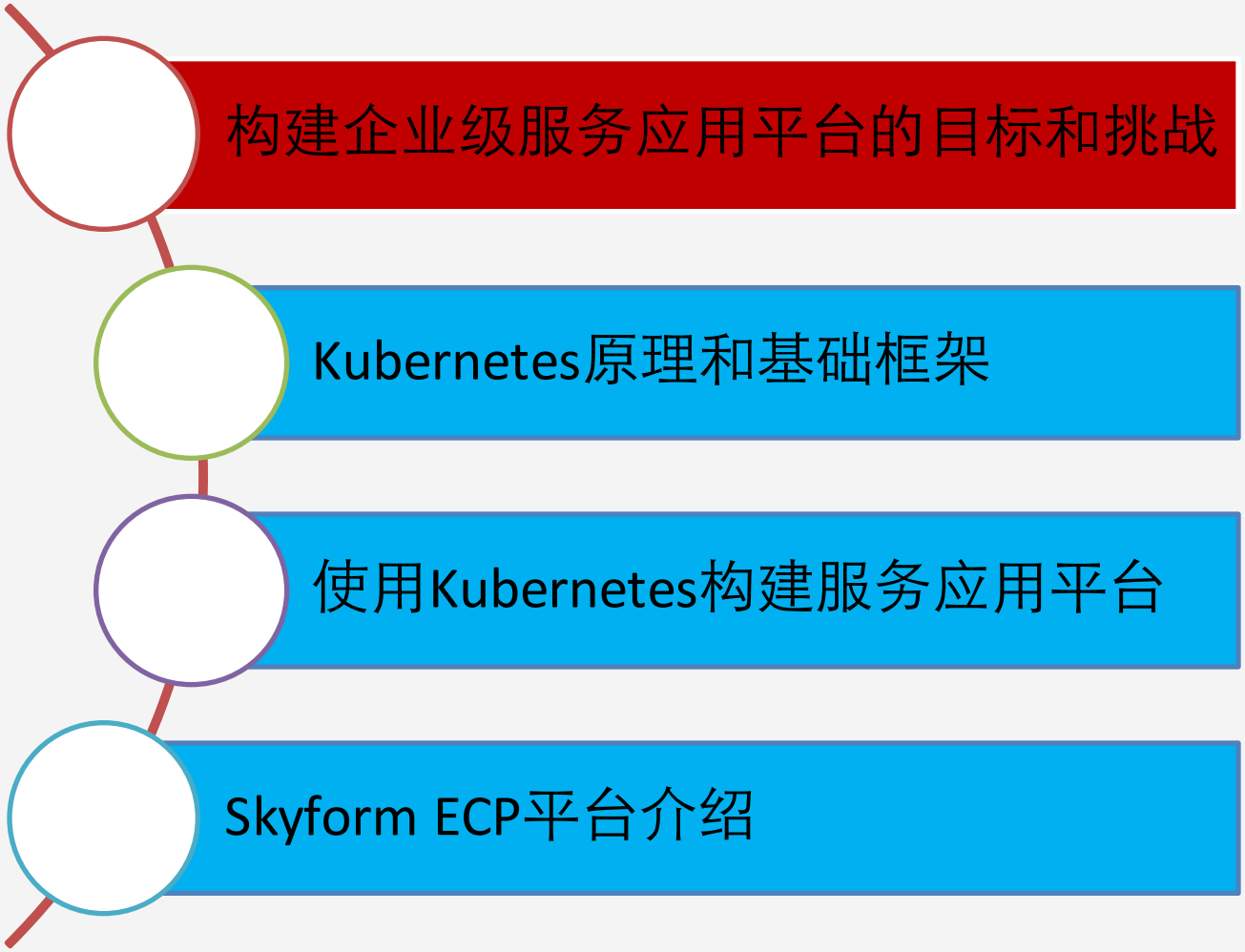
使用kubernetes构建企业级服务应用平台

张伟

Skycloud天云软件
Software

概要


构建企业级服务应用平台的目标和挑战
Kubernetes原理和基础框架
使用Kubernetes构建服务应用平台
Skyform ECP平台介绍

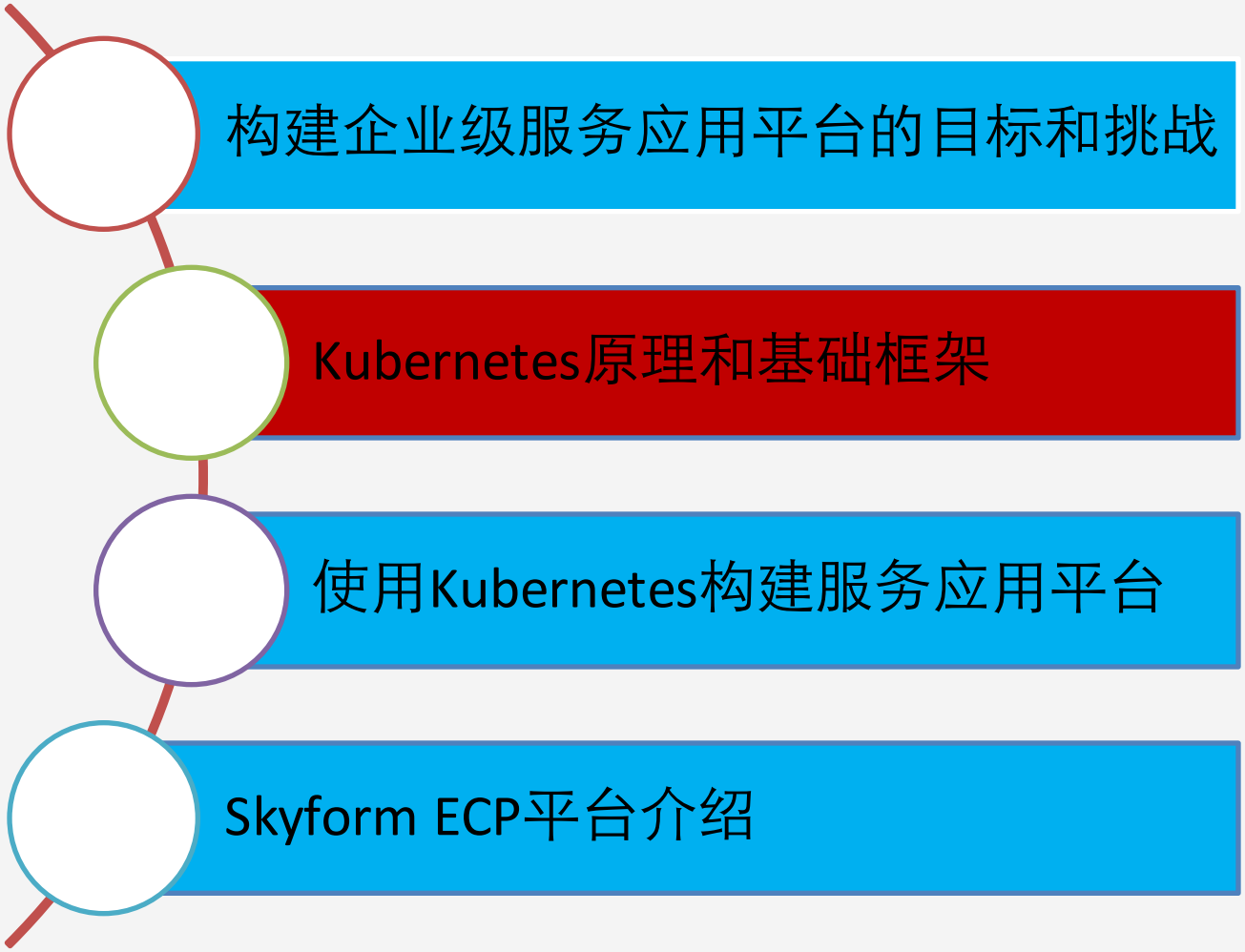
目标和挑战

目标：实现应用的生命周期管理，包括应用的开发、镜像制作、发布、运行、动态调整、业务数据管理、监控、日志、应用拓扑等等。

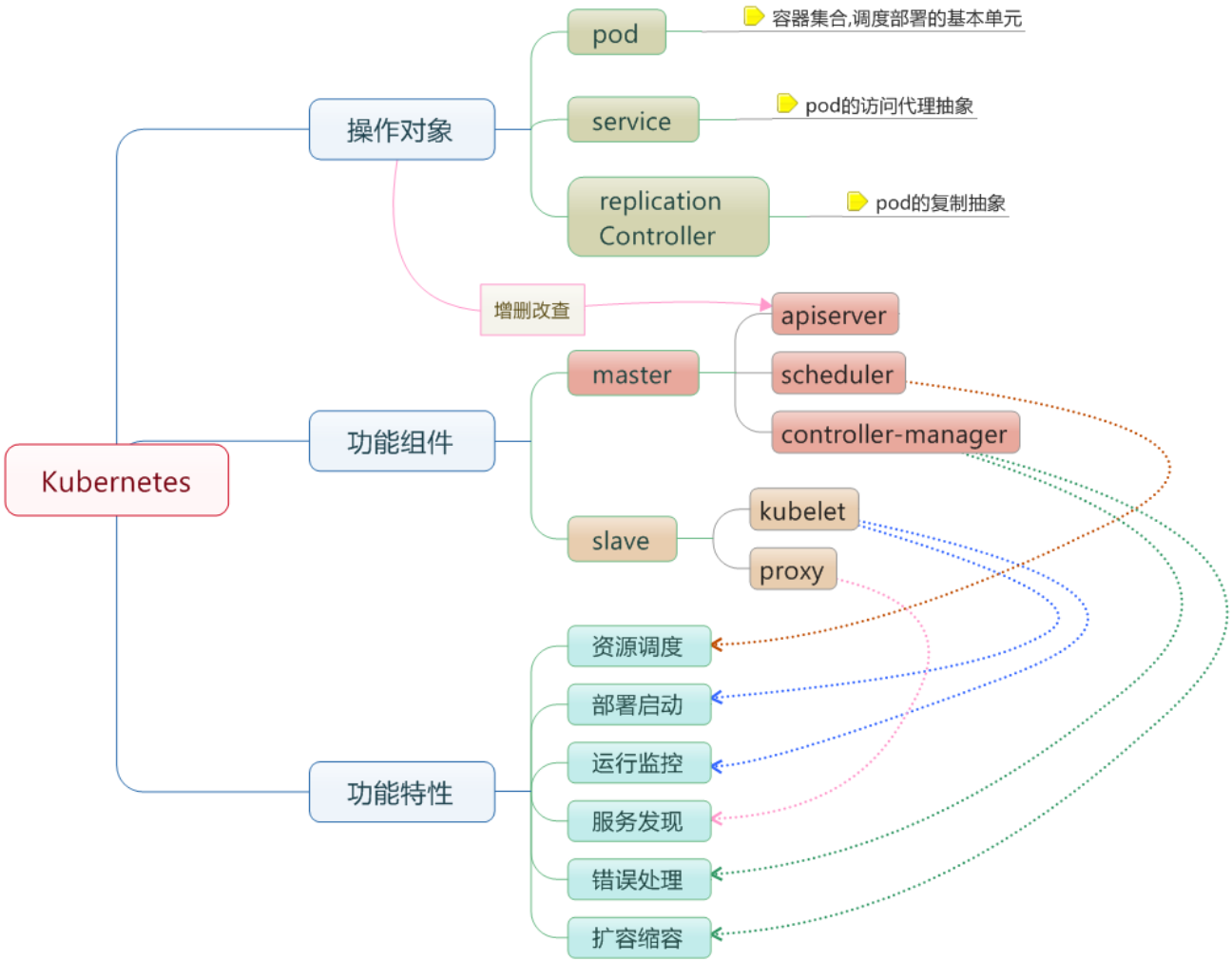
需要解决的问题：

1. 资源管理：包括集群、主机、网络、存储等等。
