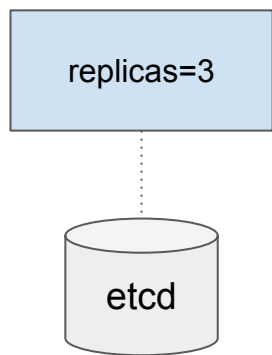


GitOps 与 OAM/KubeVela

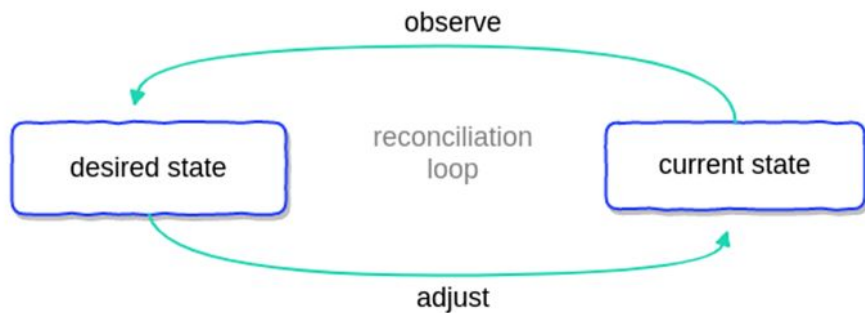
03/05/2021

到底啥是 GitOps ?

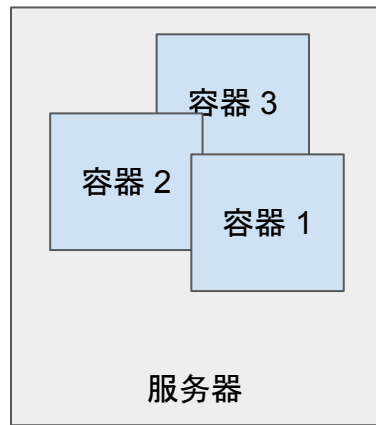
要理解 GitOps, 就必须先理解 Kubernetes **控制循环**



期望状态
(Source-of-Truth)



