QCon 全球软件开发大会
INTERNATIONAL SOFTWARE DEVELOPMENT CONFERENCE

BEIJING 2018

初创公司
通用可扩展自动化上线平台快速落地

58速运-沈剑

主办方 Geekbang. 极客邦科技 InfoQ

关于-我

- **“架构师之路”作者**，深夜写写技术文章
- 百度 - 高级工程师
- 58同城 - 高级架构师，技术委员会主席，技术学院优秀讲师
- 58到家 - 高级技术总监，技术委员会主席
- 58速运 - CTO
- 本质：技术人一枚

目录

一、缘起

- 问题的提出：如何把新系统发布到线上？
- 幸福常常千差万别，痛苦总是似曾相识

不同阶段的公司，玩法不同

- 原始阶段：**人肉型**
  - 手工执行
  - 研发人员操作
  - 单机，若是集群逐台操作
  - 随意操纵数据库
  - 每次上线要通宵
  - …
- 进阶阶段：**自动化**
  - 脚本执行
  - 运维人员操作
  - 集群，可批量
  - 研发只有数据库只读权限
  - 每次上线小时级别
  - …
- 高级阶段：**平台化**
  - 平台点按钮，“傻瓜化”
  - 测试人员操作
  - 集群，可批量
  - 专业的DBA操作数据库
  - 每次上线分钟级别
  - …

二、创业型公司玩自动化运维潜在的困难

- **需求多**：集群初始化，上线前打包，系统发布
- **语言不同**：C/C++，PHP，Java…
- 同一种语言，**类型不同**：web，service…
- 同一种类型，**框架不同**：dubbo，thrift，DSF…
- 开源软件满足一部分需求，**难以定制化**
- 创业型公司，没有人和时间，**难以像大公司一样长期投入**开发大平台

自研上线平台，如何得到老板的支持？

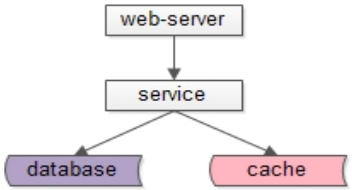
- **解决大部分需求**
- 支持多语言上线
- 支持多类型上线：web，service…
- 支持多框架上线
- **可以定制化**
- **能够快速落地**

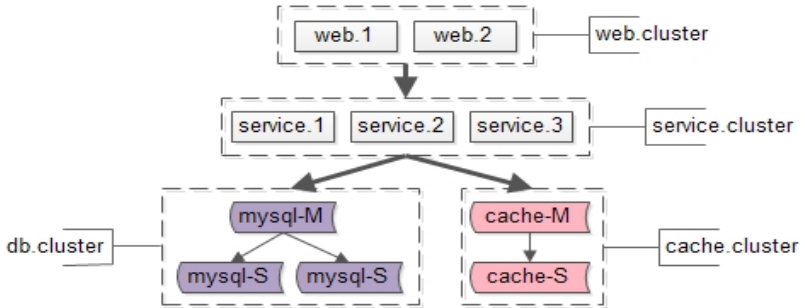
三、58到家自动化上线快速实践

抽象集群

什么是集群

- 什么是集群
- 集群是保证高可用的基础





集群属性（信息）

- 例如，一个“用户服务”集群有哪些信息？
  - **集群名称**：user.service
  - **IP列表**：ip1, ip2, ip3
  - **二进制目录**：/user.service/bin/
  - **配置目录**：/user.service/conf/
  - **日志目录**：/user.service/log/
  - **负责人列表**：shenjian, zhangsan
  - …

- 集群信息有什么用？
  - 自动化日志清理
  - 自动化备份
  - 自动化监控
  - 上下游调用（用得最多）
  - 自动化发布（今天要讲的）

集群信息用处-上下游调用

- web-X上调用service会用到哪些信息？
- **service.name** : user.service
- **service.ip.list** : ip1, ip2, ip3
- **service.port** : 8080
- 这些信息保存在哪里？
- web-X.config

