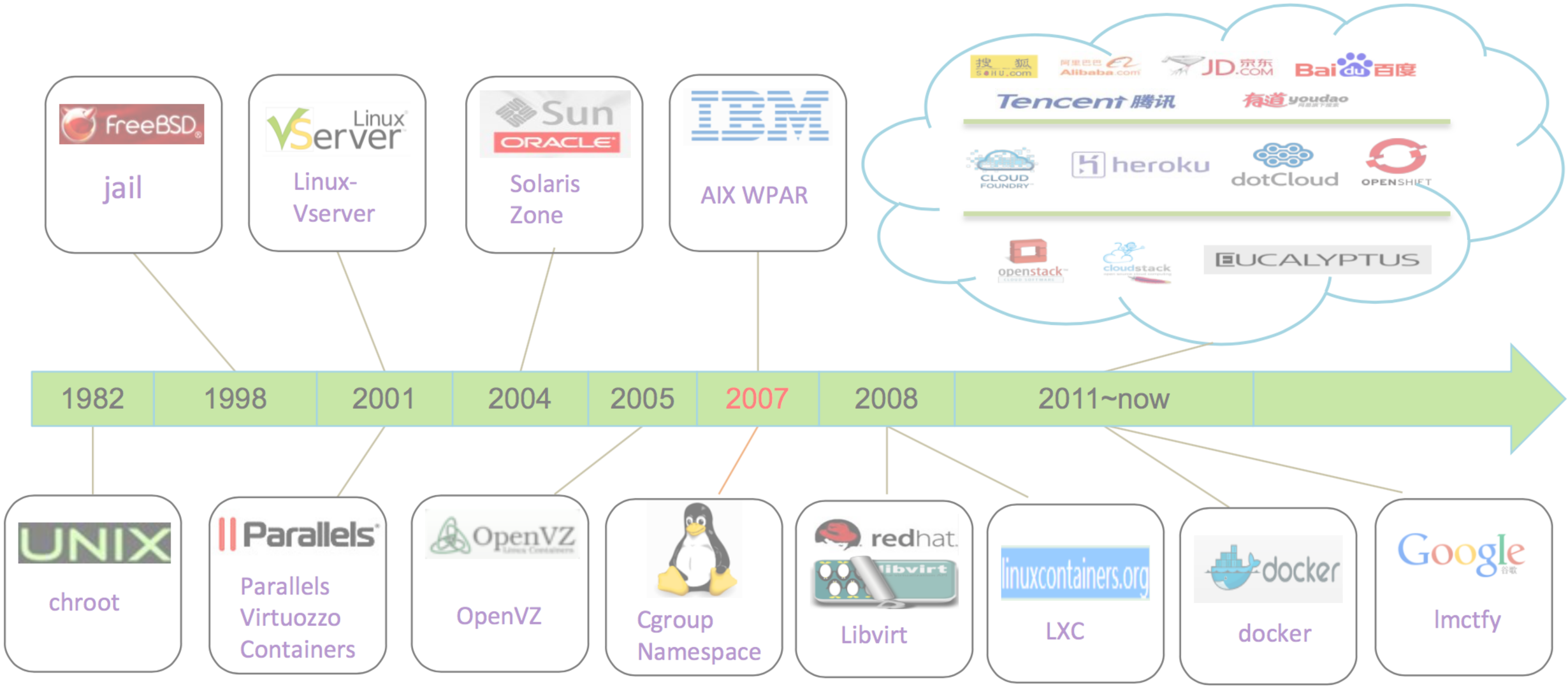
容器 - 以应用为中心的新一代虚拟化技术

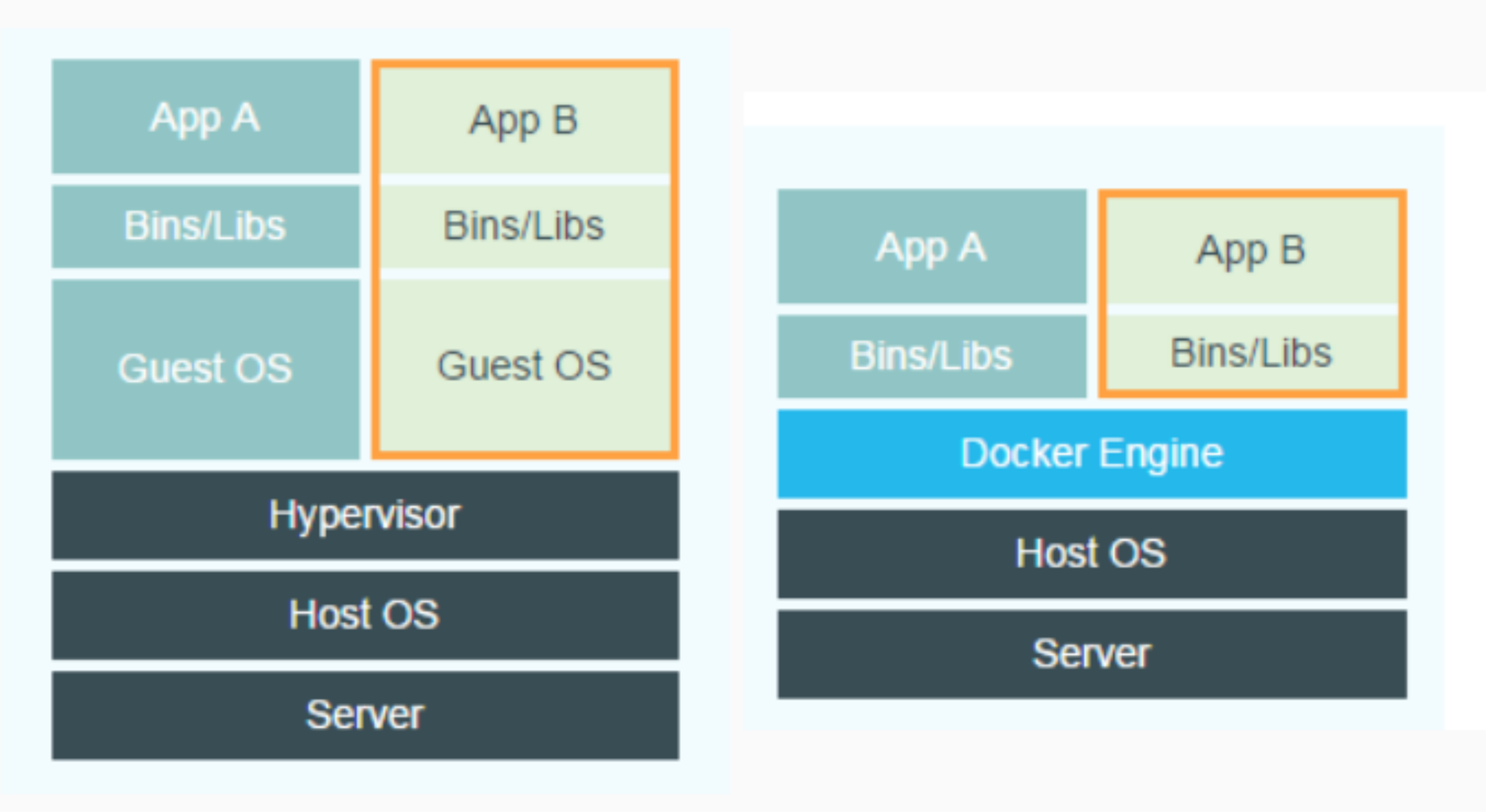
Agenda

- 初识容器技术
- 容器技术的优势
- 在企业中的应用场景
- 相关技术生态
- 未来的发展趋势

容器技术发展历史


FreeBSD
jail
Linux
VServer
Linux-
Vserver
Sun
ORACLE
Solaris
Zone
IBM
AIX WPAR
搜狐 SOHU.com
阿里巴巴 Alibaba.com
JD.京东 JD.COM
Baidu 百度
Tencent 腾讯
有道 youdao
CLOUD FOUNDRY
heroku
dotCloud
OPENSHIFT
openstack
cloudstack
EUCALYPTUS
1982
1998
2001
2004
2005
2007
2008
2011~now
UNIX
chroot
Parallels
Parallels
Virtuozzo
Containers
