



HorizonCD

云音乐在GitOps CD领域的最佳实践



朱旭

网易云音乐
云原生开发工程师



目 录

背景

01

Horizon模板体系

02

GitOps最佳实践

03

Horizon产品介绍

04

Horizon落地

05

第一部分

背景

背景

云主机时代的痛点



效率
低下

计算
焦虑

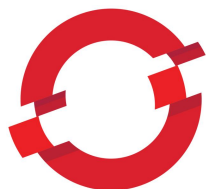
运维
繁琐

成本
高昂

2020年，云音乐开始容器化转型

背景

业界优秀产品



站在巨人的肩膀上，打造符合公司中长期发展的 DevOps CD 产品 - Horizon

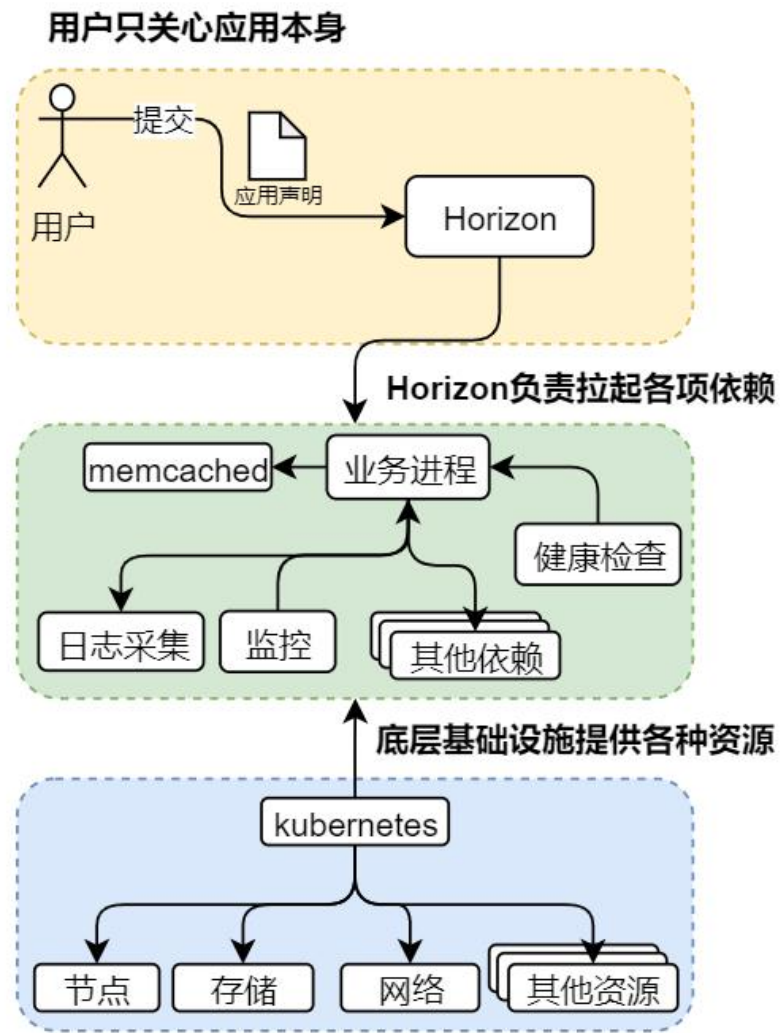
Helm Chart的理念

如何屏蔽 k8s 复杂性？

- 通用模板能力：屏蔽大部分细节，将少量配置开放给使用者
- 以应用为中心：围绕着应用，集中了应用依赖的各种资源

helm chart不足：

- 面向开发者，对普通用户不友好



第二部分

Horizon模板体系



Horizon模板结构

```
├── Chart.yaml
├── README.md
├── schema                                ## 定义各种上层用户可理解的输入（支持前端自动渲染）
│   ├── application.schema.json ## 应用部署配置的 schema 定义
├── output
│   └── outputs.yaml                ## 模板自定义输出
├── templates                          ## 定义各种下层基础设施可理解的声明式 spec
│   ├── _helpers.tpl
│   ├── hpa.yaml
│   ├── ingress.yaml
│   ├── prometheusrule.yaml
│   ├── deployment.yaml
│   └── service.yaml
├── files                              ## templates 依赖的各种文件
│   ├── http-probe.sh
│   ├── offline.sh
│   ├── online.sh
│   ├── startup.sh
│   └── status.sh
└── values.yaml
```