- 上下游调用过程是什么？
- web-X启动
- web-X读取服务集群的IP列表与端口
- web-X初始化user服务连接池
- web-X拿取连接，通过RPC接口调用

集群信息用处-自动化二进制备份

- 自动化备份会用到哪些信息？
- **name** : user.service
- **ip.list** : ip1, ip2, ip3
- **bin.path** : /user.service/bin/
- 这些信息保存在哪里？
- backup.user.service.config

- 自动化备份过程是什么？
- 依次轮询各个ip
- 建立相关备份目录
- 把二进制复制到备份目录下

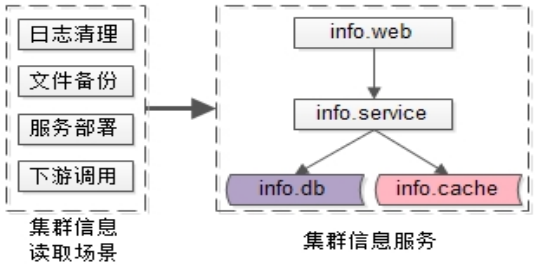
集群信息维护

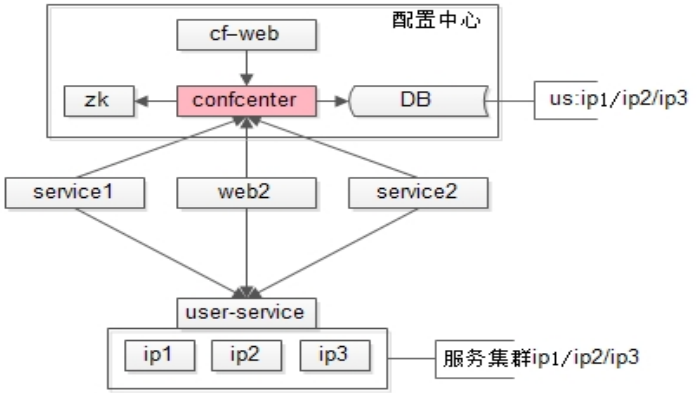
- 分散式
  - 不同的应用场景，使用的集群信息均不同
  - 集群信息都写在各自“配置文件”里
  - 集群信息冗余了很多份
- 存在的问题
  - 集群信息修改，多个地方要改（耦合）
  - 人员流动，配置在哪里就忘了
  - 时间推移，配置旧改漏了
  - …

- 集中式
  - 统一配置文件
  - 统一配置服务
  - 统一配置中心

集中管理集群配置，是自动化运维的基石

```
1  [user.service]
2  ip.list : ip1, ip2, ip3
3  port : 8080
4  bin.path : /user.service/bin/
5  log.path : /user.service/log/
6  conf.path : /user.service/conf/
7  ftp.path : ftp://192.168.0.1/user.service/user.exe
8  owner.list : shenjian, zhangsan
9
10 [passport.web]
11 ip.list : ip11, ip22, ip33
12 port : 80
13 bin.path : /passport.web/bin/
14 log.path : /passport.web/log/
15 conf.path : /passport.web/conf/
16 ftp.path : ftp://192.168.0.1/passport.web/passport.jar
17 owner.list : shenjian, lisi
18
```

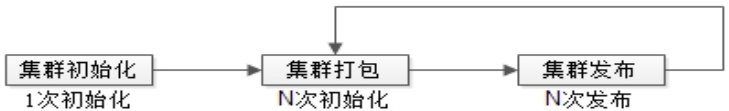




抽象自动化上线过程

过程抽象，每个过程都需要实现自动化

- 回顾一下运维日常操作，不论任何语言、类型、框架
- 首次：**集群初始化**
- 每次：**集群打上线包**
- 每次：**集群发布上线**
- 通用、可扩展源自抽象，如何抽象，才能用自动化上线平台解决上述问题