OpenVZ
OpenVZ
Cgroup
Namespace
redhat
libvirt
Libvirt
linuxcontainers.org
LXC
docker
docker
Google
lmctfy

容器 - 操作系统级虚拟化技术



虚拟机：

利用硬件虚拟化技术（VT-X, AMD-V), 通过Hyperviser层来实现对资源的彻底隔离。

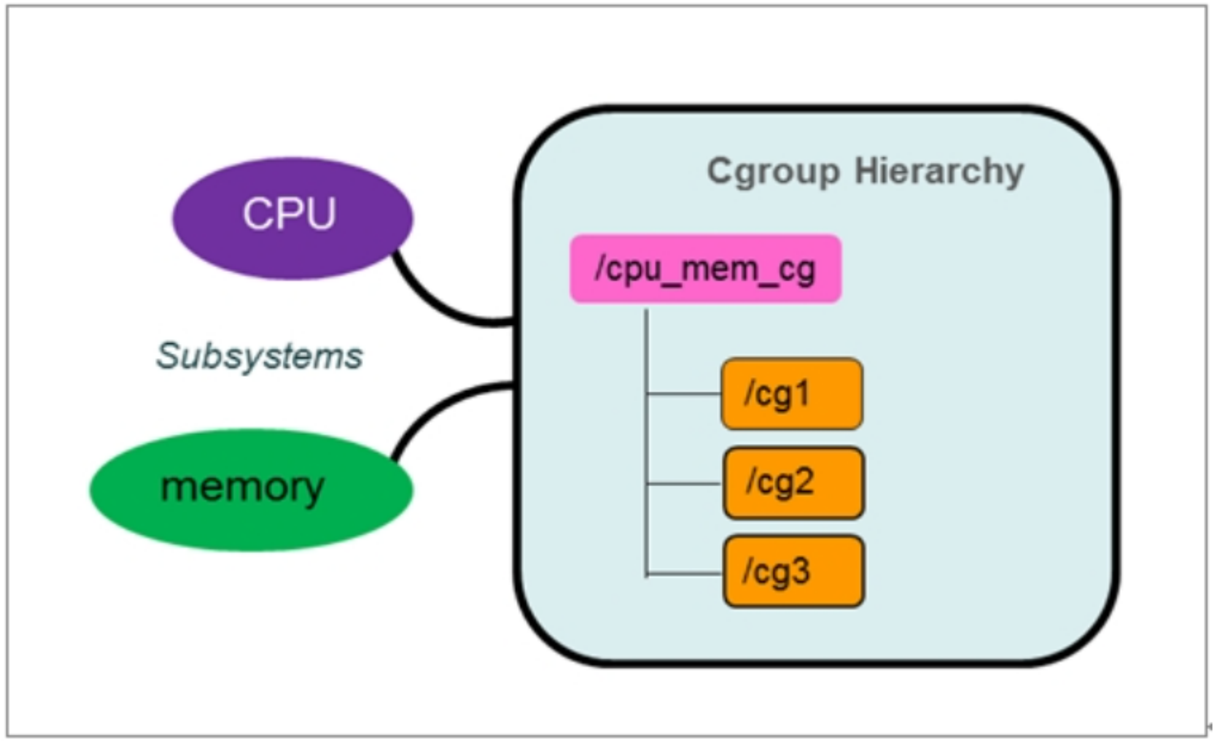
容器：

操作系统级别的虚拟化，通过利用Namespace和Cgroup进行资源隔离。

容器核心技术

- 使全局性的系统资源不再具有全局性，而是属于特定的命名空间
- 在一个容器环境内，最小化对用户其它环境的影响。

- UTS --- 隔离主机名和域名
- IPC --- 隔离IPC
- PID --- 隔离进程ID
- Mount --- 隔离文件系统挂载点
- User --- 隔离用户ID和组ID
- Net --- 隔离网络资源



容器 = namespace + cgroup + rootfs +容器引擎(用户态工具)