2. 应用管理：包括应用的发布、运行、拓扑、服务发现等等。
3. 运维管理：包括监控、日志等。
4. 服务质量保证：包括调度、自动伸缩等。

Kubernetes原理和基础框架



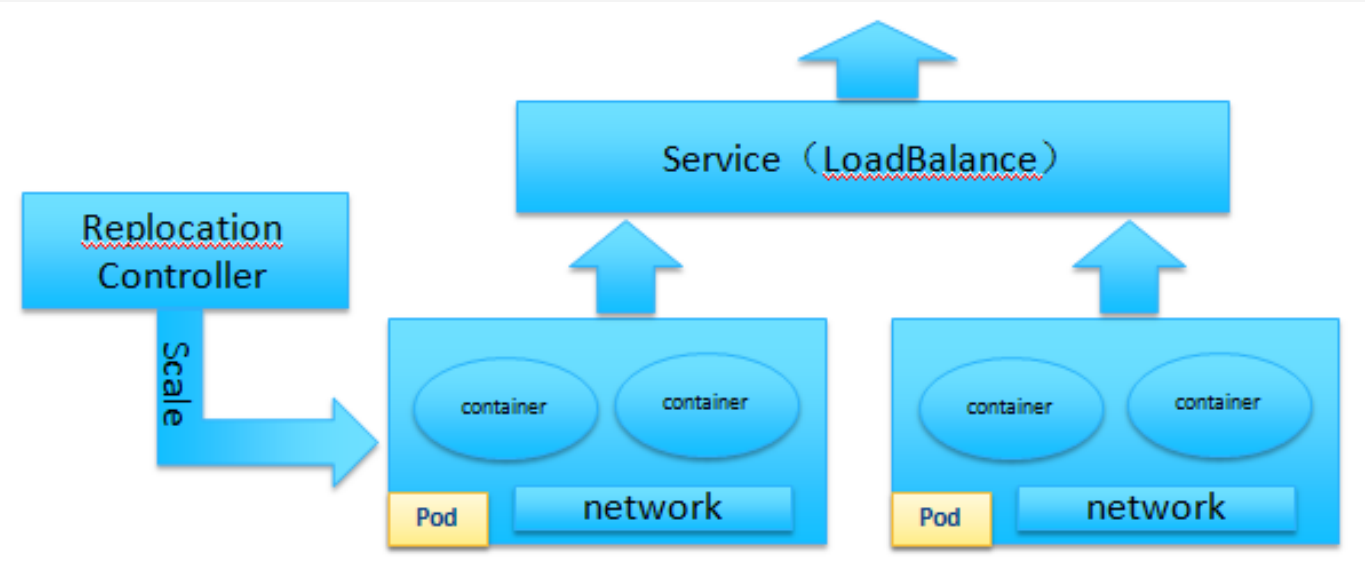
Kubernetes总体概览



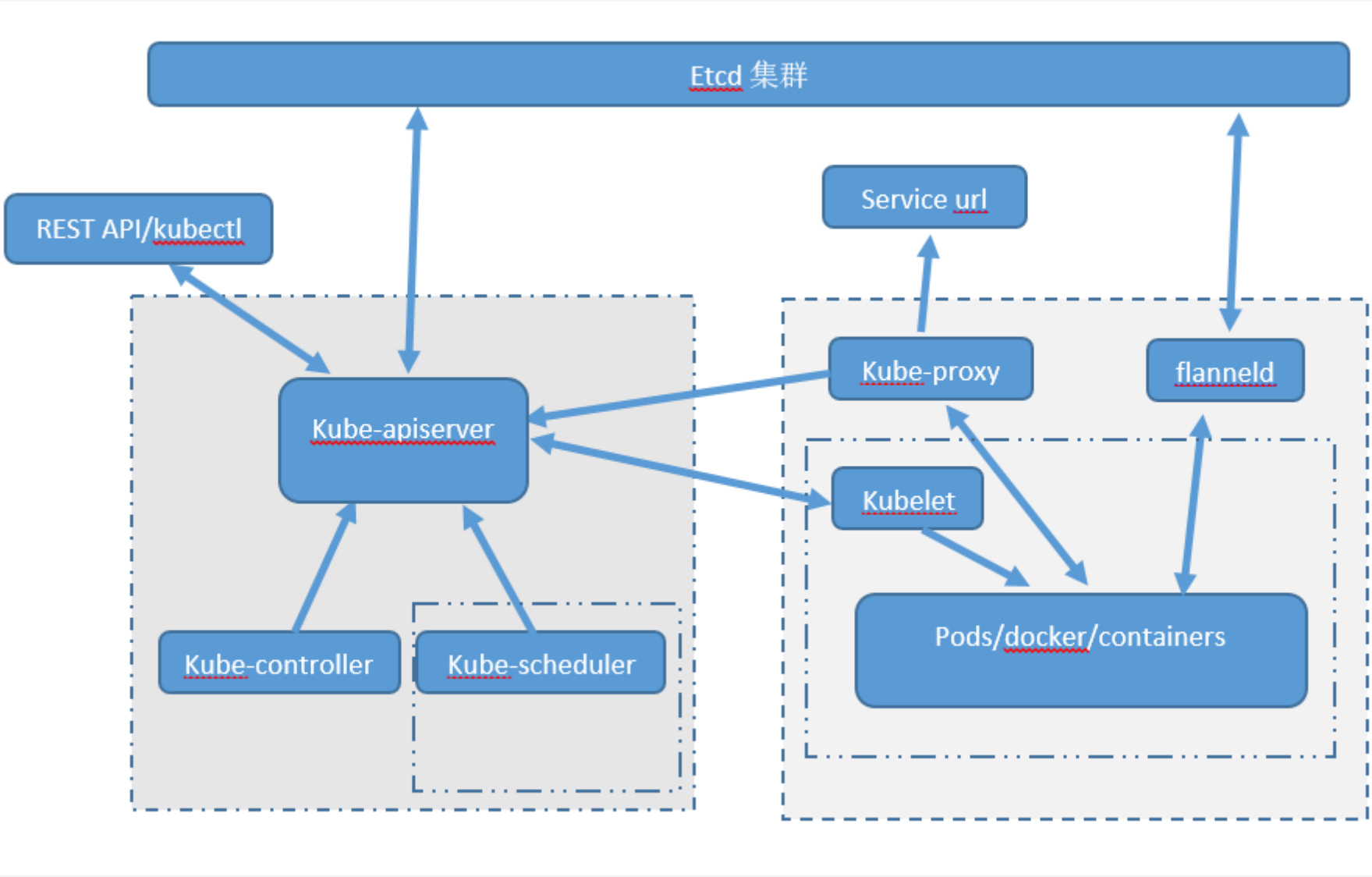
基本元素

基本元素：

- Pod（最小部署单元，可包含多个容器，共享网络）
- ReplicationController（Pod生命周期控制器，scale资源伸缩）
- Service（抽象服务出口，通过proxy对多个Pod负载均衡）
- Labels（标签，用户分类，查询筛选）



