海联金汇 HYUNION HOLDINGS | 金融科技集团 FINTECH GROUP

第一期:docker&kubernetes

docker

kubernetes

Docker

- 容器
- 什么是Docker
- 优点

小　故　事

- 本故事纯属虚构

- 小明是团队的开发，产品通知小明务必今天 18:00 前修复一个查询功能，小明开发用了1 个小时，本地自测没问题，开发环境测试没问题。

- 小明信心满满，中午吃饭前给提测了。

小　故　事

- 下午，测试MM找小明，说是WEB应用启动报错。
- 小明表示这一般是tomcat版本和jdk版本不兼容。
  1. 小明，看了下测试MM用的Tomcat版本，说用tomcat 8.0就ok
- 过一会，测试MM又来找小明了，说是这个web启动又报错。
- 小明表示这是Tomcat端口被占用了，简单，改下tomcat 的配置 就行。
  1. 小明表示，改server.xml，改成没人用的端口号，绝对没问题。
- 很好，赶在16:00，大佬没催前，测试环境终于测过了

```
used by: org.apache.tomcat.util.bcel.classfile.ClassFormatException: I
    at org.apache.tomcat.util.bcel.classfile.Constant.readConstant(C
    at org.apache.tomcat.util.bcel.classfile.ConstantPool.<init>(Con
    at org.apache.tomcat.util.bcel.classfile.ClassParser.readConstan
```

```
四月 03, 2019 11:18:08 上午 org.apache.catalina.co
严重: StandardServer.await: create[localhost:8005]
java.net.BindException: 地址已在使用
        at java.net.PlainSocketImpl.socketBind(Nat
        at java.net.AbstractPlainSocketImpl.bind(A
        at java.net.ServerSocket.bind(ServerSocket
```

小　故　事

- 上线，小明找到运维同事，说是应用紧急上线。

- 今天有毒，紧急上线的比较多，轮到小明已经到了17:30。
- 小明表示这是两个应用，四台机器，比较简单，半个小时应该足够了。

- 过一会，运维GG通知小明了，说是这个web启动失败。

- 小明一个java –version命令，一句卧槽，生产的Jdk版本 是谁 随便改的?
  1. 小明表示，我在各个应用的启动脚本指定Jdk版本吧!

- 在18:10，生产环境上完了。

- 产品同事验收

iver : Unsupported major.minor version 52.0

rce)

小　故　事

- 小明没在截止时间上完线，被扣绩效

- 环境问题有点过分
  1. 不一致
  2. 不规范

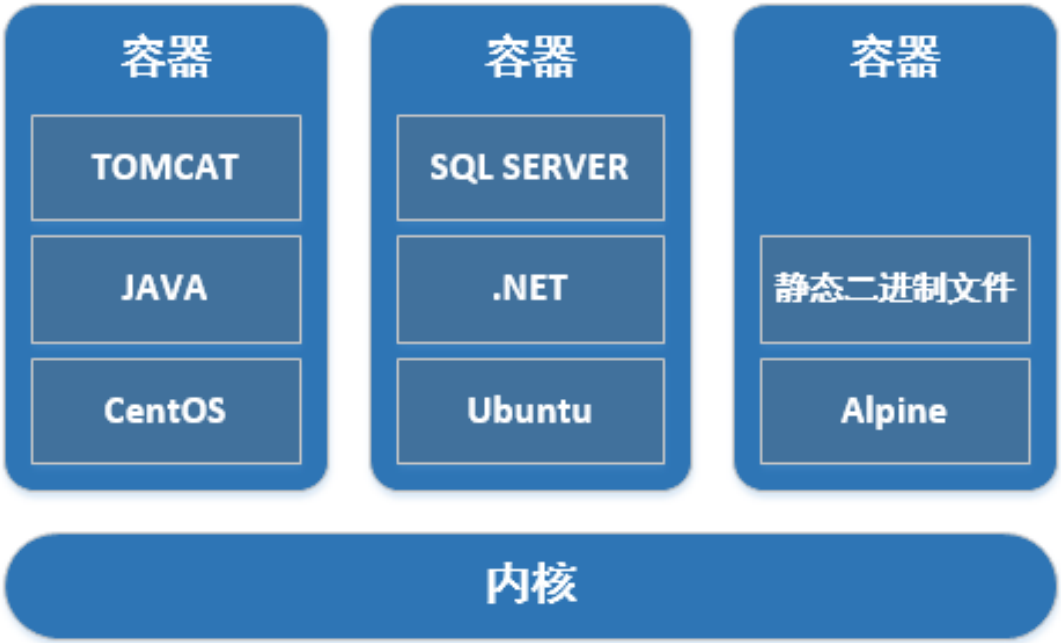
- 交付慢
  1. 出现问题，开发测试运维沟通（得找开发确认）
     - 测试不知道应用该用什么版本的tomcat
     - 运维不清楚应用该用哪个版本的JDK
  2. 人手动执行脚本
     - 熟悉应用的和不熟悉的
     - 人数量有限，同一时间执行的任务数有限

