SACC

破解阿喀琉斯之踵

--利用系统开发破解架构中的硬伤


运动啦！ 七、八年就来一次～！

我们是谁？

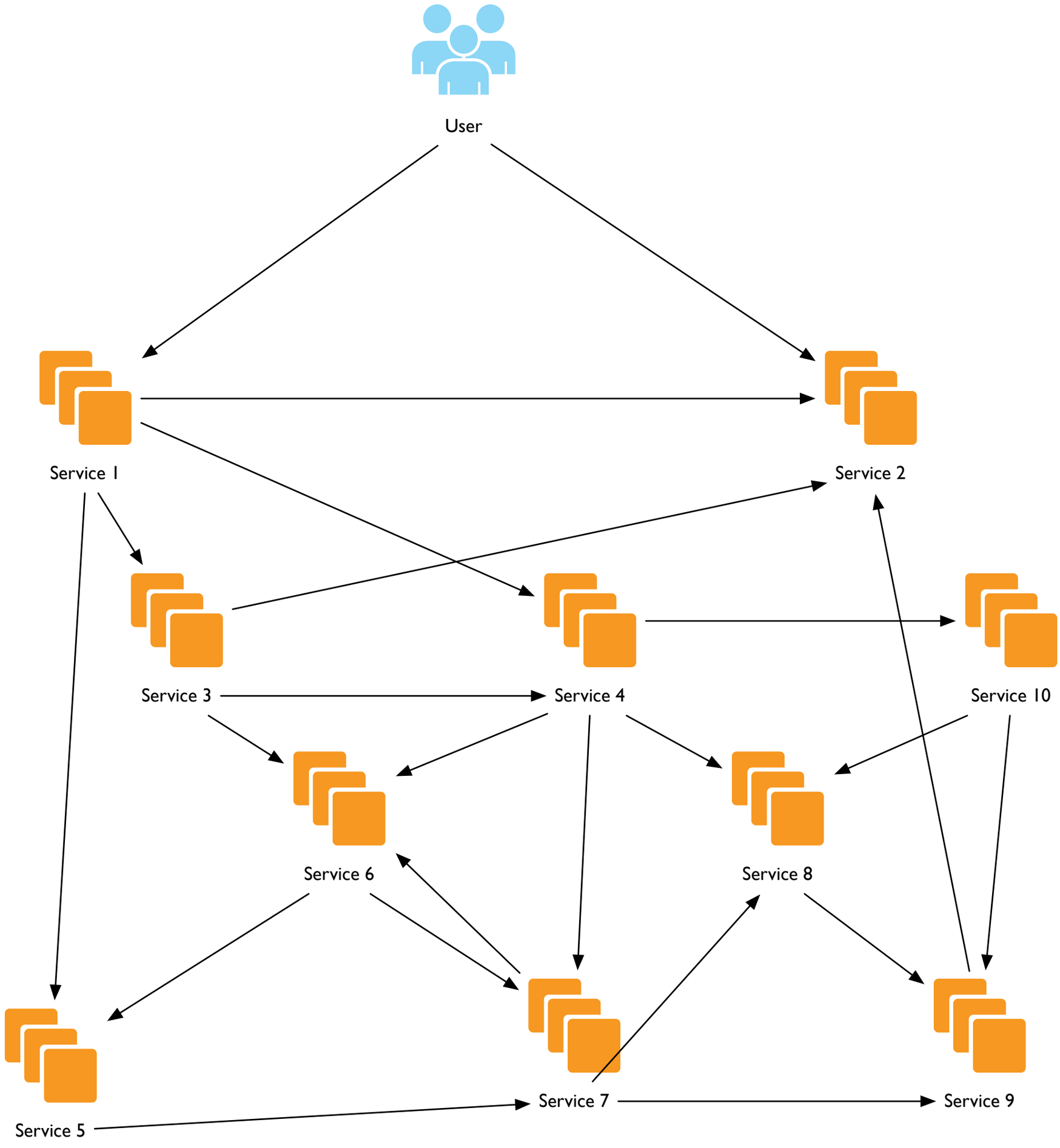
- DevOps @Sina
- Lamp项目托管平台
- 已托管项目500+
- 日均请求10B+