组件介绍


Etcd 集群
REST API/kubectl
Service url
Kube-apiserver
Kube-proxy
flanneld
Kubelet
Kube-controller
Kube-scheduler
Pods/docker/containers

组件介绍

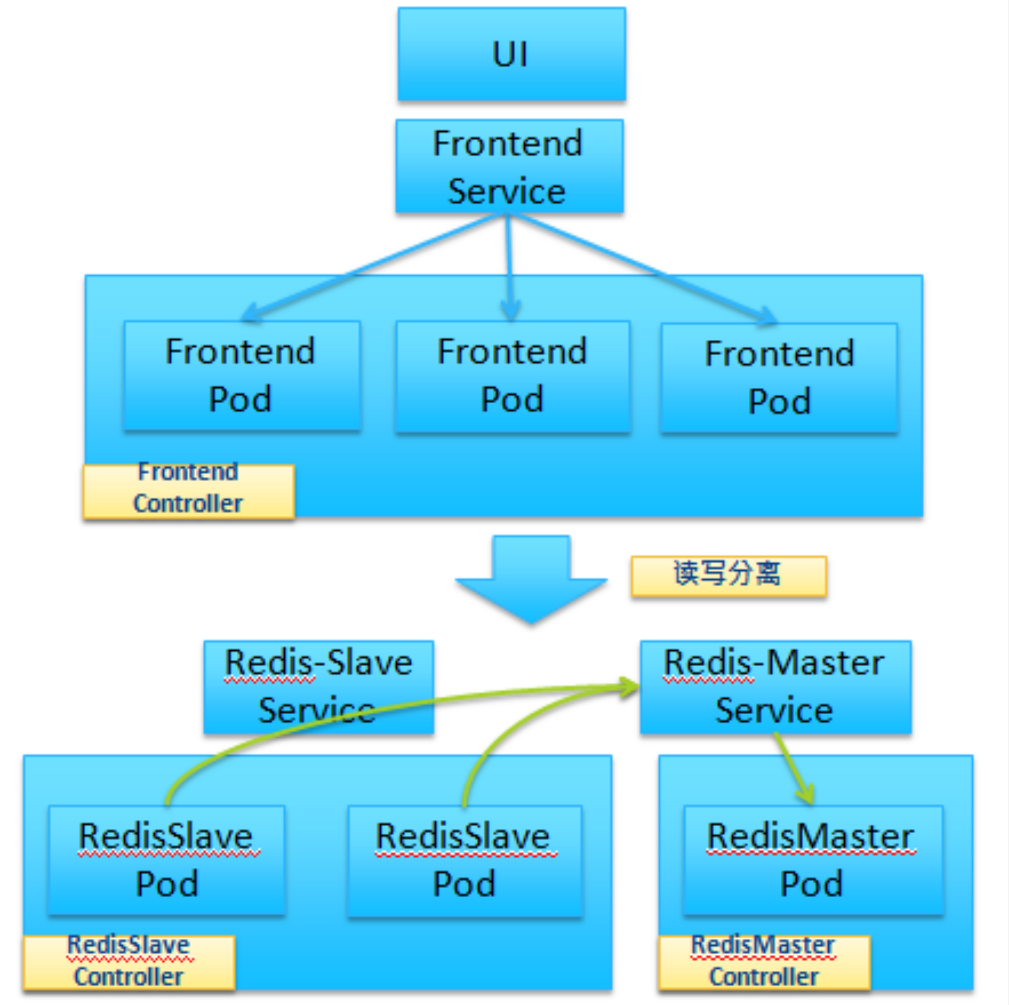
Master组件：

- **ApiServer**：作为kubernetes系统的入口，封装了核心对象的增删改查操作。
- **Scheduler**：插件式的调度器，负责集群的资源调度，为新建的pod分配机器。
- **Controller**：负责管理各种控制器。如ReplicationController，EndPointController等。

Slave组件：

- **kubelet**：负责管控docker容器，如启动/停止、监控运行状态等。
- **proxy**：负责为pod提供代理。它会定期从etcd获取所有的service，并根据service信息创建代理。

GuestBook示例



支持的服务类型:

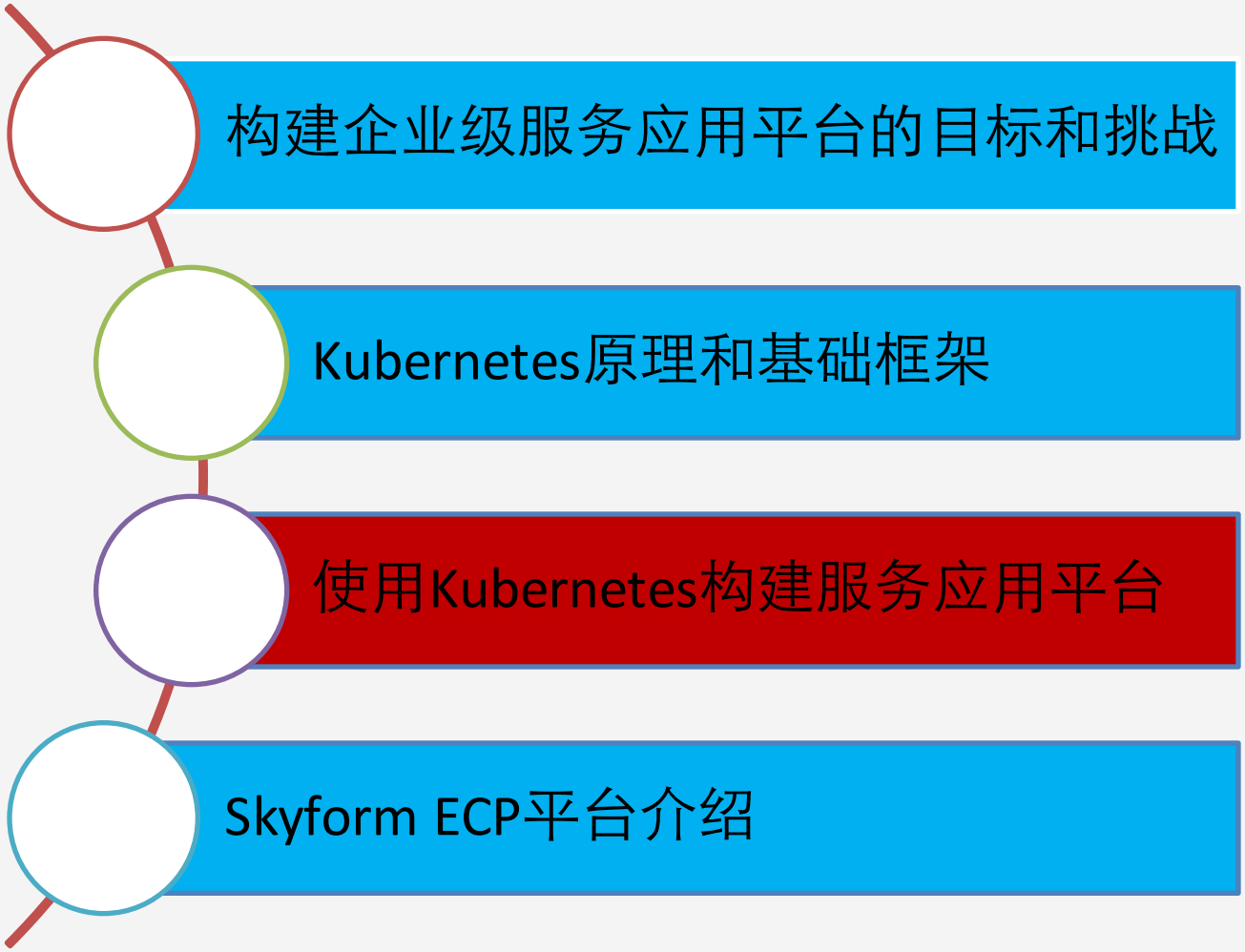
- StateFul 有状态。
- StateLess 无状态。

负载均衡:

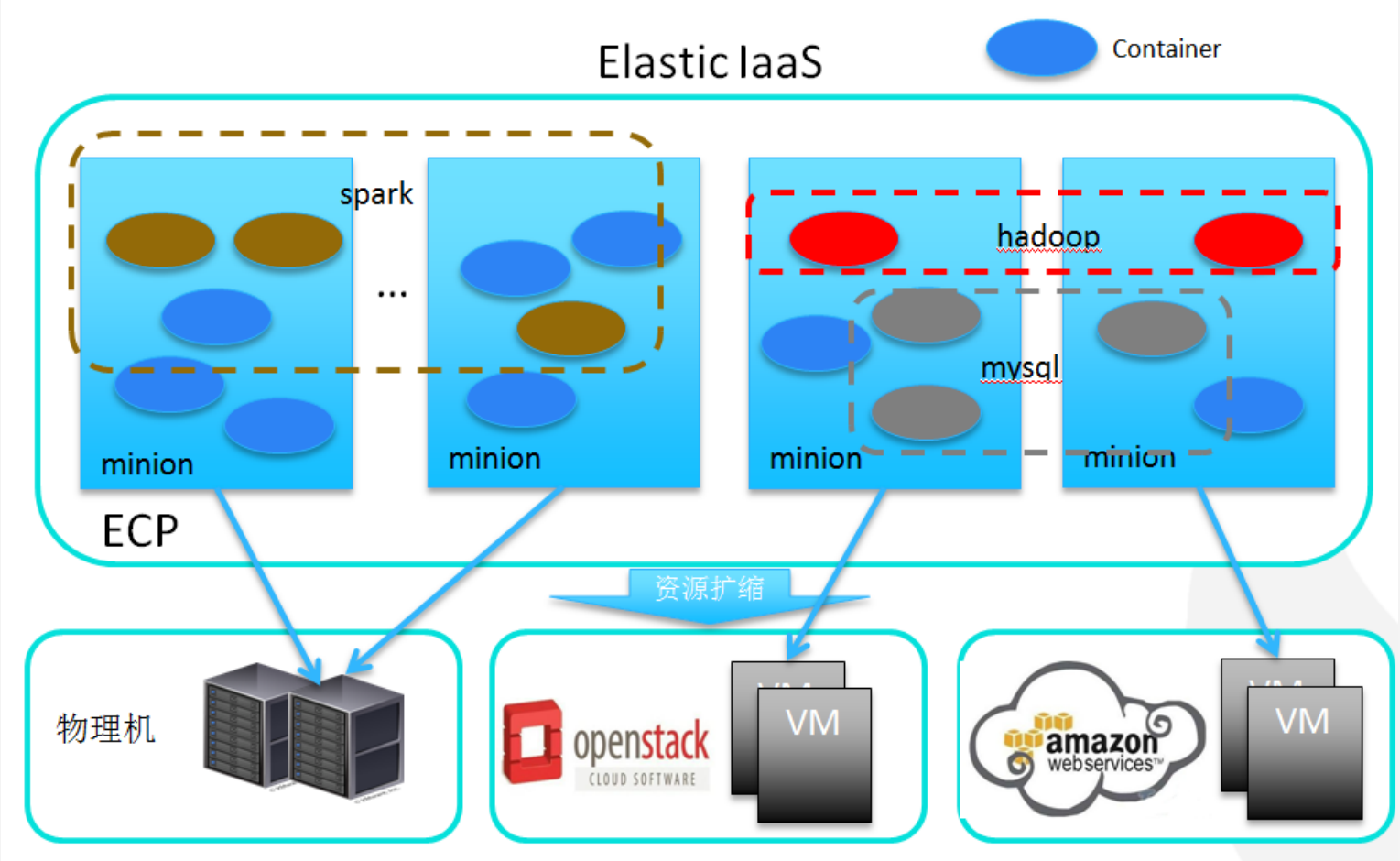
- Service 提供loadbalance能力，把流量按照策略分配给不同的后端。

SkyForm

使用Kubernetes构建服务应用平台



主机管理


Elastic IaaS
Container
spark
...
hadoop
mysql
minion
minion
minion
minion
ECP
资源扩缩
物理机
openstack
CLOUD SOFTWARE
VM
amazon
webservices
VM

网络管理

Kubernetes网络解决的问题：

Pod内容器通信、Pod和Pod通信、Pod和Service通信、外部和Service通信。

Kubernetes网络模型：

- 集群中所有container可以直接和其他任意container通信，不通过NAT。
- 任意node主机可以和任意container通信，不通过NAT。
- 任意container自己本身拥有的ip，和外界container看到它的ip一致。

实现方式：隧道、路由，Flannel、Open vSwitch、Weave等。

存储管理

Kubernetes存储解决的问题：

主要解决Pod重启数据丢失和同一Pod中的容器共享存储的问题。并且支持多种的存储类型。

支持的存储类型：

- emptyDir：随Pod删除，场景：临时存储、灾难恢复、共享运行时数据。
- hostPath：类似于Docker的本地Volume，用于访问一些本地资源。
- gcePersistentDisk： GCE disk，只有在 Google Cloud Engine 平台上可用。
- awsElasticBlockStore： 类似于GCE disk，节点必须是 AWS EC2的实例。
- nfs ： 支持网络文件系统。
- rbd： Rados Block Device – Ceph。
- persistentVolumeClaim：从抽象的PV中申请资源，而无需关心存储的提供方。

资源限制

- 多层次资源限制

Container Level:
Resource request limits

```
apiVersion: v1
kind: Pod
metadata:
  name: frontend
spec:
  containers:
  - name: db
    image: mysql
    resources:
      requests:
        memory: "64Mi"
        cpu: "250m"
      limits:
        memory: "128Mi"
        cpu: "500m"
  - name: wp
    image: wordpress
    resources:
      requests:
        memory: "64Mi"
        cpu: "250m"
      limits:
        memory: "128Mi"
        cpu: "500m"
```

Pod Level:
Limit range

```
apiVersion: v1
kind: LimitRange
metadata:
  name: default-limits
  namespace: default
spec:
  limits:
    type: Pod
  - default:
      cpu: 100m
      memory: 512Mi
    max:
      cpu: 1024m
      memory: 10240Mi
    min:
      cpu: 100m
      memory: 256Mi
```

Namespace Level:
Resource Quota

```
apiVersion: v1
kind: ResourceQuota
metadata:
  name: my-quota
  namespace: default
spec:
  hard:
    cpu: 4
    memory: 32G
    pods: 30
    services: 100
    replicationcontrollers: 80
    resourcequotas: 10
```

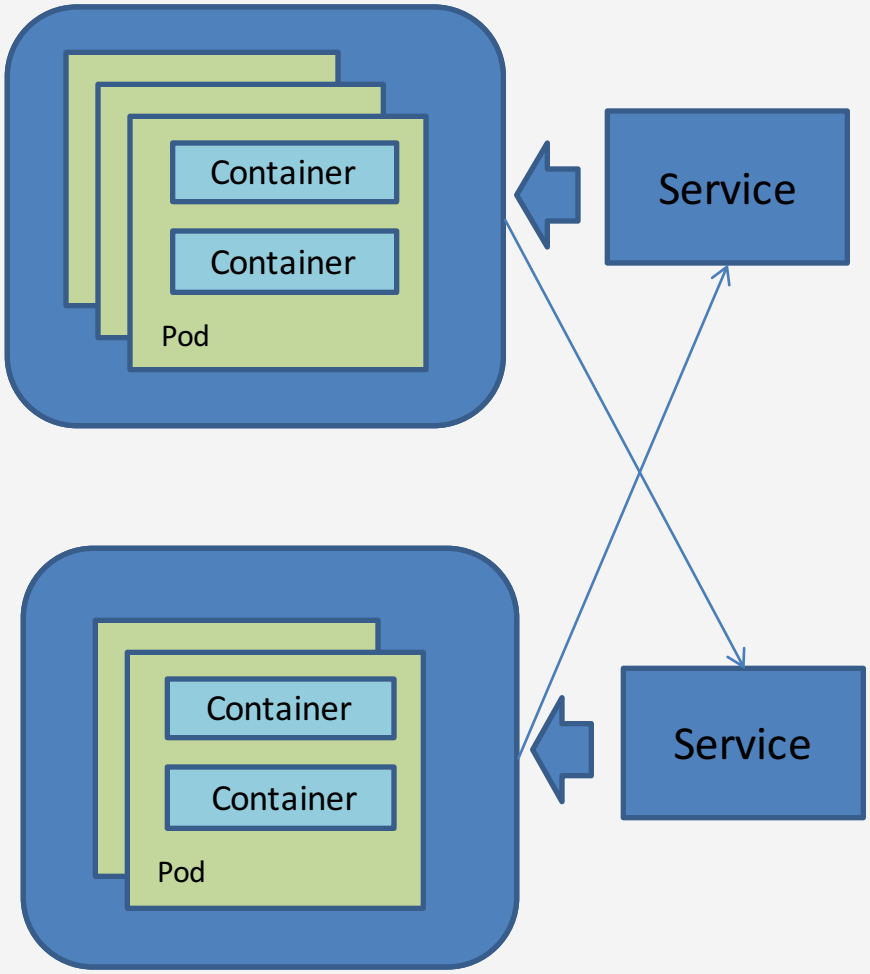
应用管理

服务编排

- 以服务构建应用
- 各服务独立部署,独立扩展
- 服务多语言开发
- 服务间轻量级交互
- 松耦合

Pod编排

- 相关的一组容器放入同一Pod
- 各容器间可以使用localhost通信
- 共享网络和存储
- 高内聚



服务发现

- 环境变量方式

  Pod启动时，kubelet把所有正在运行的服务的信息，通过环境变量的方式添加到容器上。

```
apiVersion: v1
kind: Service
metadata:
  name: elasticsearch-logging
  namespace: kube-system
  labels:
    k8s-app: elasticsearch-logging
    kubernetes.io/cluster-service: "true"
    kubernetes.io/name: "Elasticsearch"
spec:
  ports:
  - port: 9200
    protocol: TCP
    targetPort: db
  selector:
    k8s-app: elasticsearch-logging
  type: NodePort
```

