GOPS2017 Shanghai

GOPS 2017 全球运维大会 上海站

暨首届金牌运维峰会

指导单位：

主办单位：

主办单位：

承办单位：

大会时间：2017年11月17日-18日

大会地点：上海光大会展中心国际大酒店（上海徐汇区漕宝路67号）

GOPS2017
Shanghai

DevOps道法术器及全开源端到端部署流水线

赵班长@DevOps学院

讲师介绍

赵舜东

- 昵称：“赵班长”，速云科技CEO；曾在武警某部负责指挥自动化的架构和运维工作，国内首批Exin DevOps Master授权讲师、中国SaltStack用户组发起人、运维社区创始人、DevOps学院创始人；著有《SaltStack入门与实践》、《运维知识体系》、《缓存知识体系》；2008年退役后一直从事互联网运维工作，历任运维工程师、运维经理、运维架构师、运维总监。现创业专注于企业DevOps运维服务和在线教育。
- DevOps学院：http://www.devopsedu.com/
- Github：https://www.github.com/unixhot/

目录

1 DevOps的道法术器

2 如何构建一个真正的DevOps平台

3 全开源端到端部署流水线

4 全开源全链路自动化运维体系

5 你来问，我来答

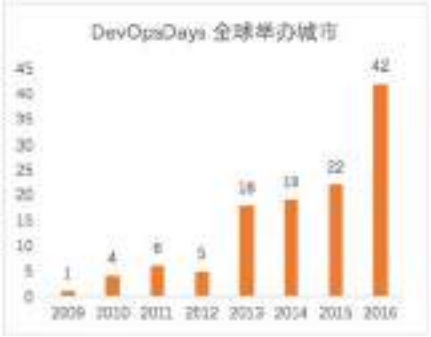
什么是 DevOps？

- DevOps 是 “开发”和“运维”的缩写
- DevOps是一组最佳实践
  - 强调IT专业人员（开发人员，运维人员，支持人员）在应用和服务生命周期中的协作和沟通
  - 强调整个组织的**合作**以及**交付**和**基础设施变更**的自动化，从而实现持续集成、持续部署和持续交付
- DevOps，双态IT的实现之道

并非开发和运维之间的
简单暧昧

DevOps 的缘起

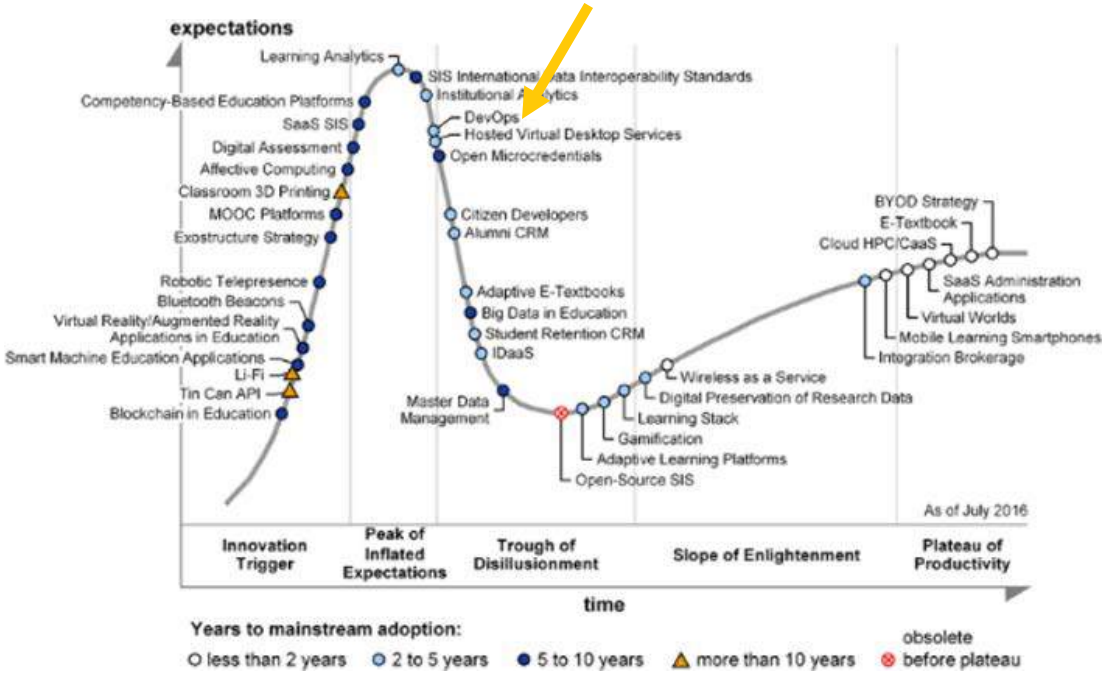
- 2008年，Andrew Clay Shafer 和 Patrick Debois 初次提出 DevOps
- 2009年，Patrick Debois 发起 DevOpsDays 运动
- 2017年，DevOpsDays 北京（318）、上海（818）
- DevOps 是什么？