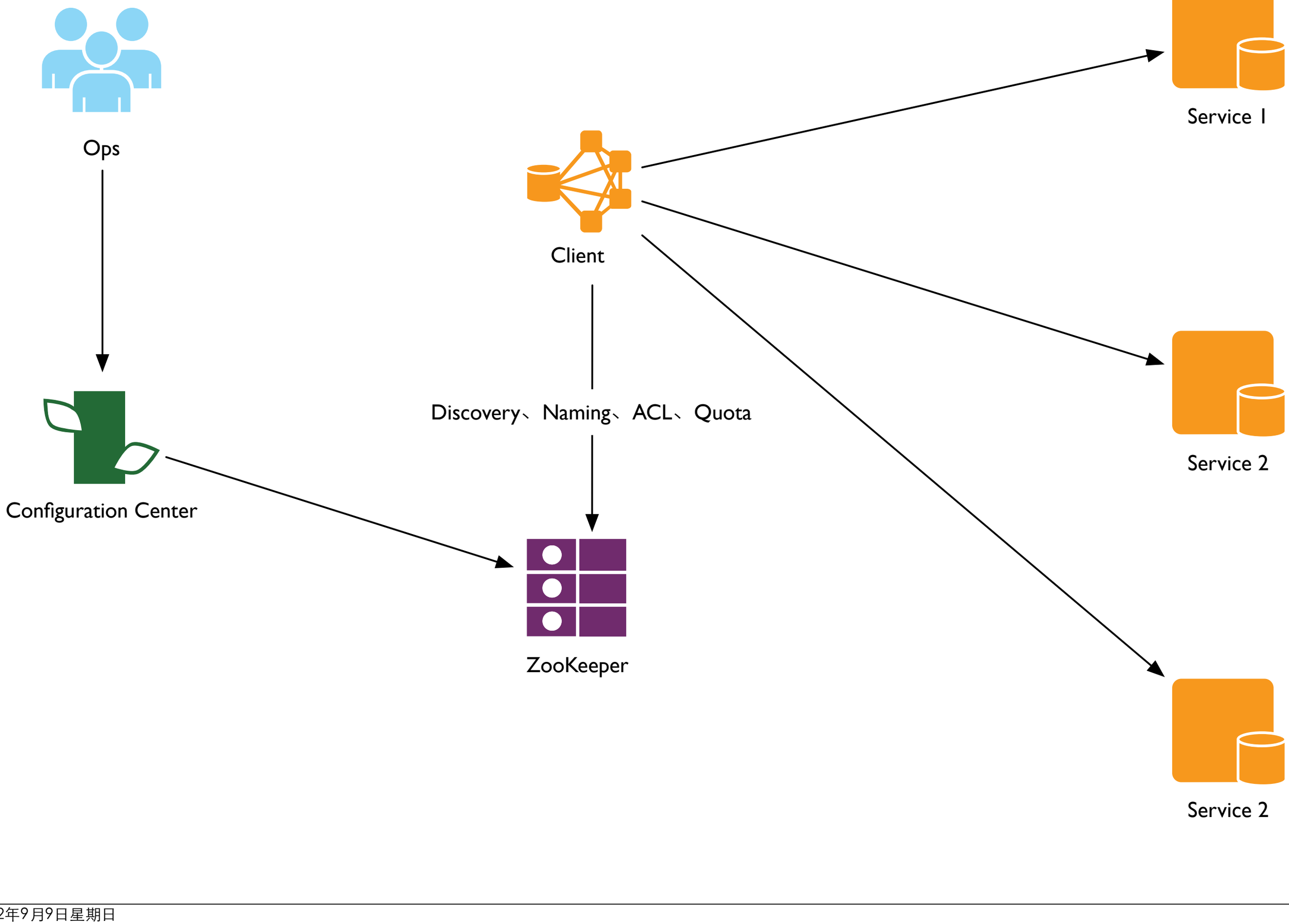
今天要破解哪些硬伤

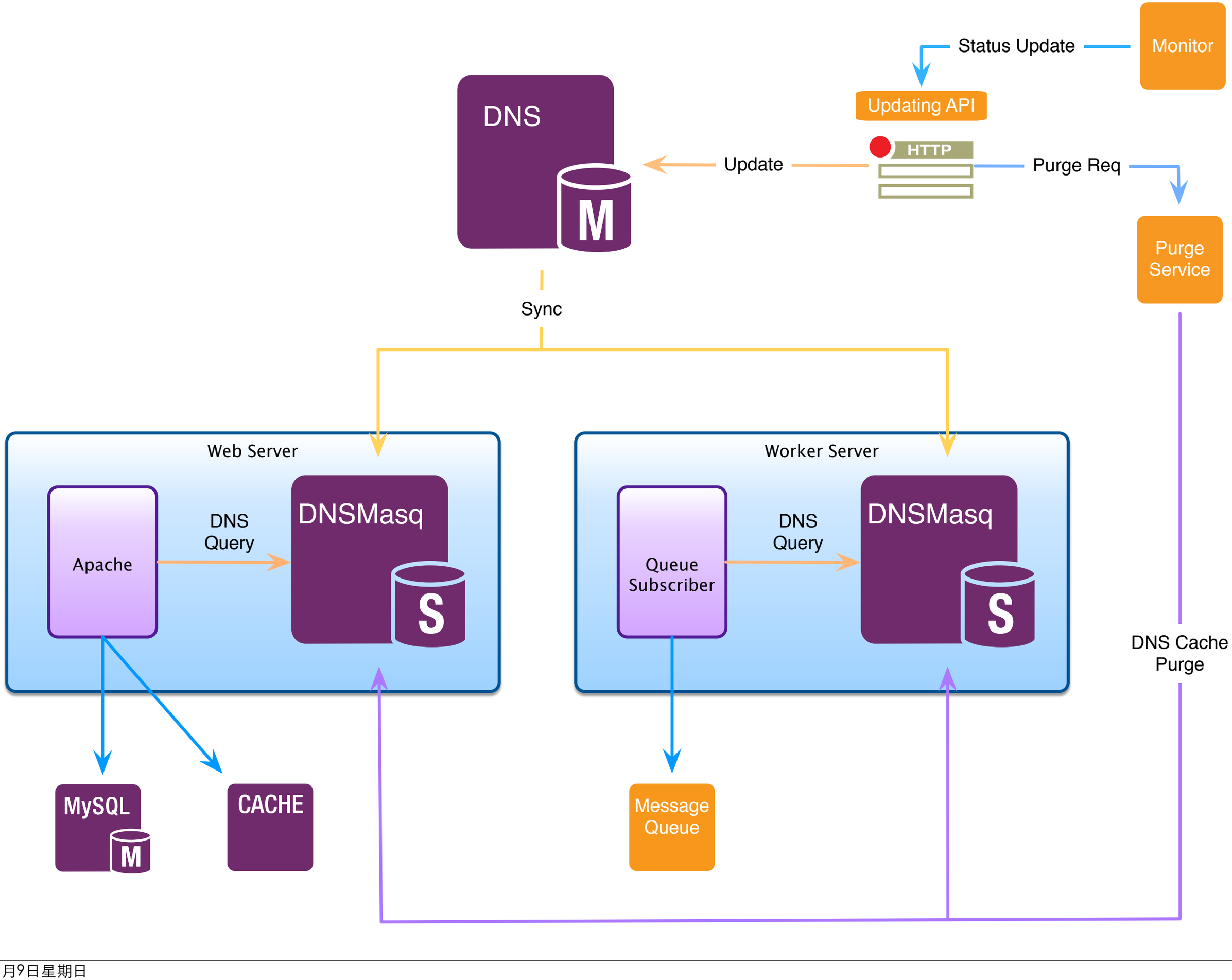
- SOA陷阱
- Memcached“太”好用了
- RDBMS难以扩展
- 困扰我们的“阿喀琉斯之踵”

我们都爱SOA

- 解耦
- 分而治之
- ...太多名词来描述优点和必要性
- 当目标太多的时候，是不是容易变成为了做而做?


User
Service 1
Service 2
Service 3
Service 4
Service 10
Service 6
Service 8
Service 5
Service 7
Service 9

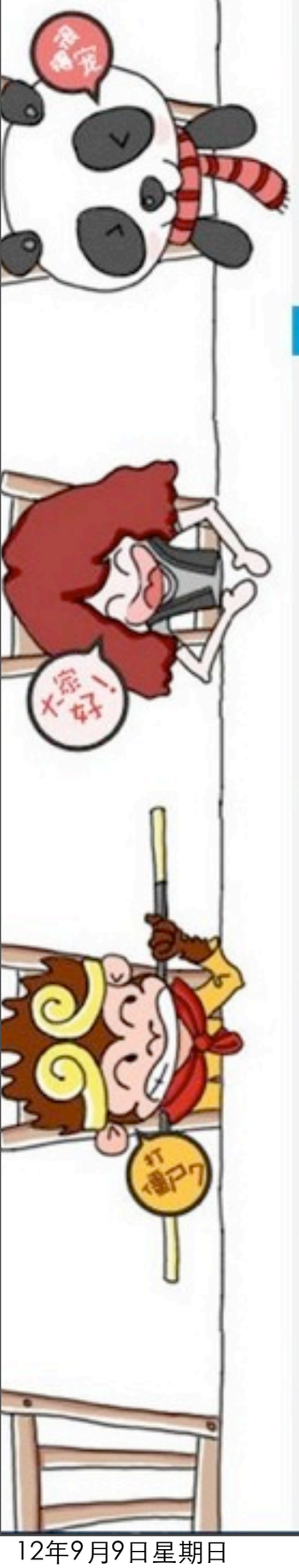

Ops
Configuration Center
Client
Discovery、Naming、ACL、Quota
ZooKeeper
Service 1
Service 2
Service 2


Monitor
Status Update
Updating API
DNS
HTTP
Update
Purge Req
Purge Service
Sync
Web Server
Worker Server
Apache
DNS Query
DNSMasq
S
Queue Subscriber
DNS Query
DNSMasq
S
DNS Cache Purge
MySQL
M
CACHE
Message Queue

通用命名服务--*Macedonia*

- C、Python、Node.JS、PowerDNS、DNSMasq、MongoDB
- DNS协议，更新时通知所有客户端，缓存一致性强
- 多样化API
- 无监控框架，需结合API设计开发
- A记录无法返回端口号

Memcached...Wow!
Memcached...OMG!



重度应用Memcached带来的挑战

- 我的首页最多要取50+1+4+2个用户的信息
- 我的首页最少要取50X2+3个计数器

- 为了访问速度大量使用并依赖Memcached
- 如果有一台Memcached服务器宕机的话...

