upyun

又拍云-企业容器私有云架构分享

hongbo.mo@upai.com

杭州 · 北京 · 上海 · 广州 · 深圳

一个小故事跟几点思考

upyun

一个小故事跟几点思考

2015 年又拍的一次线上故障导致了「重试风暴」，后端图片处理请求翻了好几倍，而当时整个图片处理集群的冗余度在 20 % 左右，是根本无法抗这么大的并发的。唯一的办法就是扩容。在整个扩容过程我们发现了几个痛点。

一个小故事跟几点思考

2015 年又拍的一次线上故障导致了「重试风暴」，后端图片处理请求翻了好几倍，而当时整个图片处理集群的冗余度在 20 % 左右，是根本无法抗这么大的并发的。唯一的办法就是扩容。在整个扩容过程我们发现了几个痛点。

痛点一、需要新的机器

一个小故事跟几点思考

2015 年又拍的一次线上故障导致了「重试风暴」，后端图片处理请求翻了好几倍，而当时整个图片处理集群的冗余度在 20 % 左右，是根本无法抗这么大的并发的。唯一的办法就是扩容。在整个扩容过程我们发现了几个痛点。

痛点二、系统环境问题

一个小故事跟几点思考

upyun

- **如何快速扩容应对突发流量?**

一个小故事跟几点思考

upyun

- **如何快速扩容应对突发流量?**
- **如何解决部署环境问题?**

一个小故事跟几点思考

upyun

- 如何快速扩容应对突发流量?
- 如何解决部署环境问题?
- 如何整合离散计算资源?

一个小故事跟几点思考

upyun

- 如何快速扩容应对突发流量?
- 如何解决部署环境问题?
- 如何整合离散计算资源?
- 如何对大规模集群统一管理?

容器很火

理想的私有云平台

upyun

理想的私有云平台

upyun

资源统一整合

部署迁移App灵活

支持资源的增删改

App间环境隔离

服务统一入口

通用监控告警

持续集成交付

通用日志采集

我们的方案

稳定性，结构清晰

我们的方案

upyun

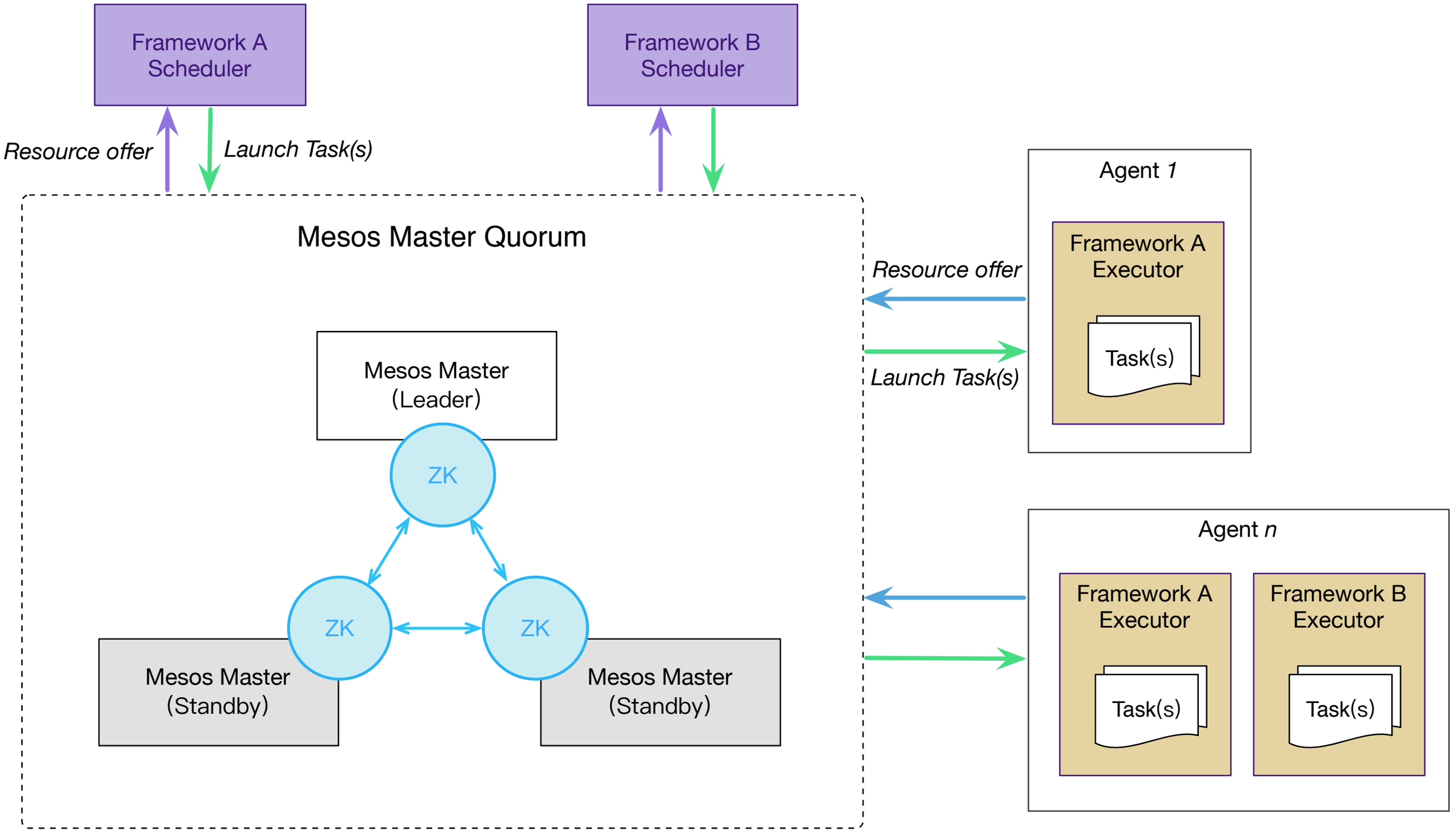
MESOS

docker

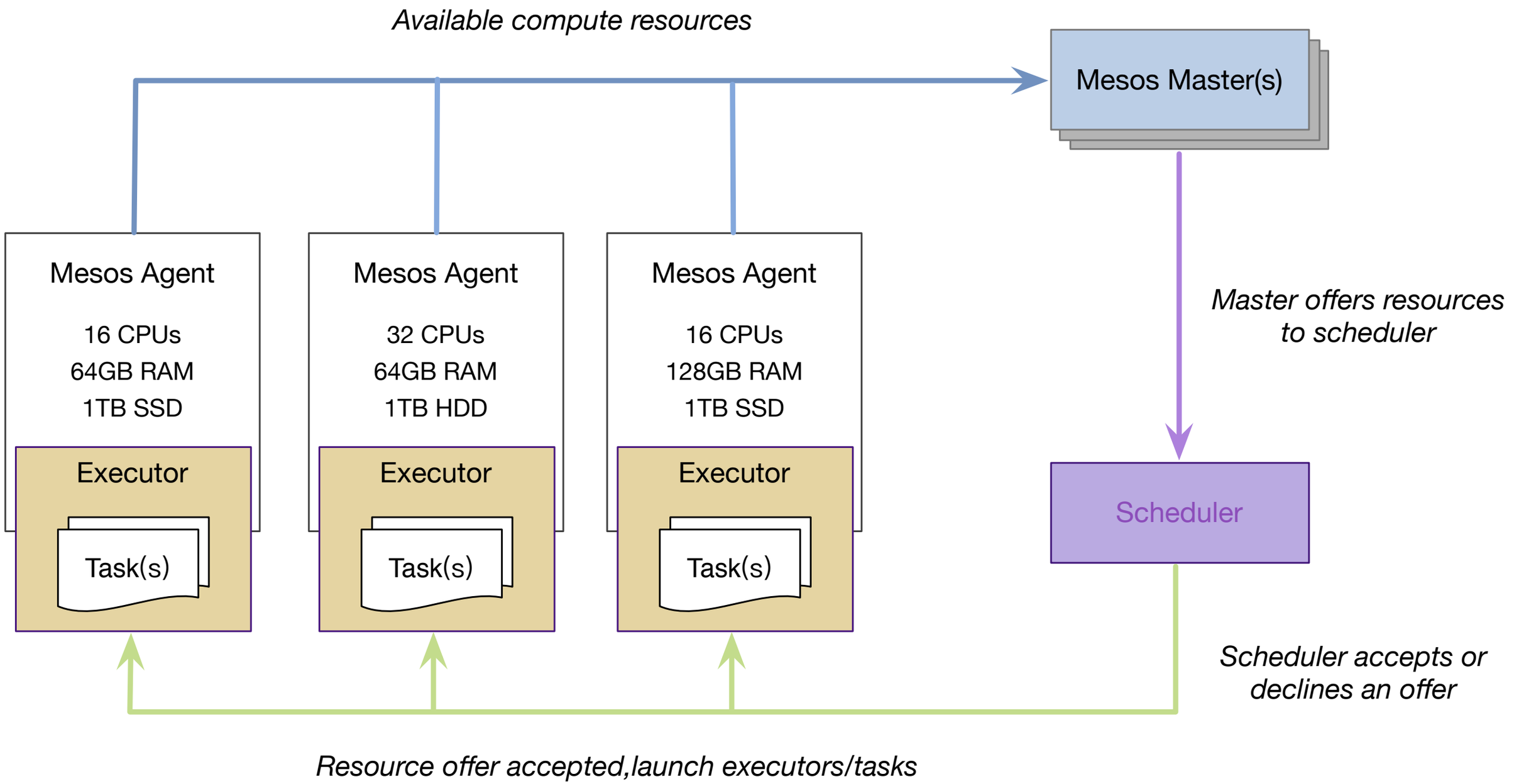
我们的方案

upyun

Mesos 基础架构


Framework A
Scheduler
Framework B
Scheduler
Resource offer
Launch Task(s)
Mesos Master Quorum
Mesos Master
(Leader)
ZK
ZK
ZK
Mesos Master
(Standby)
Mesos Master
(Standby)
Resource offer
Launch Task(s)
Agent 1
Framework A
Executor
Task(s)
Agent n
Framework A
Executor
Task(s)
Framework B
Executor
Task(s)

Mesos 内部机制


Available compute resources
Mesos Master(s)
Mesos Agent
16 CPUs
64GB RAM
1TB SSD
Executor
Task(s)
Mesos Agent
32 CPUs
64GB RAM
1TB HDD
Executor
Task(s)
Mesos Agent
16 CPUs
128GB RAM
1TB SSD
Executor
Task(s)
Master offers resources to scheduler
Scheduler
Scheduler accepts or declines an offer
Resource offer accepted,launch executors/tasks