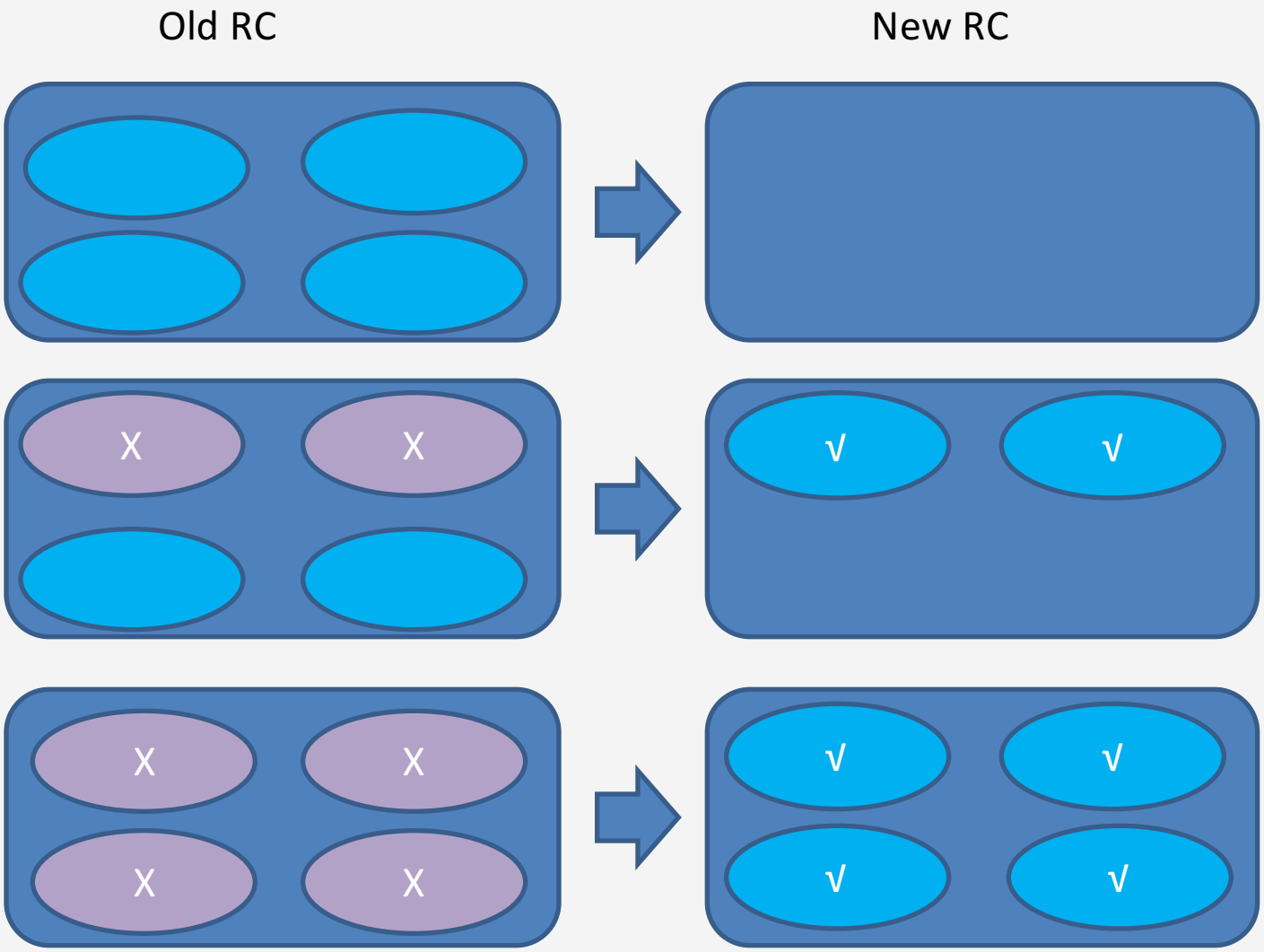
```
"Env": [
    "MONITORING_GRAFANA_PORT_80_TCP=tcp://192.168.3.33:80",
    "MONITORING_GRAFANA_PORT_80_TCP_ADDR=192.168.3.33",
    "KIBANA_LOGGING_PORT_5601_TCP=tcp://192.168.3.140:5601",
    "MONITORING_INFLUXDB_SERVICE_PORT=8083",
    "MONITORING_INFLUXDB_PORT=tcp://192.168.3.139:8083",
    "ELASTICSEARCH_LOGGING_SERVICE_PORT=9200",
    "ELASTICSEARCH_LOGGING_PORT=tcp://192.168.3.135:9200"
],
```

- DNS方式

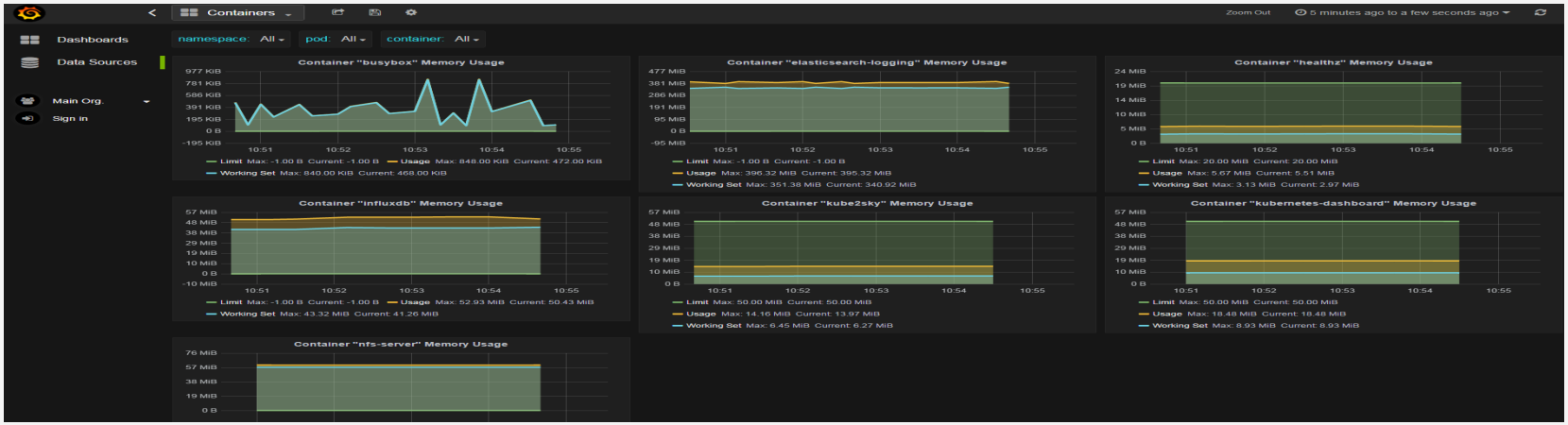
  skydns 服务为每一个service自动生成一个域名，可以使用域名访问service。

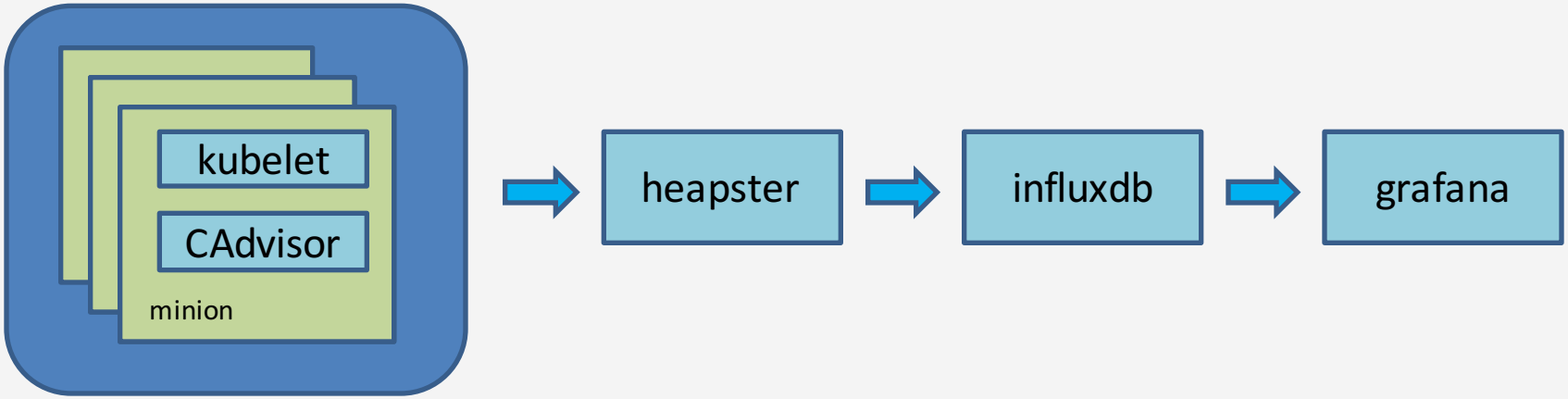
  域名命名格式：<service-name>.<namespace-name>.svc.cluster.local