Horizon 模板结构示例

模板组成：

- Helm Chart
- JsonSchema
- ReactJsonSchemaForm (RJSF)

关注点分离：

- 业务关注产品化
- 管理员关注底层实现
- 运维人员关注特殊场景的人工介入

Horizon模板最佳实践

管理员可以通过快速的配置，将一个复杂应用的关键配置快速暴露给用户使用，保障用户进行最佳实践，例如：

- 屏蔽复杂的资源配置：业务更容易理解的标准资源规格
- 底层能力快速封装：混合云、服务网格

有了这样的模板，是否能直接apply使用呢？

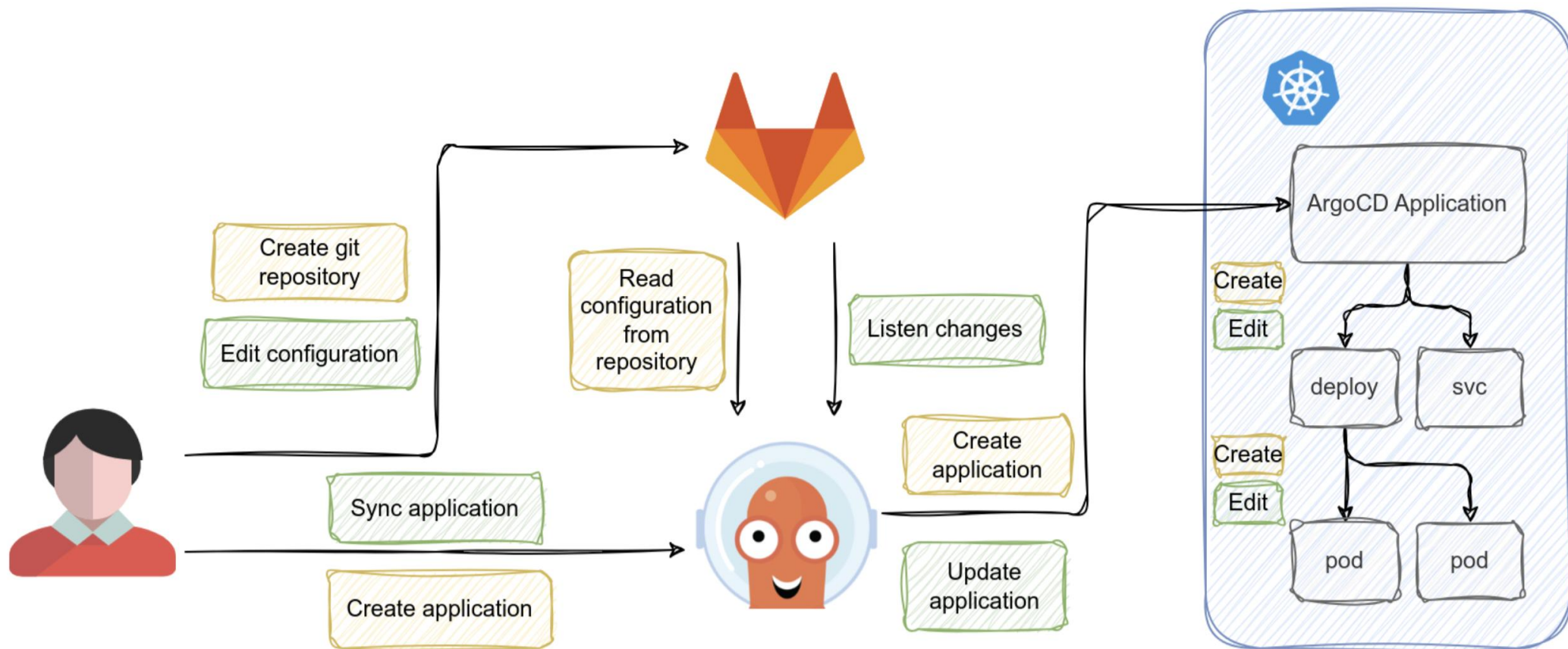
The screenshot shows the 'Horizon' management console interface. The top navigation bar includes 'Horizon', '应用实例', '应用', '分组', and '模板'. The right side of the header has links for 'SLO', 'API', '文档', '联系我们', and a search icon. The left sidebar contains a list of items: '1', '主页', '详情' (highlighted), '流水线', '监控', '成员', and '设置'. The main content area is titled '配置' and shows the configuration for 'javaapp-demo-1'. It includes sections for '应用', '规格' (with a dropdown menu showing 'x-small(1C2G)' and a note about recommended specifications), '扩缩容' (with radio buttons for '固定副本数', '弹性扩缩容', and '定时扩缩容'), '副本数' (with a text input '4' and a note), '发布策略', '暂停策略' (with a dropdown menu showing '全部暂停'), '发布批次' (with a text input '1' and a note), and '参数'.