- 有没有一种方案让我开发 在开发环境没问题，测试，生产环境也没问题（小明主要遇到的）

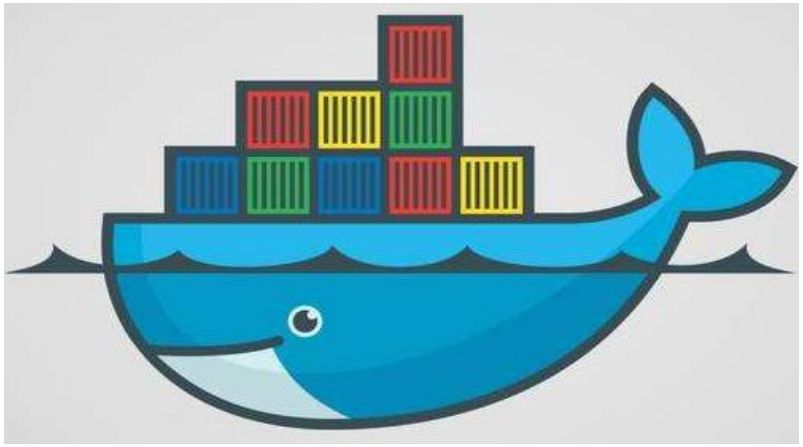
容　器　技　术

- 是技术手段
  1. 解决由于环境不一致导致应用出现的各种问题

- 怎么解决
  1. 打包（我需要的我都要）
  2. 将应用需要的Jdk，tomcat，甚至linux文件目录打包到一起
  3. 这个包能运行起来

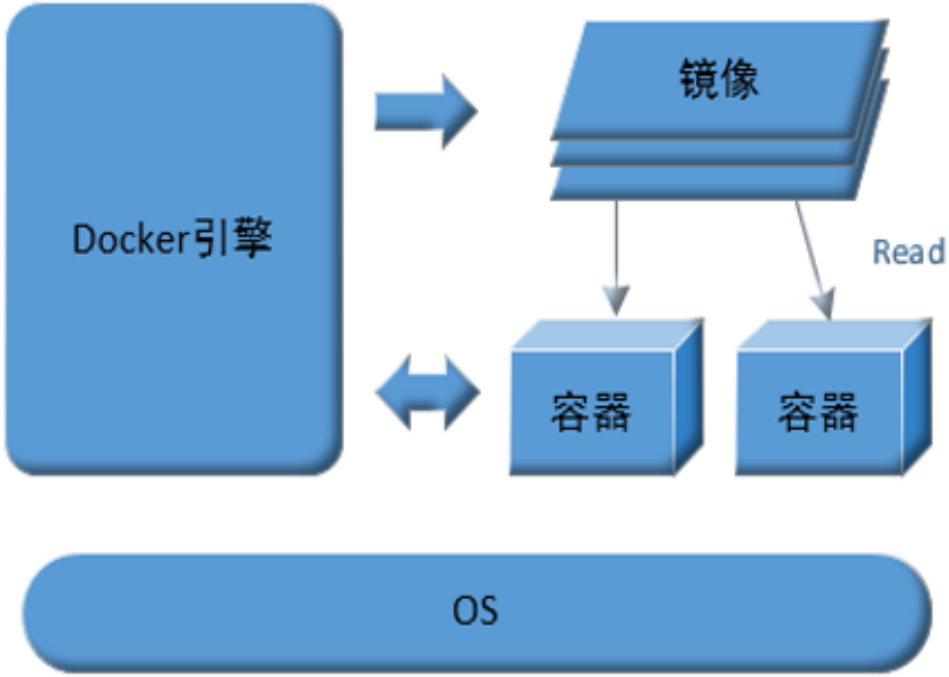


Docker



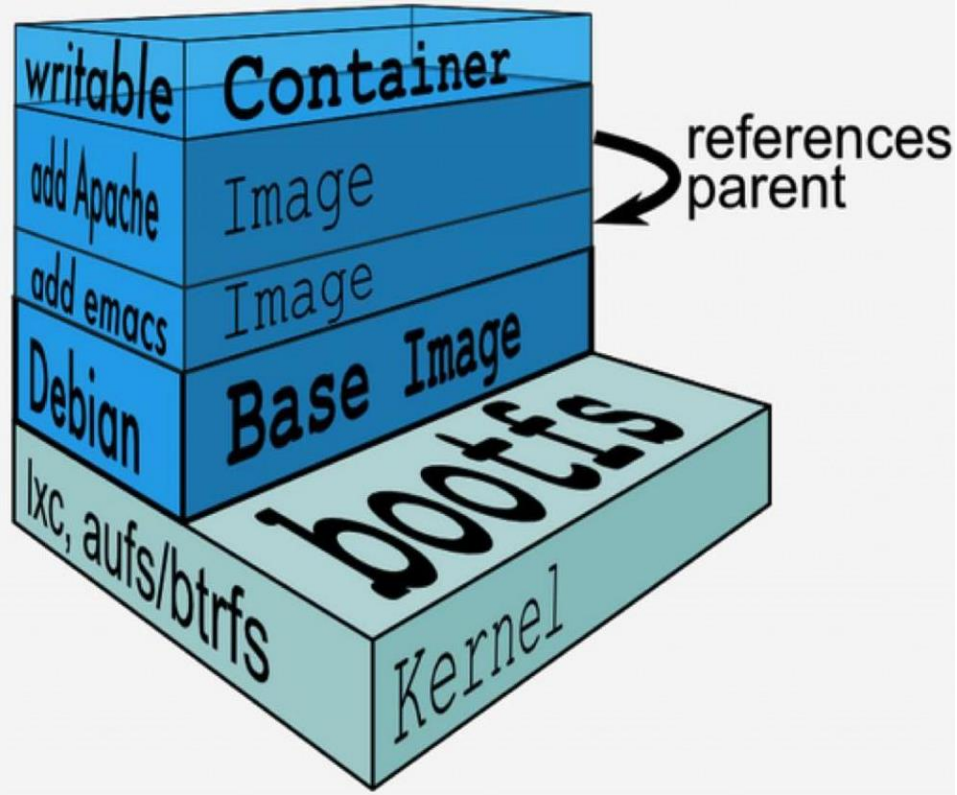
- 是工具
  1. 容器技术领先的平台
  2. 提供了简单高效使用容器技术的方式
     - ✓ 便利的打包
     - ✓ 方便的使用包，即让包运行起来，有提供服务的能力
  3. 标准化（交付镜像，到处运行）

- 说明
  1. 镜像（image）：
     - ✓ 打包后生成的东西
     - ✓ 一个可执行包
  2. 容器（container）：
     - ✓ 包运行起来可提供服务能力的实体
     - ✓ 运行时



Docker容器镜像

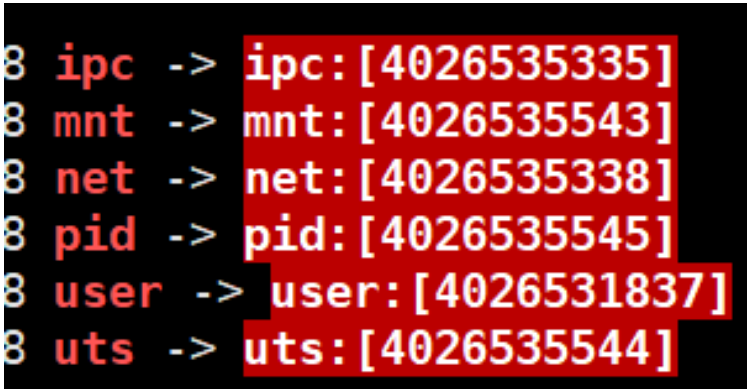
- 问题
  - 环境不一致，应用起不来
- 打包
  1. 把应用和应用需要的依赖打包封装到一起。包括操作系统的文件和目录（RootFS）
  2. 分层存储，一层一层构建
- 是静态的层 文件系统
- 保证了
  1. 强一致性
  2. 增量，复用