（访问隔离+资源控制+文件系统隔离+生命周期控制）

容器技术的优势 - 1

轻量级、速度快

	容器	*虚拟机*
大小	*几百k ~ 几百M*	*10G～几百G*
启动	*秒级*	*分钟级*
迁移	*不同Linux系统之间迁移*	*很难跨平台迁移*

- 仅需要包含最少的环境依赖，更小、更安全

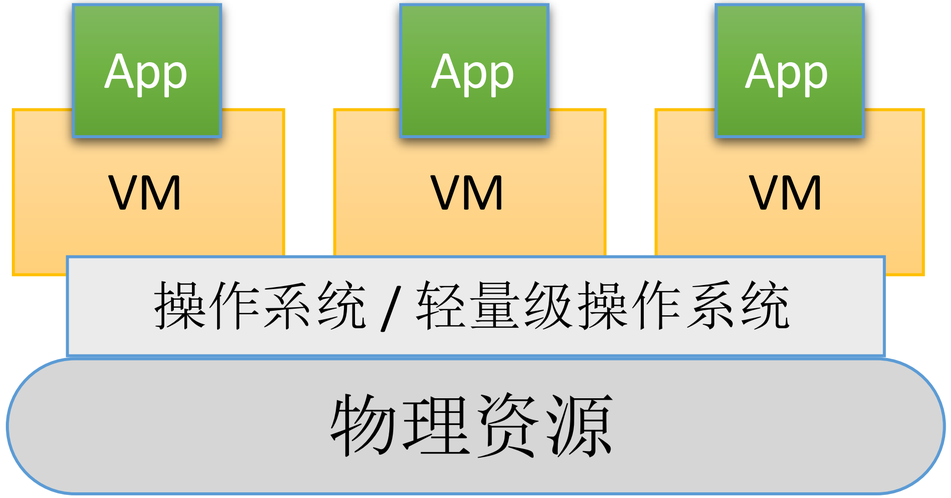
容器技术的优势 - 2

提高资源利用率、降低成本

- 共享系统内核
- 将资源更多的提供给应用
- 很小的overhead

CPU 60%
Memory 80%

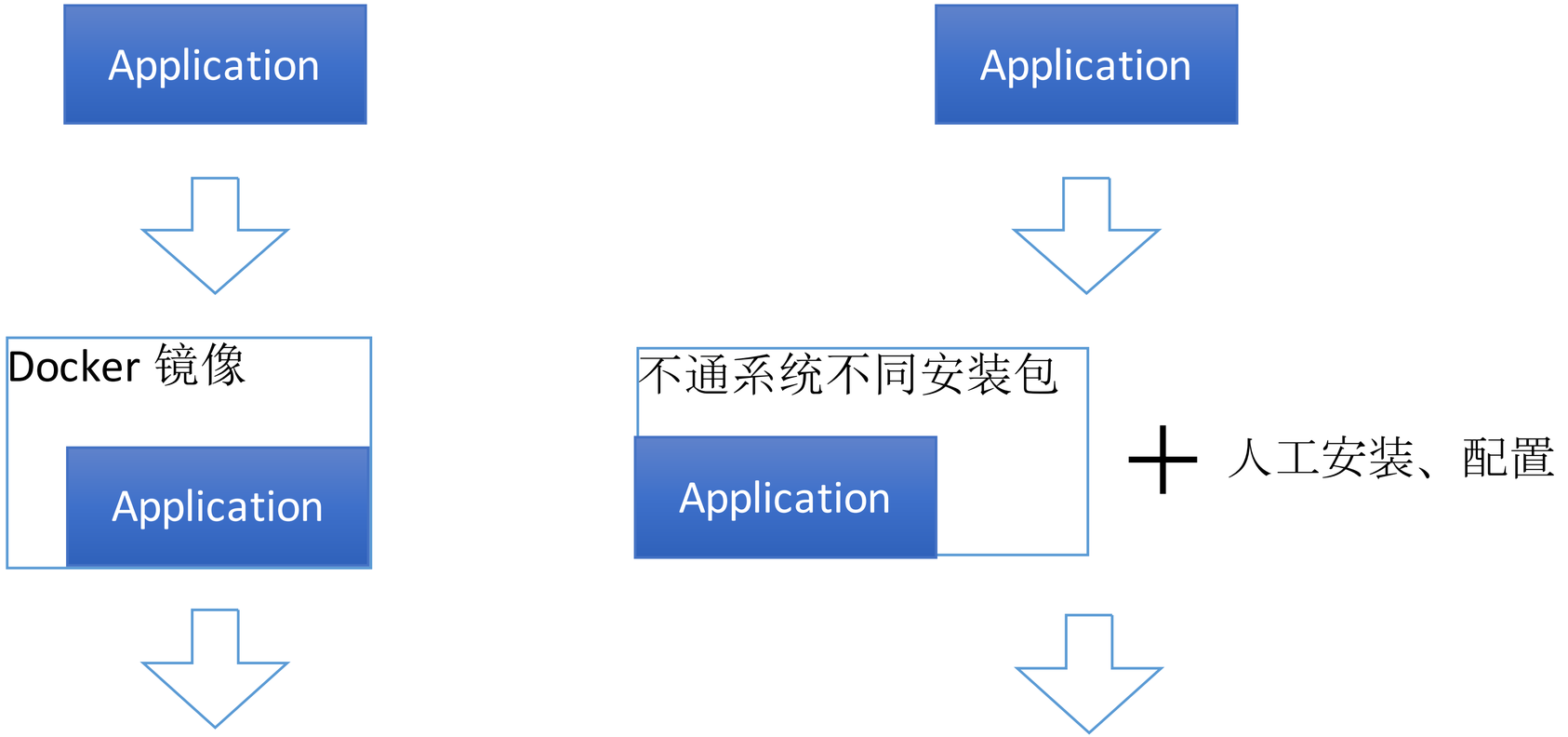
- 无法共享系统
- 虚拟机本身的overhead比较大
- 虚拟机层面的隔离仍然不能充分利用资源