不同语言，类型，框架

初始化、打包、发布的过程都不一样，怎么办？

集群需要新增一个集群类型（cluster type）属性

```
1  [user.service]
2  cluster.type : Java.service.DSF
3  ip.list : ip1, ip2, ip3
4  port : 8080
5  bin.path : /user.service/bin/
6  log.path : /user.service/log/
7  conf.path : /user.service/conf/
8  ftp.path : ftp://192.168.0.1/user.service/user.exe
9  owner.list : shenjian, zhangsan
10
11 [passport.web]
12 cluster.type : Java.web.SpringMVC
13 ip.list : ip11, ip22, ip33
14 port : 80
15 bin.path : /passport.web/bin/
16 log.path : /passport.web/log/
17 conf.path : /passport.web/conf/
18 ftp.path : ftp://192.168.0.1/passport.web/passport.jar
19 owner.list : shenjian, lisi
20
```

接下来就简单了

不同类型的集群，人肉的操作用脚本实现

初始化、打包、上线

集群自动初始化举例

- Java.web.Spring：部署tomcat，部署相关库，建立规范目录结构…
- Java.service.dubbo：部署容器，部署相关库，建立规范目录结构…
- C++.service.sofa：部署相关库，建立规范目录结构…
- PHP.web.zend：…
- …

规范目录结构非常重要，是自动化运维的基石

- 没有规范

```
[work@test01 x]$ tree
.
├── a.out
├── hello.c
├── hello.conf
├── hello.log
├── hello.log.2018012812
└── tmp
```

- 模块优先

```
[work@test01 module.first]$ tree
.
├── das
│   ├── bin
│   ├── conf
│   └── log
├── entry
│   ├── bin
│   ├── conf
│   └── log
└── logic
    ├── bin
    ├── conf
    └── log
```

- 功能优先

```
[work@test01 func.first]$ tree
.
├── bin
│   ├── das
│   ├── entry
│   └── logic
├── conf
│   ├── das
│   ├── entry
│   └── logic
└── log
    ├── das
    ├── entry
    └── logic
```

日志规范，是**日志**自动化监控/备份/清理的基石

- **日志目录规范**
- **日志分级规范**
  - user.fatal.log
  - user.error.log
  - user.debug.log
- **日志切分规范**
  - daojia.log.2018042114
  - daojia.log.2018042115
  - daojia.log.2018042116
  - daojia.log.2018042117
- **日志格式规范**