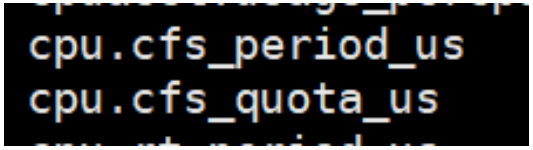


Docker容器运行时

- 是运行中的实例
  1. 是应用启动后的进程
  2. 容器可以被启动、开始、停止、删除
  3. 每个容器都是相互隔离的、保证安全的平台

- 隔离和限制
  1. Namespace
     ① /proc/[进程号]/ns
  2. Cgroups
     ① mount -t cgroup

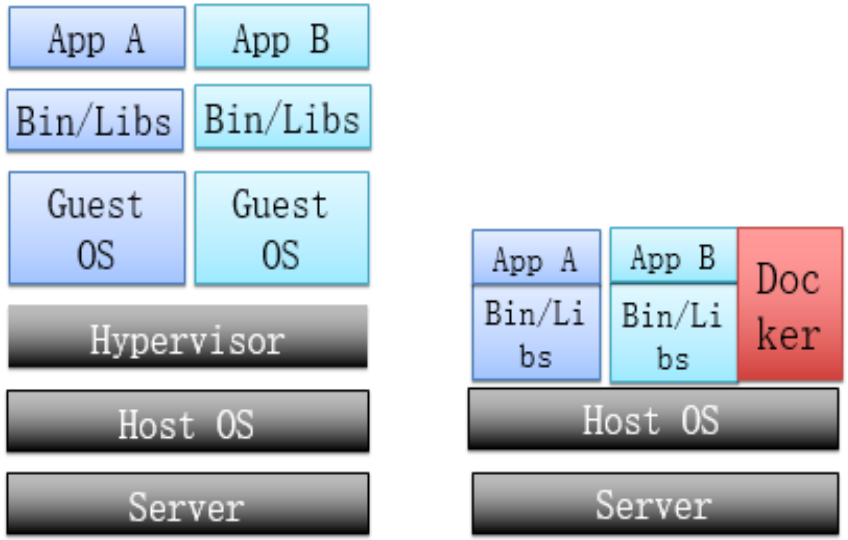





```
cgroup on /sys/fs/cgroup/memory type cgroup (rw,nosuid,nodev,noexec,relatime,memory)
cgroup on /sys/fs/cgroup/cpu,cpuacct type cgroup (rw,nosuid,nodev,noexec,relatime,cpuacct,cpu)
```

优　　点

- 一致的运行环境：Docker的镜像提供了除内核外完整的运行时环境，确保了应用运行环境一致性，从而不会再出现“这段代码在我机器上没问题啊”这类问题
- 迁移方便
- 持续交付和部署
- 隔离性
- 敏捷和高性能：直接调用宿主机操作系统，不需要经过虚拟化软件的拦截处理



虚拟机和Docker技术比较

面向Docker编程

- 实战

1. Dockerfile

   FROM：指定基础image

   CMD：设置container启动时执行的操作

2. docker build 命令

   记得带 . (点)

3. docker run 命令

```
[ tomcat]# ll
total 4
drwxrwxrwx 9 mpsp develop 220 Apr  3 16:06 apache-tomcat-7.0.92
-rwxrwxrwx 1 mpsp develop 293 Apr  3 18:16 Dockerfile
```

```
# Dockerfile for apache-tomcat-7.0.92

# Build with:
# docker build -t apache-tomcat-7.0.92:v1.0.0 .
# docker run -di -p 10080:8080 apache-tomcat-7.0.92:v1.0.0


FROM openjdk:8-jre-alpine
MAINTAINER haokailin <haokailin@umfintech.com>

RUN mkdir -p /usr/mpsp

ADD . /usr/mpsp/

EXPOSE 8080

CMD ["/usr/mpsp/apache-tomcat-7.0.92/bin/catalina.sh","run"]
```

```
[root@k8s-master tomcat]# docker ps -a|grep tomcat
2169c8add262        apache-tomcat-7.0.92:v1.0.0
     0.0.0.0:10080->8080/tcp   naughty_swartz
```

小　故　事

- 容器这么流弊，小明用上了容器技术，引入Docker，采用docker交付，部署，不再受应用版本等问题的困扰

- 但是新的风暴将要出现，此时小明却对他一无所知

小　故　事

- 随着业务量的增大，小明负责的应用拆分为数个服务，如，用户管理，账户管理，支付管理，产品管理，风险管理等，
- 20个容器，10台机器，没大问题，写好脚本的话hold住
- 安稳的过了几个月，小明接了个大工单，上面的服务都得动（双11来了）
- 11月9号，小明上线完应用，还扩了两台支付管理，根据往年来看，应该刚的住
- 领导问：小明，咱们这系统能顶住吧。
- 小明：No Problem！（内心，卧槽，领导都发问了，再扩两台，绝对没问题吧，容器宝宝们）

小　故　事

- 那么很显然，今年流量突增，半夜服务炸了。小明急忙先通知运维紧急重启，顺便赶到公司去修复，为了能及时解决问题，小明得在值班室一直盯着监控，看日志，看流量。

- 小明想申请机器，申请内存，申请cpu，申请数据库连接，提高服务质量，半夜。。。流量高峰做这些。。。

- 小明没办法，遇到问题，赶快切流量重启恢复服务。

微笑中透露着疲倦

- 早上，小明修仙归来

- 领导对小明说：昨晚辛苦啦
- 小明：还好

小　故　事

- 小明没完成定下的目标，觉悟经过上次的事情有所提高，决定自己扣绩效

- 容器管理问题
  1. 高可用（一台炸了还能有服务）
  2. 灾难恢复（一台炸了能重启恢复服务）
  3. 可伸缩性（到了一定阈值，能扩服务给分摊压力）
     服务能力不强，不够流弊
  4. 资源调度（内存，cpu等）

- 那么有没有一种方案，能自动部署，扩展和管理容器化应用程序呢？能更好的管理资源？能提供更强的服务能力？

kubernetes

- 什么是kubernetes
- 一次调度编排过程
- 优点

Kubernetes!

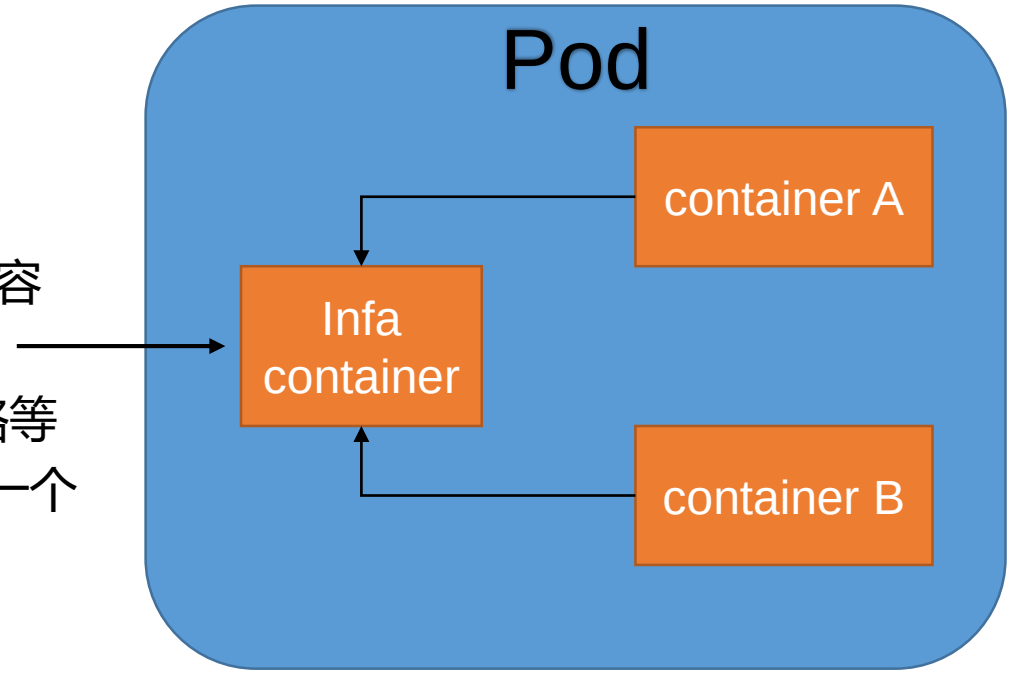
- 是一种机制
  1. 管理的对象是容器（容器化的应用程序）
  2. 应用部署，规划，更新，维护
  3. 目的：提供更强的服务能力