灰度升级

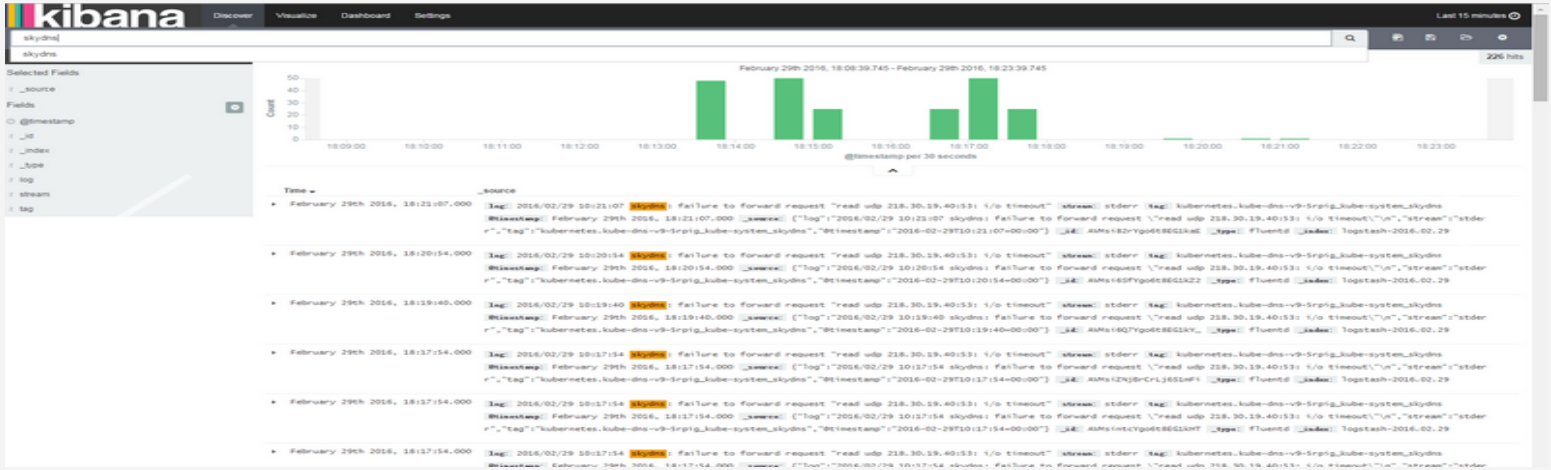


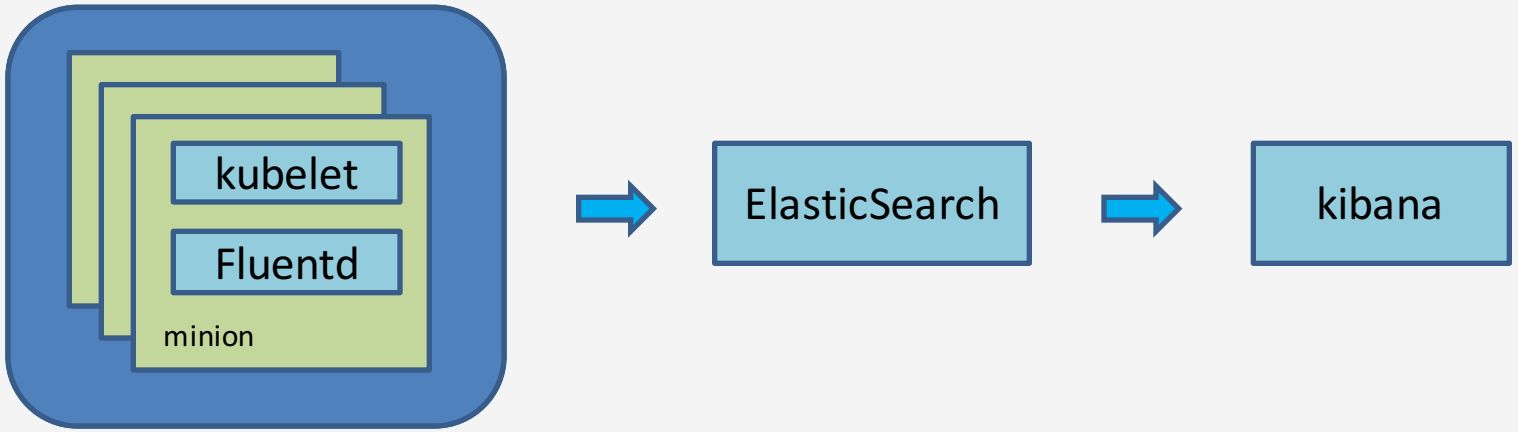
运维-监控


Containers
Zoom Out
5 minutes ago to a few seconds ago
Dashboards
Data Sources
Main Org.
Sign in
namespace: All
pod: All
container: All
Container "busybox" Memory Usage
Container "elasticsearch-logging" Memory Usage
Container "healthz" Memory Usage
Container "influxdb" Memory Usage
Container "kube2sky" Memory Usage
Container "kubernetes-dashboard" Memory Usage
Container "nfs-server" Memory Usage