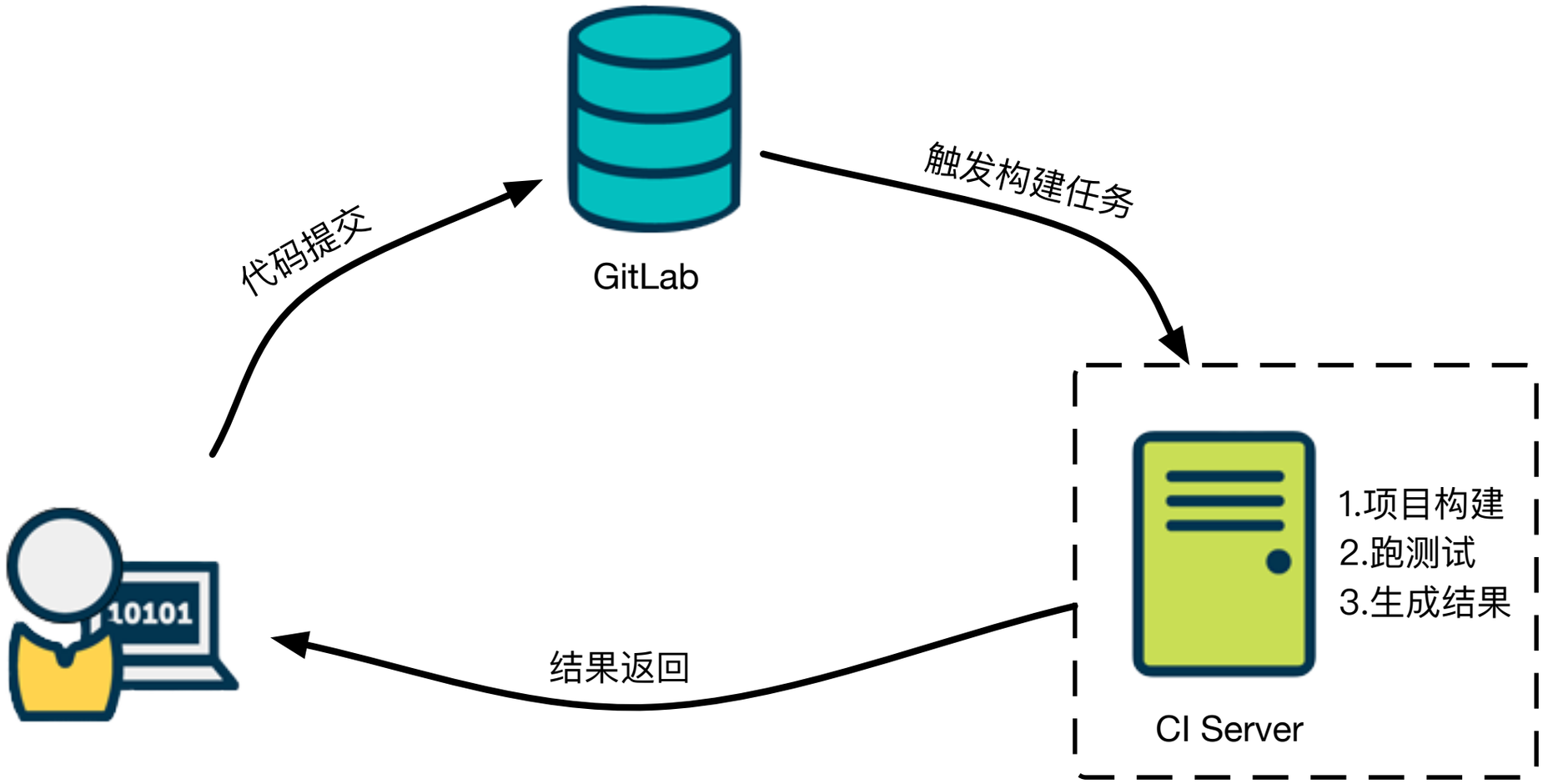
除此之外，还需要

- 镜像持续交付
- 动态服务路由
- 服务日志收集
- 服务监控告警

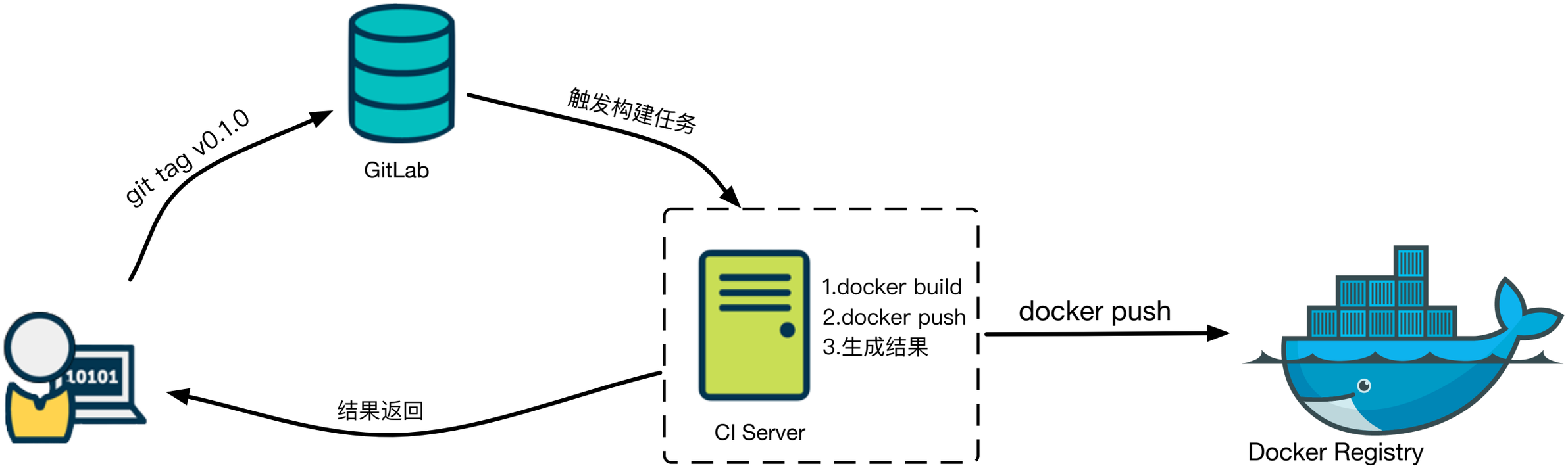
除此之外，还需要

- **镜像持续交付**
- 动态服务路由
- 服务日志收集
- 服务监控告警

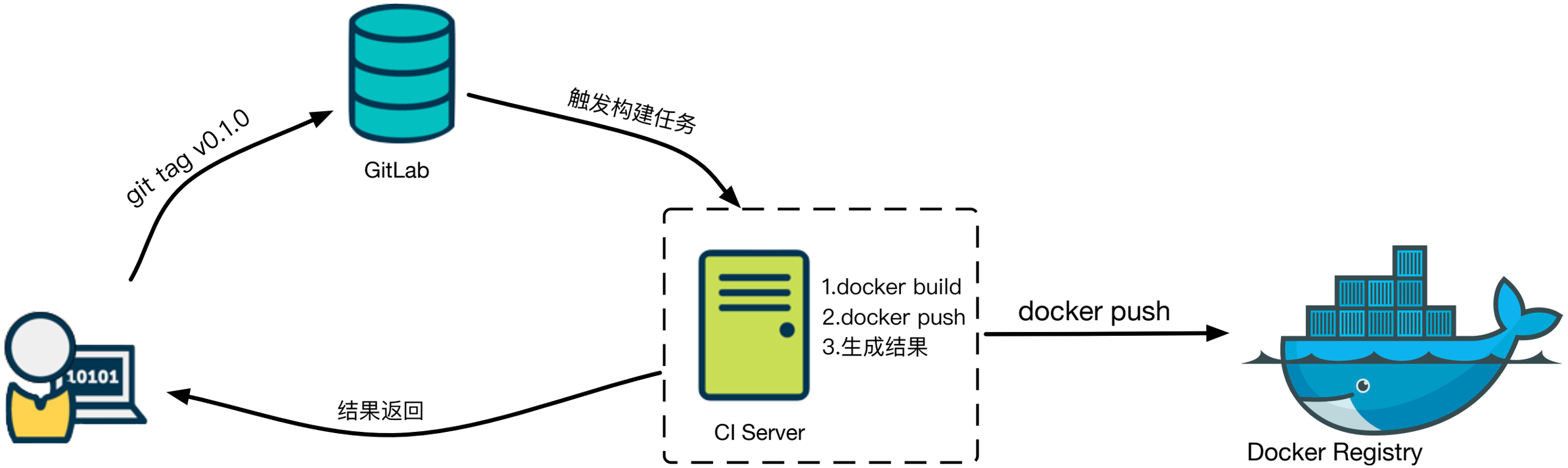
Docker 持续集成



Docker 持续交付

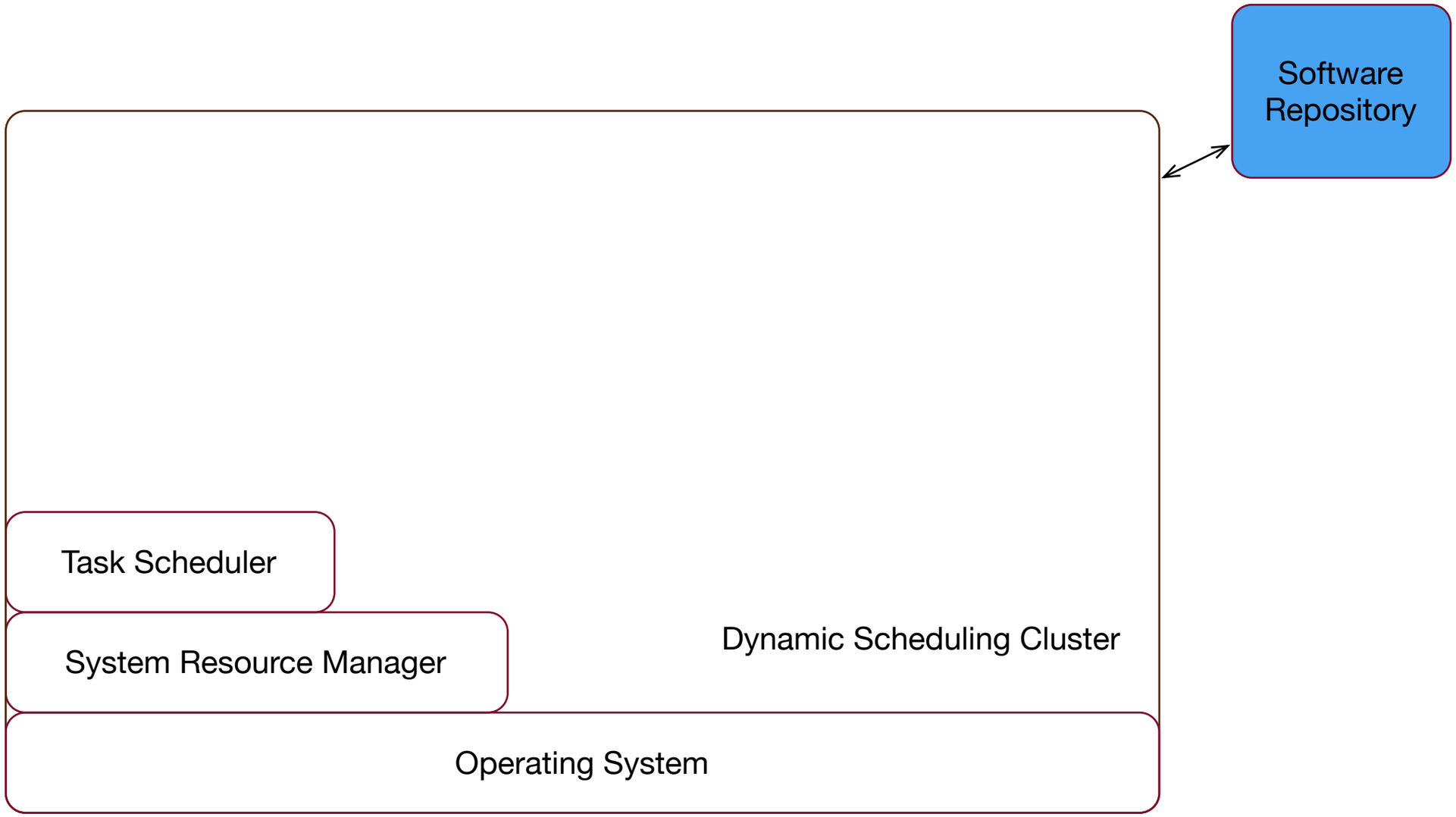

git tag v0.1.0
GitLab
触发构建任务
1.docker build
2.docker push
3.生成结果
CI Server
docker push
Docker Registry
结果返回

Docker 持续交付

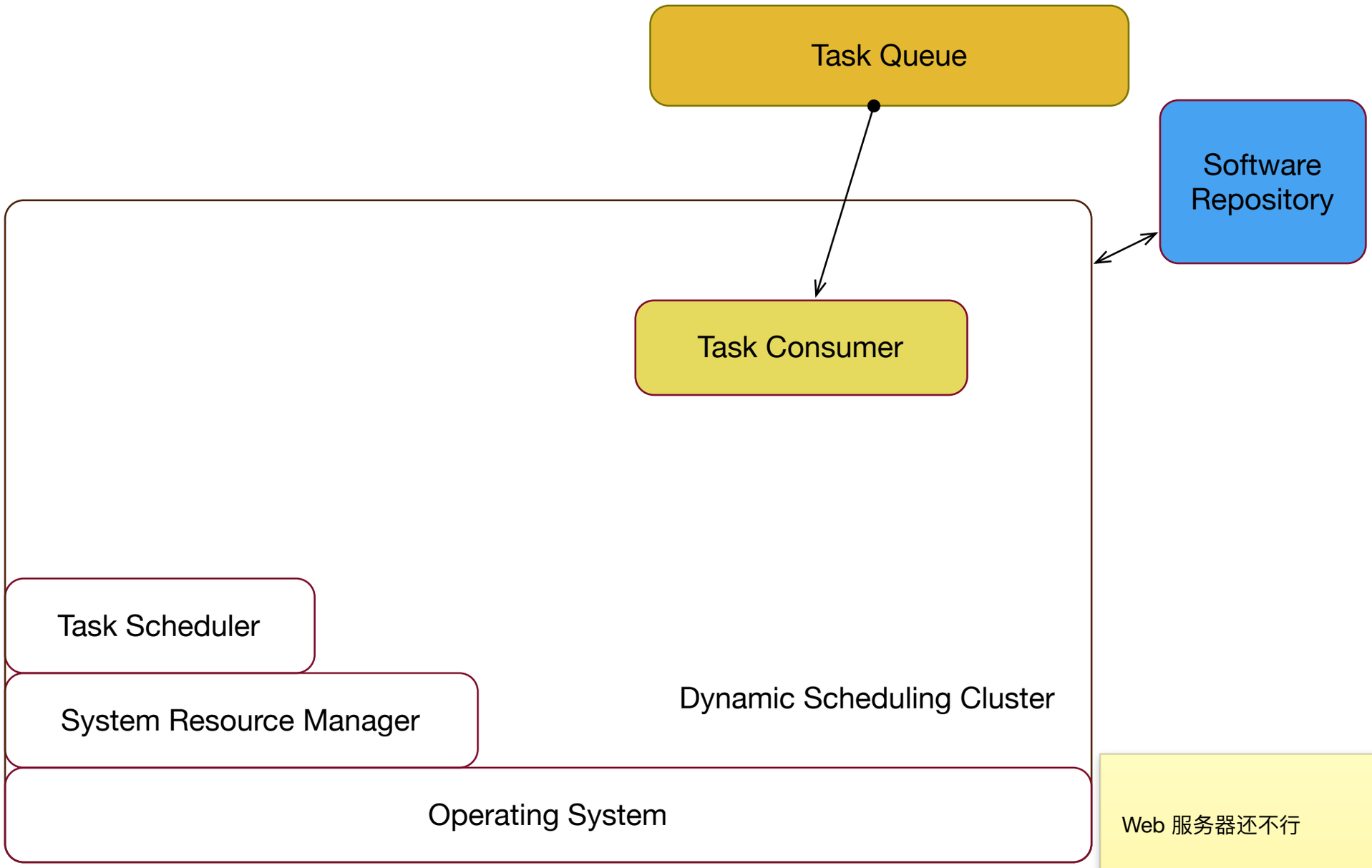



```
build:
    stage: build
    script:
        - img=repo.upyun.com:5043/echo-hello-world:$CI_BUILD_TAG
        - for i in {1..3}; do (docker build -t $img .) && break; done
        - docker push $img
    tags:
        - builder
    only:
        - tags@consumers/echo-hello-world
```

早期版本


Software Repository
Task Scheduler
System Resource Manager
Dynamic Scheduling Cluster
Operating System

早期版本



Web 服务器还不行

除此之外，还需要

- 镜像持续交付
- 动态服务路由
- 服务日志收集
- 服务监控告警

动态服务路由

upyun

HAProxy VS Nginx

动态服务路由

upyun

HAProxy VS Nginx/Slardar

动态服务路由 Slardar

upyun

NGINX + Lua

https://github.com/upyun/slardar

slardar 怎么工作的

Host 区分服务

```
$ curl -T cat.jpg 192.168.1.155:3130 \
    -H "Host: imageinfo"
$ curl -T cat.jpg 192.168.1.155:3130 \
    -H "Host: imgprocess"
```