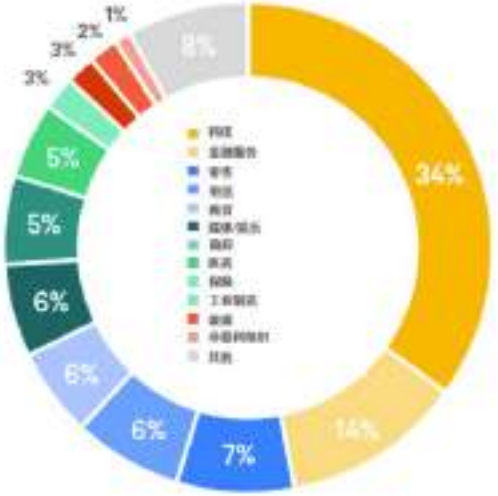



DevOps 之父

DevOps 已成发展趋势

GOPS2017
Shanghai





DevOps团队比例2014年16%，2015年19%，2016年22%，2017年27%

DevOps 钱途如何


DevOps专家
工资居然比开发高出 38%
DevOps ¥407426
¥385673
QA ¥363730
¥338310
Web ¥330657
¥295385
弯道超车！

DevOps 道法术器

GOPS2017
Shanghai

道	• 快速交付价值，灵活响应变化	大道：总括性的原则
法	• 全局打通敏捷开发 & 高效运维	法则：相关标准
术	• 系统应用指导原则、最佳实践	战术：具体的手段
器	• 端到端工具链相互联通和整合	工具：落地的实践

