没有这个底子，微服务是不可靠的，何时何地出问题都是无法预计的，出问题之后排查又是很费时的。

第二步，自动化持续集成环境，能自动化就上自动化，减少人工介入。

实践？两步走！

第一步，对微服务进行单元测试集成测试用例的补充。如果没有这个底子，微服务是不可靠的，何时何地出问题都是无法预计的，出问题之后排查又是很费时的。

第二步，自动化持续集成环境，能自动化就上自动化，减少人工介入。



GitLab CI/CD

The screenshot displays a GitLab CI/CD pipeline interface. At the top, a green status bar indicates the pipeline is 'passed'. Below this, the title 'Merge branch 'front-hotfix' into 'master'' is shown. A link to 'See merge request !1093' is provided. The pipeline summary states '7 jobs from v0.2.7-rc4 in 16 minutes and 49 seconds (queued for 3 seconds)'. The commit hash 'df72447f' is visible. The pipeline jobs are organized into three stages: 'Test' (1 job: 'test'), 'Build' (3 jobs: 'build-app', 'build-dashboard', 'build-swagger'), and 'Release' (3 jobs: 'offline-release', 'online-release', 'online-release-...'). All jobs are marked with green checkmarks, indicating successful completion.

passed Pipeline #181513 triggered 1 day ago by hongbo.mo

Merge branch 'front-hotfix' into 'master'

屏蔽团队管理资源组入口

See merge request !1093

7 jobs from v0.2.7-rc4 in 16 minutes and 49 seconds (queued for 3 seconds)

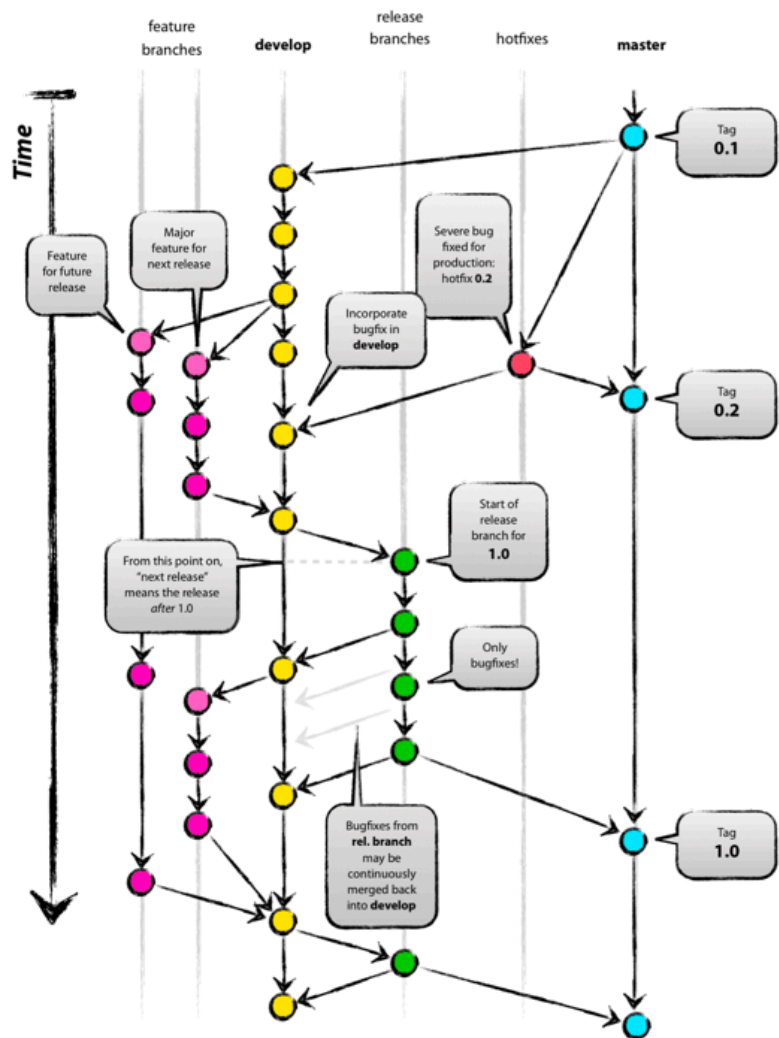
df72447f

Pipeline Jobs 7

Test	Build	Release
test	build-app	offline-release
	build-dashboard	online-release
	build-swagger	online-release-...