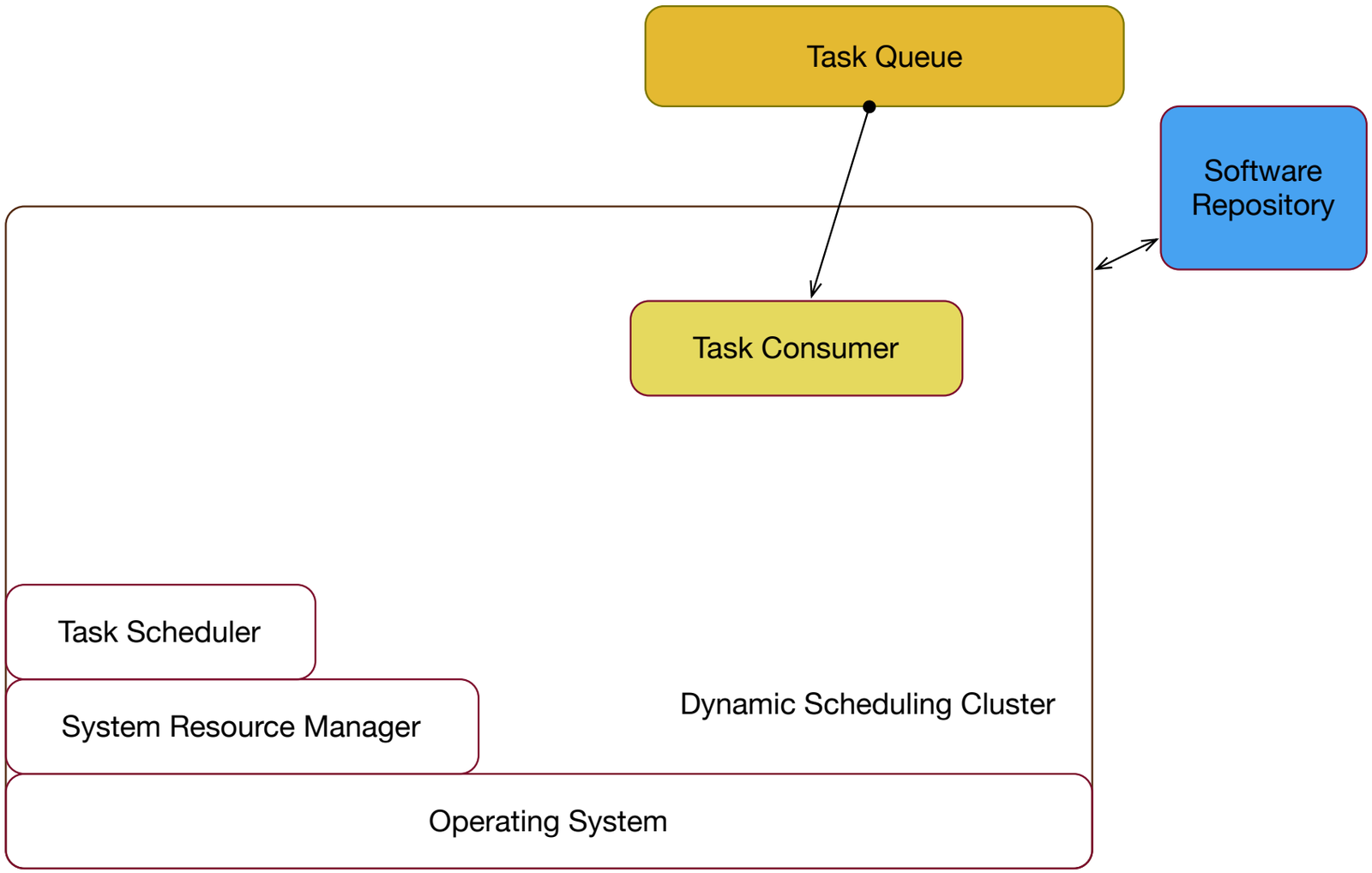
动态更新 upstream

```
curl -d \
  '{“servers”:[ \
   {"host":"10.0.5.108", "port": 4001}, \      ——> node docker 1
   {“host":"10.0.5.109", "port": 4001}], \     ——> node docker 2
   "keepalive": 20}'\
  127.0.0.1:1995/upstream/node-dev.upyun.com
```

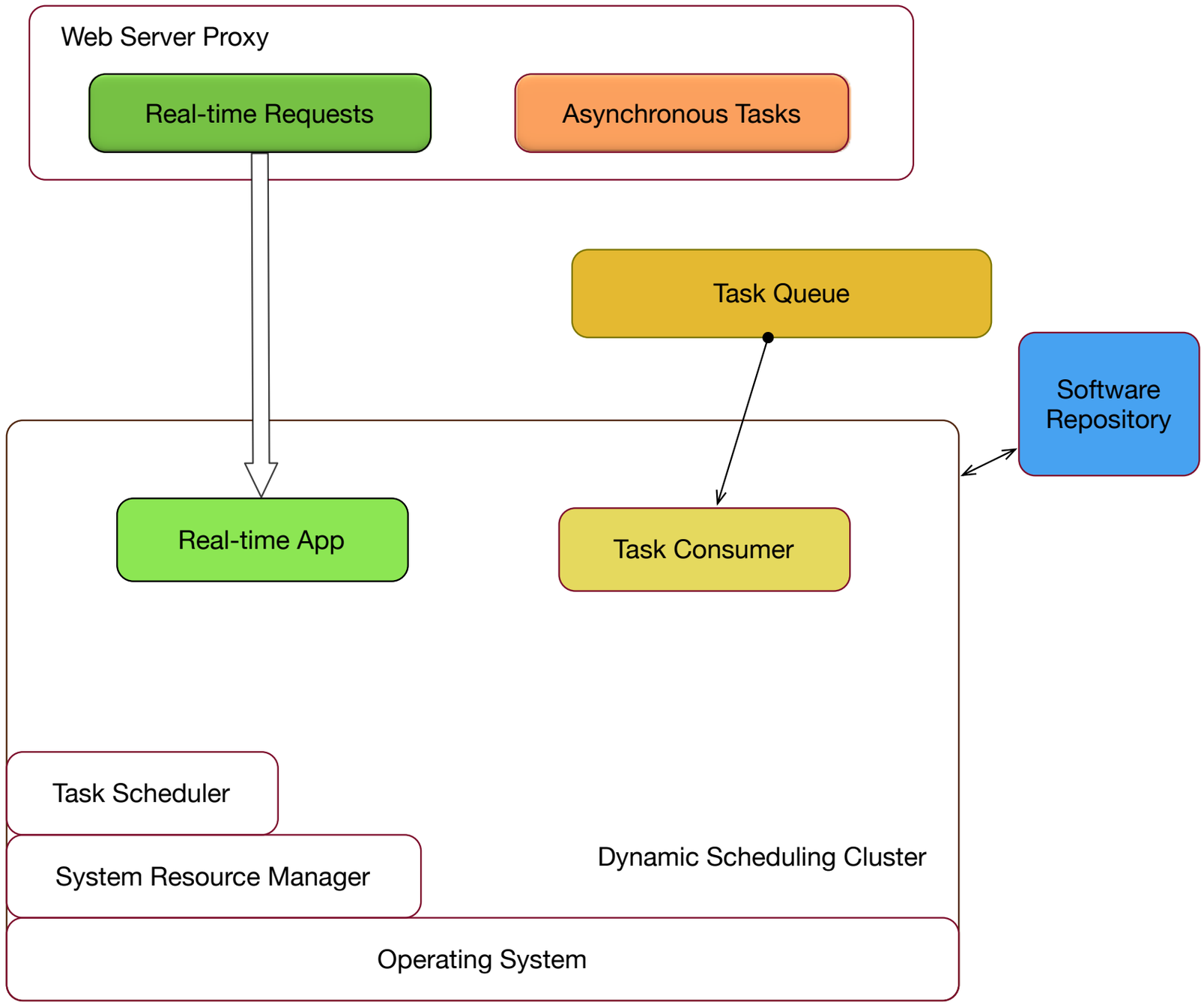
动态服务路由 Slardar


```
"cls:node-dev.upyun.com": [
  [
    {
      "server": "node-dev.upyun.com:10.0.5.108:4001",
      "msg": null,
      "weight": 1,
      "status": "ok",
      "lastmodified": "2016-11-29 09:38:34",
      "fail_num": 0
    },
    {
      "server": "node-dev.upyun.com:10.0.5.109:4001",
      "msg": "connection refused",
      "weight": 1,
      "status": "err",
      "lastmodified": "2016-11-29 09:29:14",
      "fail_num": 114
    }
  ]
],
```

统一的服务路由 Slardar



统一的服务路由 Slardar



除此之外，还需要

- 镜像持续交付
- 动态服务路由
- 服务日志收集
- 服务监控告警

日志搜集面临的几个问题

日志量大、频率高

分布在多台机器

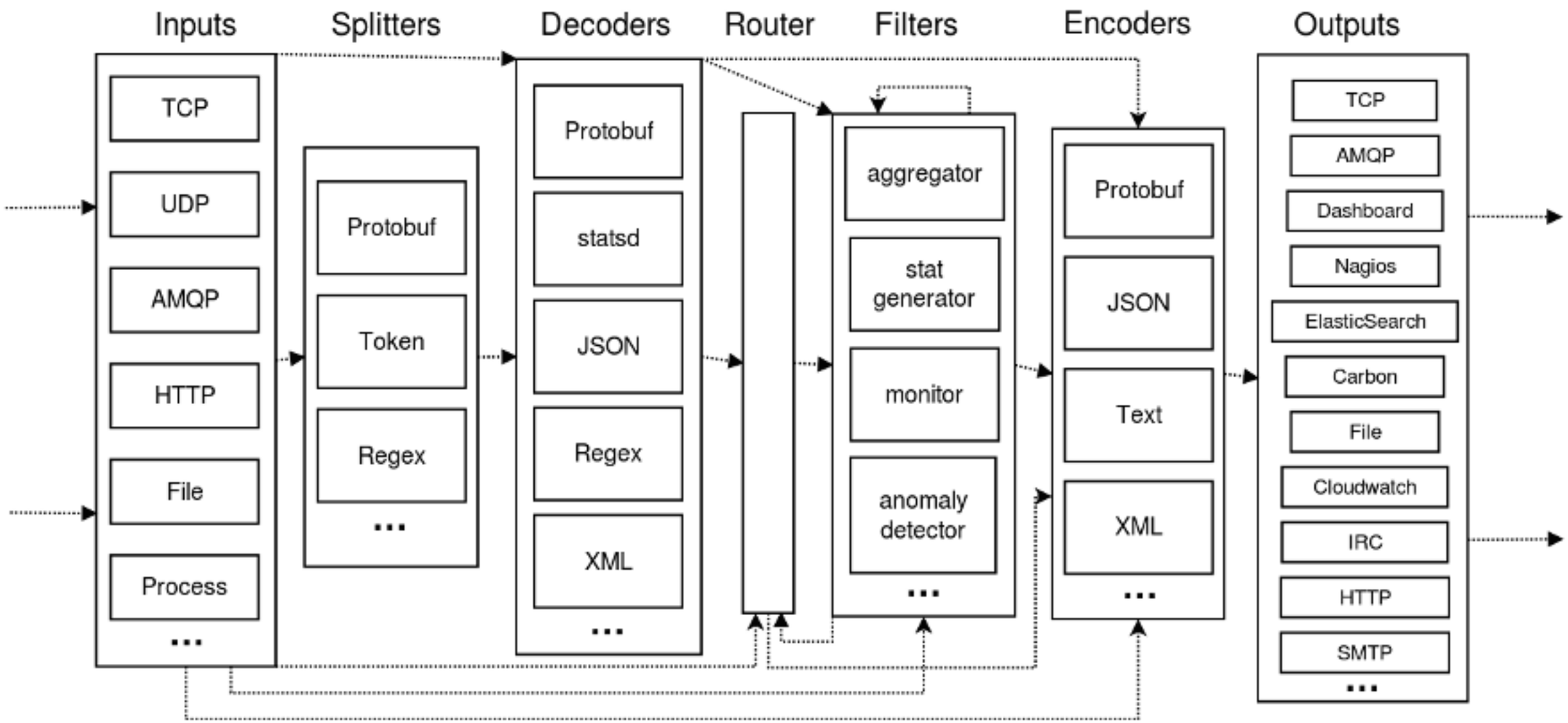
经常迁移

按服务分割

多种用途

Heka

upyun


Inputs
Splitters
Decoders
Router
Filters
Encoders
Outputs
TCP
UDP
AMQP
HTTP
File
Process
...
Protobuf
Token
Regex
...
Protobuf
statsd
JSON
Regex
XML
...
aggregator
stat generator
monitor
anomaly detector
...
Protobuf
JSON
Text
XML
...
TCP
AMQP
Dashboard
Nagios
ElasticSearch
Carbon
File
Cloudwatch
IRC
HTTP
SMTP
...