第三部分

GitOps最佳实践



ArgoCD的设计



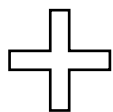
ArgoCD: 用户只需要创建 Git 仓库和 ArgoCD 应用，ArgoCD 会自动地将 Git 仓库中的配置应用到目标环境中，并实时监听 Git 仓库变化，一旦 Git 仓库中的配置发生变化，ArgoCD 也可以进行自动同步。

什么是GitOps

GitOps 是一种符合 DevOps 思想的运维方式，GitOps 以 Git 仓库作为唯一的**事实来源**，储存**声明式配置**，并通过**自动化工具**实现环境和应用的自动化管理。

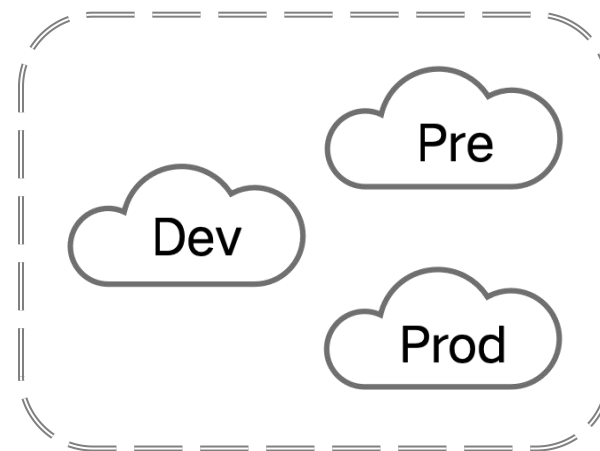
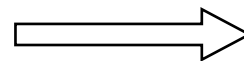


版本管理
持久化、可审计



```
apiVersion: apps/v1
kind: Deployment
metadata:
  name: demo
spec:
  replicas: 3
  ...
```