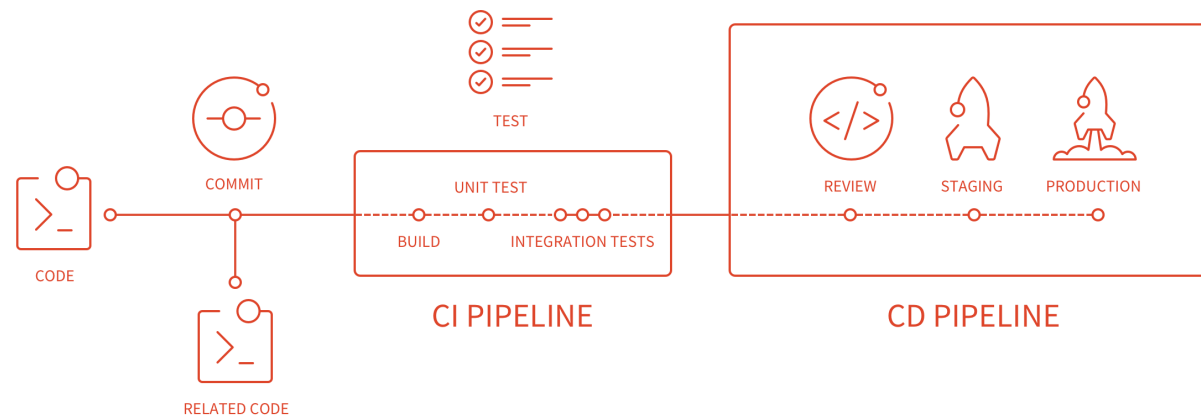
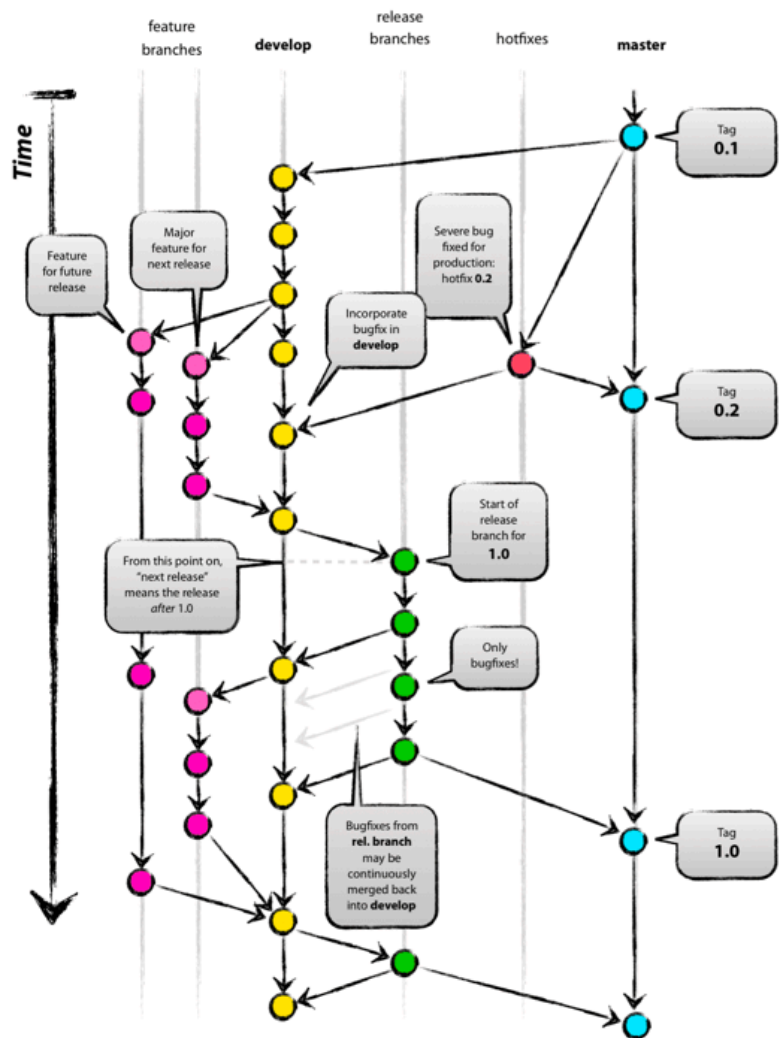
- ◆ 统一的 Web 页
- ◆ 可以在 MR 中跳转查看
- ◆ Pipeline 编排直观展示
- ◆ 所有操作都在项目中搞定
- ◆ GitLab 官方支持

GitLab WorkFlow



- ◆ master 分支上发布版本
- ◆ develop 分支上做开发
- ◆ feature 分支从 develop 分支上 checkout
- ◆ hotfix 分支需要合并到 master/develop 分支

GitLab WorkFlow



- ◆ master 分支上发布版本
- ◆ develop 分支上做开发
- ◆ feature 分支从 develop 分支上 checkout
- ◆ hotfix 分支需要合并到 master/develop 分支

.gitlab-ci.yml 代码片段

stages:

- test
- build
- release
- deploy

test:

stage: test

image: golang:1.13-alpine

services:

- **name:** mysql:5.7
- **name:** redis:5.0.7

script:

- make fasttest

build-app:

stage: build

image: golang:1.13-alpine

script:

- make app **VER=\${CI_BUILD_TAG:-dev}**

build-swagger:

stage: build

image: node:10.9-alpine

script:

- make swagger-validate

online-release:

stage: release

script:

- make docker **VER=\$CI_BUILD_TAG**

.gitlab-ci.yml 代码片段

stages:

- test
- build
- release
- deploy

test:

stage: test

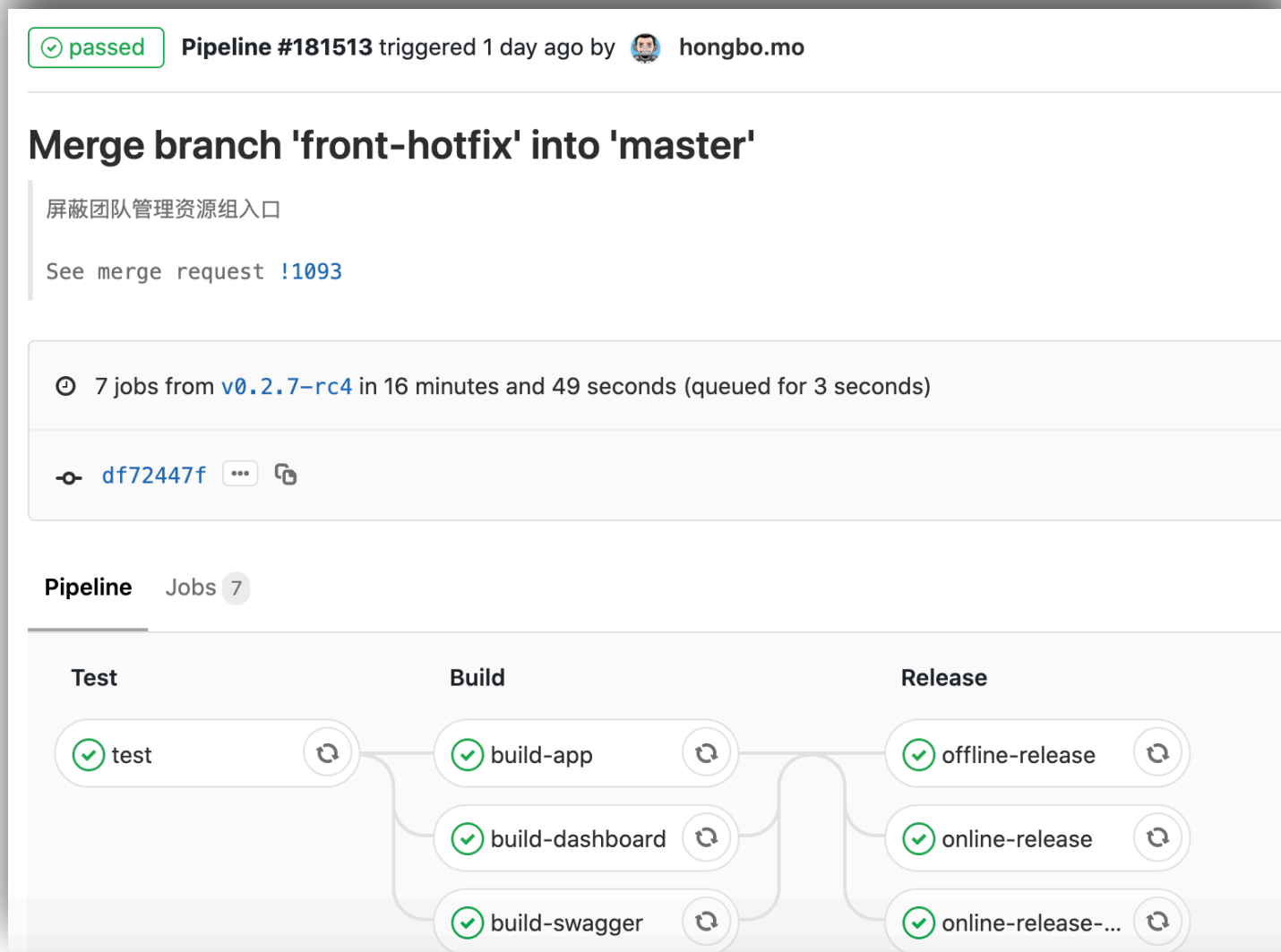
image: golang:1.13-alpine

services:

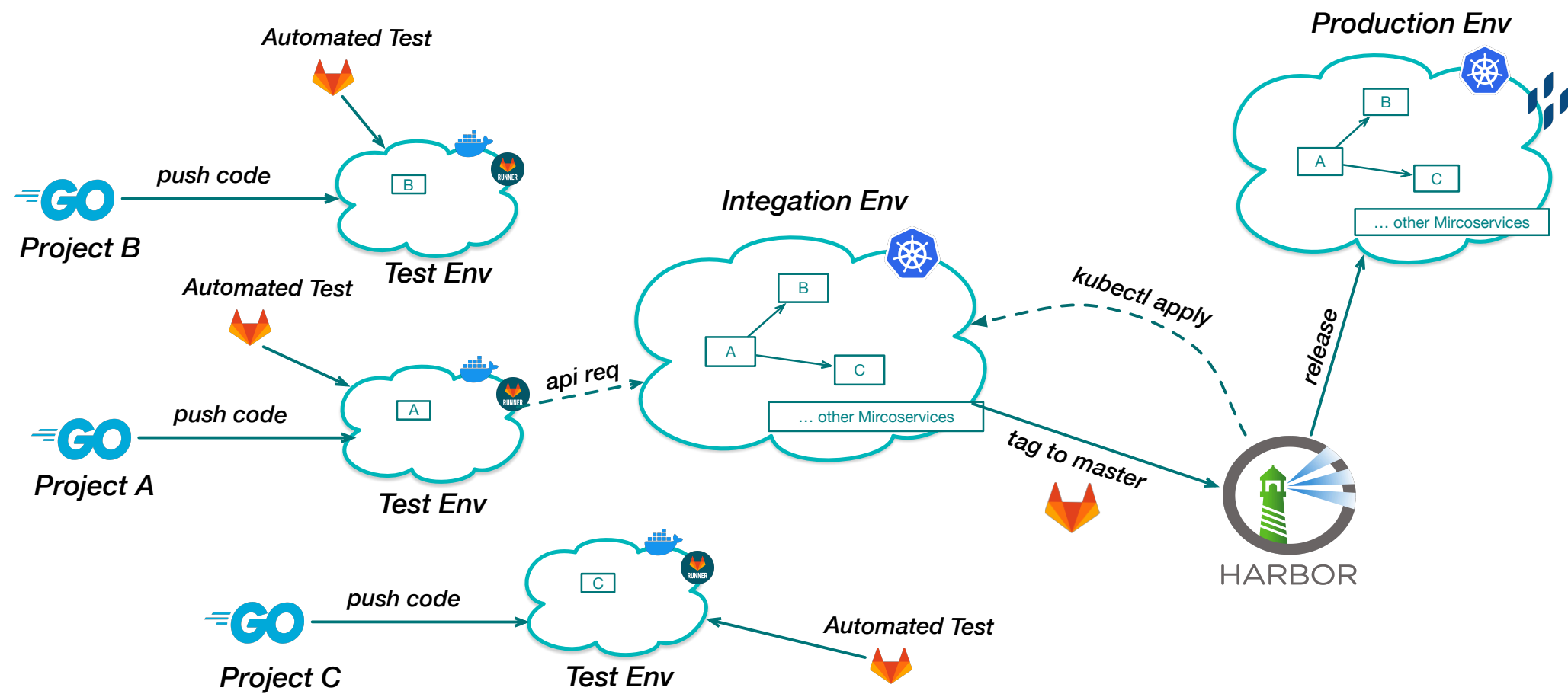
- name: mysql:5.7
- name: redis:5.0.7

script:

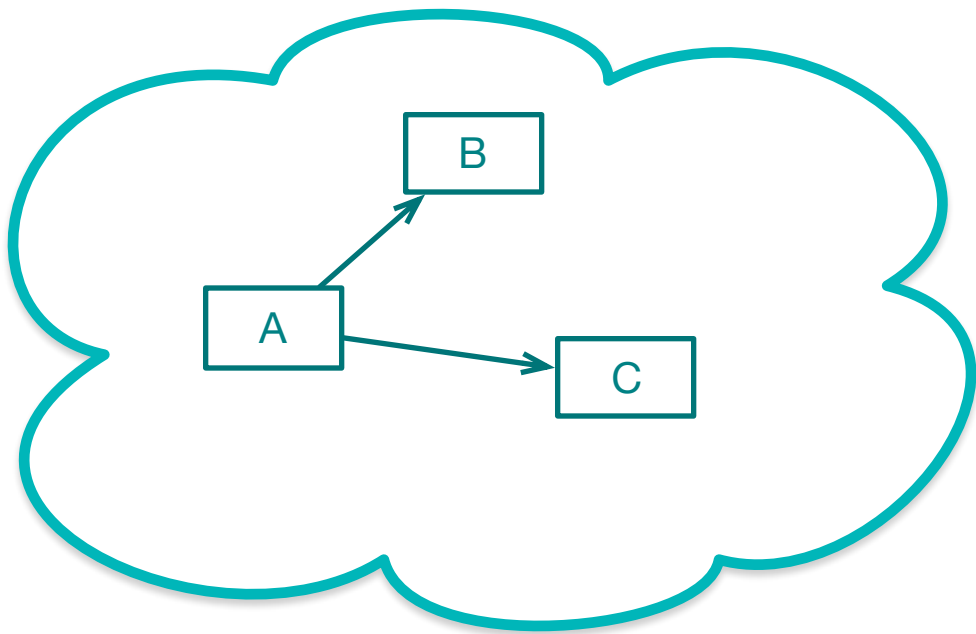
- make fasttest



微服务场景下的变形

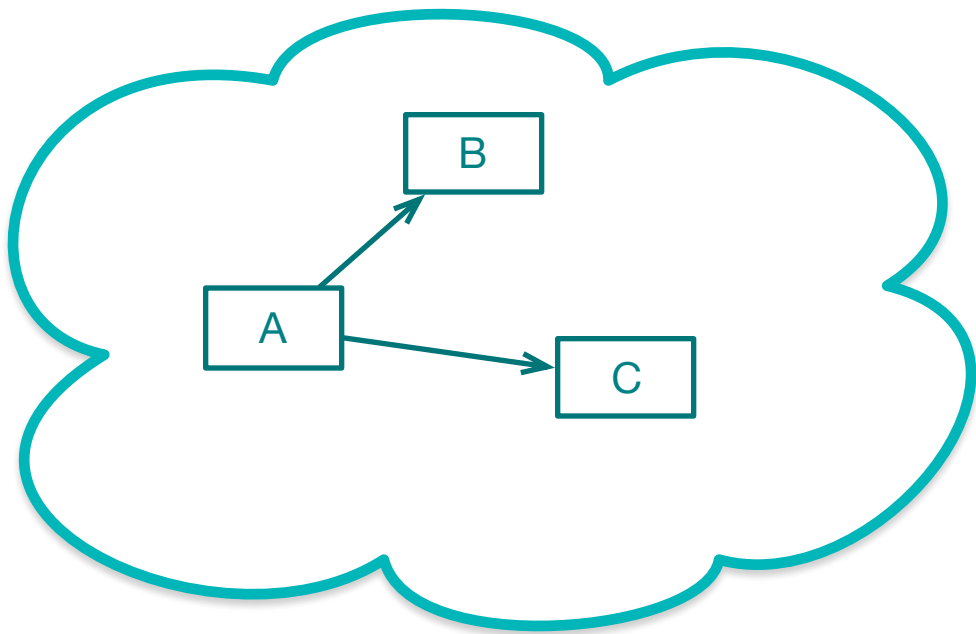


微服务场景下的变形-服务发现



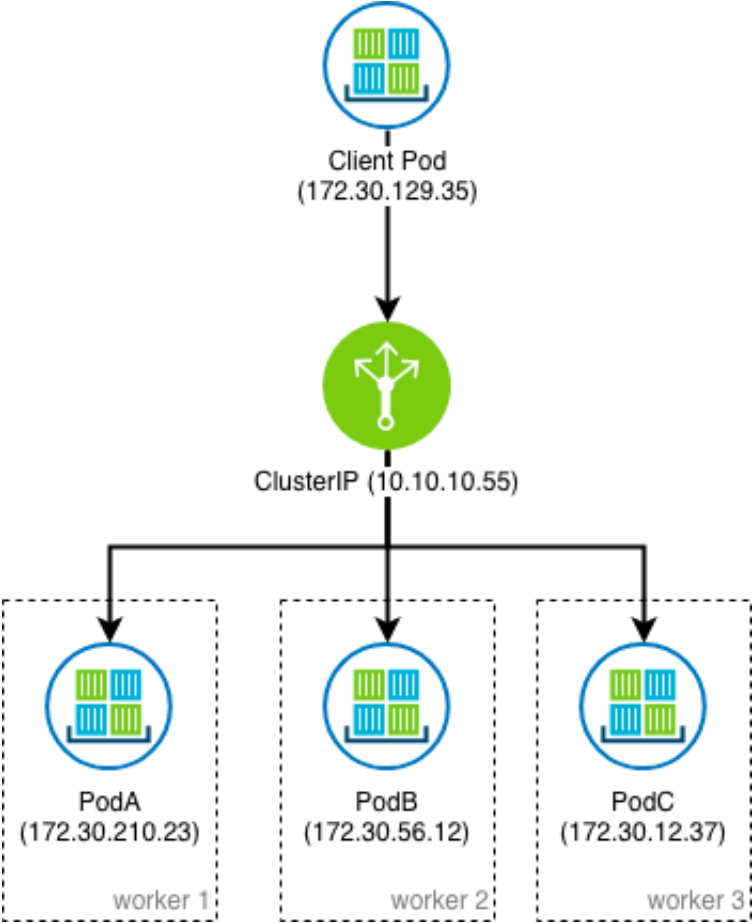
- ◆ 一台共用的测试机器
- ◆ 依赖的服务手动部署
- ◆ 项目测试写死 ip 地址

微服务场景下的变形-服务发现

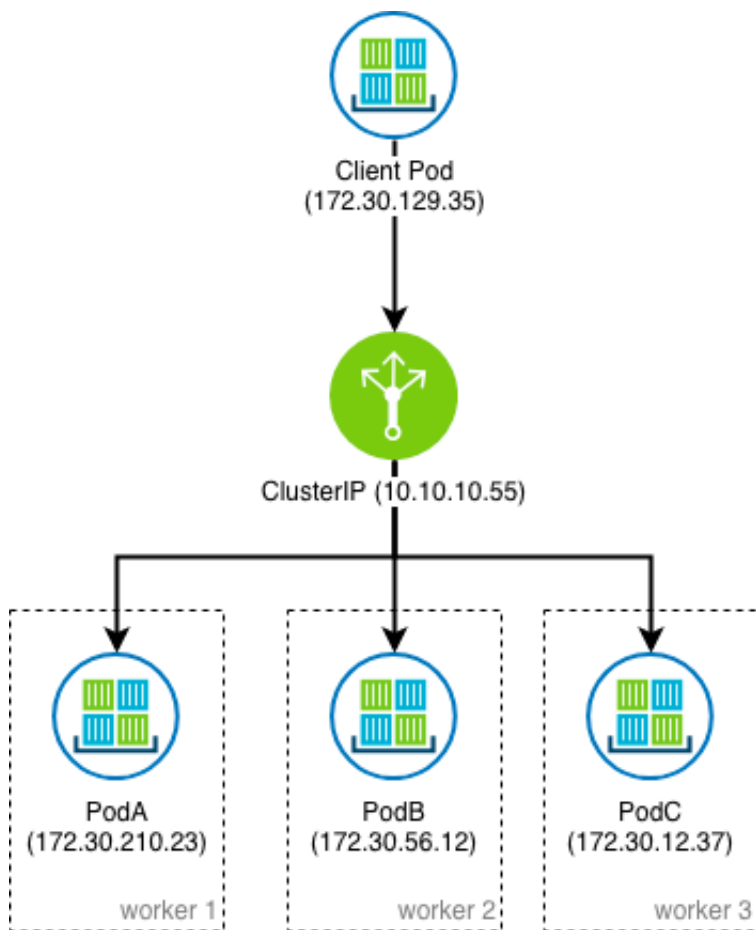


- ◆ 一台共用的测试机器
- ◆ 依赖的服务手动部署
- ◆ 项目测试写死 ip 地址
- ◆ 服务更新延迟甚至遗忘
- ◆ 环境权限混乱
- ◆ 人工操作容易出错
- ◆ 维护成本太高