Heka 的 DockerLogInput 插件

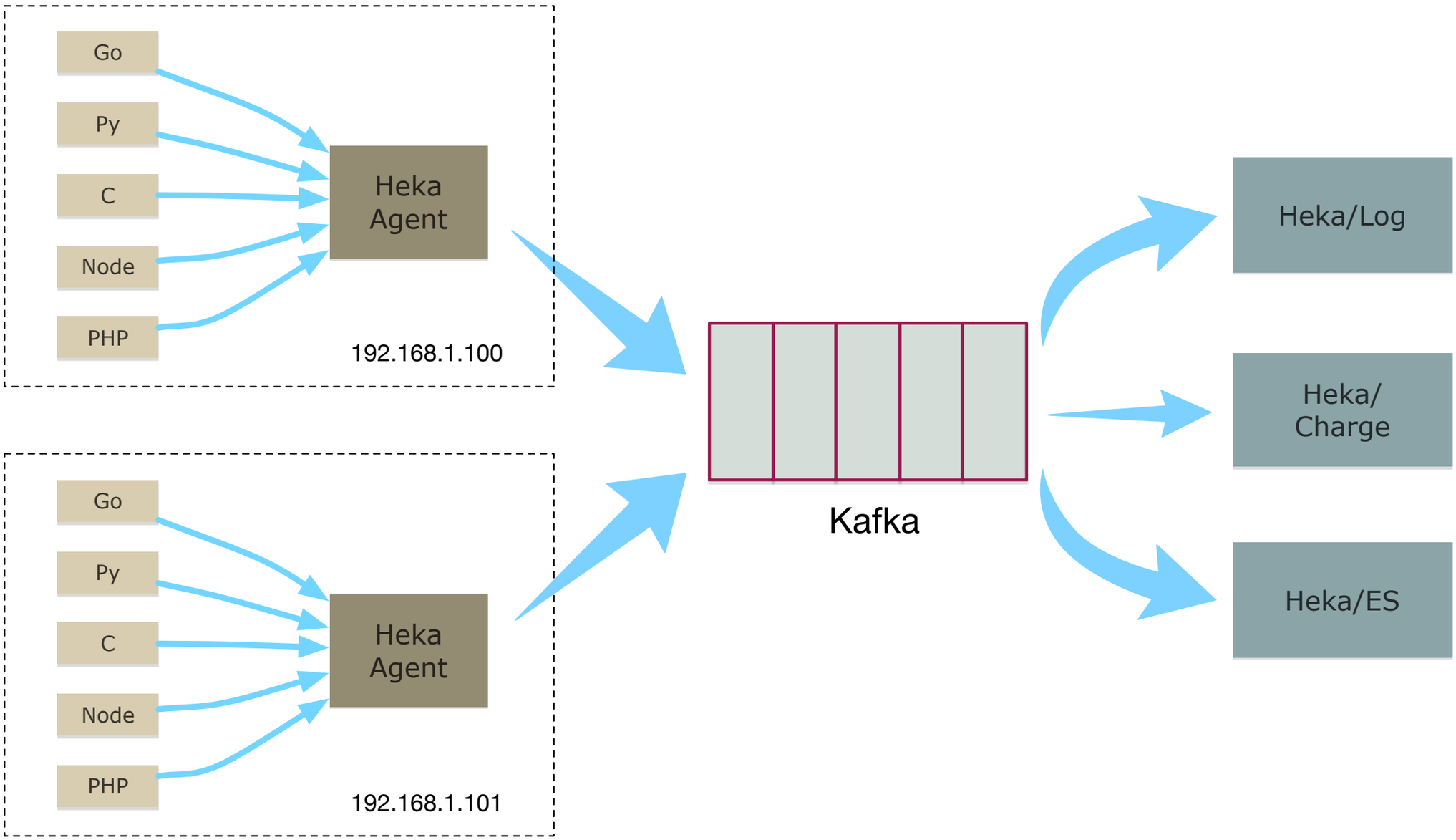
upyun

Plugin Name: **DockerLogInput**

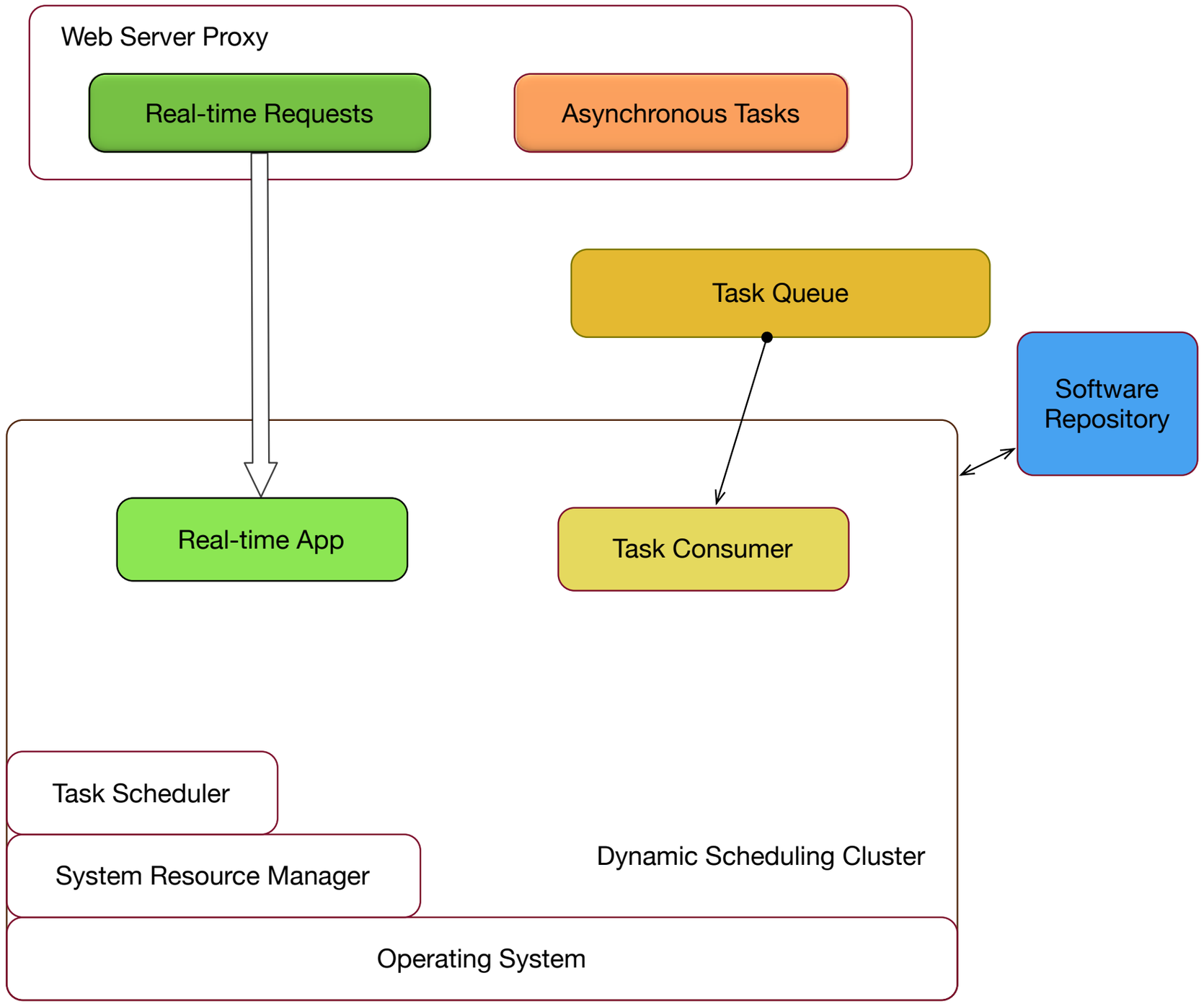
The DockerLogInput plugin attaches to all containers running on a host and sends their logs messages into the Heka pipeline. The plugin is based on Logspout by Jeff Lindsay. Messages will be populated as follows:

- Uuid: Type 4 (random) UUID generated by Heka.
- Timestamp: Time when the log line was received by the plugin.
- Type: *DockerLog*.
- Hostname: Hostname of the machine on which Heka is running.
- Payload: The log line received from a Docker container.
- Logger: *stdout* or *stderr*, depending on source.
- Fields[“ContainerID”] (string): The container ID.
- Fields[“ContainerName”] (string): The container name.