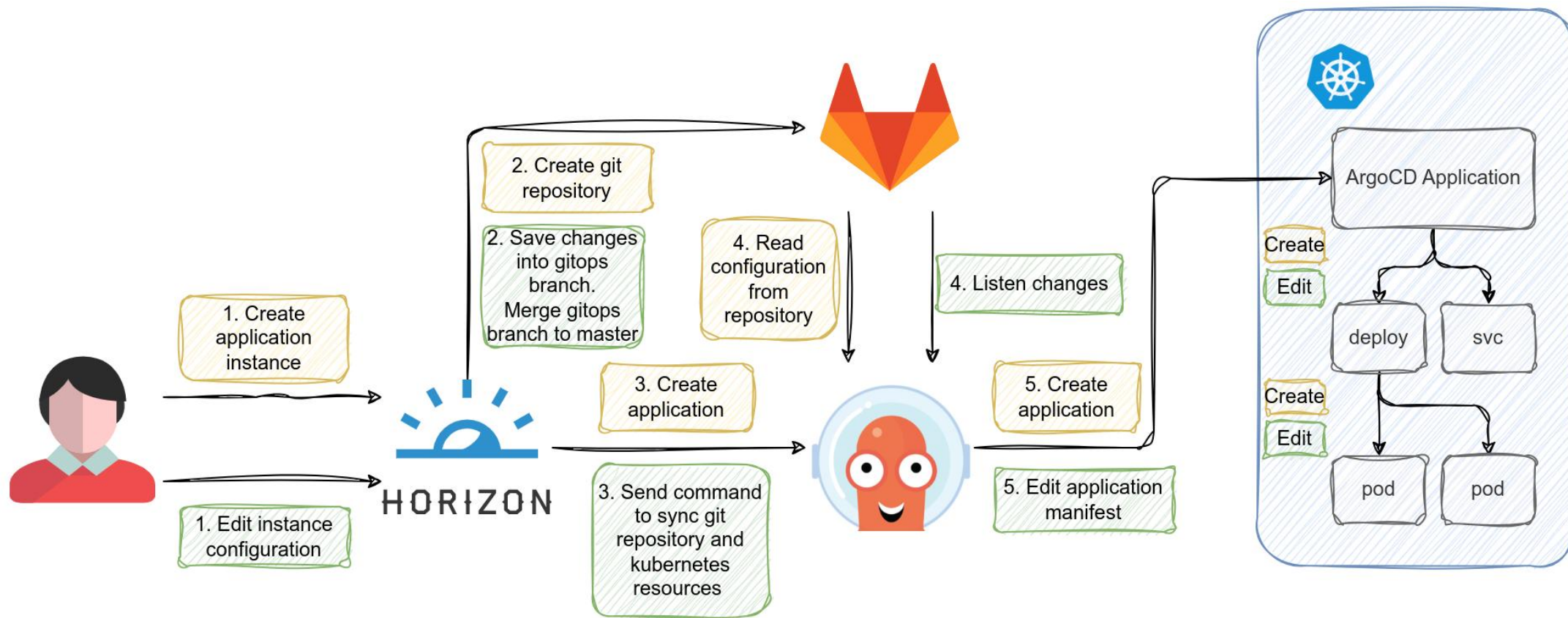
声明式配置
幂等性、事务性



自动化部署
效率提升

GitOps in Horizon

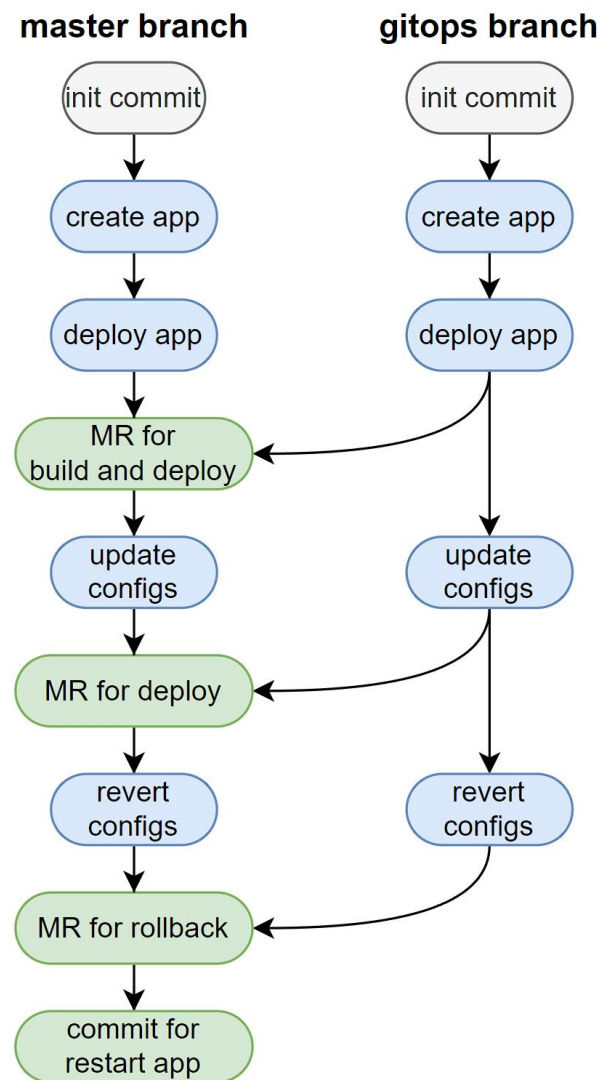
Horizon作为面向用户的统一界面，用户不感知 gitlab 和 argo CD



GitOps in Horizon

应用生命周期->代码库的生命周期

- 应用创建：平台自动创建 git 仓库
- 应用修改：git commit
- 应用发布：git merge
- 应用回滚：git revert
- 发布记录查看：git log
- ...