服务发现-Kubernetes Service



服务发现-Kubernetes Service



```
apiVersion: v1
kind: Service
metadata:
  name: holdon
  namespace: platform
  labels:
    app: holdon
spec:
  type: ClusterIP
  ports:
    - name: http
      port: 8000
      targetPort: 8000
      protocol: TCP
  selector:
    app: holdon
```

`holdon.platform.svc.cluster.local:8000`

微服务场景下的变形-持续交付

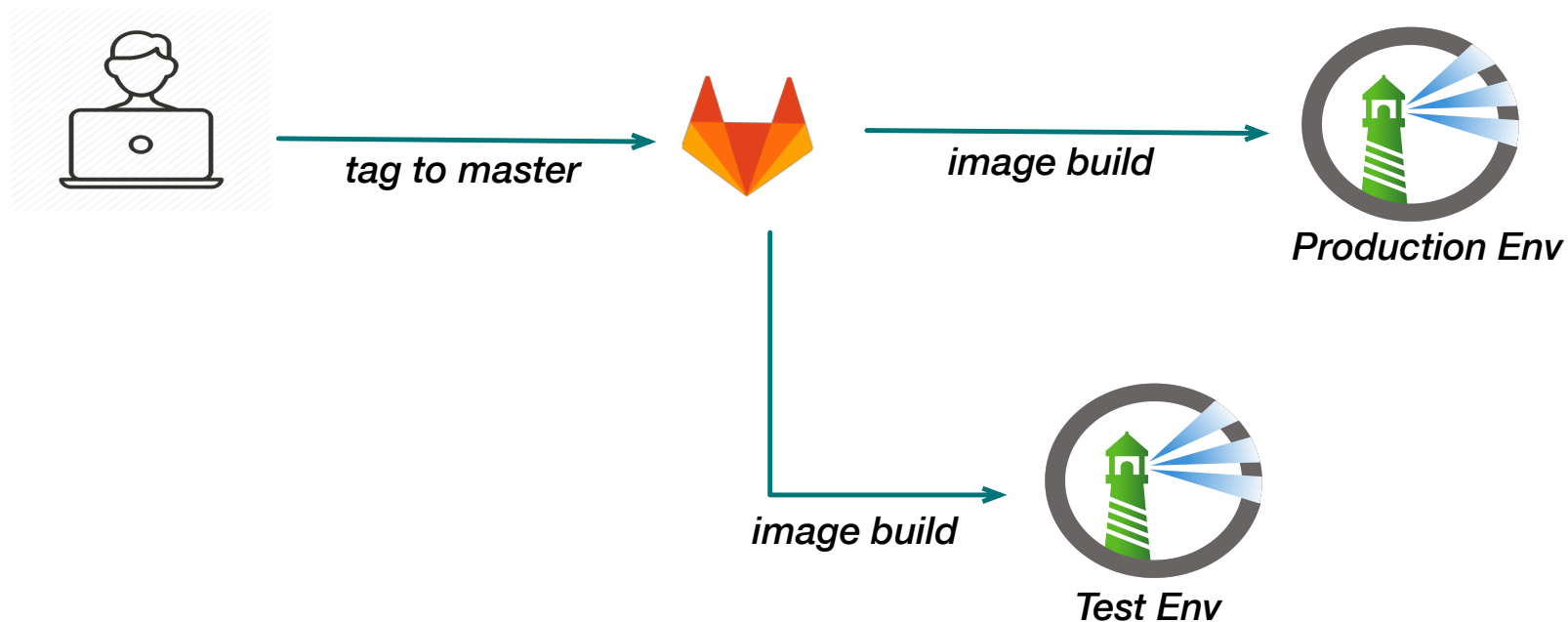
持续交付（英语：Continuous delivery，缩写为 CD）。

每个项目都要有一个 Dockerfile，提供了服务运行所需的环境，以及服务对应的软件包。当需要发版本的时候，我们会在主线上打上一个 tag，这个 tag 也会对应到镜像的版本。

微服务场景下的变形-持续交付

持续交付（英语：Continuous delivery，缩写为 CD）。

每个项目都要有一个 Dockerfile，提供了服务运行所需的环境，以及服务对应的软件包。当需要发版本的时候，我们会在主线上打上一个 tag，这个 tag 也会对应到镜像的版本。