- 诞生说明
  1. Google、RedHat 大佬支持
  2. 15年在 Google运行生产工作负载的经验
  3. 拥有允许Google每周运行数十亿个容器的相同设计原则
  4. 开源

Kubernetes对象介绍 → Pod

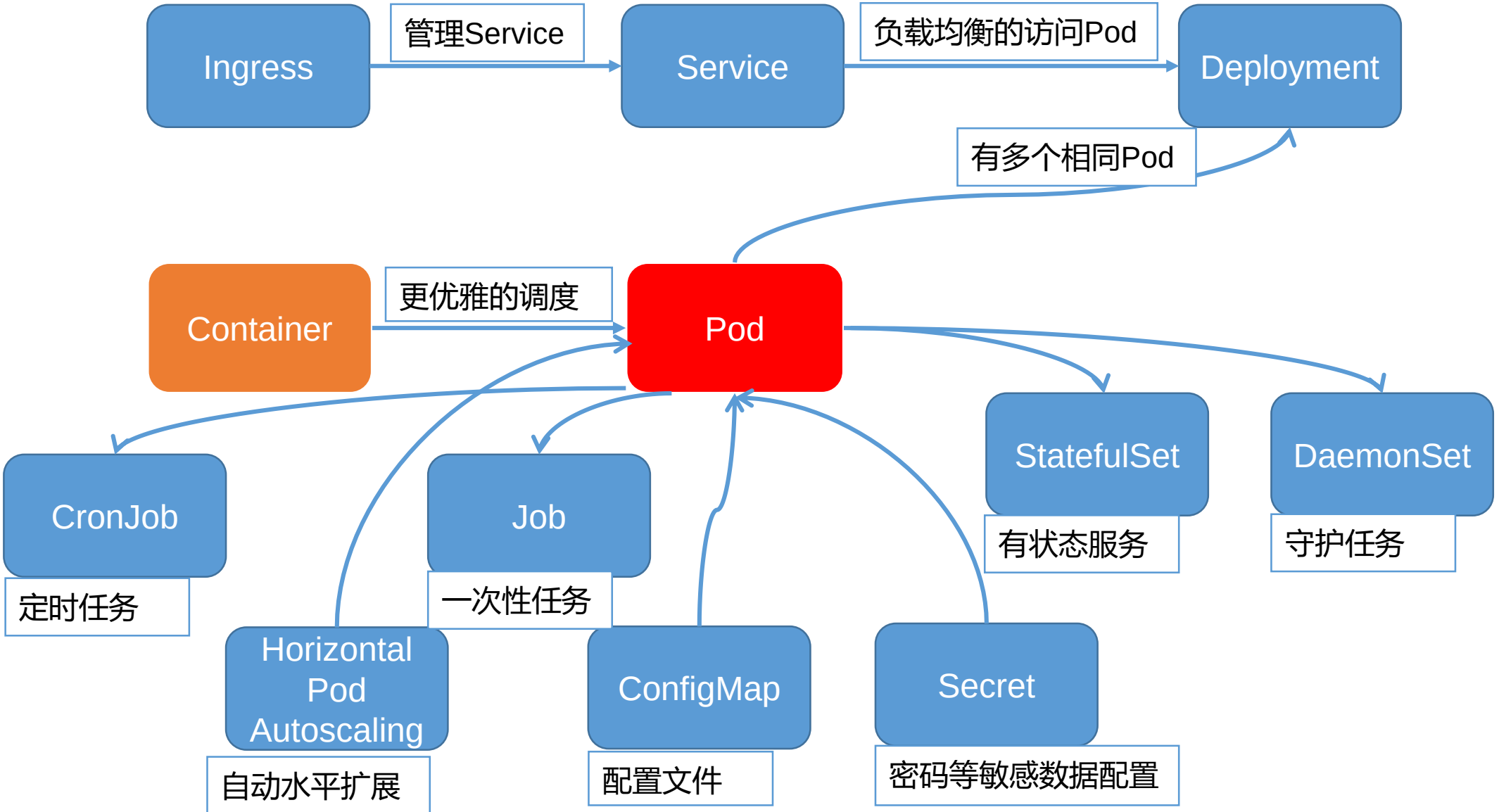
- 是kubernetes中最小调度单元
  - ◆ 管理 容器

- 需求
  1. 容器技术有Docker、rkt等，需要抽象出对容器管控的单元（接口）
  2. 管理容器使用的资源如，cpu，内存，网络等
  3. 管理容器之间的关系，例如多个容器提供一个服务（比如tomcat和war包）



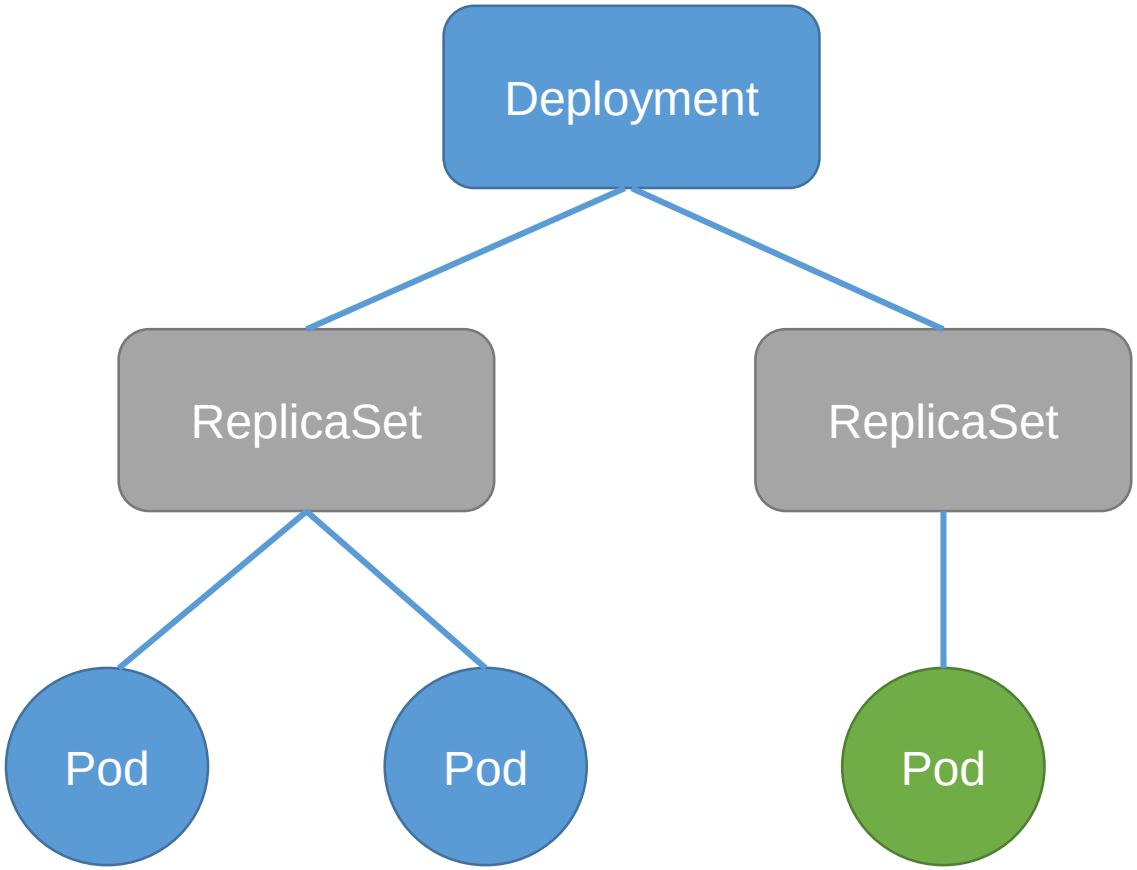
- 更比较正统的介绍
  1. 容器 --》 进程 --》人
  2. Pod ---》进程组 --》 一起工作的同事
  3. Kubernetes --》操作系统