Legend

Server

Process

Cluster

Extension

Web Front-End

Apache

php-meta-ext

php-mc-ext

metad
zk local cache

Updating
Notification

Memcached Command

Zookeeper Cluster

Node Register
&
Heartbeat

Cache Node

Watcher

MC

MC

MC

Cache Node

Watcher

MC

MC

MC

分布式缓存服务--*Babylon*

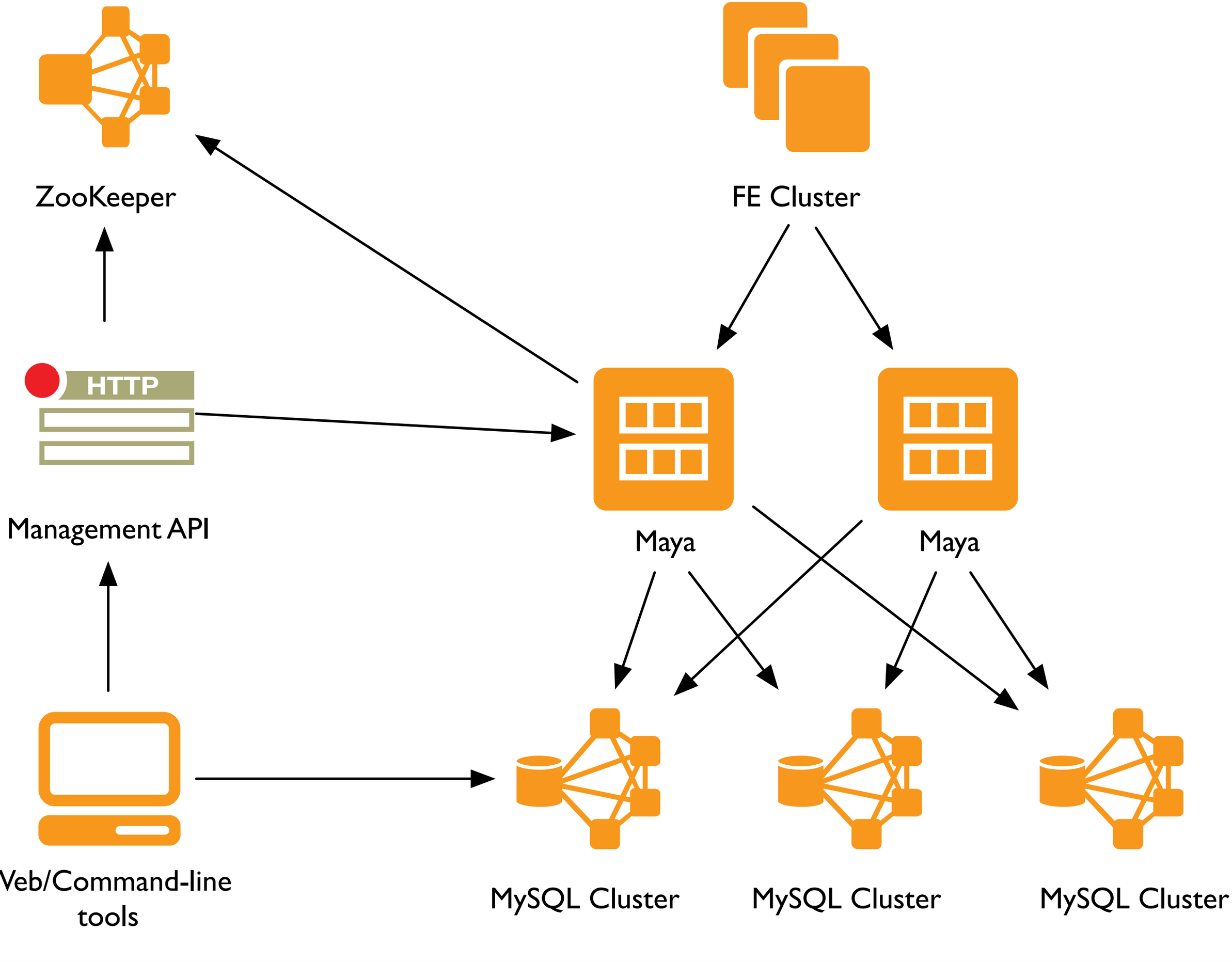
- Thrift、C、C++、Python、ZooKeeper
- 分片信息保存在ZooKeeper中，并在Metad缓存
- 支持引擎级主-主复制
- 连接数LRU（Patch from SAE）
- Memcached-Like SDK

分布式缓存服务--*Babylon*

- 未来特性：
  - 跨IDC复制（开发中）
  - 数据迁移和无缝扩容（也许）
  - FlashCached（也许）
  - 更好的管理性

DBA的烦恼

- MySQL很棒，但是...
- Range VS Hash
- 开发改造成本
- 不合“规范”的SQL
- Master HA


ZooKeeper
FE Cluster
HTTP
Management API
Maya
Maya
Web/Command-line tools
MySQL Cluster
MySQL Cluster
MySQL Cluster

数据库中间层--*Maya*

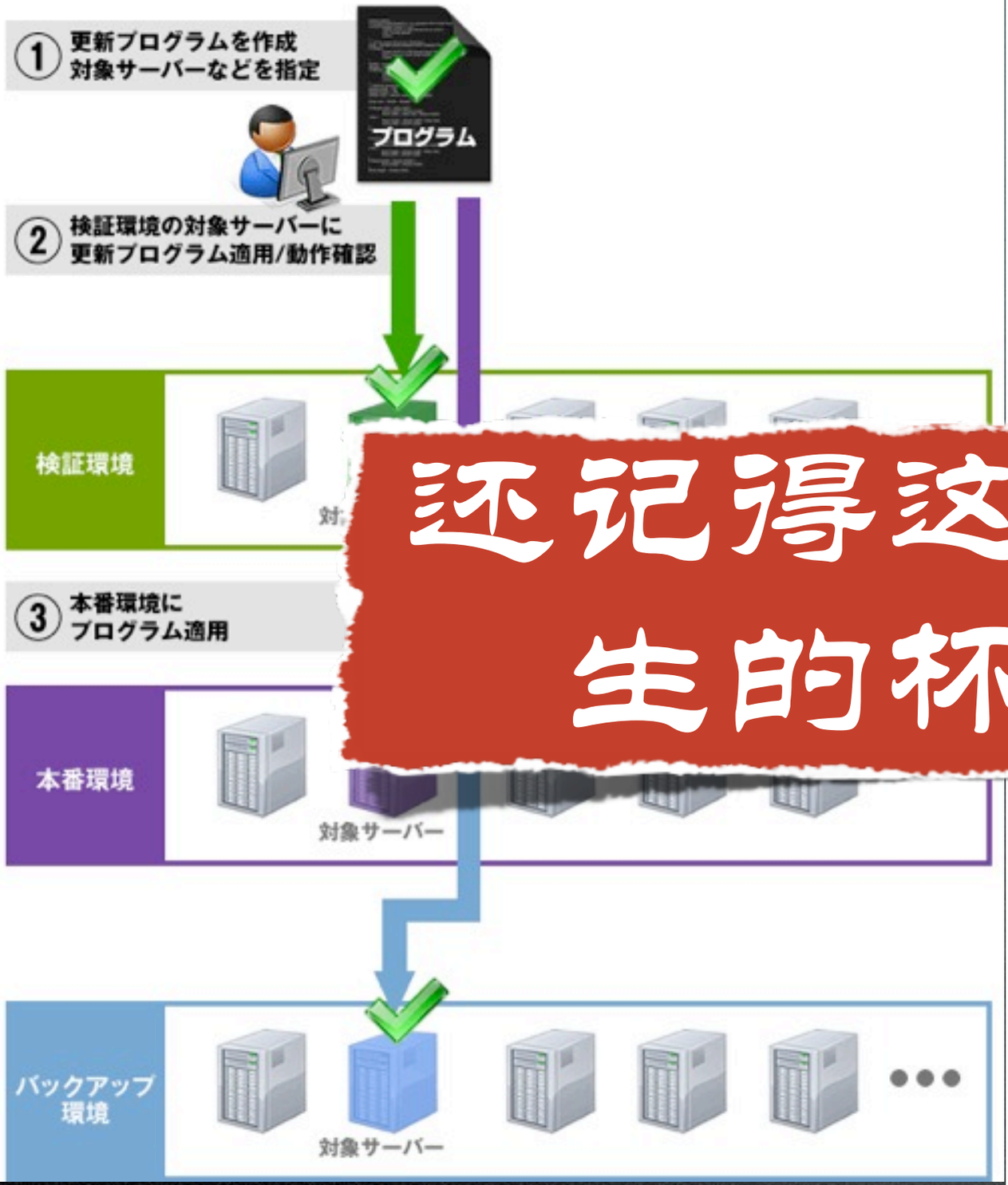
- Node.JS、C、Flex & Bison、Python、
- SQL路由
- 流量统计、配额、限制
- 数据库预拆分（不是自动拆分）
- 主库半自动切换、从库自动切换
- SQL审计、自动DDL

我们的
“阿喀琉斯之踵”