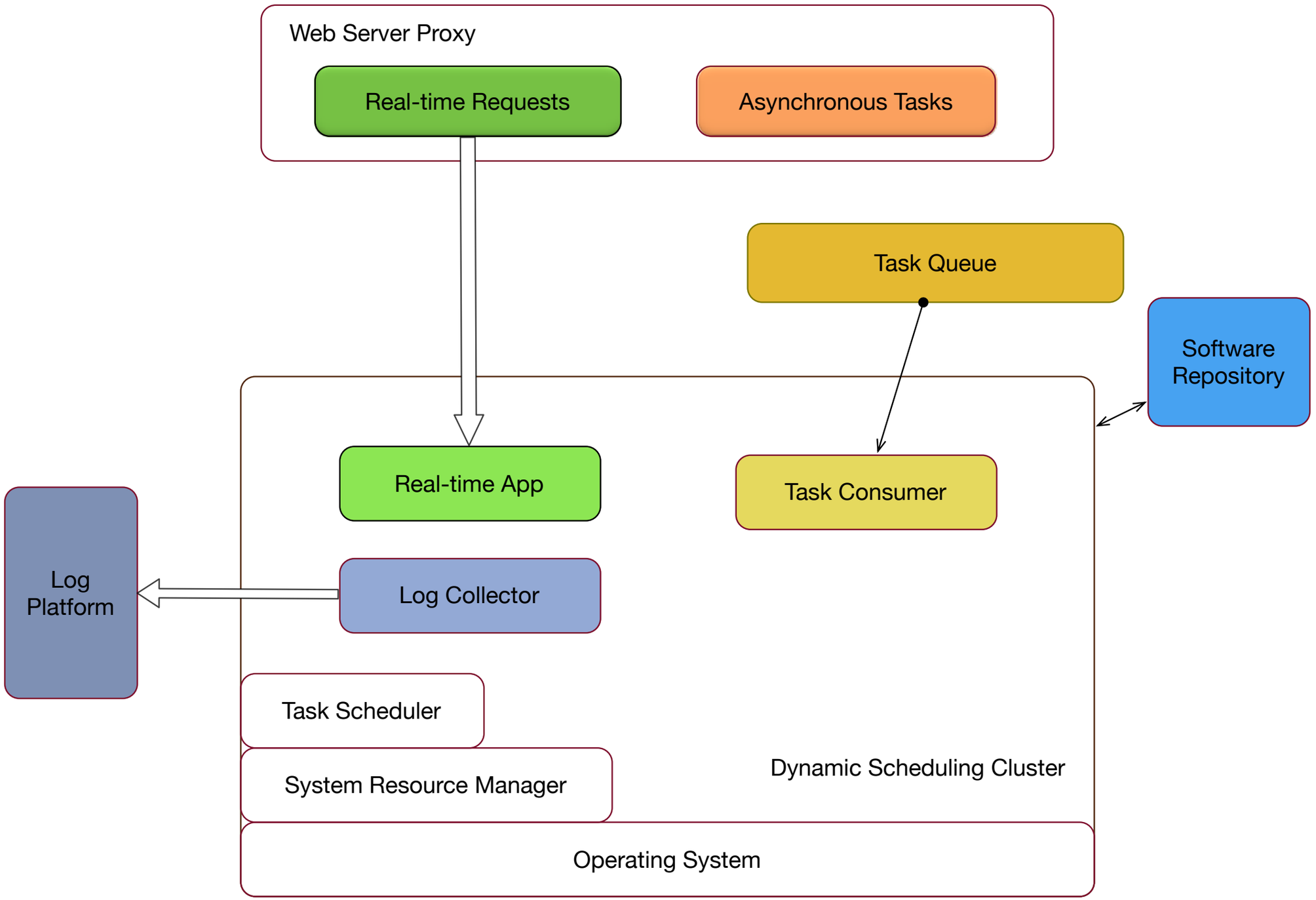
Heka 的 DockerLogInput 插件



Heka 的 DockerLogInput 插件



Heka 的 DockerLogInput 插件



除此之外，还需要

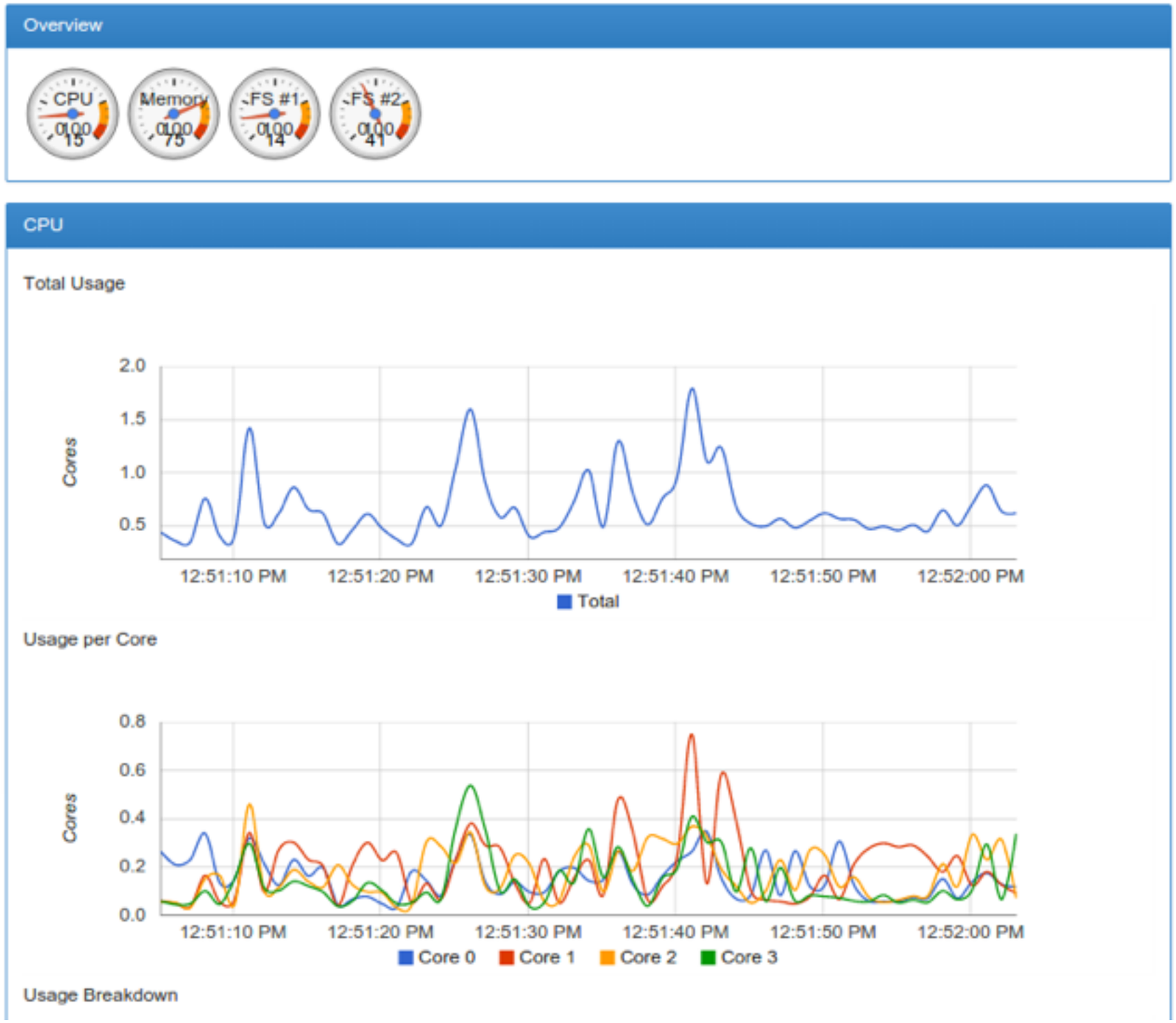
- 镜像持续交付
- 动态服务路由
- 服务日志收集
- **服务监控告警**

有所顾忌 Docker 的稳定性

DIY Docker 监控告警平台

upyun

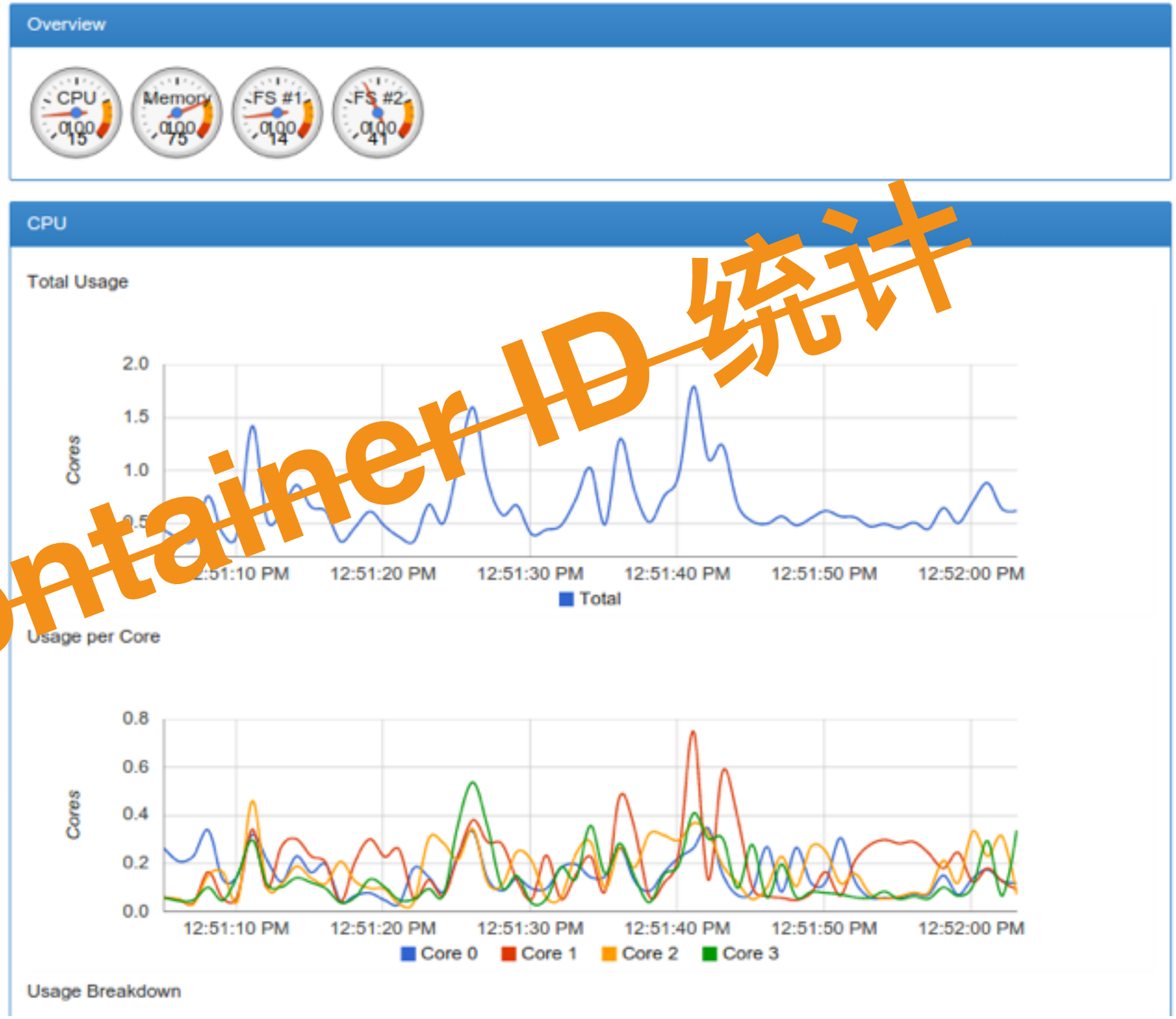
cAdvisor



DIY Docker 监控告警平台

upyun

cAdvisor

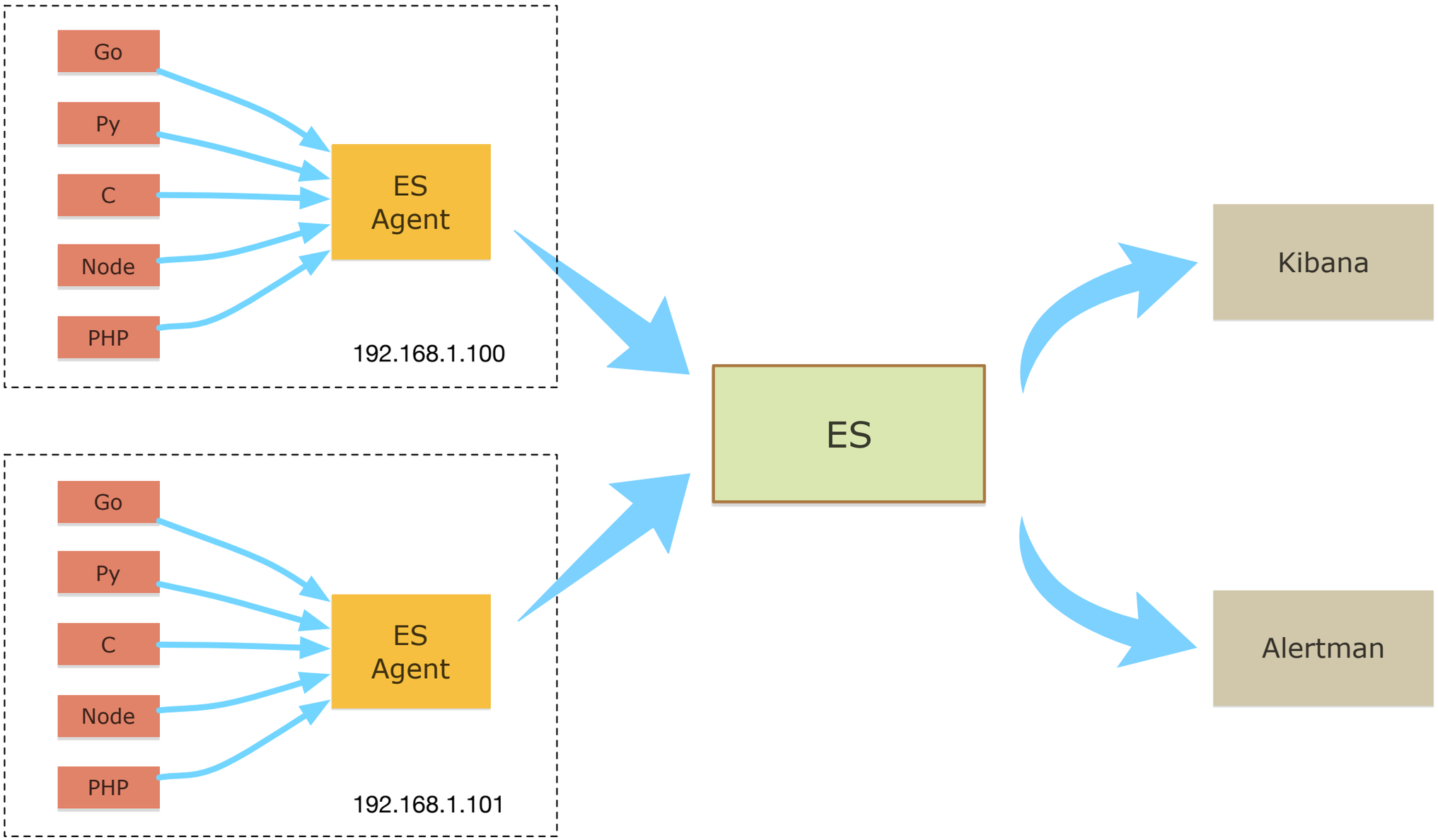


~~按 container 统计~~

粒度太细，无法对应服务名
多个实例，需要看整体状态

DIY Docker 监控告警平台

upyun



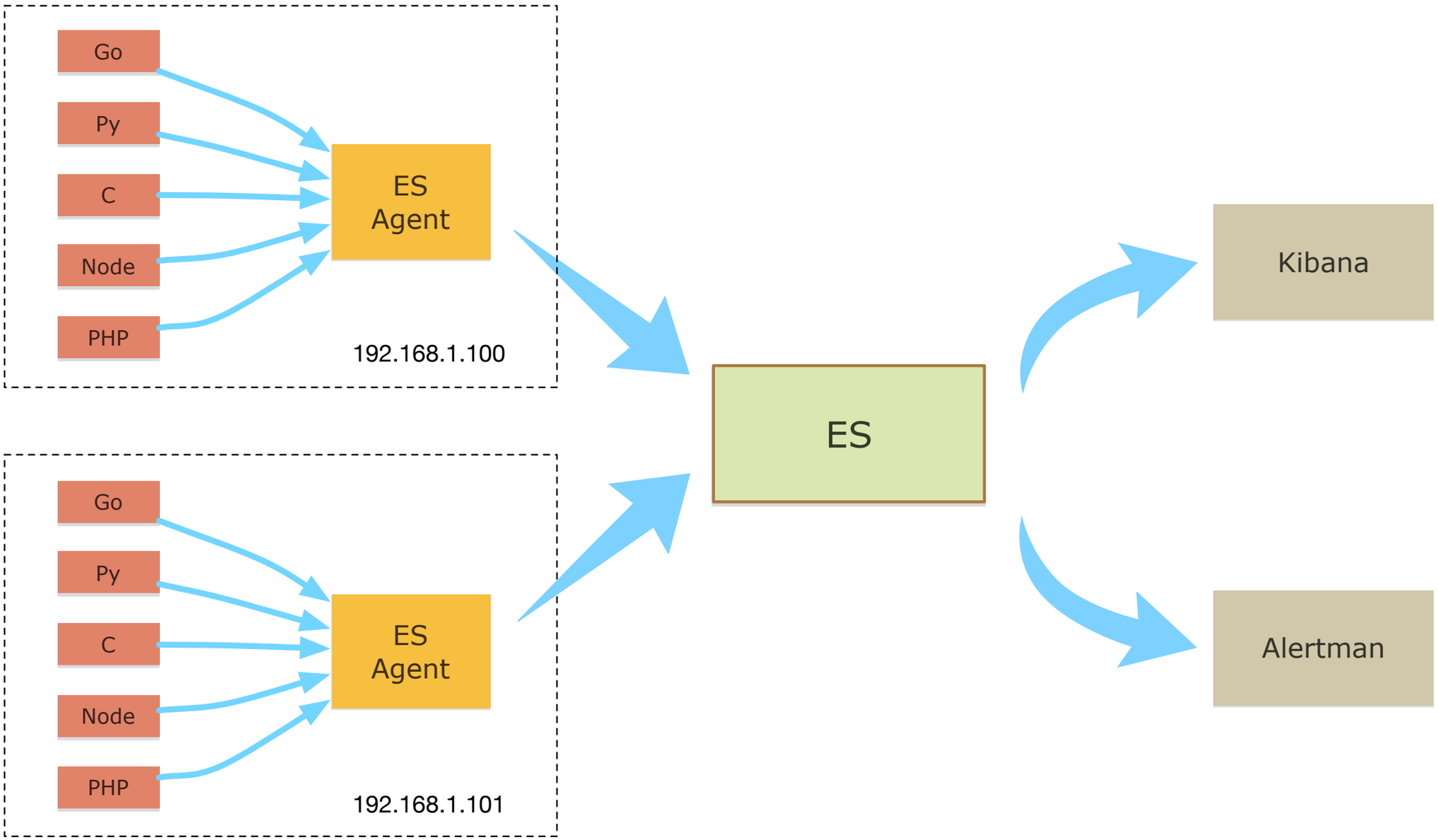
DIY Docker 监控告警平台

upyun

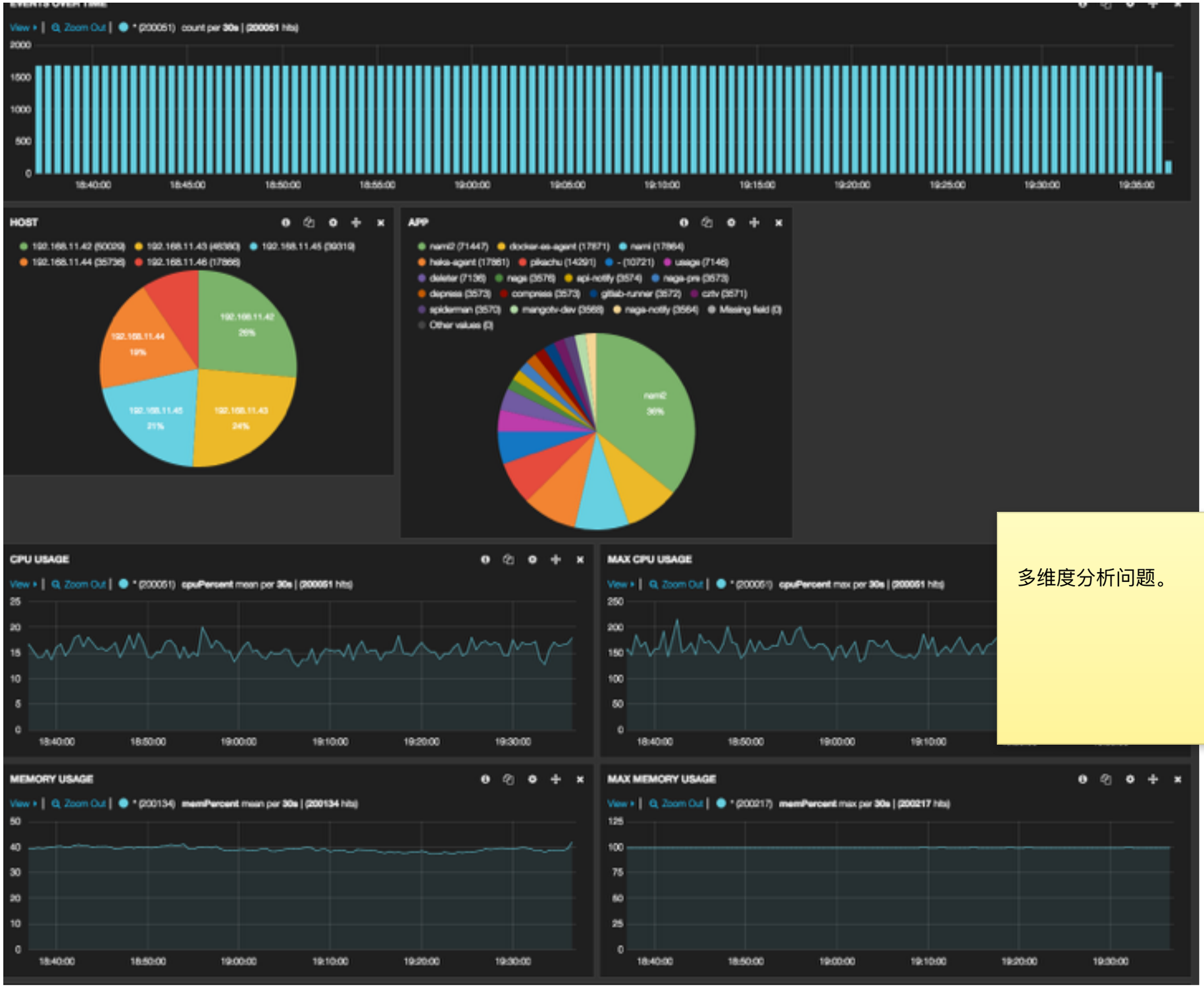
```
sendMetrix(map[string]interface{}{
        "host":         host,
        "cpuPercent":   uint64(cpuPercent),
        "memUsage":     stats.MemoryStats.Usage,
        "memLimit":     stats.MemoryStats.Limit,
        "maxMemUsage": stats.MemoryStats.MaxUsage,
        "memPercent":   uint64(memPercent),
        "rssPercent":   uint64(rssPercent),
        "blkRead":      blkRead,
        "blkWrite":     blkWrite,
        "netRx":        netRx,
        "netTx":        netTx,
        "name":         name,
        "cID":          cID,
        "appID":        appID,
        "stats":        stats,
})
```