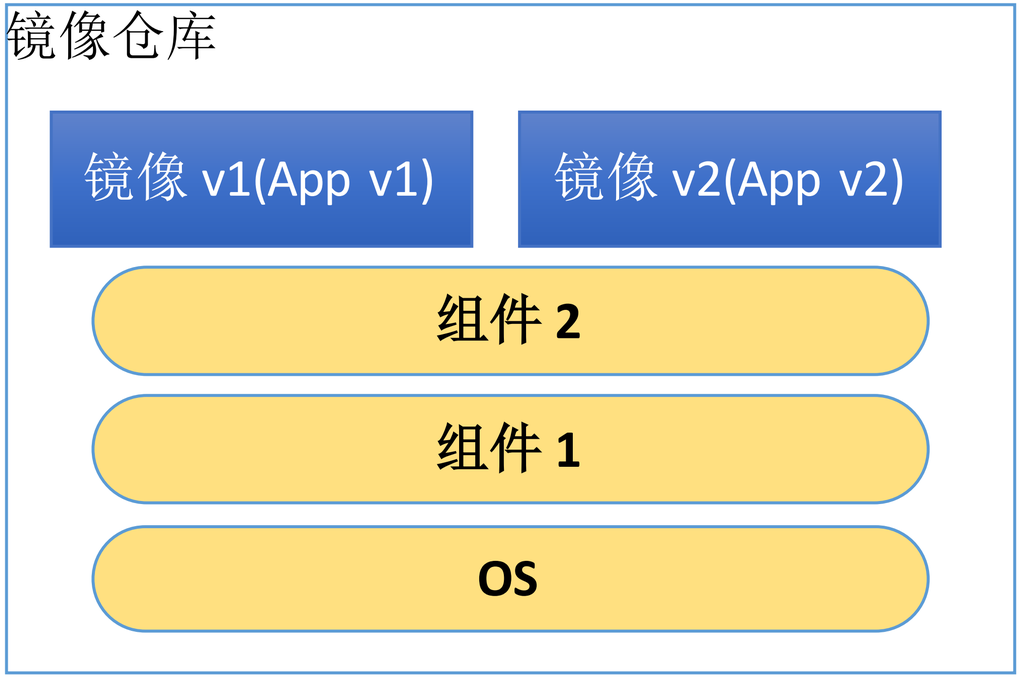
CPU ~15%
Memory ~ 50%

容器技术的优势 - 3

标准化、快速应用发布



容器技术的优势 - 4



- 标准镜像格式、分层共享
- 轻松实现应用版本切换
- 一致的发布、部署方式
- 开发、运维统一视角

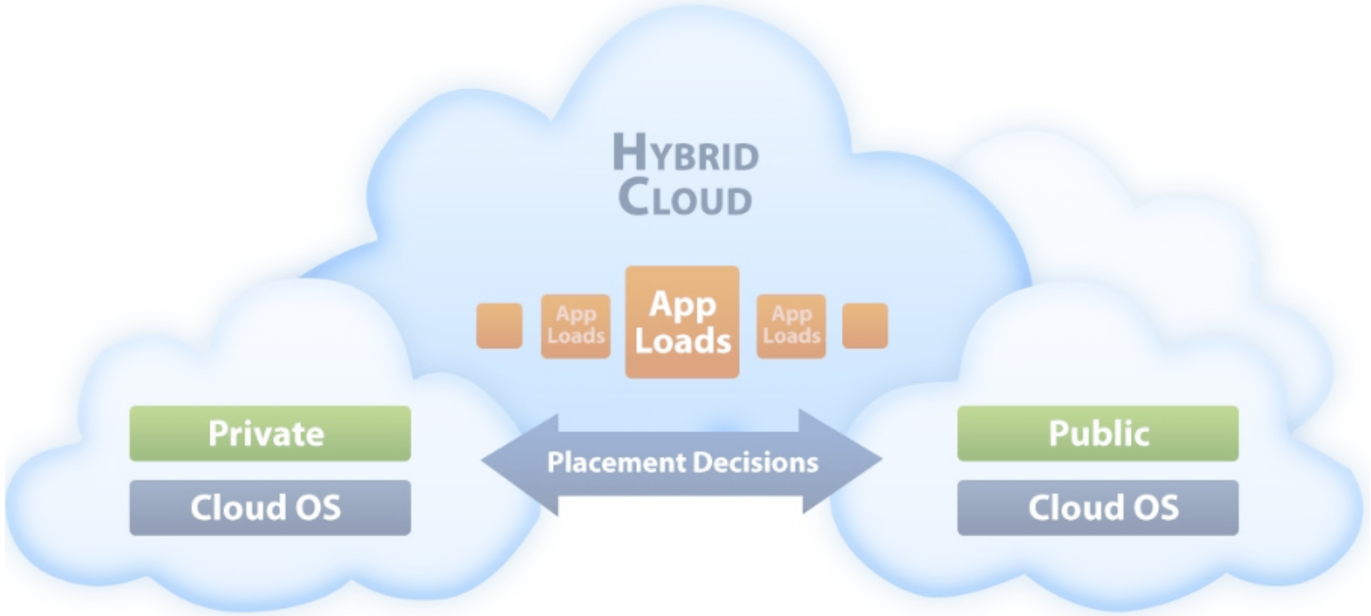
- 不同格式，缺少版本控制仓库
- 不同安装、部署方式
- 很难快速切换版本
- 维护性差

Agenda

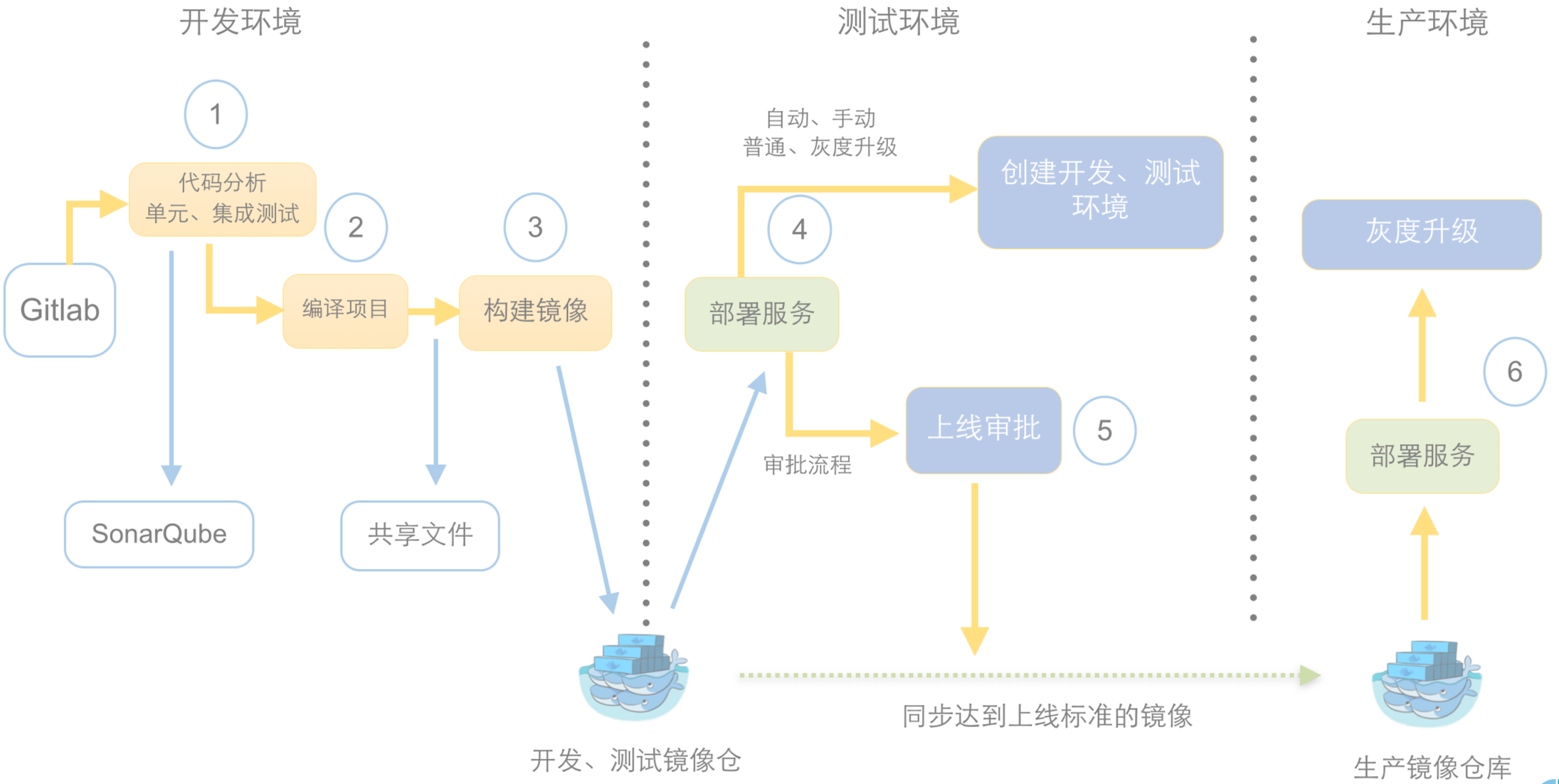
- 初识容器技术
- 容器技术的优势
- **在企业中的应用场景**
- 相关技术生态
- 未来的发展趋势