微服务场景下的变形-持续部署

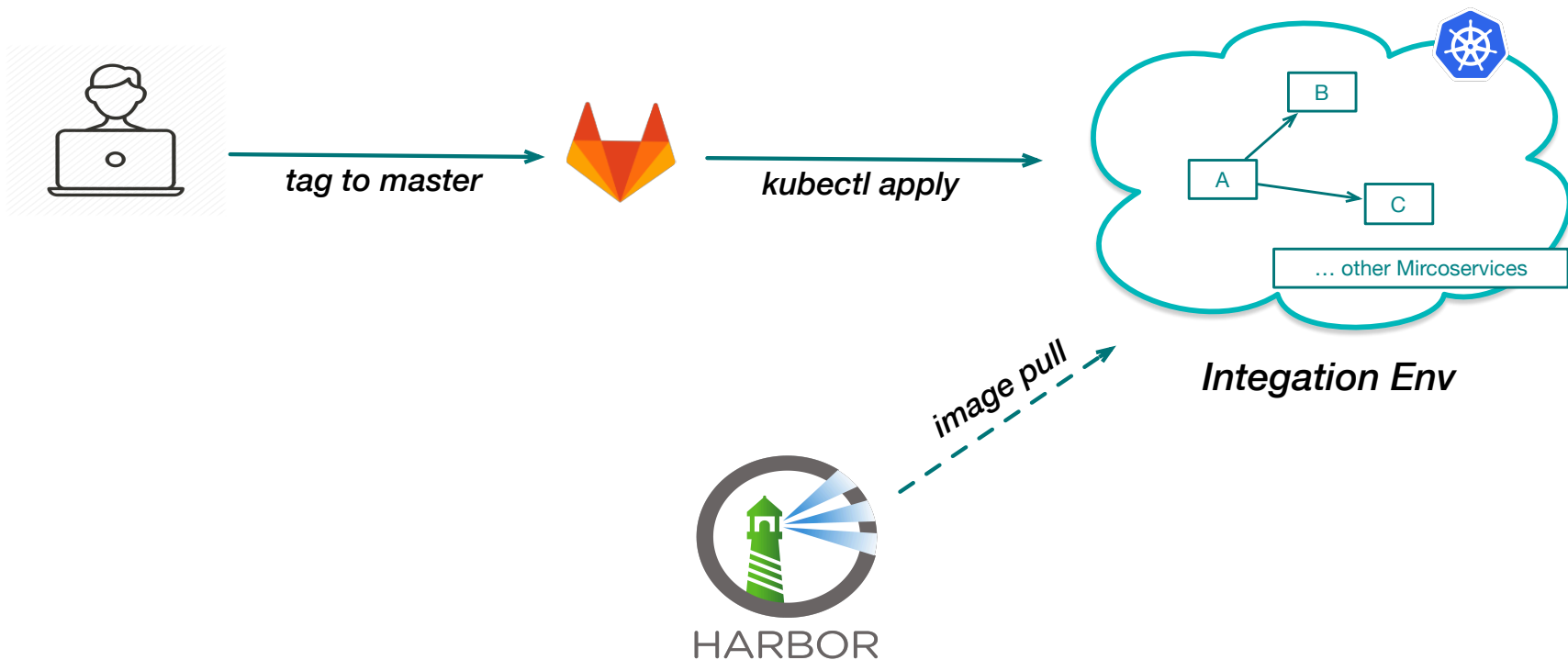
持续部署（英语：Continuous deployment，缩写为CD）。

只针对集成测试环境。给项目增加一个 k8s-deploy.yaml 的文件，里面包含了服务相关的配置，部署方式，访问方式等等，等待 镜像构建完成后，apply 这个文件就可以了