DIY Docker 监控告警平台

upyun



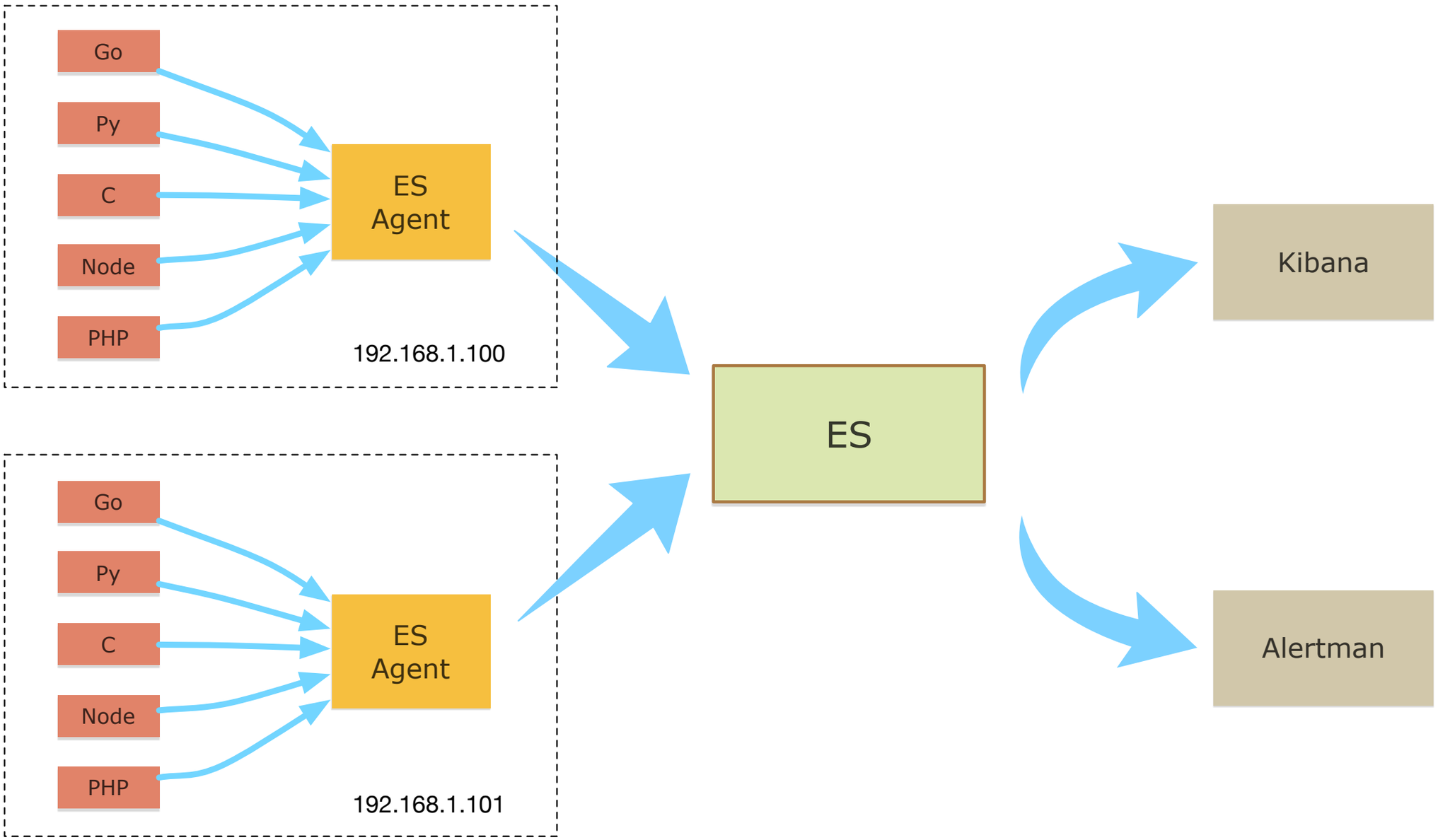
DIY Docker 监控告警平台



多维度分析问题。

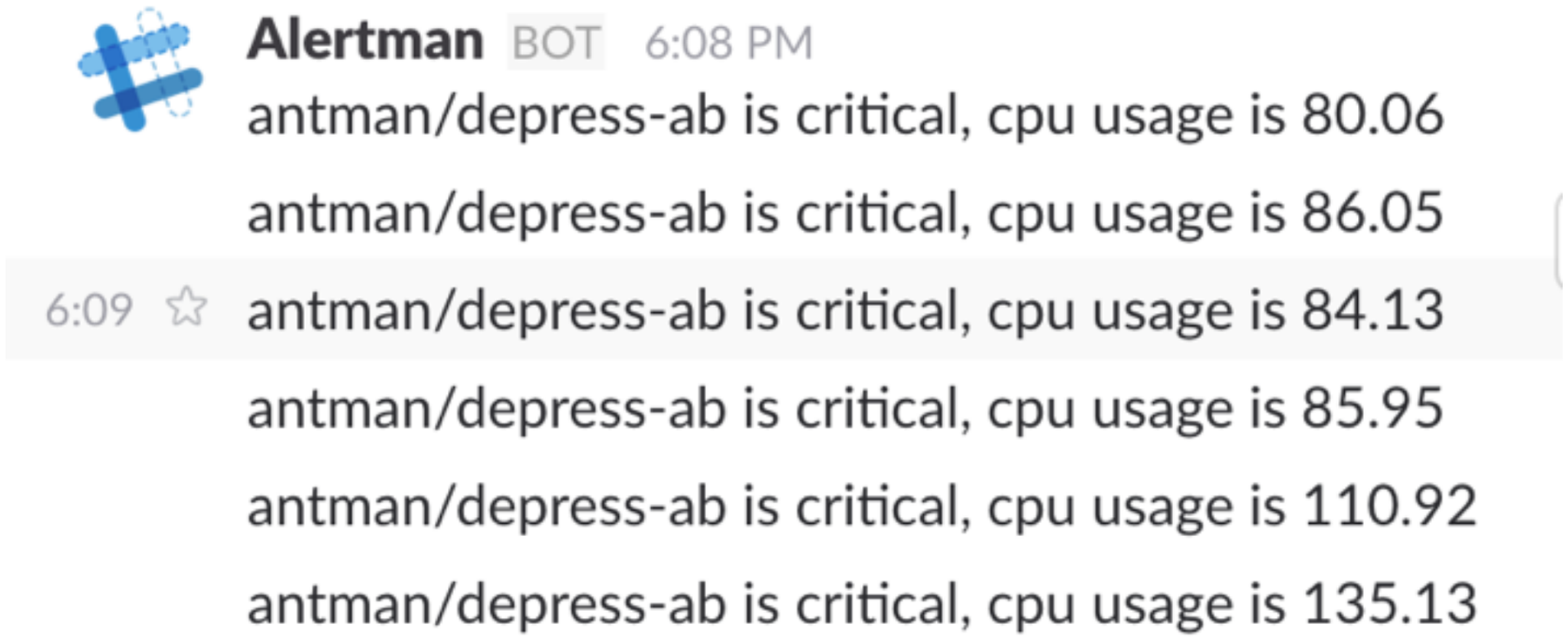
DIY Docker 监控告警平台

upyun



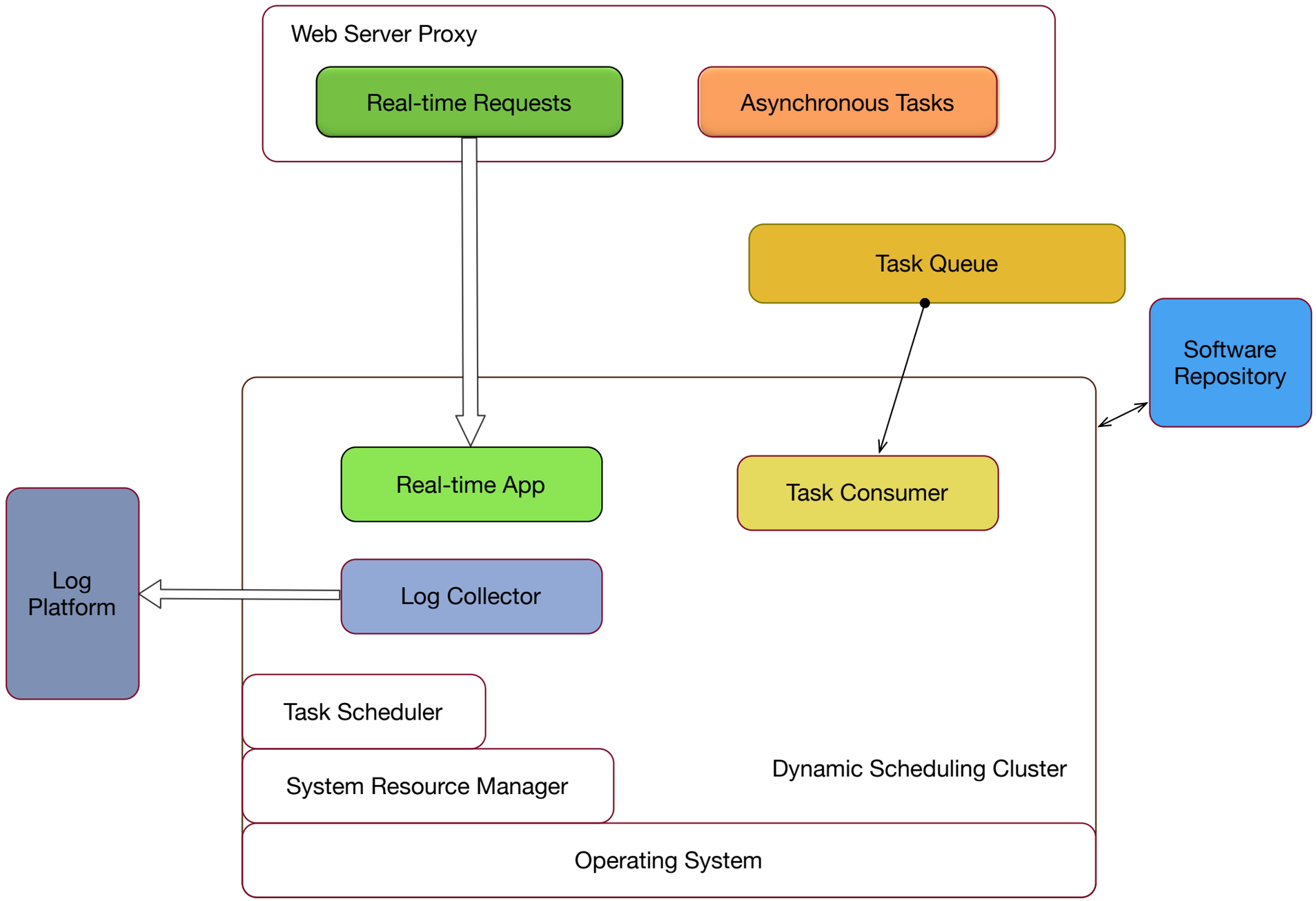
DIY Docker 监控告警平台

upyun



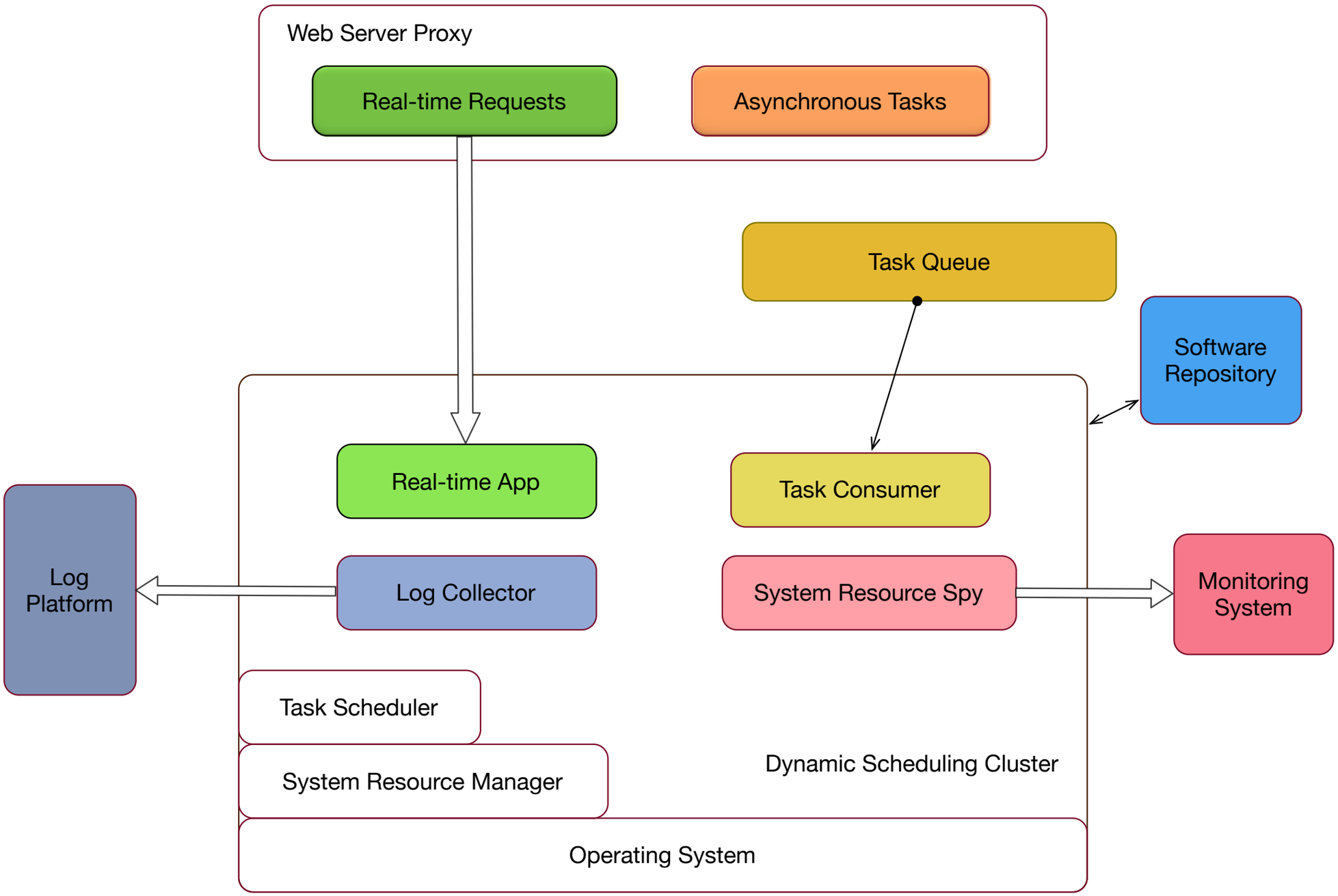
DIY Docker 监控告警平台

upyun

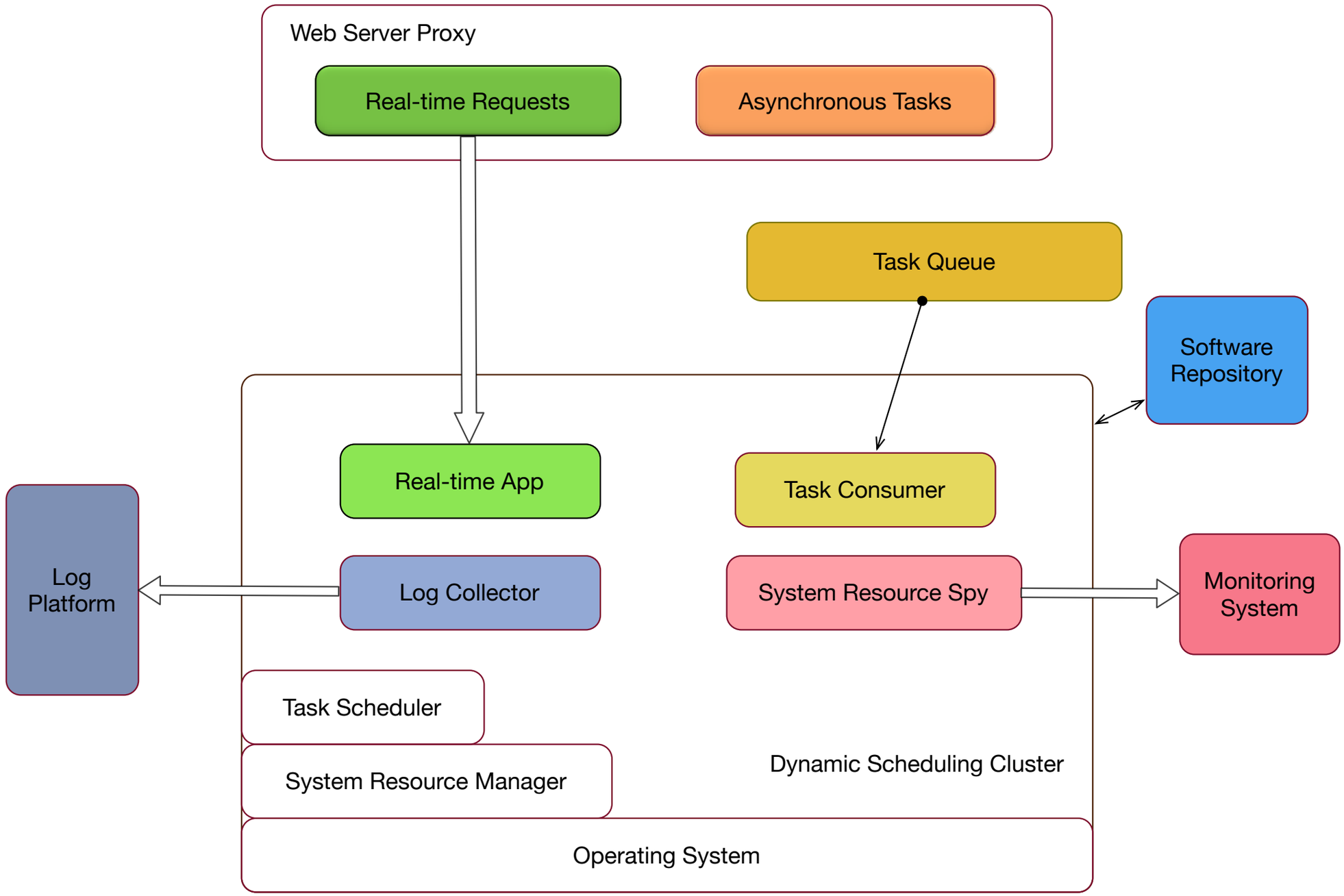


DIY Docker 监控告警平台

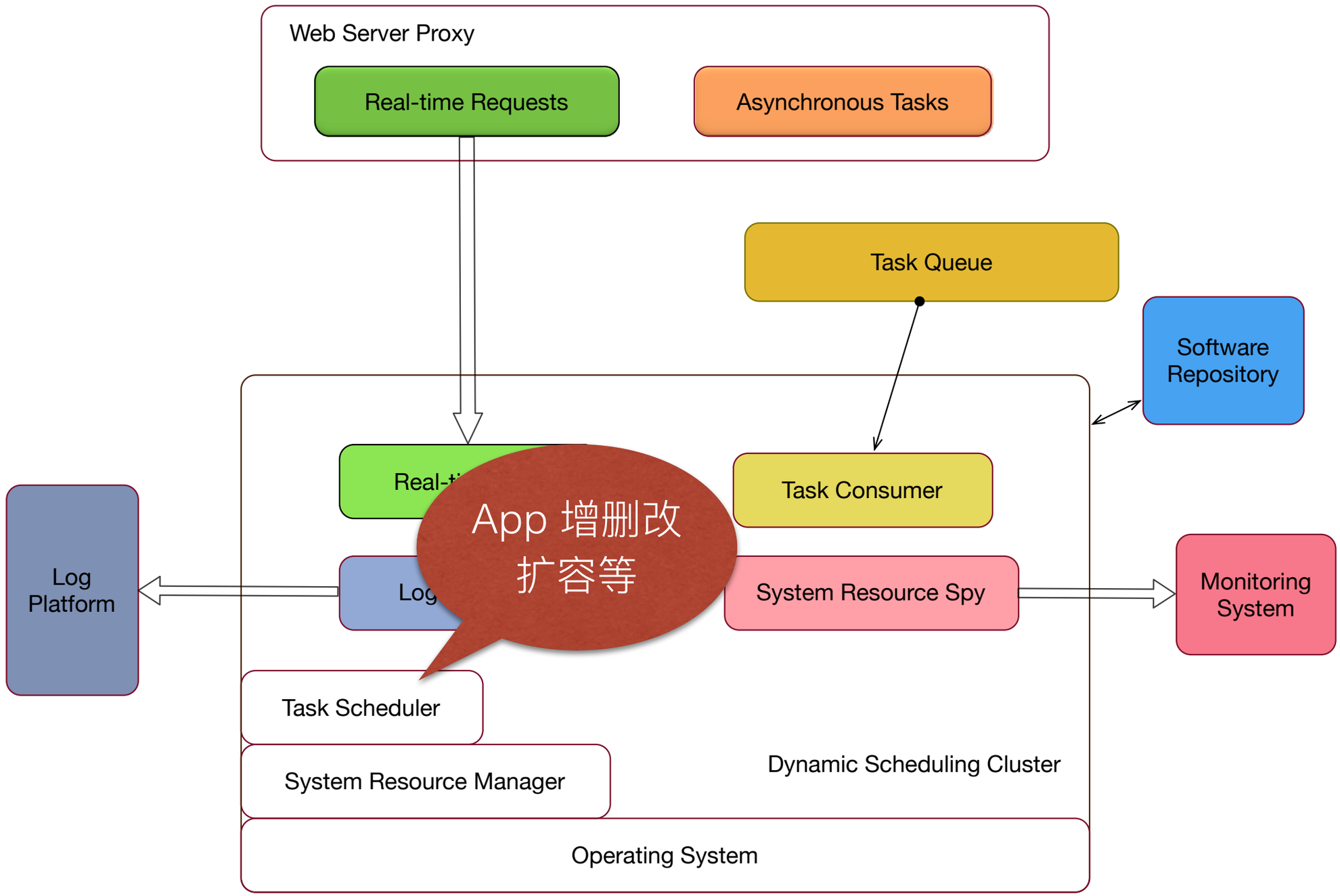
upyun



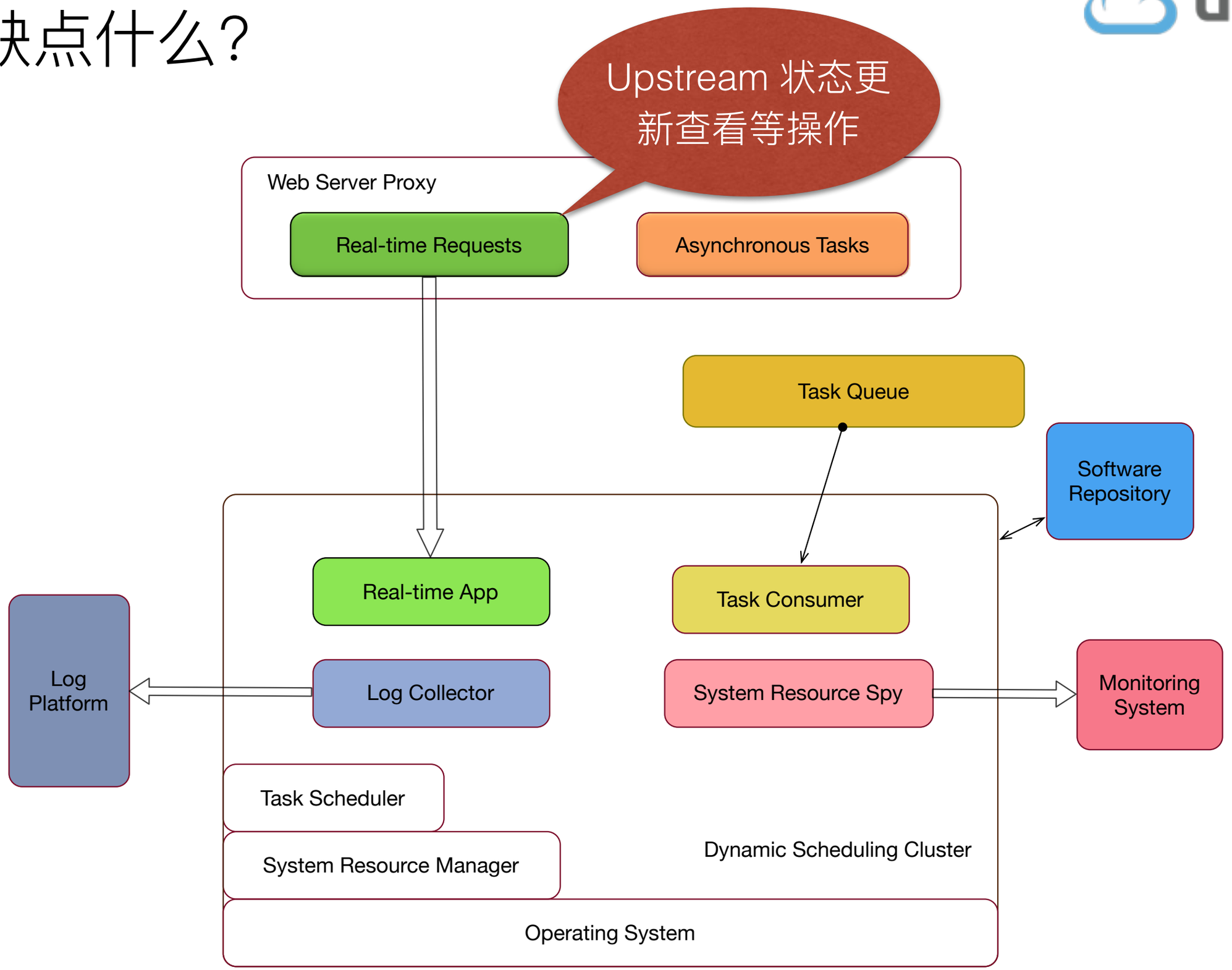
还缺点什么?


Web Server Proxy
Real-time Requests
Asynchronous Tasks