混合云方案

- 使用容器、镜像技术的应用可以在任何基础设施之上进行迁移
- 避免被基础设施、云厂商绑架
- 在适合的时间、场景使用适合的资源

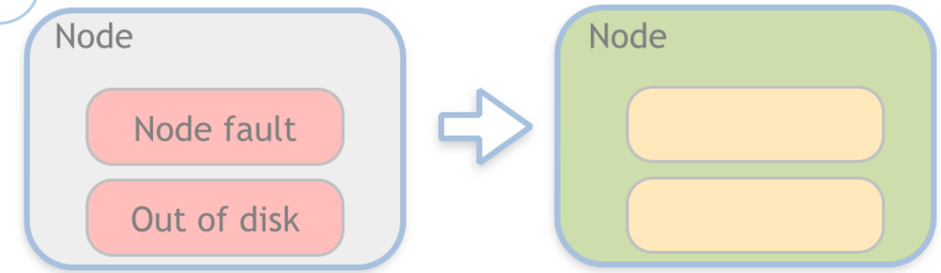


开发测试自动化、标准化



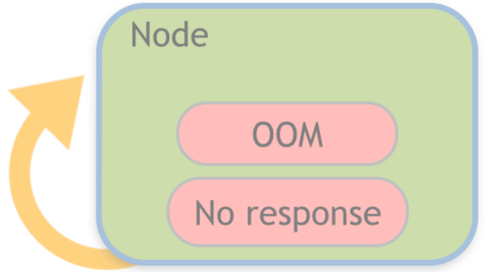
应用高可用

1



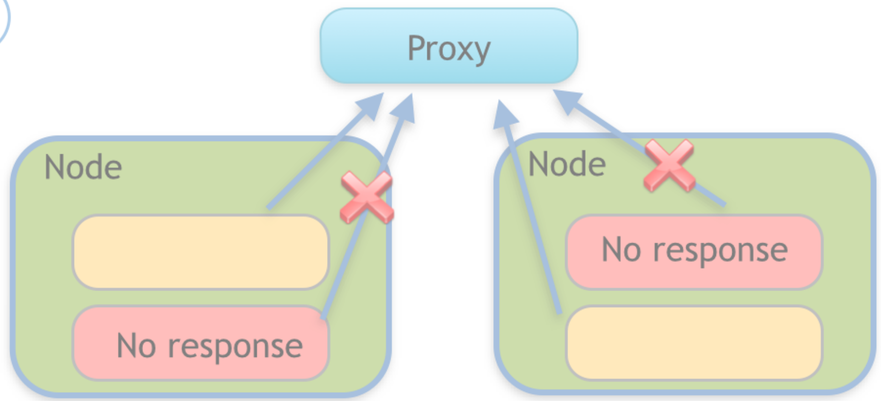
应用将迁移到其他可用节点

2



应用优先在当前节点重启

3



应用节点将被从路由记录中删除

4

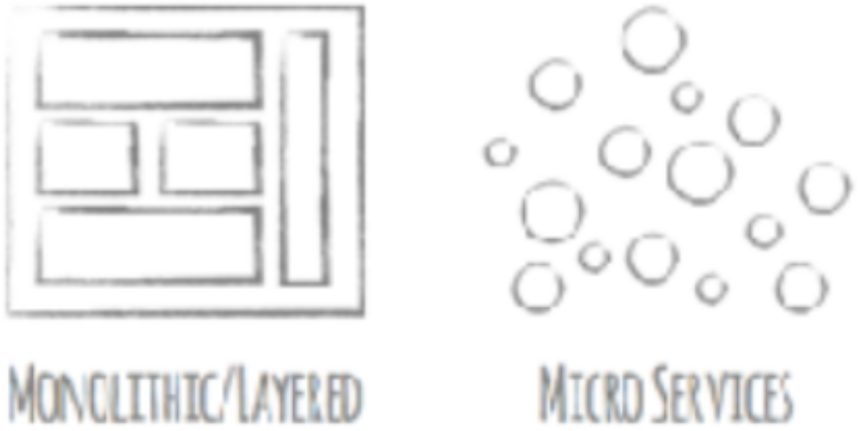
Master HA

5 Multiple zones - 跨区高可用

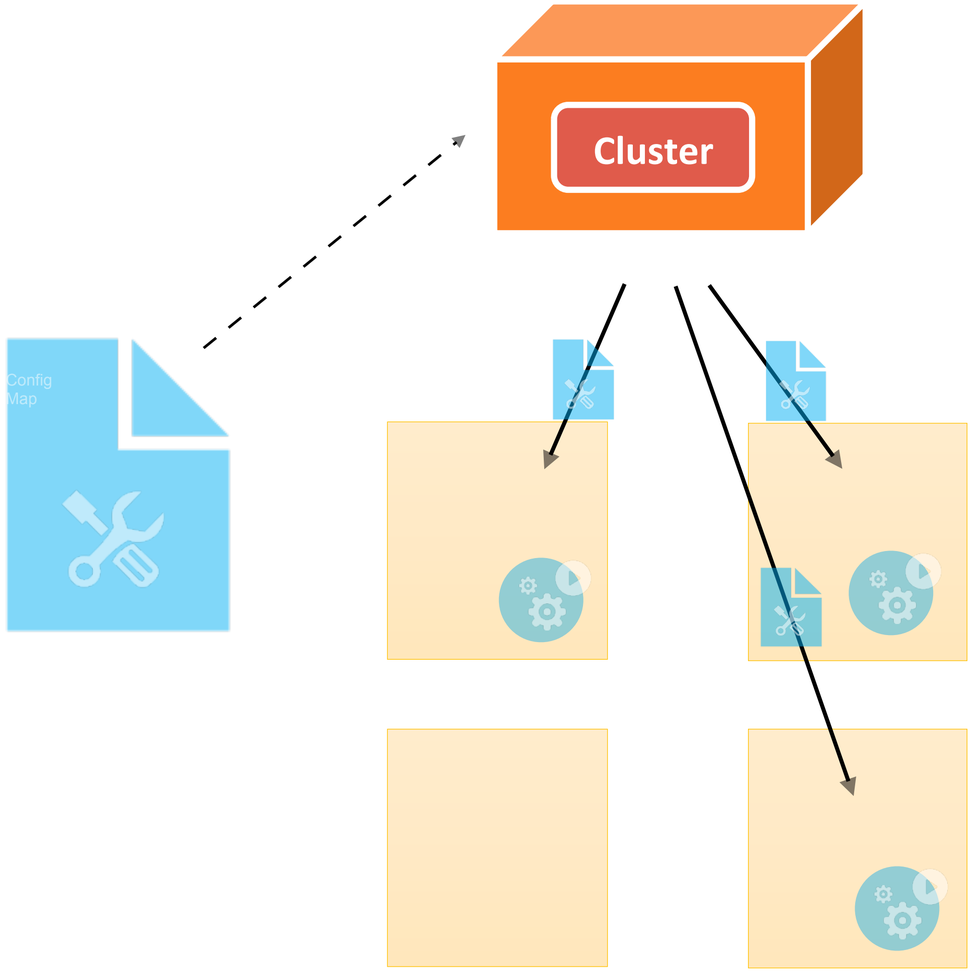
6 Federation clusters - 联邦集群

微服务架构

- 开发测试效率低 – 单独编译、验证
- 不利于多种技术栈的运用 – 可以选择最合适的技术
- 不利于模块化扩展 – 仅扩展有瓶颈的服务
- 容错能力
- 编排式服务开发能力



云平台的配置管理



集群范围内配置的集中化管理

主要特色：

- ✔ 平支级支持应用镜像和配置文件的松耦合
- ✔ 支持通过环境变量、命令行参数、或者数据卷的方式使用配置
- ✔ 配置文件更新，自动同步到引用该配置的所有应用节点

