# devops的一些想法

内容目录

[devops的一些想法 1](#__RefHeading___Toc109_1711635334)

[术语 1](#__RefHeading___Toc111_1711635334)

[关系 1](#__RefHeading___Toc113_1711635334)

[分支模式 2](#__RefHeading___Toc115_1711635334)

[应用的流程 2](#__RefHeading___Toc117_1711635334)

[应用流程规范 3](#__RefHeading___Toc119_1711635334)

[UAT 应用 3](#__RefHeading___Toc121_1711635334)

## 术语

该文会用到以下名词：

1、项目（project）: 以客户角度分类。

例：唯品会project,邮政project

2、应用 （app）: 项目的模块就是应用 。

例：wes app, rcs app

3、环境 （environment） ： 环境对应K8S的namespace。

例： DEV (开发环境) 、QA （测试环境）、release (发布环境)

4、分支模式：一个应用有两种模式选择。

aoneflow模式，trunk模式

5、应用模板： 一个应用在gitlab中对应一个模板，模板的定义由devops制定

例： wes的模板，rcs模板

6、应用域名：每一个应用在k8s中都有一个域名来标明，用户通过域名来访问应用。

**域名命名规则：project\_slug.app\_name.environment\_name.k8s.flashhold.com**

例：唯品会项目的wes应用在QA环境的应用访问域名：

vipshop.wes.qa.k8s.flashhold.com

7、平台： 将来会有一个快仓devops可视化平台

## 关系

project和app的关系： 一对多

app和environment的关系： 一对多

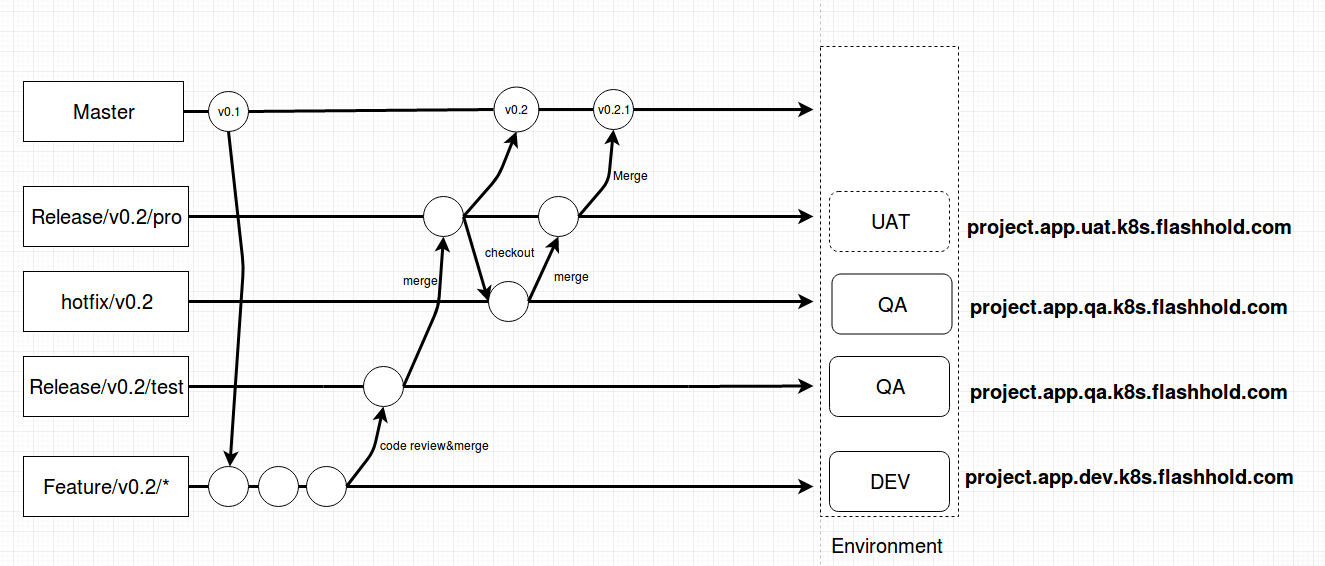
分支和environment的关系： 一对一

app和应用域名的关系： 一对多

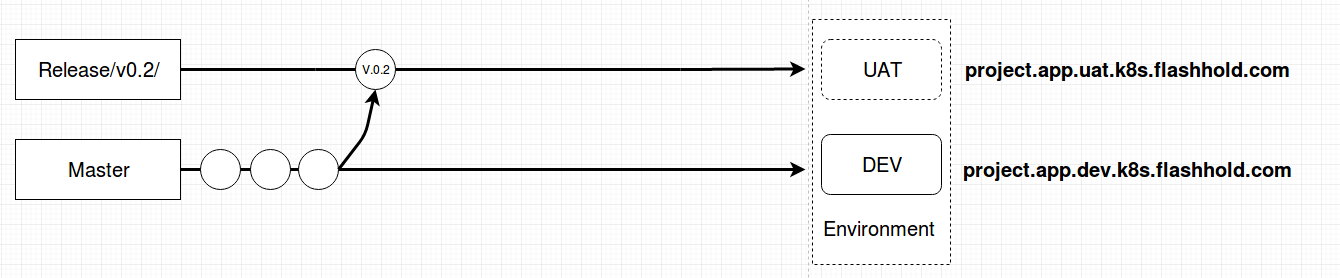
分支和应用域名的关系：一对一

## 分支模式

aone模式：



Trunk模式



## 应用的流程

应用在每一条分支上都由自动化构建pipeline每次提交到远程都会自动生成一个镜像和chart,然后推送到harbor上。最后发一份邮件给提交该commit的人员。

镜像命名规范：项目名-应用名-环境名：commit\_id

chart命名规范： 项目名-应用名-环境名/ 项目名-应用名-环境名

邮件规范：

dear aaa：

xxx项目的bbb应用在ccc环境中已经开始部署。请您于2分钟后，访问xxx.bbb.ccc.k8sflashhold.com来验证服务

例：

小明在远程feature分支上提交了一个唯品会rcs app的commit。commit\_id为123456。那该远程feature上会生成一个镜像和chart命名规则为

镜像名：vipshop-rcs-dev:123456

chart名:vipshop-rcs-dev/vipshop-rcs-dev

应用域名：vipshop.rcs.dev.k8s.flashhold.com

邮件内容为：

dear 小名：

vipshop项目的rcs app已经开始部署。请您于2分钟后，访问vipshow.rcs.dev.flashhold.com来访问服务

## 应用流程规范

应用的流程规范与分支模式密切相关

在aoneflow模式中：

feature到release/test的合并需要该app的项目负责人codereview，通过后在release/test分支构建后由qa测试。测试通过后由qa在平台上点发布。把release/test分支合并到release/pro分支构建进入uat环境。平台通过api接口触发UAT应用的构建。

## UAT 应用

这个应用很特殊：

一般应用和项目的关系是多对一

uat应用和项目的关系是一对一

一个项目只有一个uat项目。他是这个项目所有项目的粘合剂

**本质上UAT项目是一个helm chart。他来管理项目各个app的关系和顺序**

uat app由平台api触发。

例：

vipshop/

rcs\_app

register\_app

wcs\_app

uat\_app(有点像docker-compose)