集群构建+打包自动化举例

- PHP：copy + tar相关代码
- Java：Maven
- C++：Make
- …

集群发布自动化举例

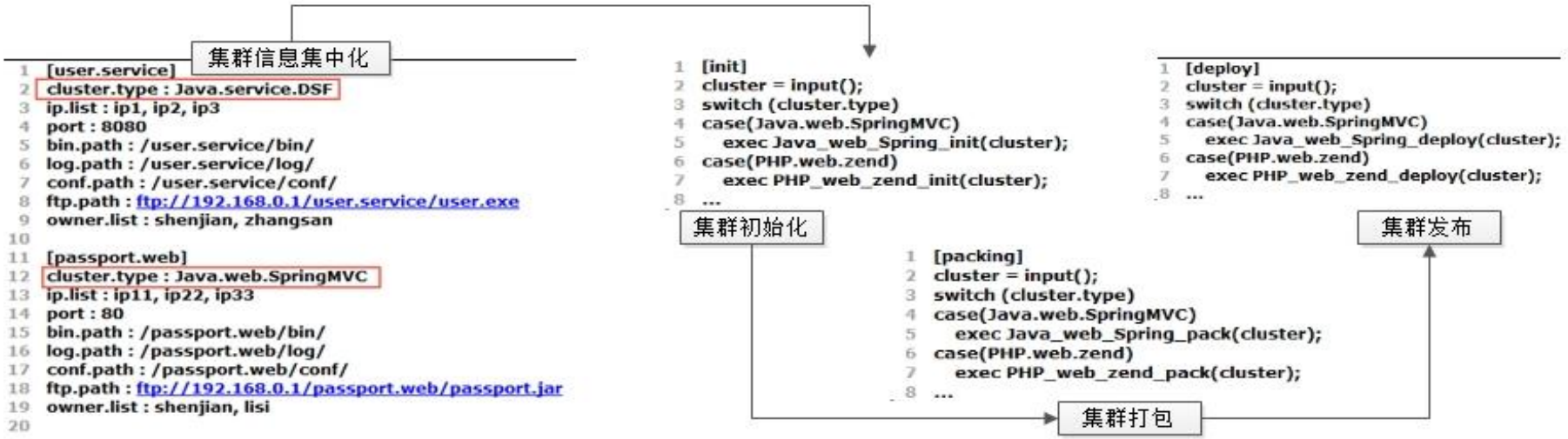
- Java.web.DWF为例：

- 将包从FTP上拉取下来（需要读取集群信息）
- 读取集群IP列表（需要读取集群信息）
- 通知nginx，将一台tomcat的流量切走
- 停止tomcat
- 备份原web，更新新web（需要读取集群信息）
- 重启tomcat
- 通知nginx，将流量切回
- loop

每一个集群类型的初始化，打包，发布步骤

将手动执行的命令脚本化，自动化

再回顾一下整个过程


集群信息集中化
[user.service]
cluster.type : Java.service.DSF
ip.list : ip1, ip2, ip3
port : 8080
bin.path : /user.service/bin/
log.path : /user.service/log/
conf.path : /user.service/conf/
ftp.path : ftp://192.168.0.1/user.service/user.exe
owner.list : shenjian, zhangsan
[passport.web]
cluster.type : Java.web.SpringMVC
ip.list : ip11, ip22, ip33
port : 80
bin.path : /passport.web/bin/
log.path : /passport.web/log/
conf.path : /passport.web/conf/
ftp.path : ftp://192.168.0.1/passport.web/passport.jar
owner.list : shenjian, lisi
[init]
cluster = input();
switch (cluster.type)
case(Java.web.SpringMVC)
exec Java_web_Spring_init(cluster);
case(PHP.web.zend)
exec PHP_web_zend_init(cluster);
...
集群初始化
[packing]
cluster = input();
switch (cluster.type)
case(Java.web.SpringMVC)
exec Java_web_Spring_pack(cluster);
case(PHP.web.zend)
exec PHP_web_zend_pack(cluster);
...
集群打包
[deploy]
cluster = input();
switch (cluster.type)
case(Java.web.SpringMVC)
exec Java_web_Spring_deploy(cluster);
case(PHP.web.zend)
exec PHP_web_zend_deploy(cluster);
...
集群发布

体现的核心架构设计思路

- **中心化管理**：保证数据一致
- **抽象**：提供共性的地方，尽量通用
- **分解**：整体很复杂，分治更简单
- **可扩展**：通过service_type保持个性可扩展，整体流程不用变

Now，终于自动化了

所谓平台化，待未来有空的话...

- 集群信息统一配置，升级为：配置服务，配置中心，配置全部存入数据库
- 手动执行的init.sh pack.sh deploy.sh，升级为：
- 平台点击按钮
- 从数据库读取数据
- 传入脚本自动执行
- 随着公司技术栈的扩展：支持更多的cluster.type
- 支持越来越多的功能...

总结

初创公司，通用，可扩展，快速落地，自动化上线

- 架构设计，讲究抽象
  - 抽象集群
  - 抽象上线过程（初始化，打包，发布）
- 自动化运维最佳实践
  - 集群信息统一维护（配置，服务，中心）
  - 集群类型决定初始化、打包、发布的具体执行
  - 注意人肉工作的脚本化
- 越早规范化，对运维自动化越有利
  - 集群配置规范化，统一化
  - 目录规范化
  - 日志规范化
- 递归分布实现，别想一口吃成胖子
  - 进阶路线：人肉 -> 自动化 -> 平台化
  - 分步路线：初始化 -> 打包 -> 发布

Q&A

“架构师之路”

Thanks!

INTERNATIONAL SOFTWARE DEVELOPMENT CONFERENCE

主办方 Geekbang. 极客邦科技 InfoQ