Agenda

- 初识容器技术
- 容器技术的优势
- 在企业中的应用场景
- 相关技术生态
- 未来的发展趋势

容器领域

LXC

CF - Warden

MS Hyper-V

Hyper

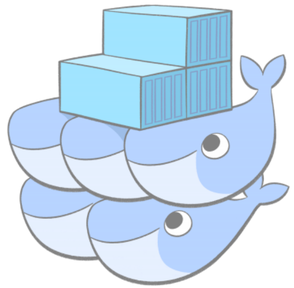
Mesos & Docker Swarm

MESOS

A distributed systems kernel

hadoop Spark kafka

- 2011 年之前就已经开始了
- 有自己的容器运行时
- Marathon、Chronos



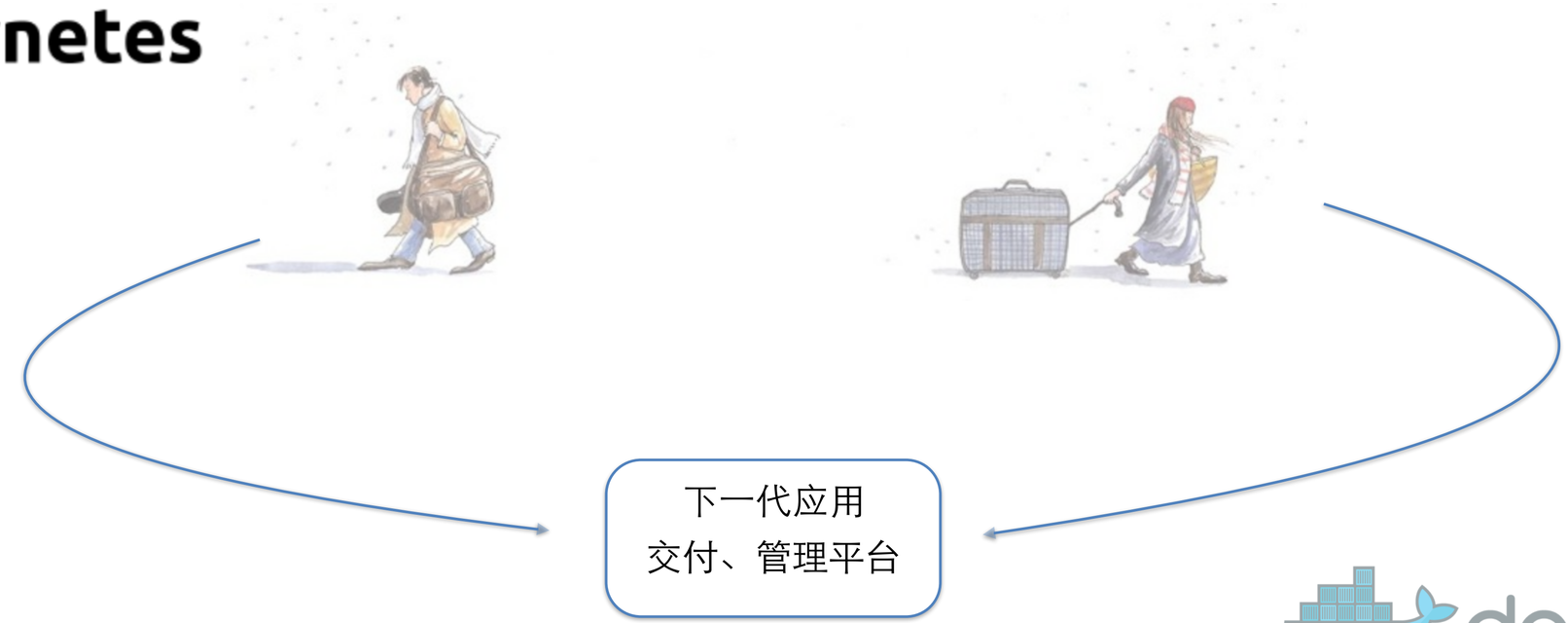
Docker Swarm

- Swarm
- Swarm mode in 1.12
- Swarmkit - swarmctl