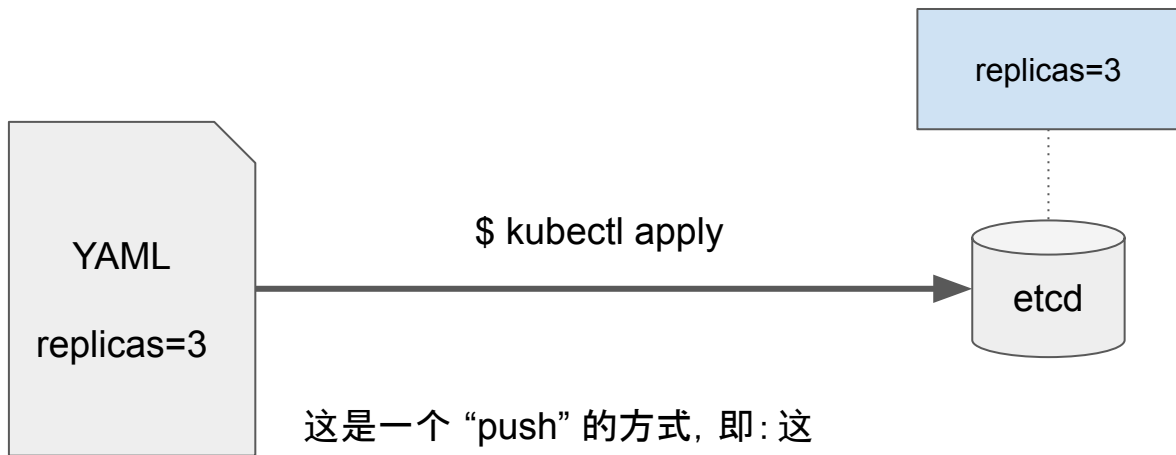
Kubernetes 控制循环



实际状态

那么 :replicas=3 又是谁指定的？

废话，当然是 YAML ！



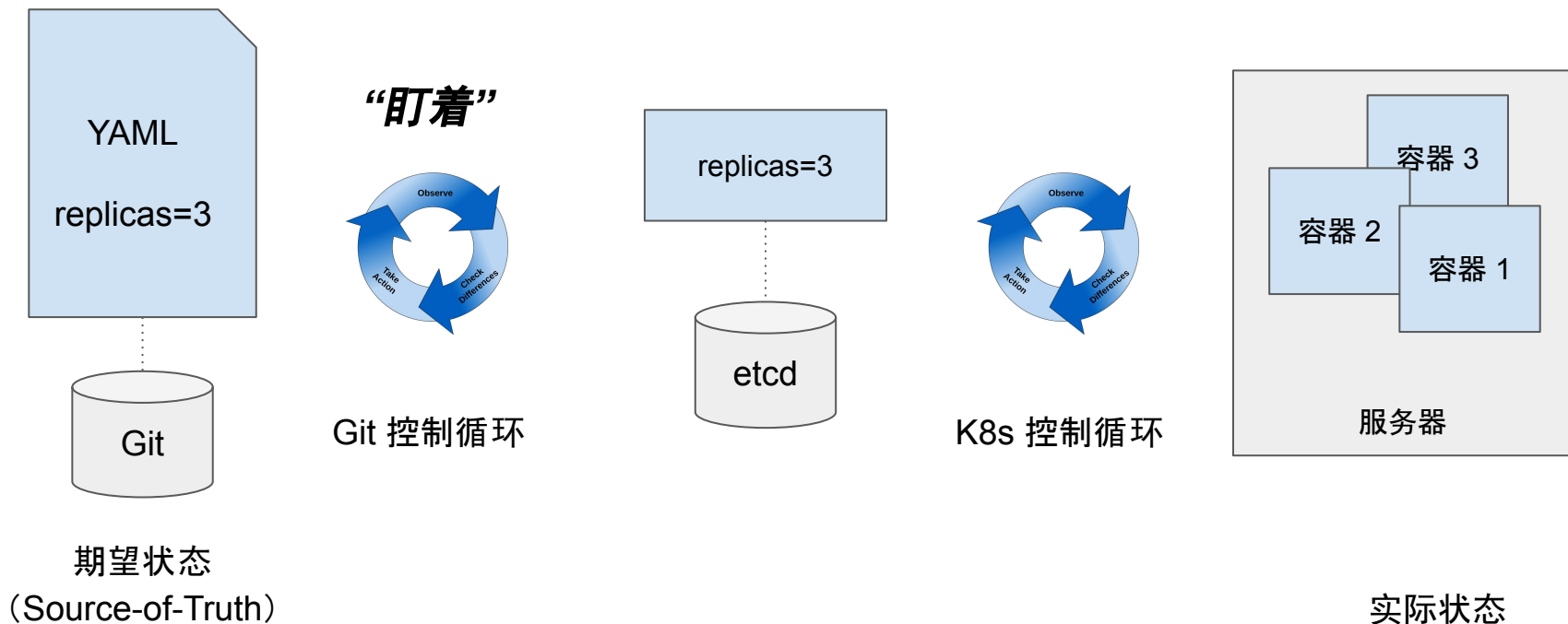
这是一个“push”的方式，即：这个 YAML 在 apply 过之后就沒用了，k8s 从始至终都不知道这个 YAML 的存在

问题来了：

如果 etcd 里的数据被别人改了怎么办？（比如 \$ kubectl edit）

要不，找人**盯着**这个变化？

解决方案: 引入一个 YAML 与 etcd 数据间的控制循环



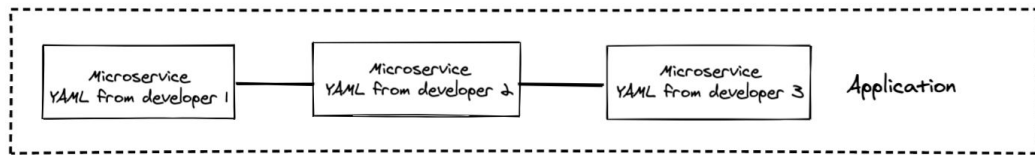
但这有啥好处啊？

- 持续交付 (CD)
 - 将应用的**描述**, “变成”目标环境中**正常运行的实例**, 并且确保这个过程的**持续成功**
- 传统方式
 - CI 流水线将 YAML (应用描述) **push** (比如 `kubectl apply`) 到目标 K8s 集群中
 - 如果 PUSH 失败了怎么办？
 - 写一些脚本来检查、重试、回滚 - CI/CD 脚本
- 有了 Git 控制循环之后
 - CI 流水线只负责生产软件制品 (比如容器镜像), 不会连接和操作任何目标 K8s 集群
 - 运维负责维护 YAML (应用描述) 在某个 Git 仓库中
 - Git 控制循环负责把 YAML 从 Git 仓库 **pull** 到目标 K8s 当中, 并确保 K8s 中保存的应用描述与 Git 仓库永远一致
- 所以, GitOps 的本质是:
 - **通过 K8s 控制器来保证应用交付的正确性、一致性和持续性, 并以此免去了编写 CI/CD 脚本的负担**
 - 这其实也是任何 K8s Controller 的主要功能 (比如 MySQL Operator 其实就是把 MySQL 运维逻辑从脚本转移到了 K8s 控制器里)