Kubernetes对象介绍

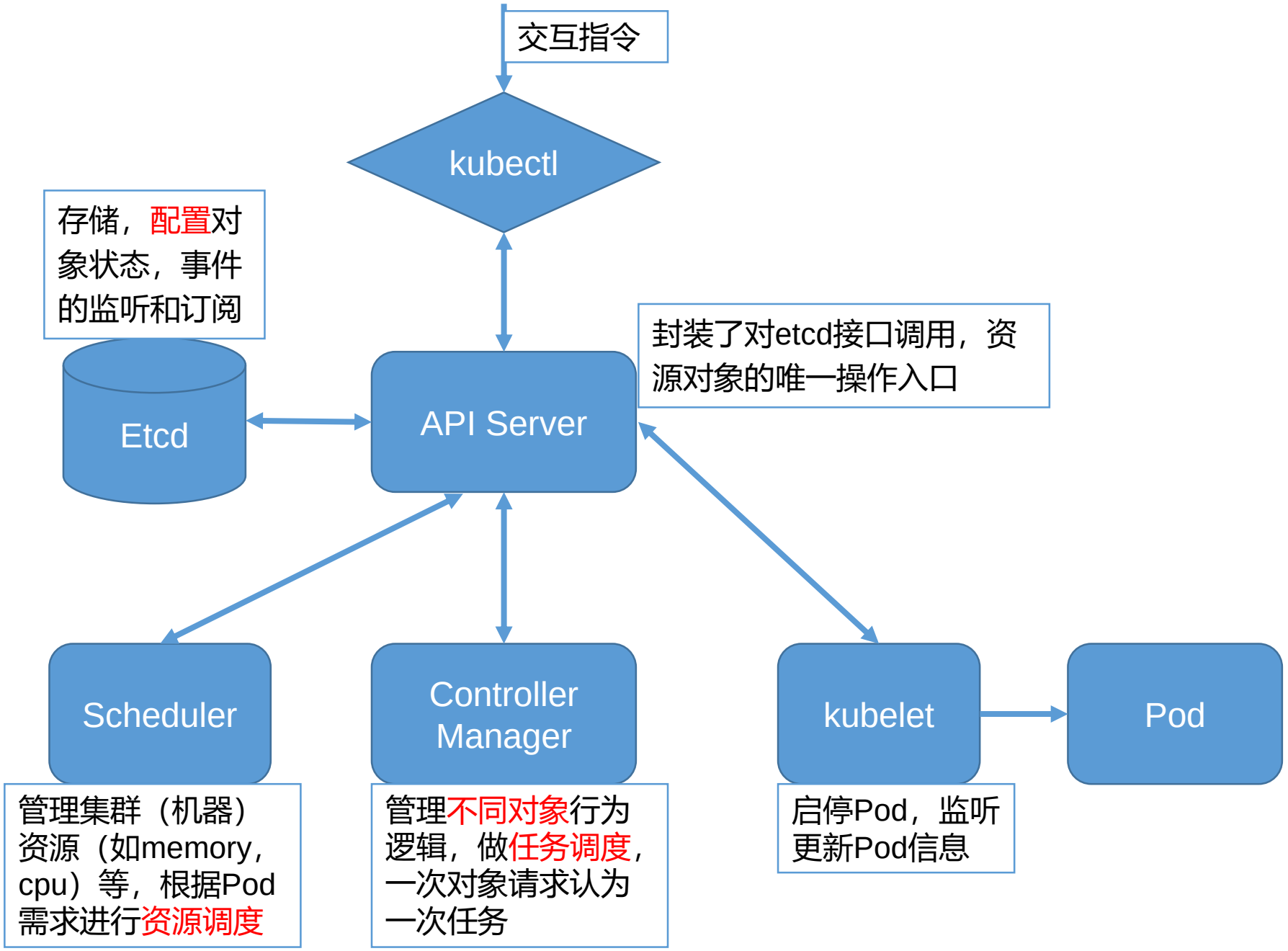


Kubernetes对象介绍 → Deployment

- 需求
  - 灾难恢复，水平扩缩（Pod数量控制）
    ReplicaSet控制运行Pod的数量
  - 版本控制
    Deployment控制ReplicaSet的个数，描述应用的版本。
- 两层控制