GitOps in Horizon

用户的每一次发布动作都会记录为一条 Pipelinerun 记录，代表了一次流水线运行实例。

Horizon

应用实例

应用

分组

模板

更多

SLO

API

文档

联系我们

D demo-0907-test

主页

详情

流水线

监控

成员

设置

horizon / demo / demo-0907 / demo-0907-test / pipelines

所有

可回滚

未发布

状态	流水线	标题	触发者	触发类型	创建时间	操作
成功	#2509702	123		构建发布	2023-10-23 17:19:20	
失败	#2509584	123		构建发布	2023-10-23 16:44:41	
成功	#2310210	rollback		回滚	2023-09-07 13:53:09	回滚
成功	#2310196	restart		重新启动	2023-09-07 13:51:07	
成功	#2309905	扩容		直接发布	2023-09-07 12:07:47	回滚
成功	#2309903	1		构建发布	2023-09-07 12:04:52	回滚



GitOps in Horizon

用户可以查看发布配置的变更情况，并且可以快速地选择流水线进行回滚

Horizon

应用实例

应用

分组

模板

更多

SLO

API

文档

联系我们

D demo-0907-test

变更

流水线日志

主页

详情

流水线

监控

成员

设置

配置变更

Files changed (1)

application.yaml

+7-5

application.yaml

CHANGED

@@ -15,14 +15,16 @@ rollout:

15

expectedStartTime: 200

15

expectedStartTime: 200

16

offline: /health/offline

16

offline: /health/offline

17

online: /health/online

17

online: /health/online

18

- port: 8080

18

+ port: 8888

19

status: /health/status

19

status: /health/status

20

spec:

20

spec:

21

- autoscale: none

21

+ autoscale: hpa

22

- replicas: 1

22

+ cpuUtilization: 30

23

23

+ maxReplicas: 4

24

24

+ minReplicas: 2

25

25

+ resource: small

26

strategy:

26

strategy:

25

- pauseType: all

27

+ pauseType: first

26

stepsTotal: 1

28

stepsTotal: 1

G

CN

第四部分

Horizon产品介绍



Horizon介绍

- 定位：
 - 基于 Kubernetes 的一站式云原生 DevOps CD 平台
 - 旨在为基于 Kubernetes 的云原生应用部署提供可靠、安全、高效的标准化方案
- 产品理念：
 - 以应用为中心：面向业务视角，关注点分离
 - 全面践行 GitOps：持久化、可审批、可回滚
 - 声明式 API 和控制器模式：屏蔽底层细节、减轻使用负担
- 官网：<https://horizoncd.github.io>



Horizon平台能力

System Management

envoriment

multi-cloud

IDP & OIDC

user

Horizon Template

json schema form

helm template

CICD

build Type

git

Dockerfile

Image

workload

rollout

serverless

service catalog

database

middleware

Permission

member

RBAC

Integration

Open API

Oauth 2.0

Access Token

Webhook

Trigger

Horizon App

第五部分

Horizon落地

Horizon落地

网易内部：

- **大规模应用：**Horizon 已在**云音乐、传媒**等事业部大规模用于生产环境，基于 Horizon 日平均应用发布**数千次**
- **全面容器化转型：**Horizon帮助云音乐全面进行容器化转型，支持多种工作负载场景
- **降本增效：**各个机房的CPU峰值利用率由不足 20% 提升至 **45%+** 以上

外部企业：

已在一家外部企业（中税税务）成功落地：[Who is using horizon?](#)

- **痛点：**原先使用华为云基于 jenkins 部署业务，**无法管理服务生命周期；**
自动化程度**低**，运维成本**高**



Q & A

欢迎加入Horizon社区



x.zhu