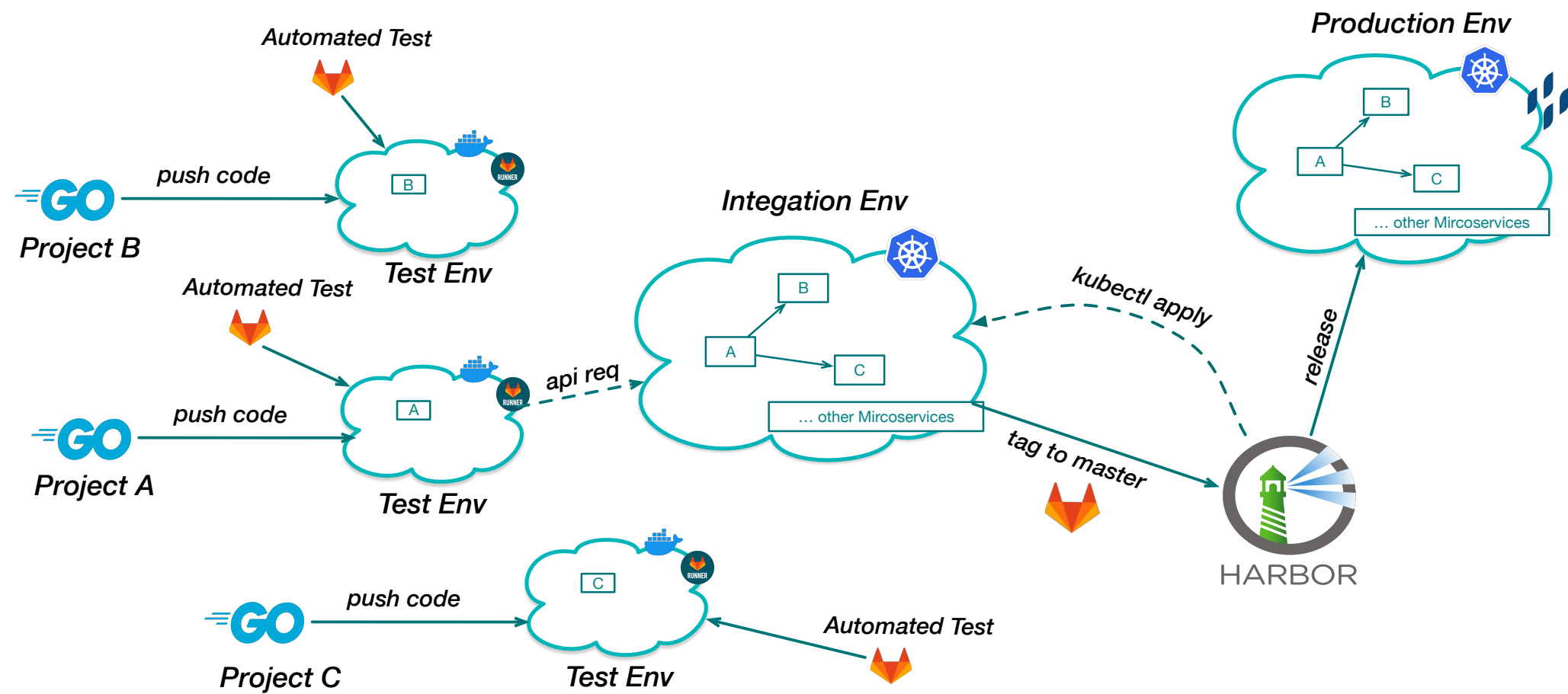
微服务场景下的变形-持续部署

持续部署（英语：Continuous deployment，缩写为CD）。

只针对集成测试环境。给项目增加一个 k8s-deploy.yaml 的文件，里面包含了服务相关的配置，部署方式，访问方式等等，等待 镜像构建完成后，apply 这个文件就可以了



微服务场景下的变形-回顾



成果展示

A collection of graphs regarding Continuous Integration

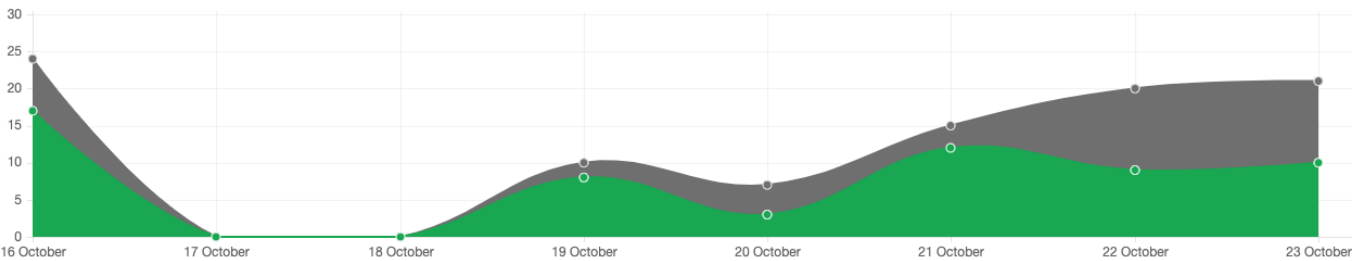
Overall statistics

- Total: **4503 pipelines**
- Successful: **3242 pipelines**
- Failed: **1012 pipelines**
- Success ratio: **76%**

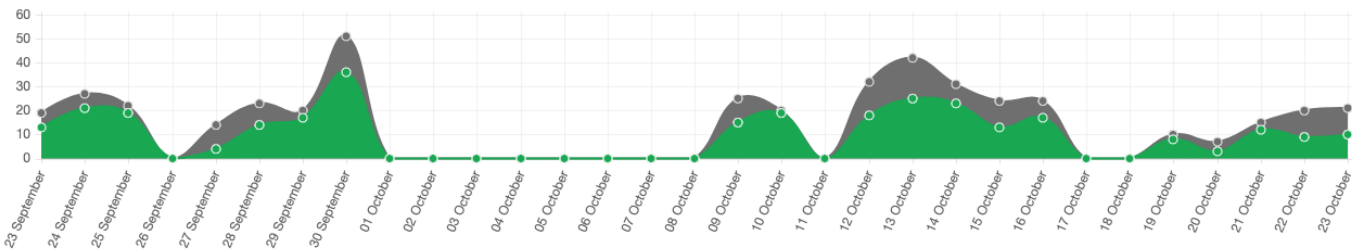
Pipelines charts

● success ● all

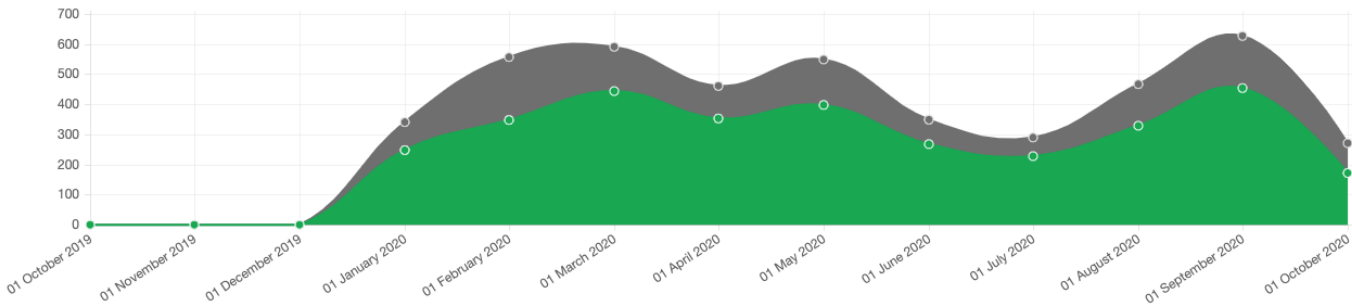
Pipelines for last week (16 Oct - 23 Oct)



Pipelines for last month (23 Sep - 23 Oct)



Pipelines for last year





关注又拍云微信公众号，
获取更多干货！

THANKS!