Kubernetes 架构设计！



面向kubernetes编程

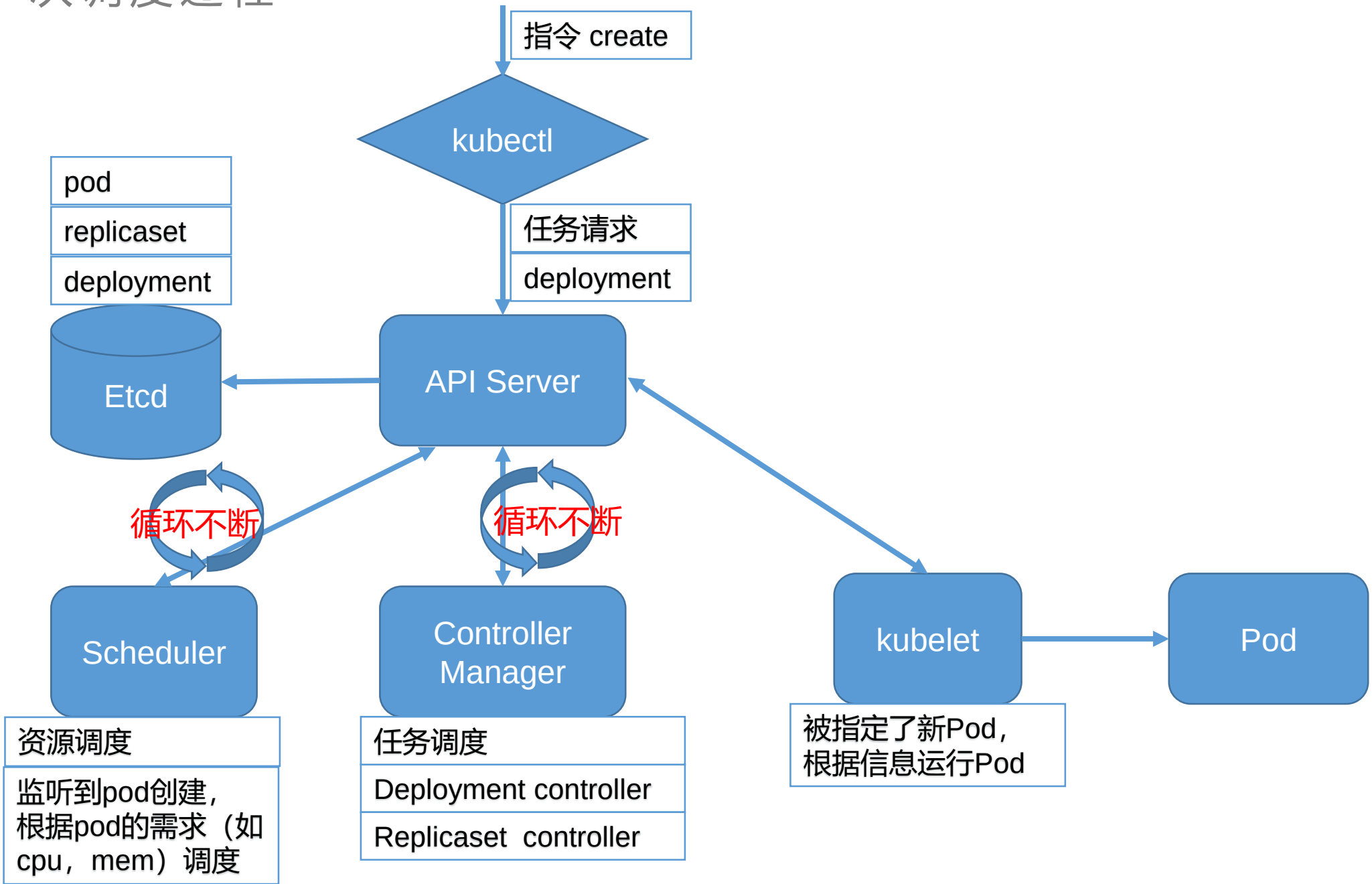
- Yaml文件

kubectl create –f tomcat_deployment.yaml

kubectl apply

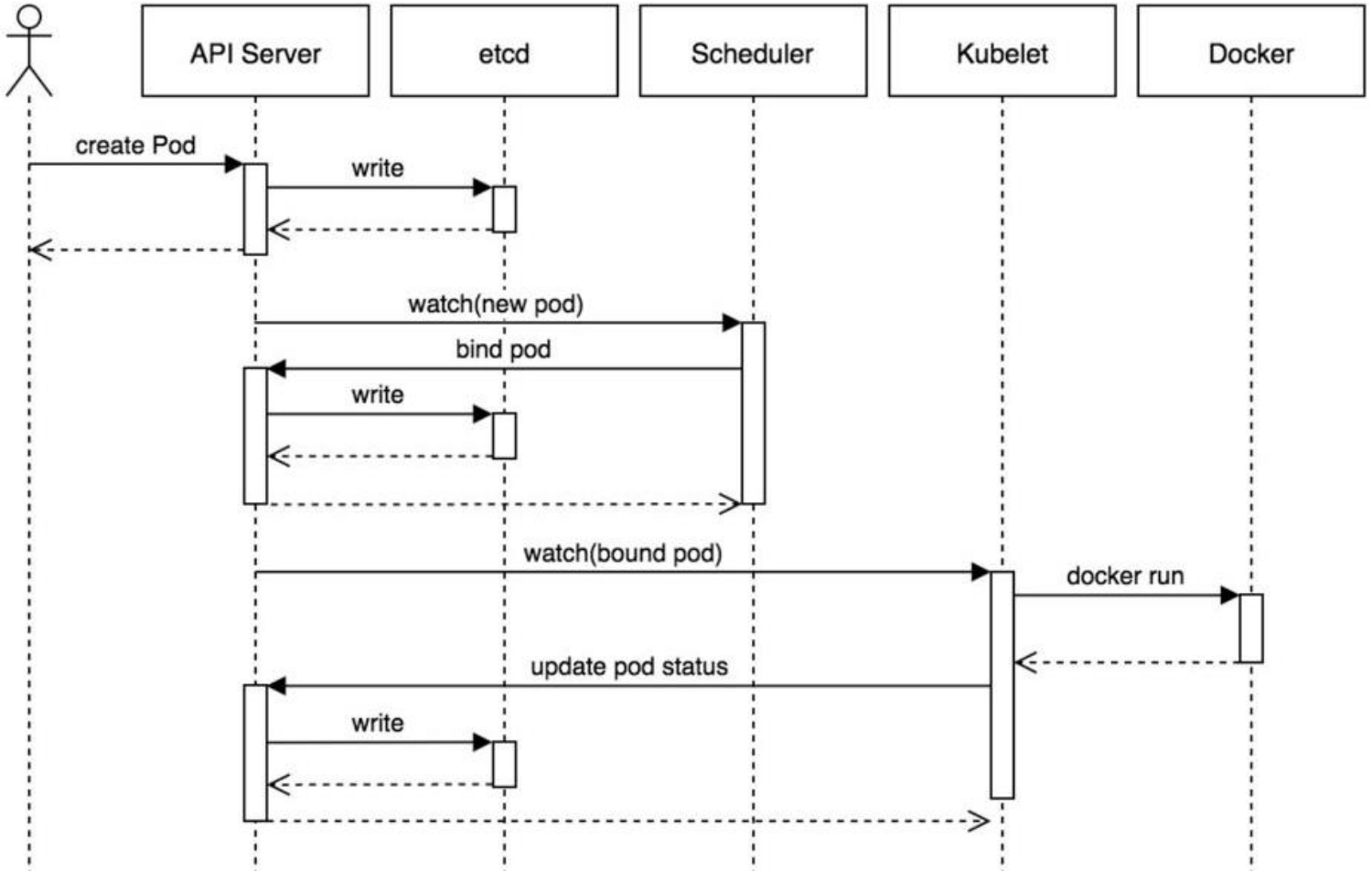
```
apiVersion: apps/v1
kind: Deployment                #类型
metadata:
  name: my-tomcat               #Deployment名称
  #namespace: my-tomcat         #命名空间
spec:
  selector:
    matchLabels:
      run: my-tomcat
  replicas: 2                   #两个pod
  template:
    metadata:
      labels:
        run: my-tomcat
    spec:                       #定义容器模板，该模板可以包含多个容器
      containers:
      - name: my-tomcat                                          #镜像名称
        image: 10.10.56.148:5000/apache-tomcat-7.0.92:v1.0.0     #镜像地址
        imagePullPolicy: IfNotPresent                            #镜像拉取策略
        resources:                                               ##CPU内存限制
          limits:
            memory: 0.5Gi
            cpu: 200m
          requests:
            memory: 0.5Gi
            cpu: 200m
        ports:
        - containerPort: 8080                                    #暴露端口
```

一次调度过程


指令 create
kubectl
pod
replicaset
deployment
任务请求
deployment
Etcd
API Server
循环不断
循环不断
Scheduler
Controller Manager
kubelet
Pod
资源调度
监听到pod创建，根据pod的需求（如cpu，mem）调度
任务调度
Deployment controller
Replicaset controller
被指定了新Pod，根据信息运行Pod