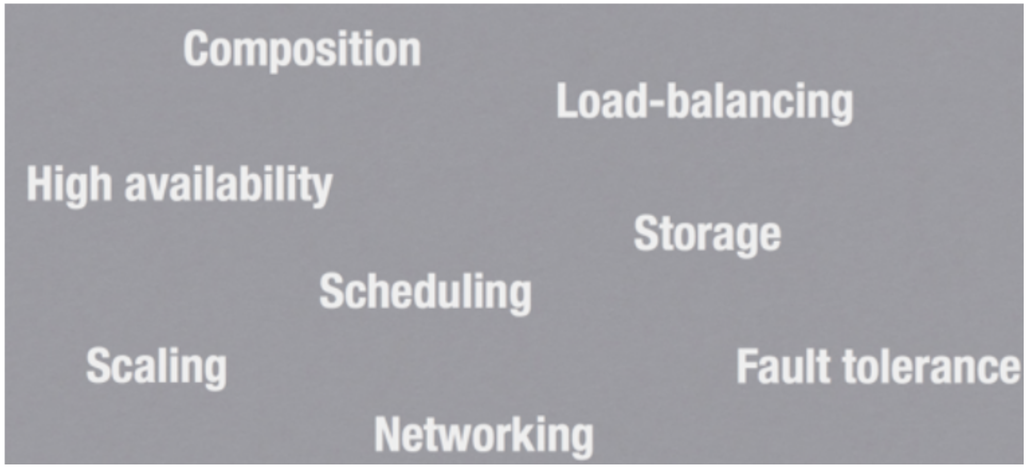
- 把 Swarm 引入Docker engine
- 引入Service对象，扩展更多能力
- 简化安装、部署

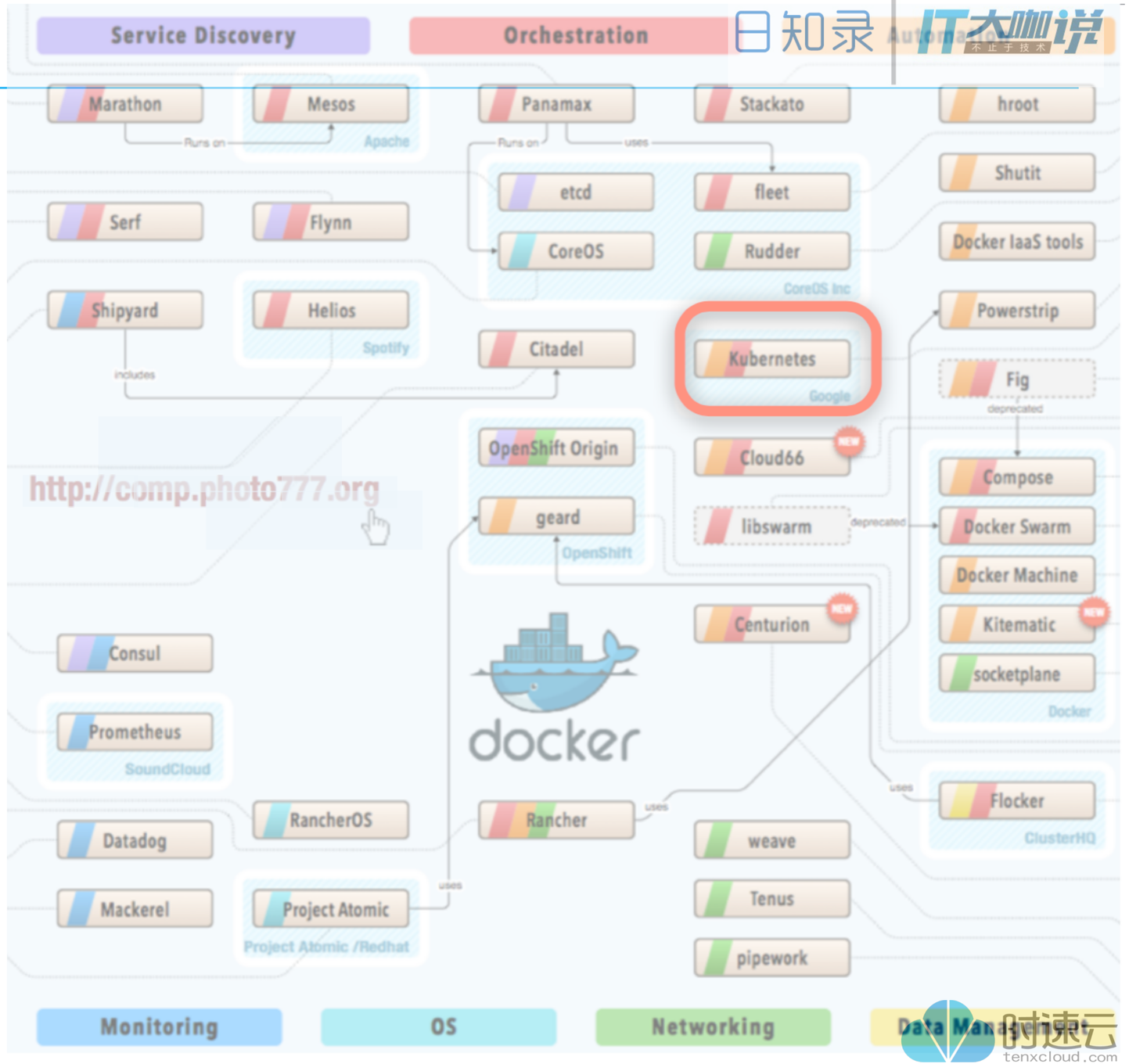
Kubernetes

- 借鉴 Google Borg 的经验、教训
- 可以在物理机、虚拟机，更好适配云端环境
- 支持多种容器运行时
- 更完善的应用生命周期管理理念



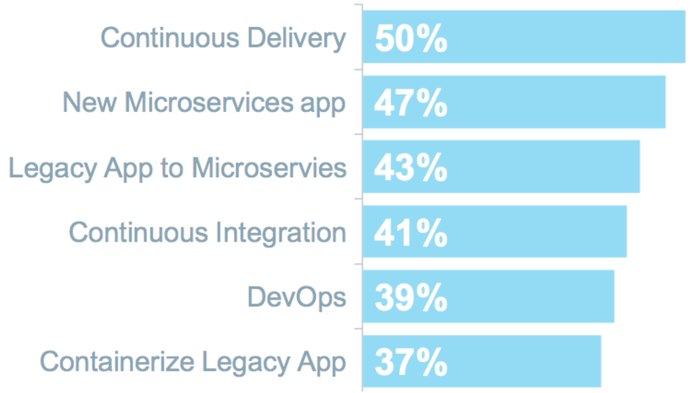
容器生态圈


Composition
Load-balancing
High availability
Storage
Scheduling
Scaling
Fault tolerance
Networking