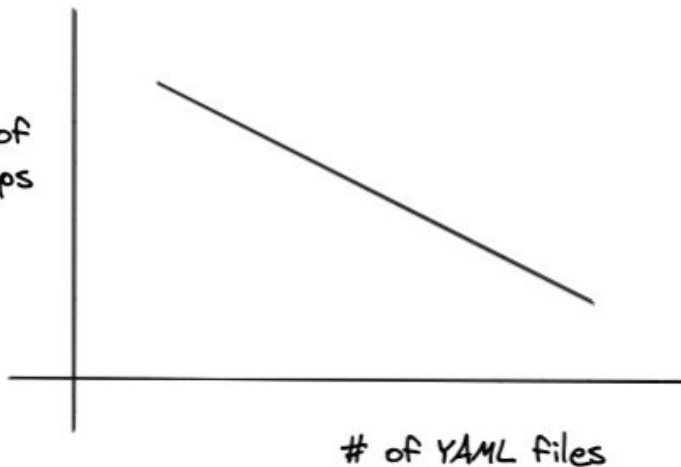
所以, GitOps 的核心不在 Git

- 所谓 PR/commit 驱动、Git Log 审计等只是因为选择 Git 作为 YAML 存储的附带价值
- **GitOps 的核心是应用描述与目标 K8s 集群之间这个控制循环的存在**
 - 所以你可以放心大胆的去搞 SVN-Ops, OSS-Ops, CMDB-Ops (本条不作为选型建议)
 - 你也可以在 Git 里托管一切(而不只是应用描述), 比如放一个 Tekton Pipeline 进去让 GitOps 去同步到目标集群
 - 你也可以用其它 Controller/Pipeline 去管理 Git 控制循环
 - 暂停 Git 控制循环(比如做 Debug 等)
 - 触发不同的控制循环(比如先启动 Staging 环境的同步, 满足一定条件再启动 Production 环境的同步 - Promotion)

但 GitOps 的缺点也是显而易见的



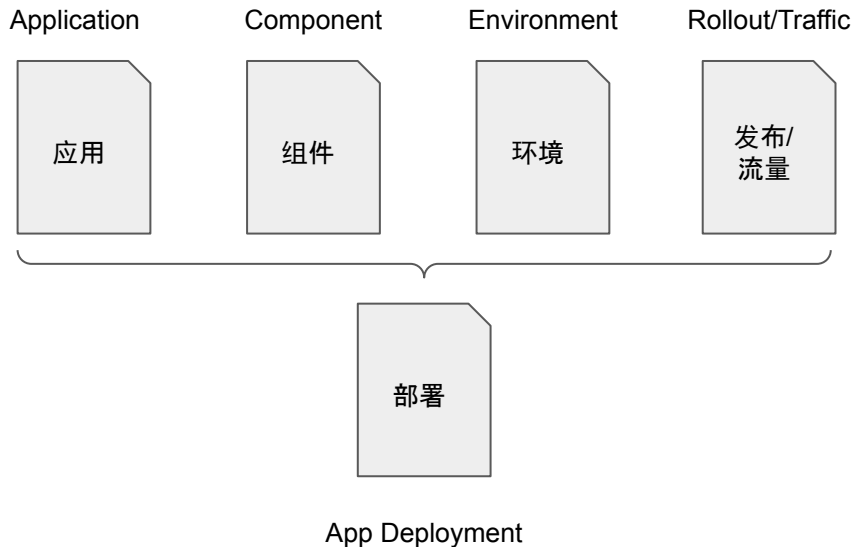
Satisfaction of
choosing GitOps



如果我有 10 个环境, 3000 个应用, 还能 GitOps 吗?

- 企业 PaaS/应用平台很难完全以 Git 为中心建设
- 完全通过 Git Layout 描述多环境、多集群、部署 Patch、发布策略等, 门槛很高
- 大规模场景下(比如部署环境和应用数量巨大), 上述问题会变得极其复杂