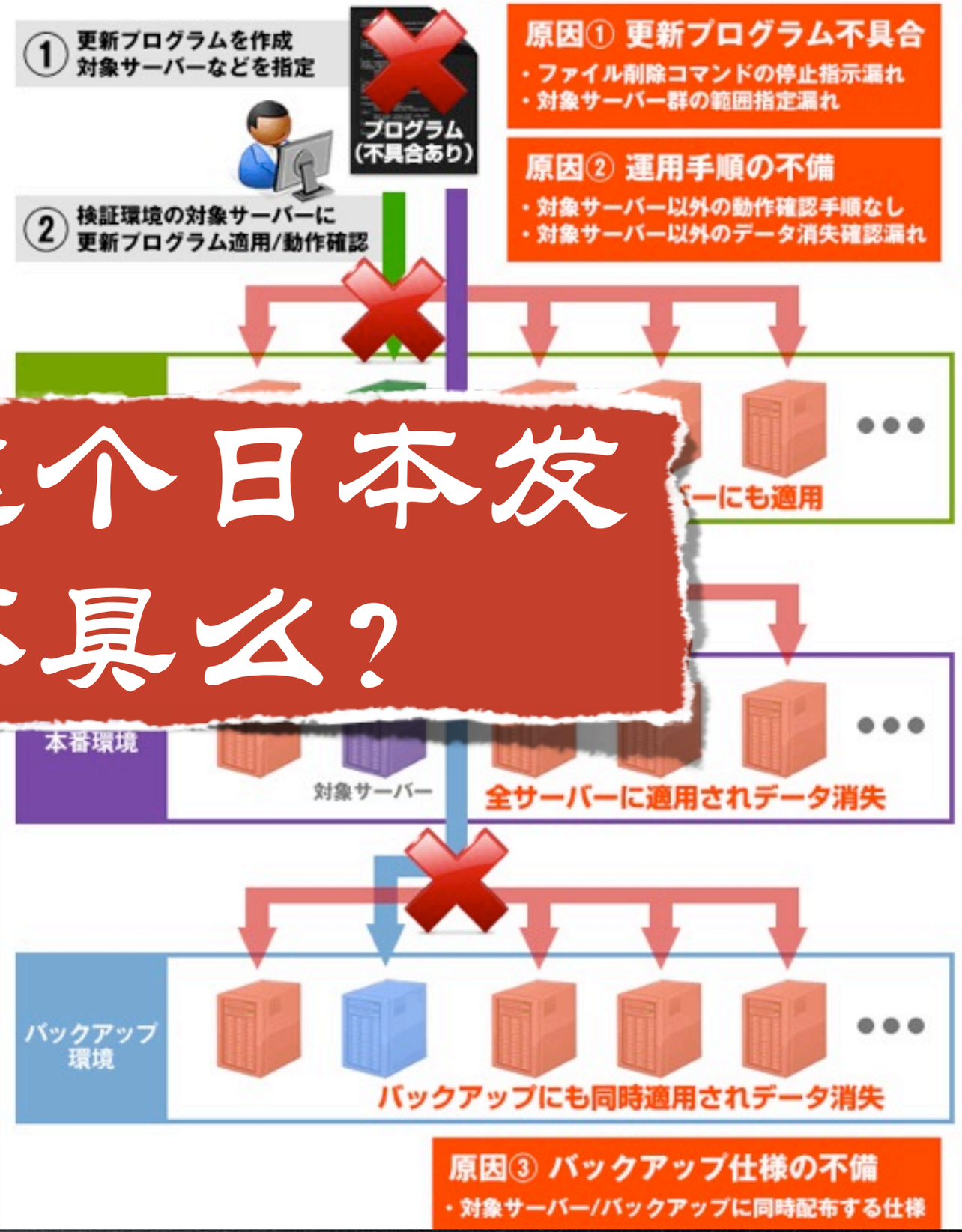
- 每年新增PHP项目100+
- 3000+台前端服务器 100亿+ Hits/天
- 45000+行虚拟主机配置 30G代码（GZip后）
- 3000+台数据库服务器 200亿+ 数据库请求/天
- 6000+ 个MySQL实例 总存储量2P+

SACC

通常の状態



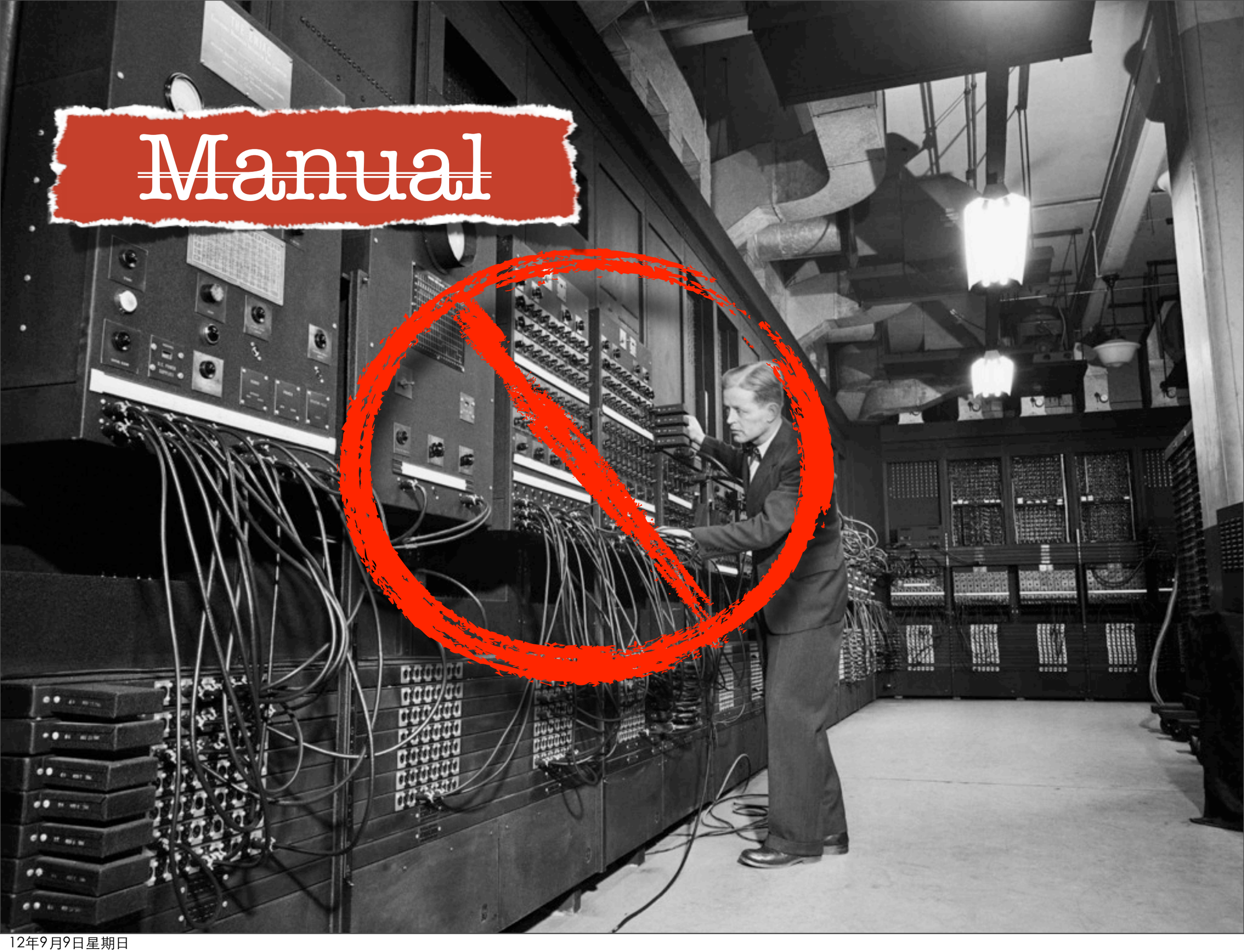
今回の事故の原因



还记得这个日本发生的杯具么？


OPERATOR!
Western Electric
MICROPHONIC
Make
Disasters
PRODUCED BY
AUDIO PRODUCTIONS, INC.