Task Queue
Software Repository
Real-time App
Task Consumer
Log Platform
Log Collector
System Resource Spy
Monitoring System
Task Scheduler
System Resource Manager
Dynamic Scheduling Cluster
Operating System

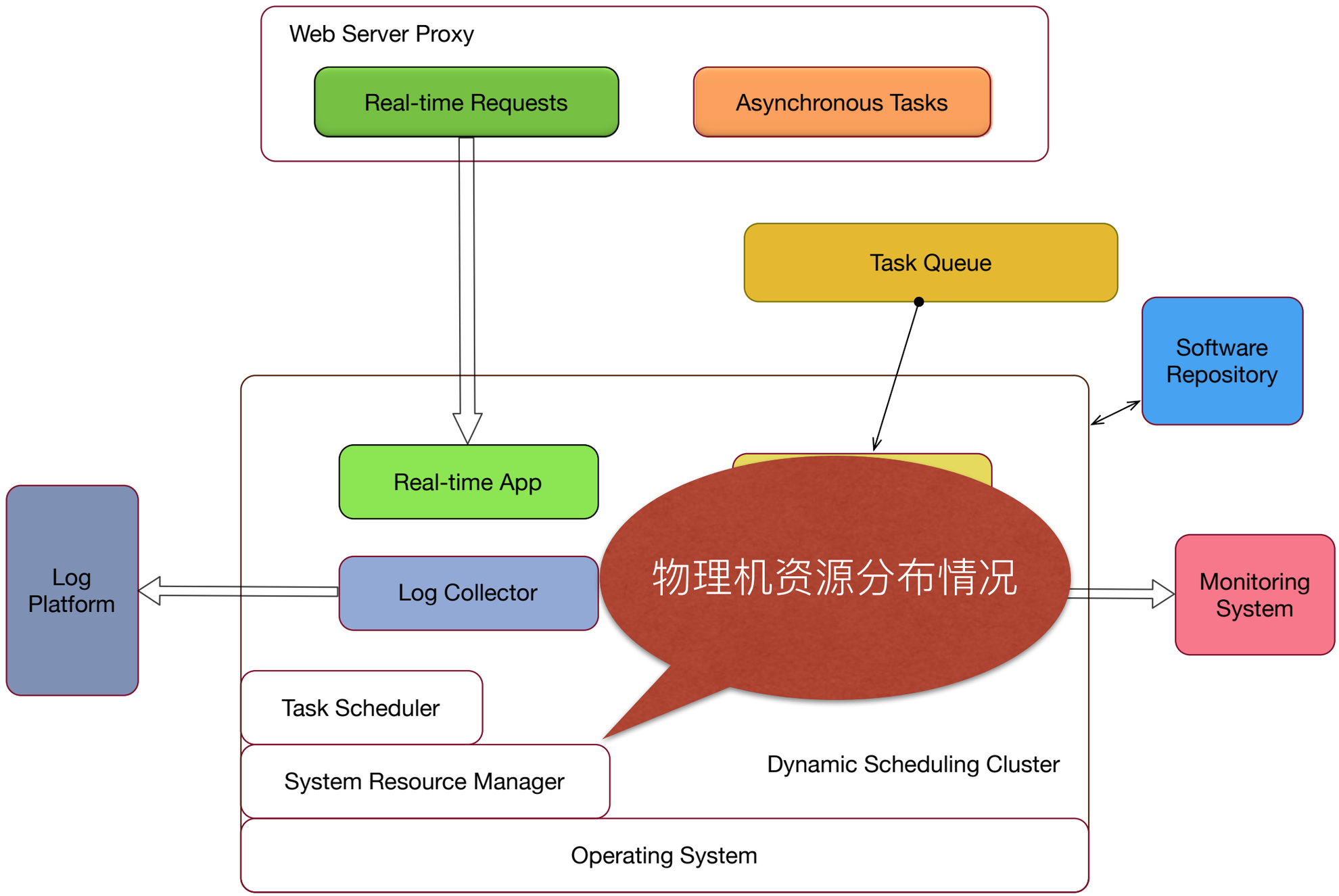
还缺点什么？


Web Server Proxy
Real-time Requests
Asynchronous Tasks
Task Queue
Software Repository
Task Consumer
App 增删改扩容等
Log Platform
System Resource Spy
Monitoring System
Task Scheduler
System Resource Manager
Dynamic Scheduling Cluster
Operating System