Pod 调度过程

- 实例

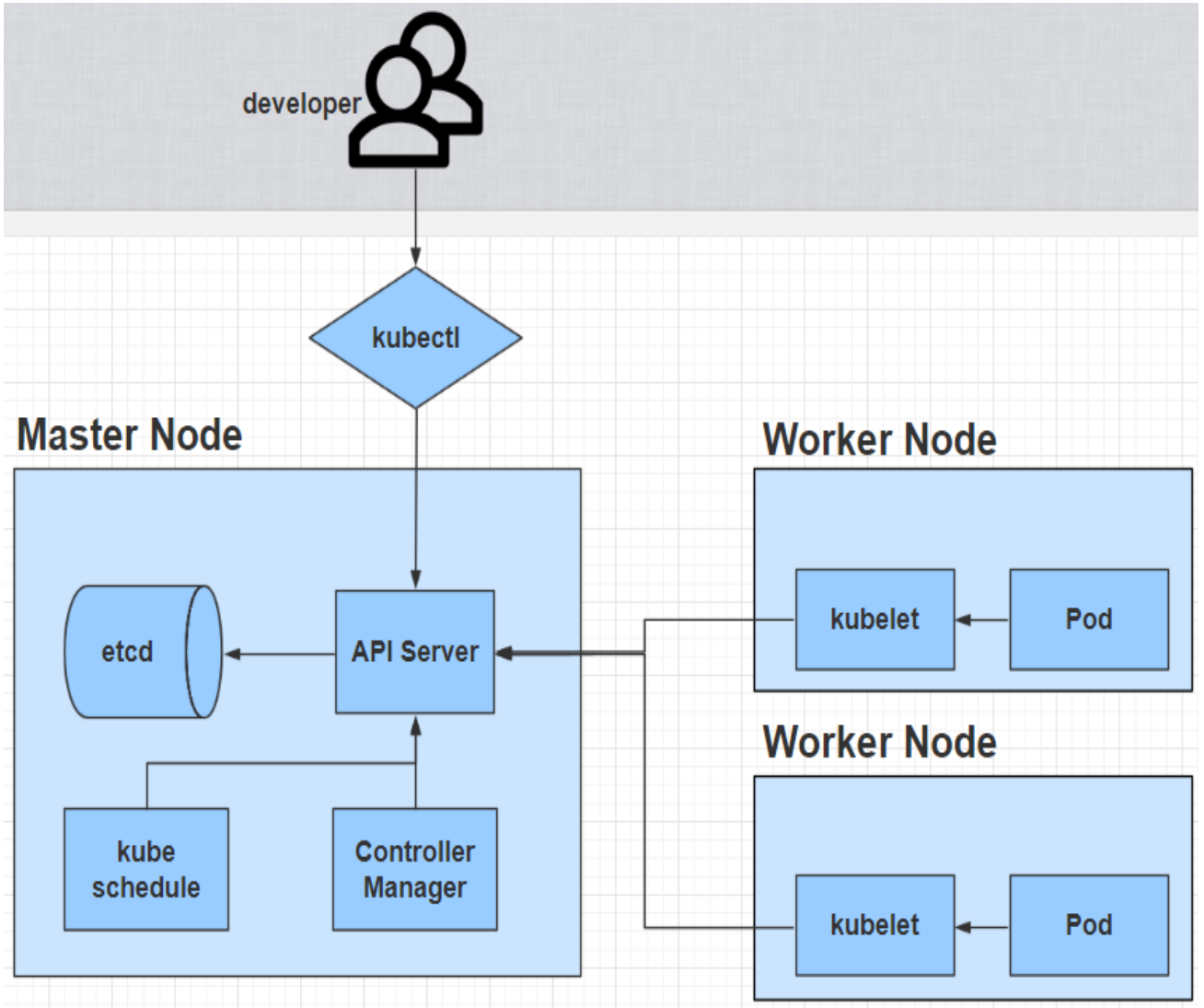

API Server
etcd
Scheduler
Kubelet
Docker
create Pod
write
watch(new pod)
bind pod
write
watch(bound pod)
docker run
update pod status
write

Kubernetes 集群架构!