Manual



自动化运维萌芽

- 配置与部署越统一越容易自动化
- 将服务器按照功能划分角色
- 为每一个角色指定相应的配置文件
- 角色和配置文件的分类尽量简洁
- 差异配置利用程序来自动化处理

httpd-vhost.conf_tmpl

```
...

SetEnv SINASRV_ZONE_IDC @@XD@@
SetEnv SINASRV_ZONE_ISP @@CNC@@
SetEnv SINASRV_ZONE_ID  @@010201@@


...
```

vhost_config.pl

```
...

%idc_rule = (
                zjm => {
                        SINASRV_ZONE_IDC => "ZJM",
                        SINASRV_ZONE_ISP => "CNC",
                        SINASRV_ZONE_ID => "010101",
                       }
                )
...

   elsif ($line =~ m!^\s*SetEnv\s+SINASRV_ZONE_IDC.+!) {
        push @vhost_configfile, "SetEnv SINASRV_ZONE_IDC ". $idc_rule{$idc}{SINASRV_ZONE_IDC};
        next;
     } elsif ($line =~ m!^\s*SetEnv\s+SINASRV_ZONE_ISP.+!) {
        push @vhost_configfile, "SetEnv SINASRV_ZONE_ISP ". $idc_rule{$idc}{SINASRV_ZONE_ISP};
        next;
     } elsif ($line =~ m!^\s*SetEnv\s+SINASRV_ZONE_ID.+!) {
        push @vhost_configfile, "SetEnv SINASRV_ZONE_ID ". $idc_rule{$idc}{SINASRV_ZONE_ID};
        next;
     }
   )

...
```

FE Node

```
/etc/httpd-vhost.conf
```

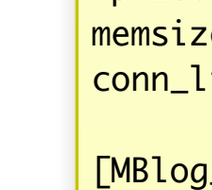
对于管理性低的开源软件

- 那就创造一个配置文件吧
- 当然要尽量创造一个全局统一的
- 索性与监控程序结合起来怎么样?
- INI VS YAML
- 就比如Memcached!

mc.conf

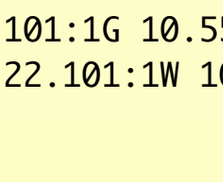
```
[MBlog_Userinfo]
ip=10.55.22.100:10000 10.55.22.101:10000 10.55.22.102:10000 10.55.22.103:10000
memsize=10.55.22.100:1G 10.55.22.101:1G 10.55.22.102:1G 10.55.22.103:1G
conn_limit=10.55.22.100:1W 10.55.22.101:1W 10.55.22.102:1W 10.55.22.103:1W

[MBlog_Counter]
ip=10.55.22.100:10001 10.55.22.101:10001 10.55.22.102:10001 10.55.22.103:10001
memsize=10.55.22.100:1G 10.55.22.101:1G 10.55.22.102:1G 10.55.22.103:1G
conn_limit=10.55.22.100:1W 10.55.22.101:1W 10.55.22.102:1W 10.55.22.103:1W
```



MC Node

```
/etc/init.d/memcached
/etc/cron.d/check_mc
```



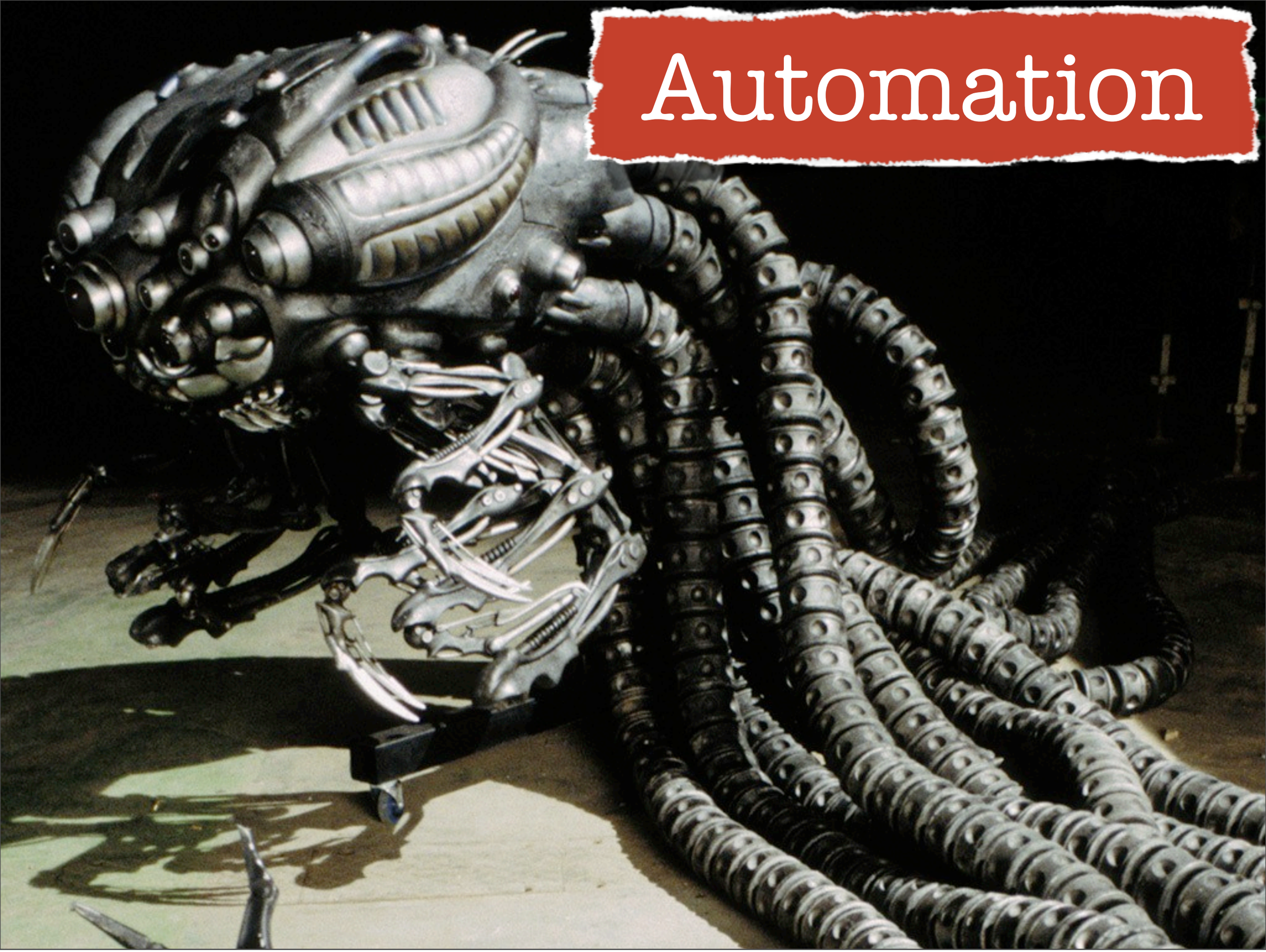
Monitor Node

```
/etc/monitor.d/check_mc.py
```


Reporting Node

```
/etc/reporting.d/get_mc_stats.py
```

Automation



12年9月9日星期日

动态平台管理中心

» 编辑配置文件 - wbtest-httpd-vhost.conf

名　　称: wbtest-httpd-vhost.conf (该名称是为了方便查看，与实际部署无关) wbtest-httpd-vhost.conf

目标路径: /etc/dAppCluster/httpd-v (绝对路径，包括文件名，指在目标机器上的存放路径) /etc/dAppCluster/httpd-vhost.conf_dpool2

文件权限: 0644 (默认"0644")　　文件所有者: root:root (默认"root:root")

关联模块： 添加一行 添加DB组模块 添加DPOOL组模块

模块0： 微博外包前端　　类型： 全部

模块1： 微博外包管理机　　类型： 全部

模块2： 外包微博测试池前端　　类型： 全部

配置文件下发前要执行的检查命令(可选，用于执行配置检查，如果该命令未能正确执行(即$?不为0)，则拒绝替换。配置文件地址请用${conf_path}代替):

配置文件下发后要执行的命令(可选，建议有的操作：ps查看进程，kill -HUP重启服务，tail查看访问/错误日志):
/etc/dAppCluster/vhost_config_unite.pl