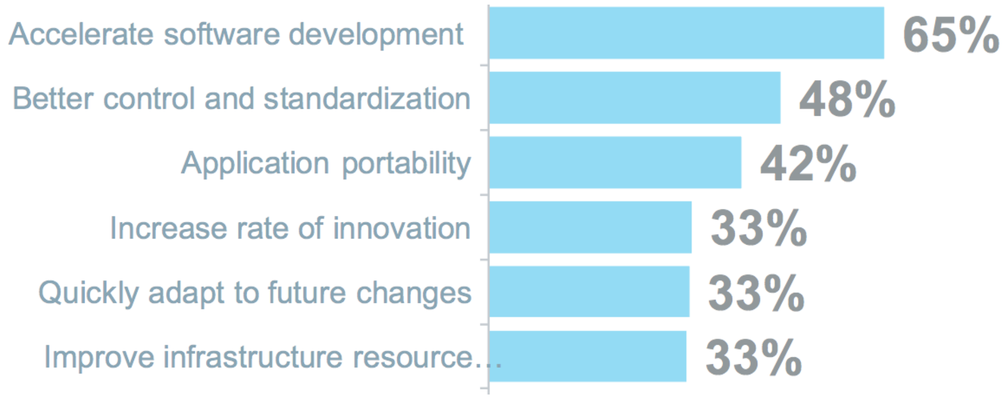

Service Discovery
Orchestration
Marathon
Mesos
Apache
Runs on
Panamax
Stackato
Runs on
uses
hroot
Shutit
etcd
fleet
Serf
Flynn
CoreOS
Rudder
CoreOS Inc
Docker IaaS tools
Shipyard
Helios
Spotify
Citadel
Kubernetes
Google
Powerstrip
includes
Fig
deprecated
OpenShift Origin
Cloud66
NEW
Compose
http://comp.photo777.org
geard
OpenShift
libswarm
deprecated
Docker Swarm
Docker Machine
Kitematic
NEW
Centurion
NEW
Consul
socketplane
Docker
Prometheus
SoundCloud
docker
uses
Flocker
ClusterHQ
RancherOS
Rancher
uses
Datadog
weave
Mackerel
Project Atomic
uses
Tenus
Project Atomic /Redhat
pipework
Monitoring
OS
Networking
Data Management

技术角度看未来趋势

使用场景

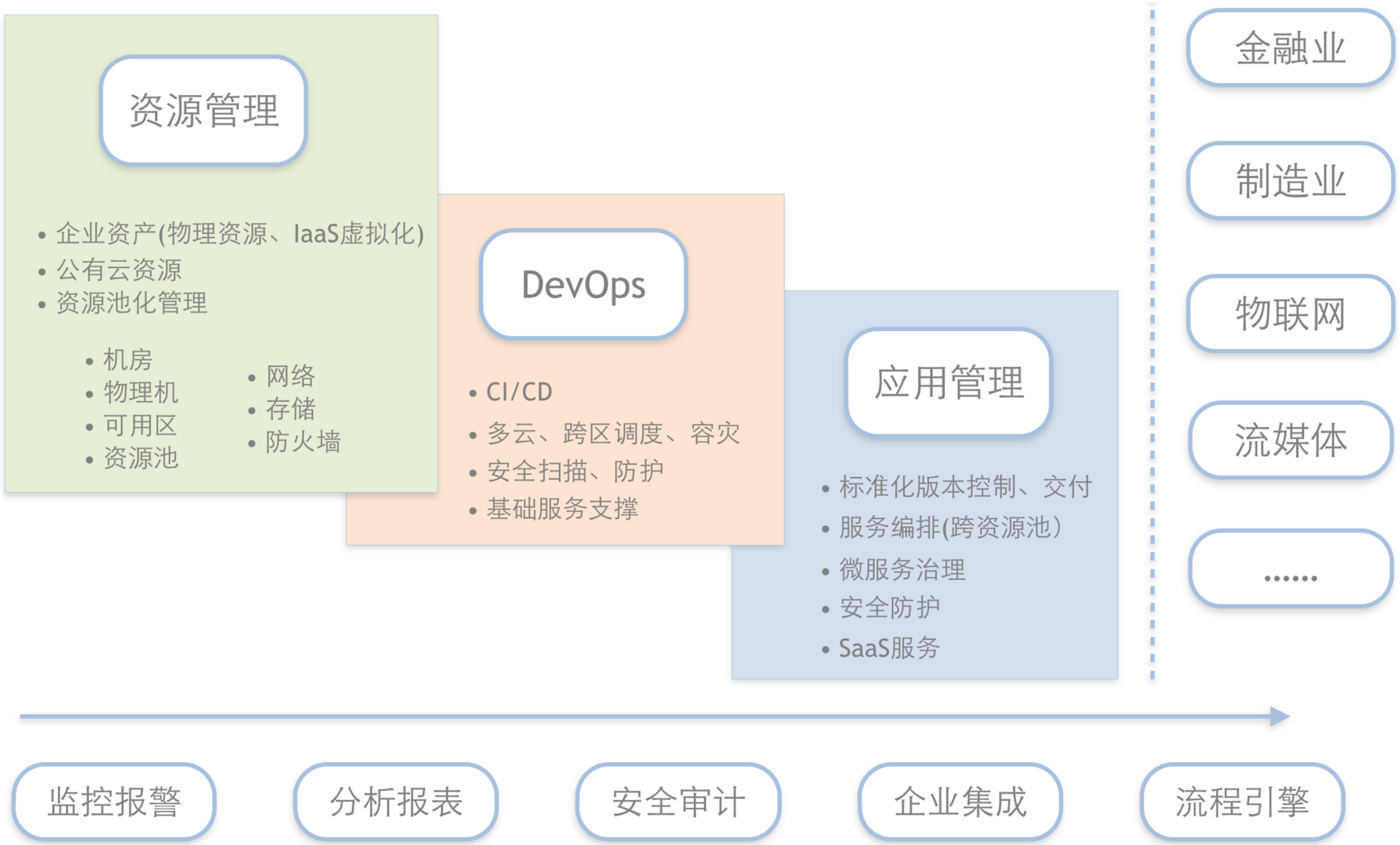


期望产出



- 应用的持续迭代、发布能力 - CI/CD/DevOps
- 微服务架构 - 服务治理、API 网关、事件驱动
- 遗留系统应用容器化
- 同IaaS的融入、互补

云平台诉求



谢谢！

https://www.tenxcloud.com/