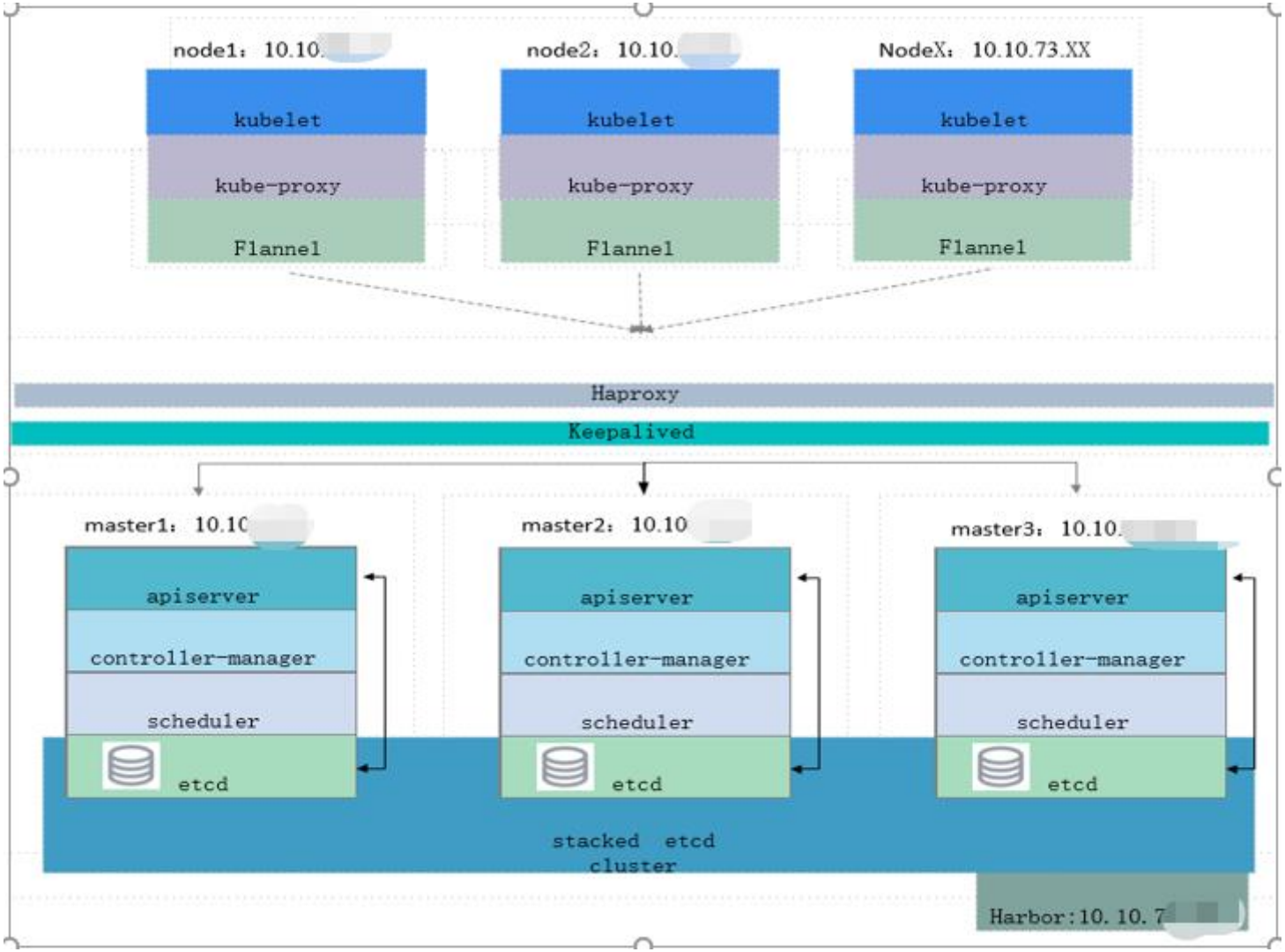
- 是集群架构

1. Master Node（主节点），总控中心，集群管理节点

2. Worker Node（工作节点），理解为一台虚拟机或者物理机，“逻辑主机”，业务应用容器跑在Node上面



Kubernetes 高可用集群架构！



K8s交付

- 思考
- Kubernetes交付
- 推荐

常 见 的 问 题 ？

- 开发
  1. 开发环境（redis，zk，etcd等）炸了
  2. 有人正在打包，这套环境 xxx 占用了
  3. 本地调试贼慢，启动贼慢
  4. …..
- 测试
  1. 没机器，这一套环境有人在用
  2. 有环境，调用其他CRM，通道等服务不通
  3. 中间件（redis，zk）炸了
  4. 数据库存储满了
  5. ……
- 表现在外：慢
- 上线
  1. 有人在上线，等上线完这个就上你的
  2. 这边出了个生产问题
  3. 你的机器部署在哪里？怎么切负载?
  4. 部署平台上线慢（改造慢，配置机器慢，上线慢，出问题解决还慢）
  5. ….
- 解决问题
  1. 下一个就是你的日志，发邮件
  2. 应用炸了（redis，zk，业务应用），来看下存活监控，正在……恢复应用
  3. 数据库连接不够
  4. ……

思　　考

完成一个应用

- 开发需要打包应用，调环境，申请机器自测
- 测试需要协调资源，协调其他业务线（万一服务又不通了），根据应用情况控制排期
- 运维需要监控应用健康状态，调整任务优先级别（又炸了），优先保证能正常提供服务
- ……

开发者 在疯狂的解决问题

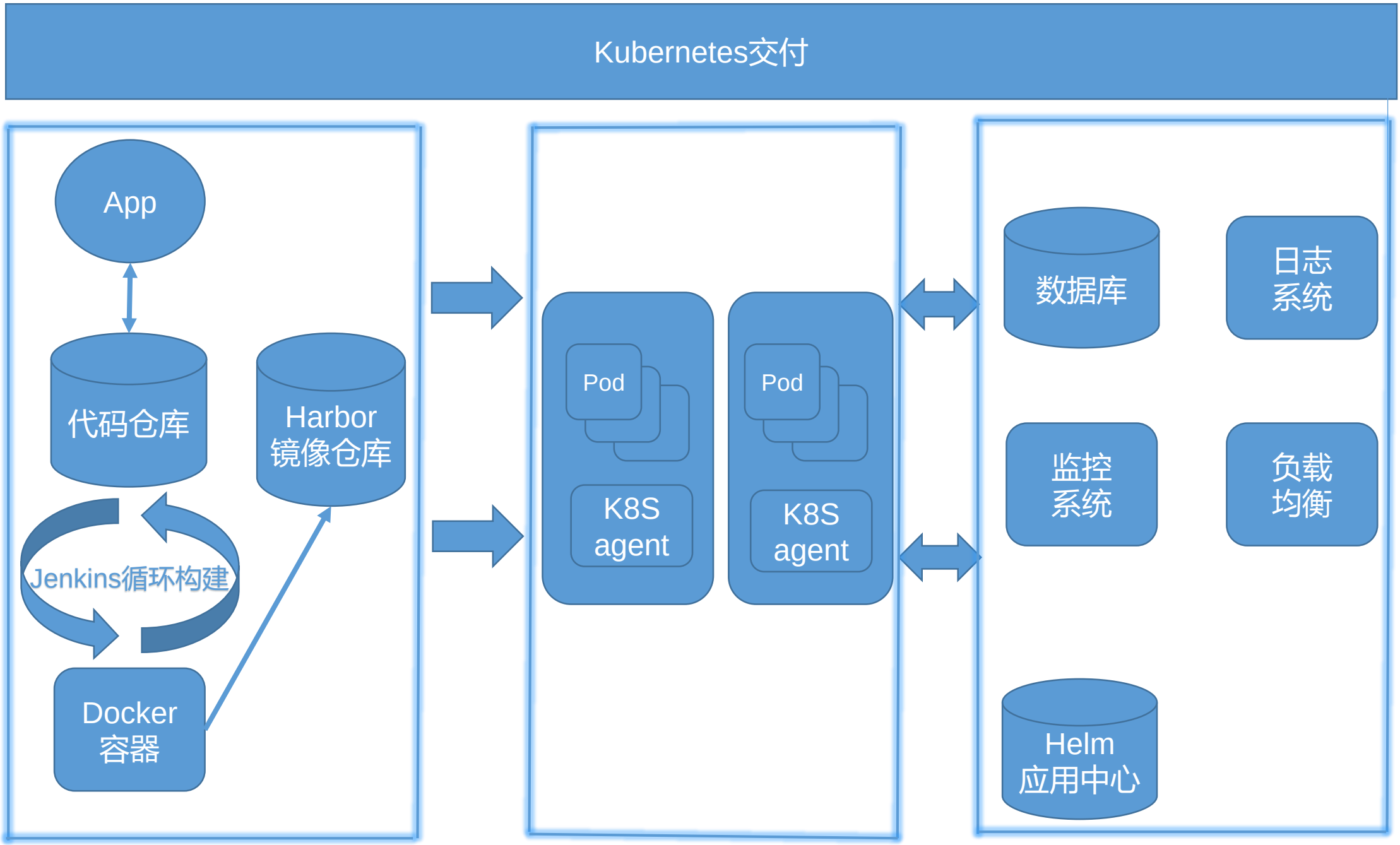
- Docker火的原因
- Kubernetes火的原因
- 结论

解放开发者的生产力

Docker解决打包难题

K8s解决管理难题

开发者关注开发或业务本身，构建杰出的软件


Kubernetes交付
App
代码仓库
Harbor
镜像仓库
Jenkins循环构建
Docker
容器
Pod
K8S
agent
Pod
K8S
agent
数据库
日志
系统
监控
系统
负载
均衡
Helm
应用中心

推　荐　！

- 链接推荐：
  1. Kubernetes官网
  2. Kubernetes中文社区
  3. 云栖社区k8s技术汇

- 重磅推出：
  1. 《CNCF x Alibaba 云原生技术公开课》
  2. 生于疼痛的阿里云

海联金汇k8s技术沟通群

Q & A


海联金汇
HYUNION HOLDINGS
共创·共享·共赢