配置文件内容

```
MaxClients @@800@@

<IfModule vhost_limit_module>
   MaxVhostClients @@200@@
</IfModule>

SetEnv SINASRV_GLOBAL_MEMCACHED_SERVERS "@@10.44.6.44:7601 10.44.6.45:7601 10.44.6.46:7601 10.44.6.47:7601 10.44.6.48:7601 10.44.6.49:7601 10.44.6.50:7601
SetEnv SINASRV_MEMCACHED_SERVERS "@@10.44.6.232:7601 10.44.6.234:7601 10.44.6.238:7601 10.44.6.240:7601@@"
SetEnv SINASRV_MEMCACHED_HOST 127.0.0.1
SetEnv SINASRV_MEMCACHED_PORT 7600
SetEnv SINASRV_DATA_DIST_SERVER  @@10.44.6.43@@
SetEnv SINASRV_DATA_DIST_PORT 8080

Header add DPOOL_HEADER "Hello World!"

SetEnv SINASRV_ZONE_IDC @@XD@@
SetEnv SINASRV_ZONE_ISP @@CNC@@
SetEnv SINASRV_ZONE_ID  @@010201@@
SetEnv SINASRV_ROLE web
SetEnv SINASRV_OUTIP outip
```

和最新线上版本的配置差异

```
--- /var/tmp/httpd-vhost.conf_dpool2.old.tmp    2012-05-24 18:00:02.000000000 +0800
+++ /var/tmp/httpd-vhost.conf_dpool2.new.tmp    2012-05-24 18:00:02.000000000 +0800
@@ -205,6 +205,7 @@
         SetEnv SINASRV_NDATA_CACHE_URL "http://cache.@@mars@@.weibo.com/nd/photoweibo/"

         SetEnv SINASRV_MEMCACHED_ALBUMS_SERVERS "10.75.31.39:7122 10.75.31.38:7122 10.75.31.37:7122 10.75.31.36:7122 10.77.7.85:7122 10.77.7.86:712
+        SetEnv SINASRV_MEMCACHED_NEW_ALBUMS_SERVERS "10.75.31.40:7122 10.75.31.39:7122 10.75.31.38:7122 10.75.31.37:7122 10.75.31.36:7122 10.77.7.8



@@ -328,6 +329,7 @@
```

动态平台管理中心

线上变更管理
- 工作流管理
- 工作流日志
- 工作流执行日志
- 配置文件管理
- 包部署管理
- WebShell
- 用户临时授权

运维信息管理
- 产品单元管理
- 环境设置
- 模块设置
- IDC设置
- 负载均衡

监控报警管理
- 性能监控
- 报警管理
- 基础监控模板设置

隐藏左栏

资源资产管理
- 节点管理
- DB资产管理

数据库应用
- 端口管理
- 域名管理
- 备份管理

实验特性专区
- Cron管理

系统管理专区
- 管理日志
- 账户设置
- 修改资料
- 注销登录

» 节点管理(默认只显示30条数据，请善用"查找"功能)

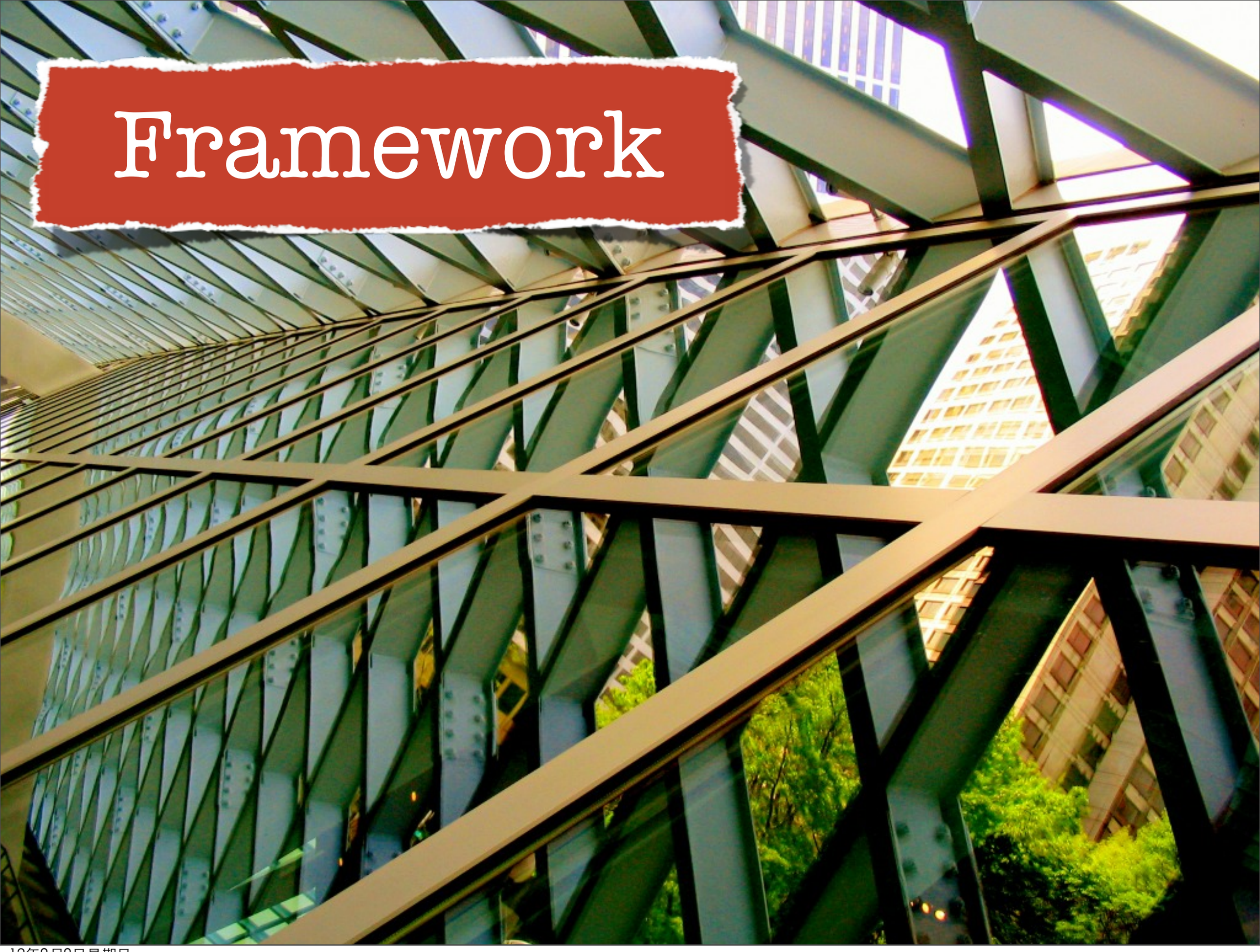
环境： 所有环境 模块： 数据库从库 IDC： 北京土城 关键字： IP 查找 共166条结果

状态及操作	内网IP	外网IP	环境	初始化状态	模块			
					模块操作	模块名	端口	产品单元
					[初始化]	Fusion_io(fusion_io)		------
					[初始化]	硬件RAID(hwraid)		------
					查看端口 待监测 扩容 [初始化]	MytriggerQ MySQL 从库(mysql-innodb)	3618	数据库平台/微博客/基础服务/myfeed
					查看端口 待监测 扩容 [初始化]	数据库从库(mysql-innodb)	3618	数据库平台/微博客/基础服务/myfeed
					查看端口 待监测 扩容 [撤销初始化状态]	mytriggerMySQL从库(t)	3618	数据库平台/微博客/基础服务/myfeed
☐ ✓ 操作	10.73.11.119	10.72.11.119	线上	✓				
[编辑] [下线] [批注] [删除] [收集硬件信息] [下发配置] [撤销初始化状态] [监控设置] [发起报修]								
					查看端口 待监测 扩容 [初始化]	主模块:数据库从库(mysql-innodb)	4624	数据库平台/微博客/基础服务/关注
☐ ✓ 操作	10.73.25.86	10.72.25.86	线上	✓	[初始化]	SSD群集(ssd)		------
					[初始化]	硬件RAID(hwraid)		------
					查看端口 待监测	主模块:数据库	462	数据库平台/微博客/