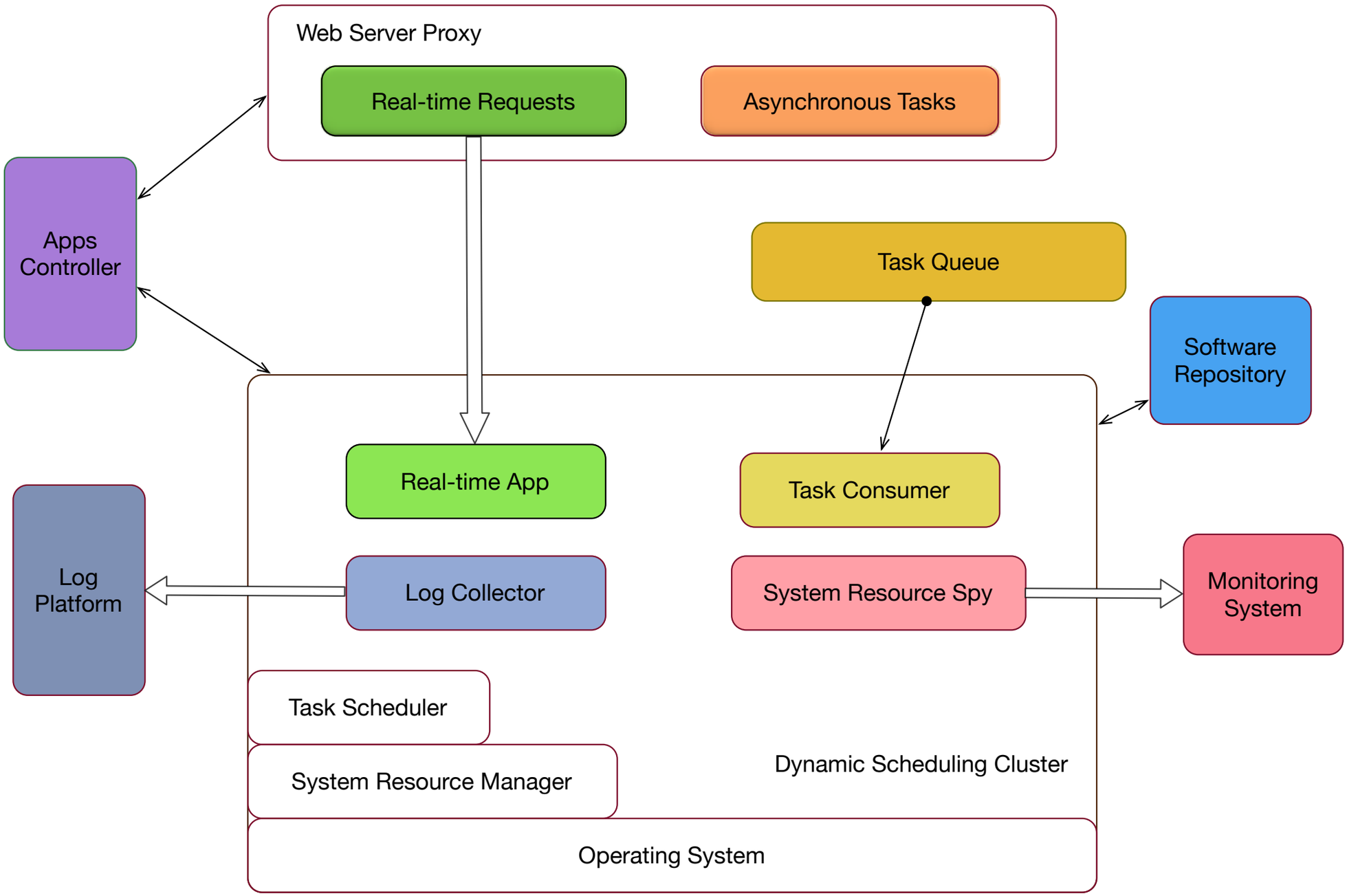
还缺点什么？


Upstream 状态更新查看等操作
Web Server Proxy
Real-time Requests
Asynchronous Tasks
Task Queue
Software Repository
Real-time App
Task Consumer
Log Platform
Log Collector
System Resource Spy
Monitoring System
Task Scheduler
System Resource Manager
Dynamic Scheduling Cluster
Operating System

还缺点什么?


Web Server Proxy
Real-time Requests
Asynchronous Tasks
Task Queue
Software Repository
Real-time App
Log Platform
Log Collector
物理机资源分布情况
Monitoring System
Task Scheduler
System Resource Manager
Dynamic Scheduling Cluster
Operating System

UPONE


Web Server Proxy
Real-time Requests
Asynchronous Tasks
Apps Controller
Task Queue
Software Repository
Real-time App
Task Consumer
Log Platform
Log Collector
System Resource Spy
Monitoring System
Task Scheduler
System Resource Manager
Dynamic Scheduling Cluster
Operating System

UPONE

UPONE

upyun

```
# required
group: guest
name: name
image: nginx

# optional
instances: 1
cpu: 0.1
memory: 64
network: bridge
privileged: false

command: /bin/bash
ports:
    - 8080

environment:
    APP_NAME: nginx
    APP_CONFIG: default
    CONSUL_HTTP_ADDR: 127.0.0.1:8500
    MY_ENV1: VAL1

zones:
    - slardar
    - slardar_test
services:
    - s1.upyun.com:8080
    - s2.upyun.com:8081
constraints:
    - hostname:UNIQUE
```

UPONE

```yaml
# required
group: guest
name: name
image: nginx

# optional
instances: 1
cpu: 0.1
memory: 64
network: bridge
privileged: false

command: /bin/bash
ports:
    - 8080

environment:
    APP_NAME: nginx
    APP_CONFIG: default
    CONSUL_HTTP_ADDR: 127.0.0.1:8500
    MY_ENV1: VAL1

zones:
    - slardar
    - slardar_test
services:
    - s1.upyun.com:8080
    - s2.upyun.com:8081
constraints:
    - hostname:UNIQUE
```

- 新建 App

```
$ upone init
$ upone deploy
$ upone app NAME sync
```

UPONE

```yaml
# required
group: guest
name: name
image: nginx

# optional
instances: 1
cpu: 0.1
memory: 64
network: bridge
privileged: false

command: /bin/bash
ports:
    - 8080

environment:
    APP_NAME: nginx
    APP_CONFIG: default
    CONSUL_HTTP_ADDR: 127.0.0.1:8500
    MY_ENV1: VAL1

zones:
    - slardar
    - slardar_test
services:
    - s1.upyun.com:8080
    - s2.upyun.com:8081
constraints:
    - hostname:UNIQUE
```

- 新建 App

```
$ upone init
$ upone deploy
$ upone app NAME sync
```

- 弹性扩容

```
$ upone app NAME scale 15
```

UPONE

```yaml
# required
group: guest
name: name
image: nginx

# optional
instances: 1
cpu: 0.1
memory: 64
network: bridge
privileged: false

command: /bin/bash
ports:
    - 8080

environment:
    APP_NAME: nginx
    APP_CONFIG: default
    CONSUL_HTTP_ADDR: 127.0.0.1:8500
    MY_ENV1: VAL1

zones:
    - slardar
    - slardar_test
services:
    - s1.upyun.com:8080
    - s2.upyun.com:8081
constraints:
    - hostname:UNIQUE
```

- 新建 App

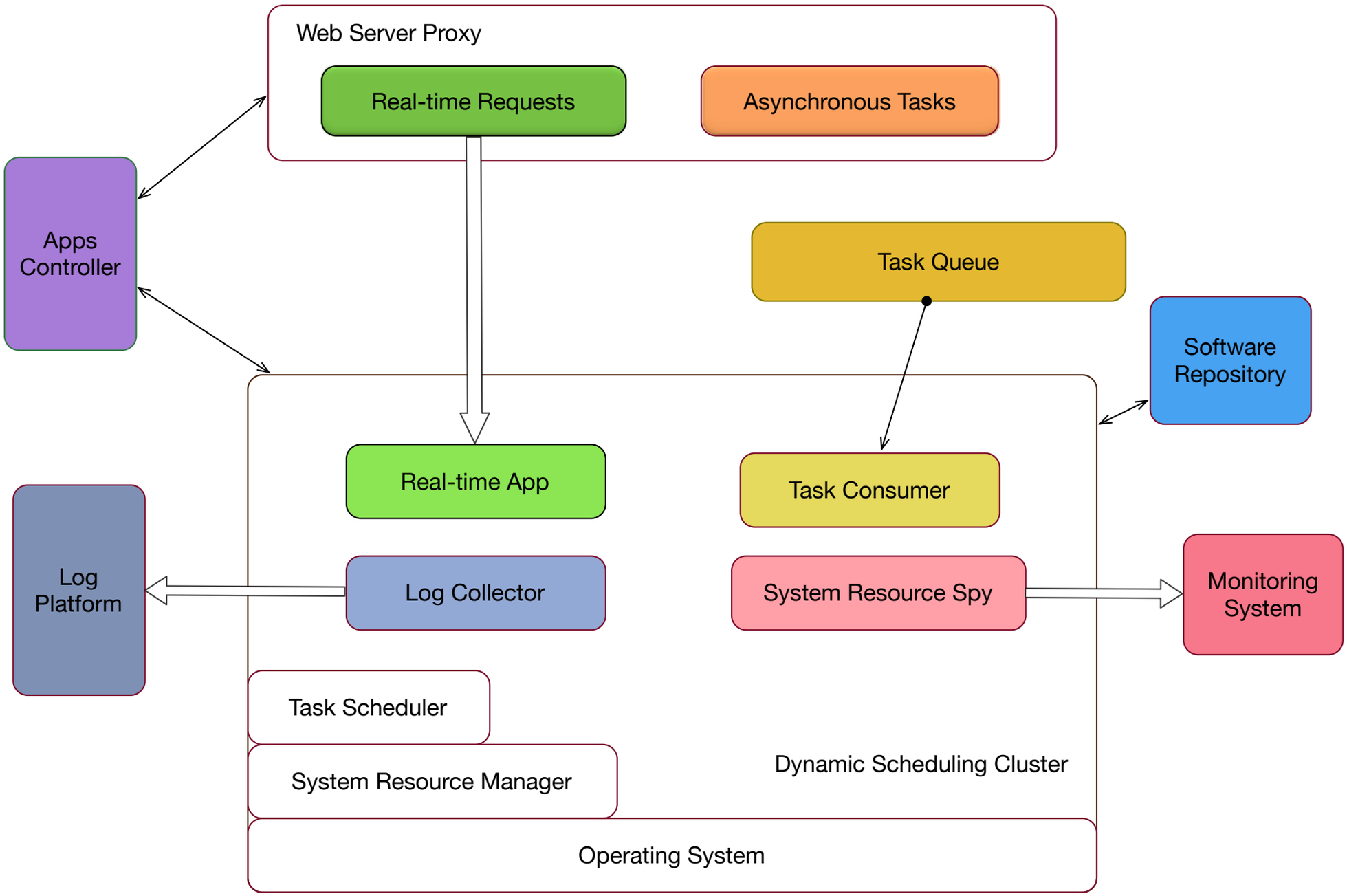
```
$ upone init
$ upone deploy
$ upone app NAME sync
```

- 弹性扩容

```
$ upone app NAME scale 15
```

- App 更新

```
$ upone app NAME upgrade
      NEW IMAGE
```


Web Server Proxy
Real-time Requests
Asynchronous Tasks
Apps Controller
Task Queue
Software Repository
Real-time App
Task Consumer
Log Platform
Log Collector
System Resource Spy
Monitoring System
Task Scheduler
System Resource Manager
Dynamic Scheduling Cluster
Operating System

可以更加智能

Nginx 日志

ES 数据

队列

.....

数据分析

可以更加智能

Nginx 日志

ES 数据

队列

……

数据分析

自动扩容

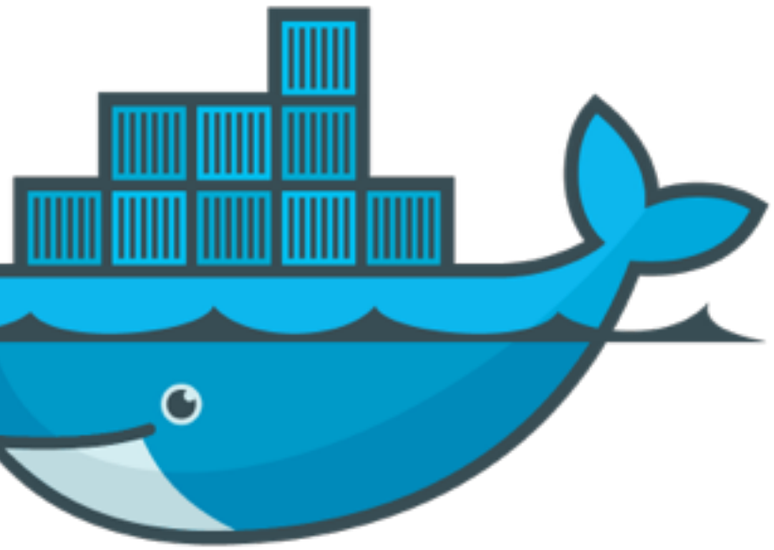
异常节点摘除

整合资源

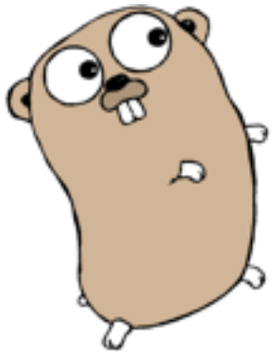
总结

upyun

kibana



NGINX



upyun

Q&A