一种解决办法：面向多环境持续交付的封装与抽象

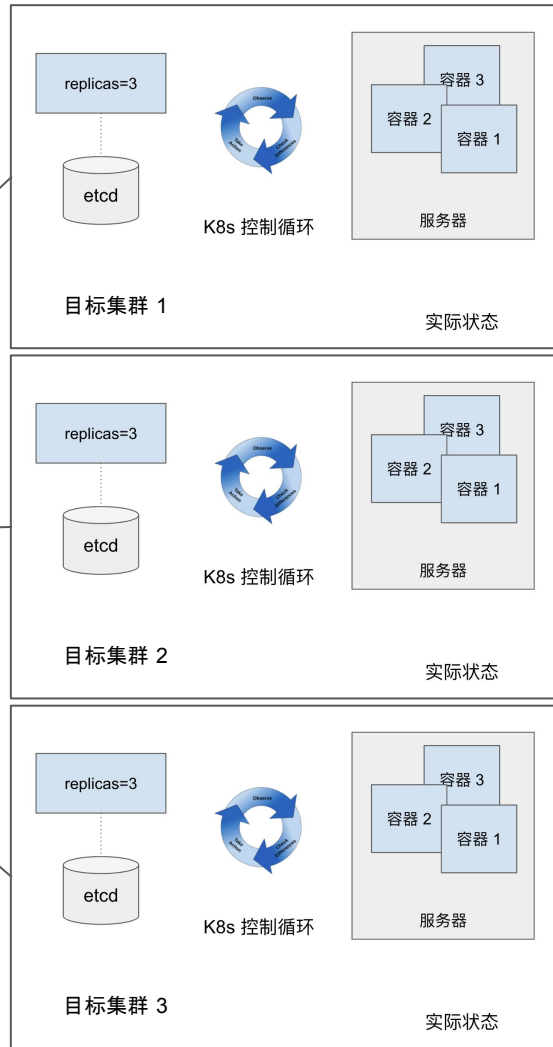
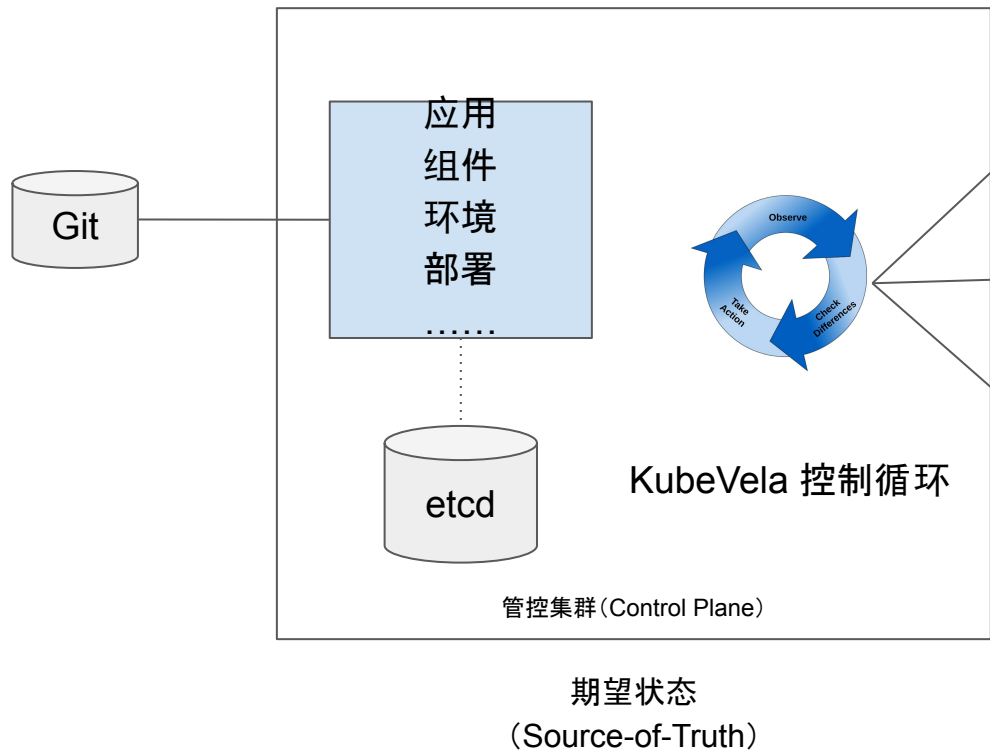


无论有多少个部署环境、部署 Patch 和发布策略，每个应用部署只需要有限的几个 YAML 文件即可完成描述，不依赖任何 Git Layout 或外部工具

```
kind: ApplicationDeployment
name: example-deploy
spec:
  appRevisions:
    - name: v1
      template: # applicaiton template
        spec:
          components:
            - name: backend
              type: worker
              settings:
                image: myimage-v1
                cmd: ...
              traits:
                - name: autoscaler
                  properties:
                    min: 1
                    max: 10
          rollout: # progressive rollout plan
            replicas: 10
          placementStrategies: # placement strategy
            - clusterSelector:
                labels:
                  usage: production
          ...
```


“不依赖 Git 的 GitOps”


KubeVela 取代 Git 控制器来充当应用描述与目标集群之间的控制循环



OAM 社区交流群

2150 人



 扫一扫群二维码，立刻加入该群。

Thank You