足够自动化了吗？

- 自动化了很多个点
- 但是运维是一个体系
- 但当我们尝试把点联起来的时候
- 我们发现不是点不够多
- 就是重复发明了轮子

Framework



我们发现问题在出发点

- 自动化运维不是“另一个”系统
- 它是对你架构可运维性的更高要求

- 需要将可运维性作为软件和架构设计的主要考量因素之一
- 运维工具也是软件和架构的一部分

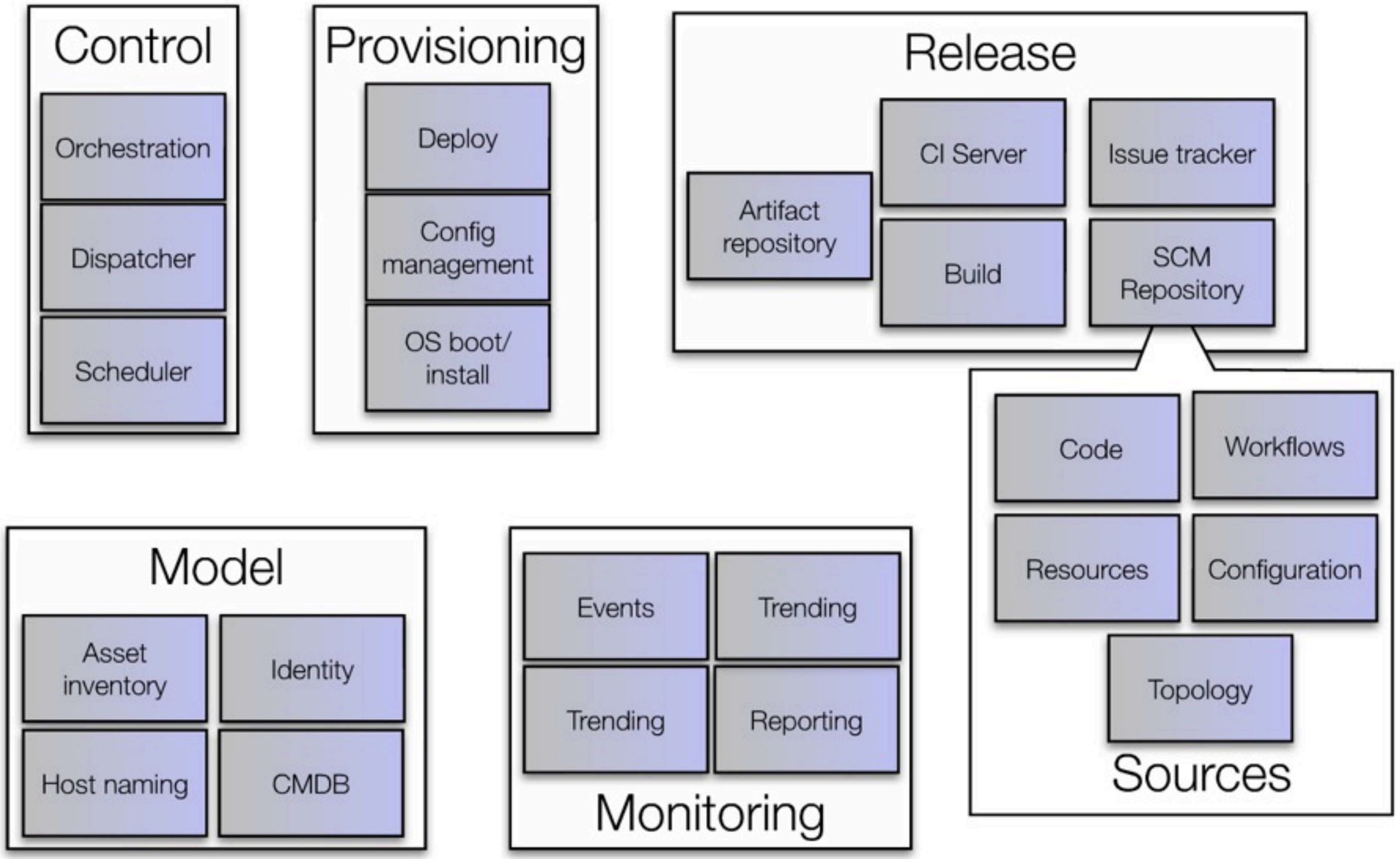
it's

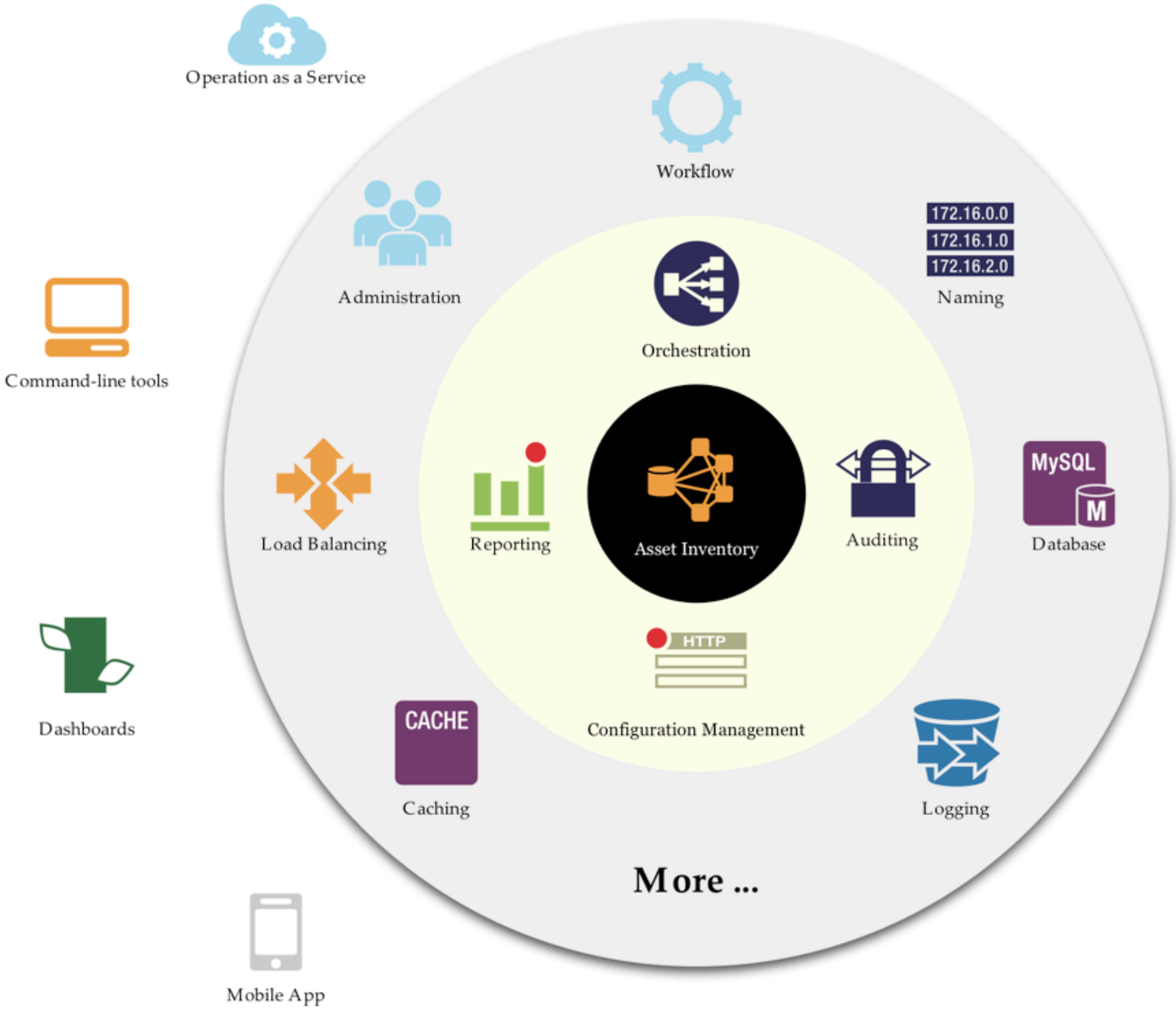
Revolution **O**f **M**artyred **E**lites

SACC

ROME

Generalized architecture




Operation as a Service
Workflow
Administration
172.16.0.0
172.16.1.0
172.16.2.0
Naming
Command-line tools
Orchestration
Load Balancing
Reporting
Asset Inventory
Auditing
MySQL
M
Database
Dashboards
HTTP
Configuration Management
CACHE
Caching
Logging
More ...
Mobile App

no chocolate without cacao

the web is... integration without coupling

performance through scalability through uniformity

you might not need that!

is it restful?

```
POST /REST.svc

<PlaceOrder>
  <Product> 123 </Product>
  <Price> 19.90 </Price>
  <Qty> 2 </Qty>
</PlaceOrder>
```

```
POST /REST.svc

{
  PlaceOrder: {
    Product: 123,
    Price: 19.90,
    Qty: 2
  }
}
```

Level 3: hypermedia
Level 2: http
Level 1: uri
Level 0
Richardson REST maturity model

303 see other

Tale of the ice-cream maker by Sebastien Lambla

thank you!