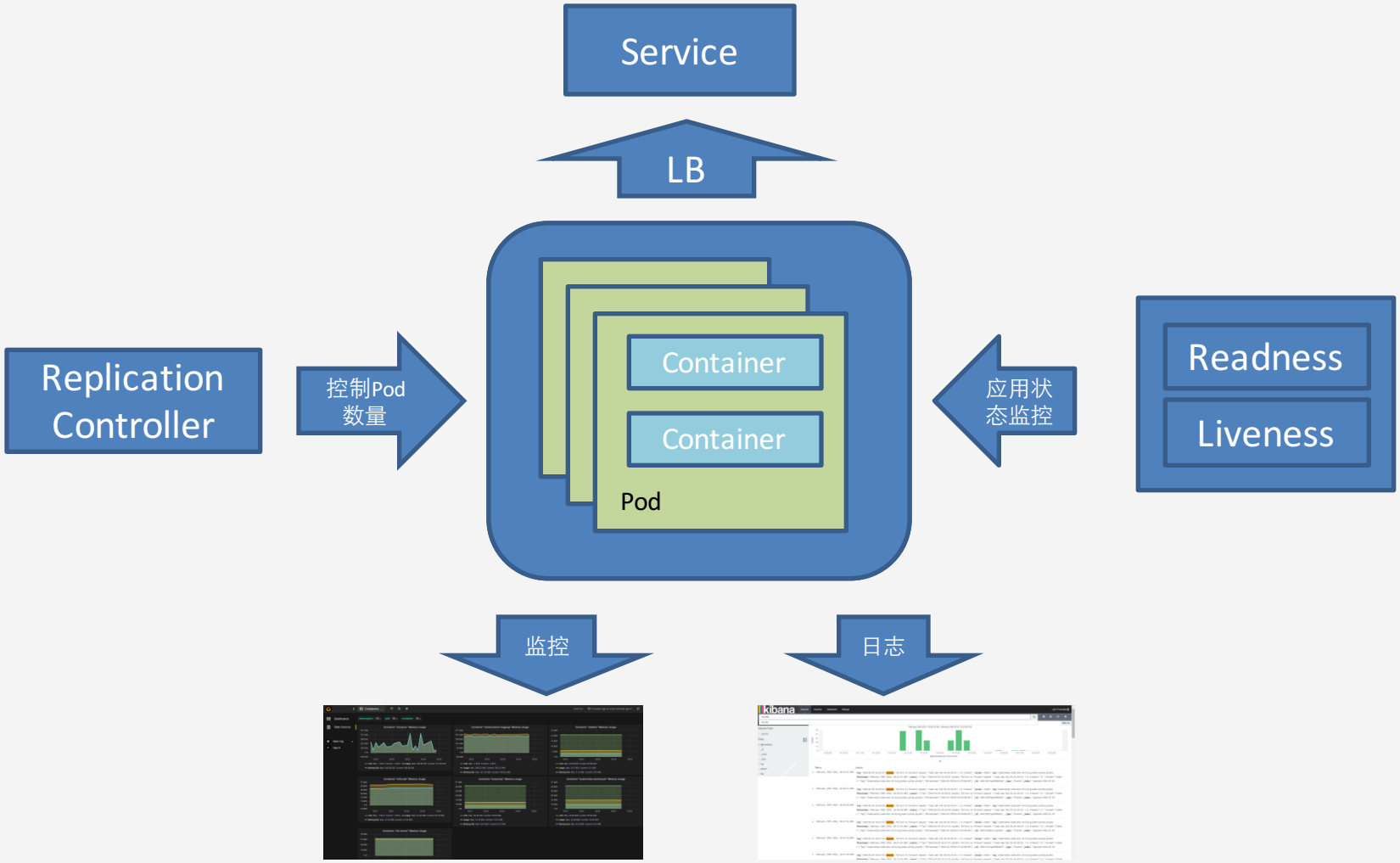

kubelet
CAdvisor
minion
heapster
influxdb
grafana

运维-日志


kibana
Discover
Visualize
Dashboard
Settings

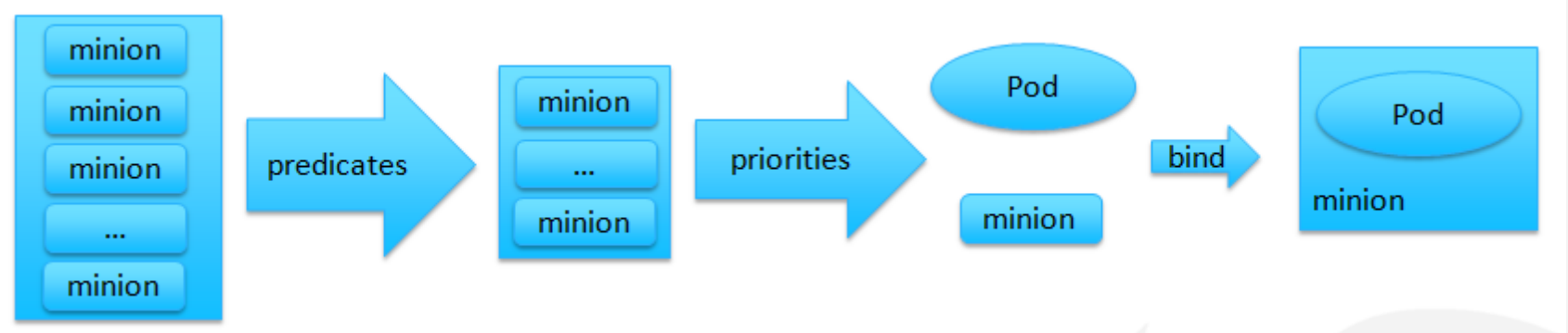

kubelet
Fluentd
minion
ElasticSearch
kibana

应用发布运行

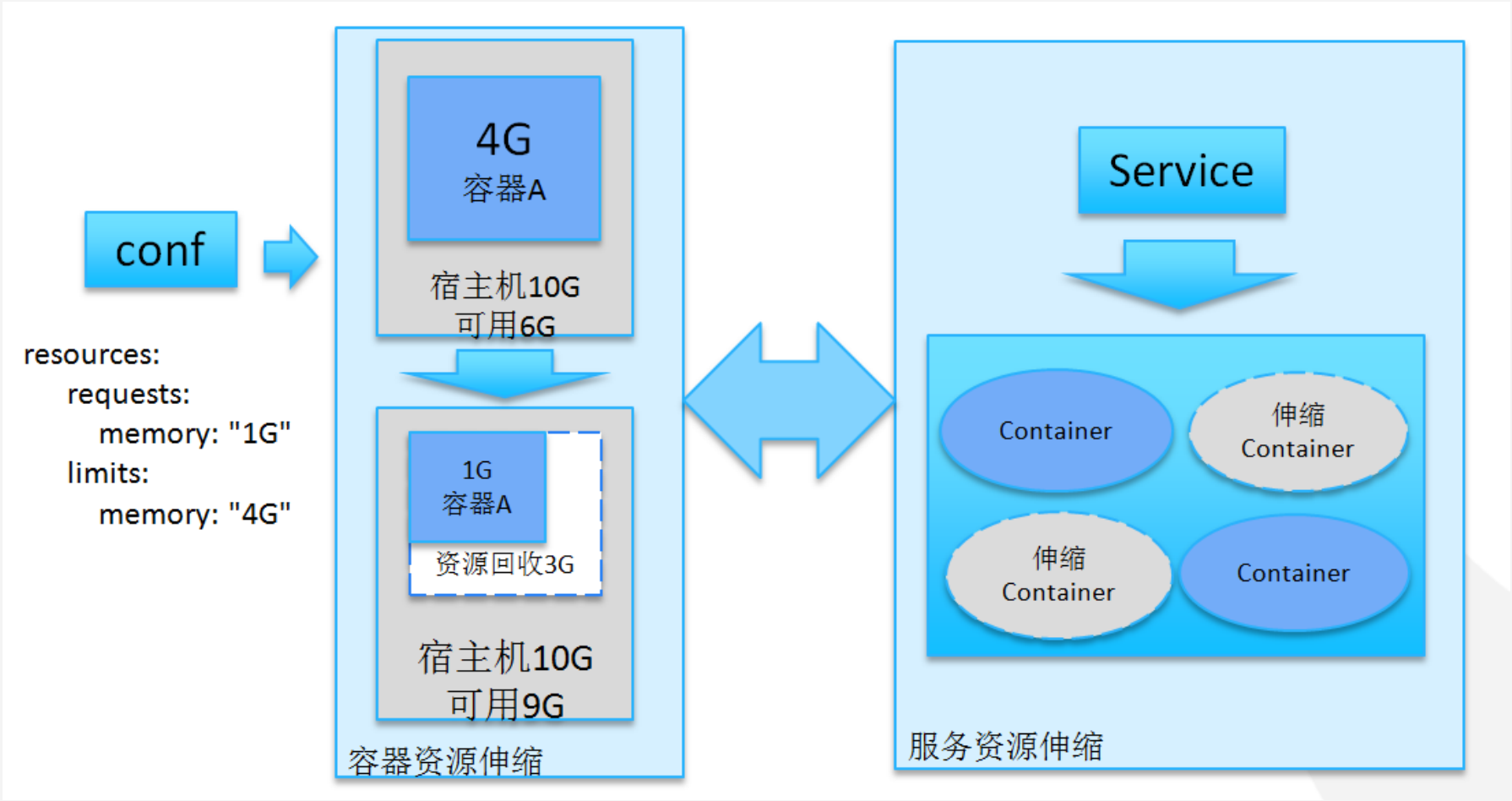

Service
LB
Replication Controller
控制Pod数量
Container
Container
Pod
应用状态监控
Readness
Liveness
监控
日志
kibana

服务质量保证-Scheduler

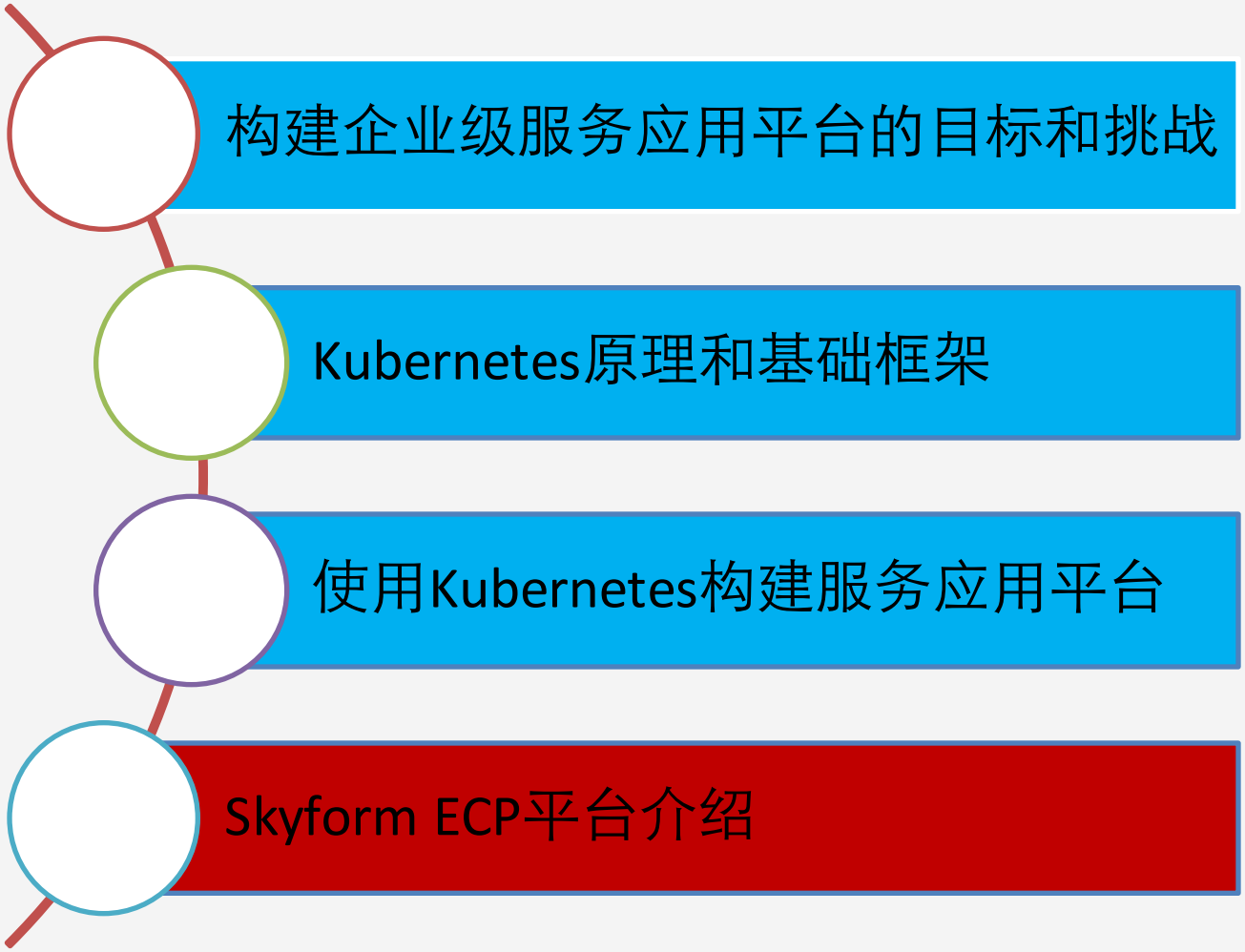
1. 完全插件化
2. Predicates：筛选合格的minion
   - 容器申请的主机端口是否可用
   - 可用CPU和内存是否满足Pod需求
   - Volume目录是否冲突
   - 是否匹配用户指定的Label
   - 是不是指定的HostName
3. Priorities：对筛选合格的minion打分
   - 选择空闲资源更多的机器
   - 不同类型的Pod尽量交错部署。
   - 属于同一个RC的Pod尽量分布在不同的机器上