REST

APIs

REST in Practice
by Jim Weber, Savas Parastatidis, Ian Robinson

@getsomerestbook on twitter
by Steve Klabnik

REST: you're doing it wrong
Sergey Shishkin
http://shishkin.github.com/presentations/rest
@sshishkin

is it restful?

```
/products/books/can-haz-rest

/products?title=can-haz-rest

/Y2FuLWhhei1yZXN0%3D

/rest.svc?op=Order&
  product=123&qty=2
```

uri's are opaque

what about uri templates?

```
/user/{id}
```

rpc & uri tunneling

make them discoverable

uri identifies resource, not representation

```
GET /product/123

Accept:
  application/vnd.example+xml;q=1,
  application/vnd.example+xml,
  application/xml
Accept-Language: de, fr, en
```

```
200 OK

Content-Type:
  application/vnd.example+xml;q=1
Language: de-DE

<body/>
```

hypermedia representations

resources

UI is already restful

hypermedia is links & forms

```
/product/123

...
<a rel="product-order"
  href="/product/123/order">
    Order
</a>
...
```

```
/product/123/order

<form action="/orders"
  method="POST">
  <input type="hidden"
    name="product" value="123" />
  <input type="text"
    name="quantity" value="1" />
...
```

design api with resources not http verbs

beware of CRUD

GET
idempotent, safe

POST
non-idempotent, unsafe

PUT
idempotent, unsafe

DELETE
idempotent, unsafe

embrace http

get un-applied changes

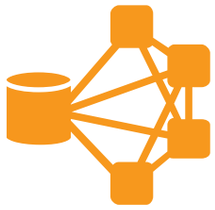
1

```
readRole(
  criteria={ 'applied': false },
  fields={ 'included': true }
)
```

3 create orchestration job

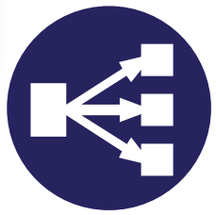
```
createOrchestration(
  targets=[
    '10.71.1.2',
    '10.71.1.3'
  ],
  commands=[
    'update vhost.conf',
    'restart httpd'
  ]
)
```

2 retrieve the related info



Asset Inventory

Configuration Management



Orchestration

```
readAsset(
  'criteria'={ 'cmdb': 'D2012887' },
  'fields'={ 'addresses.internal': true }
)
```

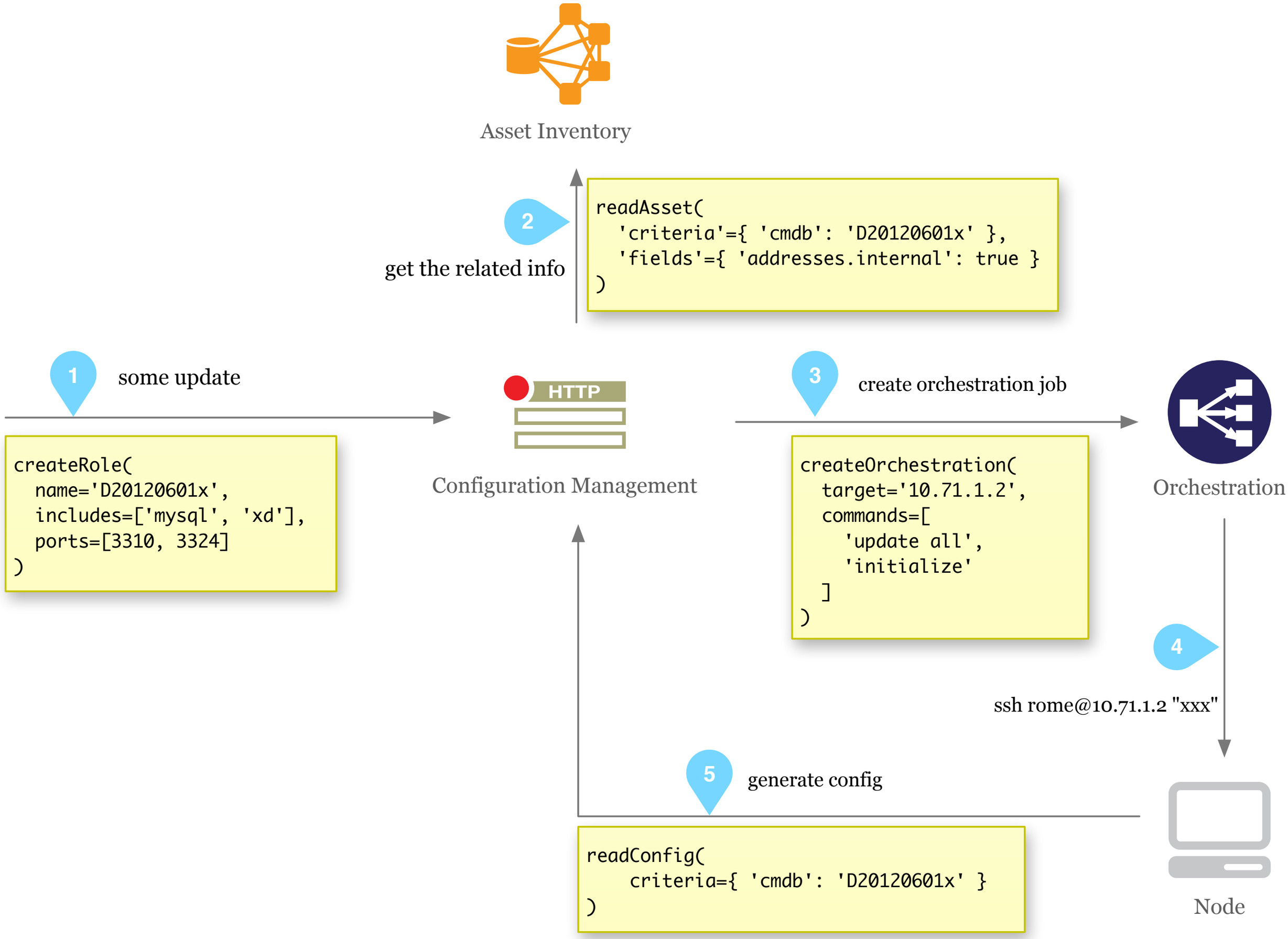
4

ssh rome@10.71.1.2 "xxx"

5 generate config

```
readConfig(
    criteria={ 'cmdb': 'D2012887' }
    files=['vhost.conf']
)
```

Node


Asset Inventory
2
get the related info
readAsset(
'criteria'={ 'cmdb': 'D20120601x' },
'fields'={ 'addresses.internal': true }
)
1
some update
createRole(
name='D20120601x',
includes=['mysql', 'xd'],
ports=[3310, 3324]
)
HTTP
Configuration Management
3
create orchestration job
createOrchestration(
target='10.71.1.2',
commands=[
'update all',
'initialize'
]
)
Orchestration
4
ssh rome@10.71.1.2 "xxx"
5
generate config
readConfig(
criteria={ 'cmdb': 'D20120601x' }
)
Node

我们用了什么技术？

- HTTP-based RESTful API with JSON payload
- A variety of MongoDB-like CRUD
- Node.JS
- Connect & Express
- MongoDB

我们如何开发？

- Ops->DevOps
- Scrum
- BDD
- CI
- 工具大部分来自于Atlassian(比如Jira)

谢谢

Q&A