服务质量保证-动态伸缩

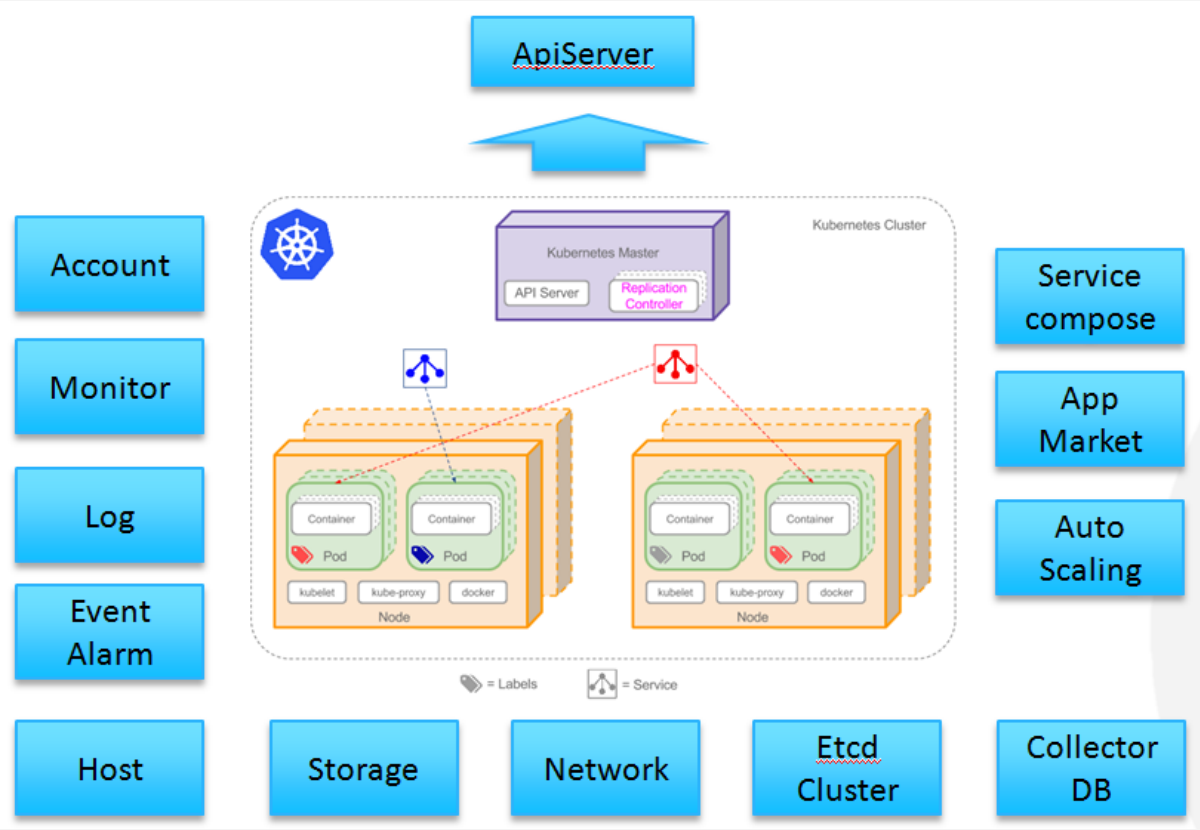

conf
resources:
requests:
memory: "1G"
limits:
memory: "4G"
4G
容器A
宿主机10G
可用6G
1G
容器A
资源回收3G
宿主机10G
可用9G
容器资源伸缩
Service
Container
伸缩
Container
伸缩
Container
Container
服务资源伸缩

Skyform ECP平台介绍

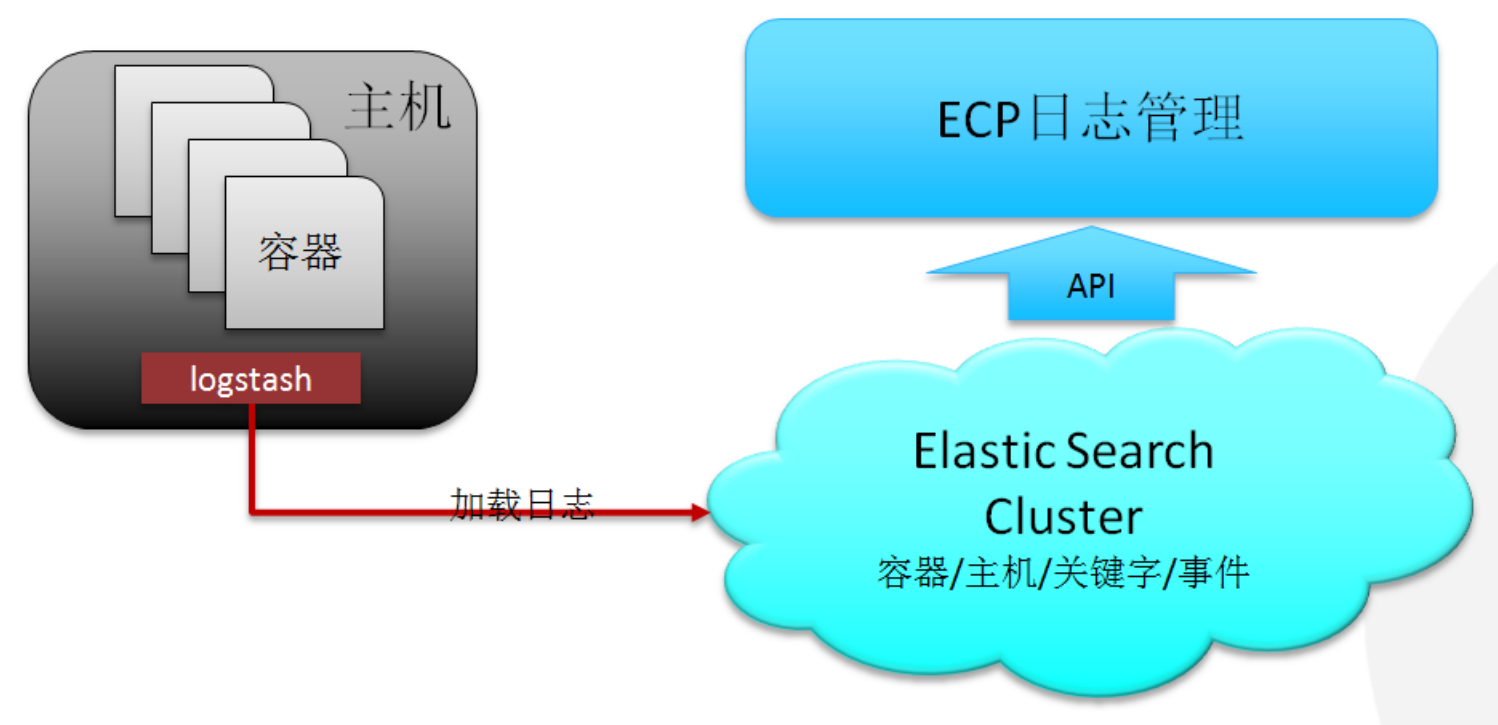


SkyForm ECP平台

SkyForm ECP：是企业级的全方位容器化应用管理平台，提供以容器技术为基础的应用程序全生命周期管理。ECP覆盖应用容器镜像管理、应用编排、应用运行期管理、计算存储网络基础资源管理，以及开箱即用的丰富的基础应用封装。



ECP 日志


主机
容器
logstash
加载日志
ECP日志管理
API
Elastic Search
Cluster
容器/主机/关键字/事件

谢谢

Skycloud天云软件
Software