【法】DevOps 标准：研发运营一体化能力成熟度模型

GOPS2017 Shanghai

主管单位：国家级智库 中国信息通信研究院（工信部下属单位，负责可信云等）

发起单位：高效运维社区

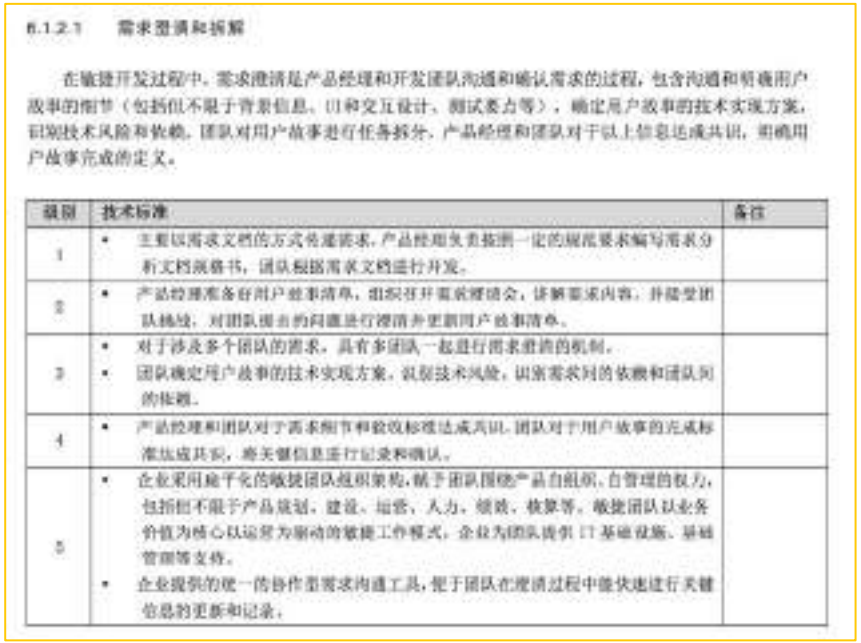
目前起草：腾讯、京东、高效运维社区、浙江移动、华泰证券、中国太保等。

目前排期：2017年11月17日发布第一版

一、研发运营一体化（DevOps）过程															
敏捷开发管理			持续交付								技术运营				
需求管理	计划管理	过程管理	配置管理	构建与持续集成	测试管理	部署与发布管理	环境管理	数据管理	度量与反馈	安全管理与服务	监控服务	数据服务	容量服务	连续性服务	运营反馈
需求收集	需求澄清和拆解	迭代管理	版本控制	构建实践	测试分级策略	部署与发布模式	环境供给方式	测试数据管理	度量指标	安全开发	应用监控	数据收集能力	容量规划能力	高可用规划	业务知识管理
需求分析	故事与任务排期	迭代活动	版本可追踪性	持续集成	代码质量管理	持续部署流水线	环境一致性	数据变更管理	度量驱动改进	运营安全	质量体系管理	数据处理能力	容量平台服务	质量体系管理	项目管理
需求与用例	计划变更	过程可视化及流动			测试自动化					数据安全	事件响应及处置	数据告警能力	运营成本管理		业务连续性管理
需求验收		度量分析								风险与威胁模型	监控平台				运营服务管理
二、研发运营一体化（DevOps）应用架构															
三、研发运营一体化（DevOps）组织结构															

【法】DevOps 标准：研发运营一体化能力成熟度模型

GOPS2017
Shanghai



6.1.2.1 需求澄清和拆解

在敏捷开发过程中，需求澄清是产品经理和开发团队沟通和确认需求的过程，包含沟通和明确用户故事的细节（包括但不限于背景信息、UI和交互设计、测试要点等），确定用户故事的技术实现方案，识别技术风险和依赖，团队对用户故事进行任务拆分，产品经理和团队对于以上信息达成共识，明确用户故事完成的定义。

级别	技术标准	备注
1	• 主要以需求文档的方式传递需求，产品经理负责按照一定的规范要求编写需求分析文档或者书，团队根据需求文档进行开发。	
2	• 产品经理准备好用户故事清单，组织召开需求澄清会，讲解需求内容，并接受团队挑战，对团队提出的问题进行澄清并更新用户故事清单。	
3	• 对于涉及多个团队的需求，具有多团队一起进行需求澄清的机制。 • 团队确定用户故事的技术实现方案，识别技术风险，识别需求间的依赖和团队间的依赖。	
4	• 产品经理和团队对于需求细节和验收标准达成共识，团队对于用户故事的完成标准达成共识，将关键信息进行记录和确认。	
5	• 企业采用扁平化的敏捷团队组织架构，赋予团队围绕产品自组织、自管理的权力，包括但不限于产品规划、建设、运营、人力、预算、核算等。敏捷团队以业务价值为核心以运营为驱动的敏捷工作模式，企业为团队提供 IT 基础设施、基础管理等支持。 • 企业提供的统一协同协作需求沟通工具，便于团队在澄清过程中能快速进行关键信息的更新和记录。	

6.2.4.2 持续部署流水线

持续部署技术既是DevOps的核心实践，通过可重复可靠的流水线，打通每到每份组织交付，实现交付过程中各个环节活动的自动化和可视化，部署流水线通过将复杂的软件交付流程细分为多个阶段，每个阶段层层递进，既开软件交付质量信心，并且在流水线过程中提供快速反馈，减少后端环节浪费。

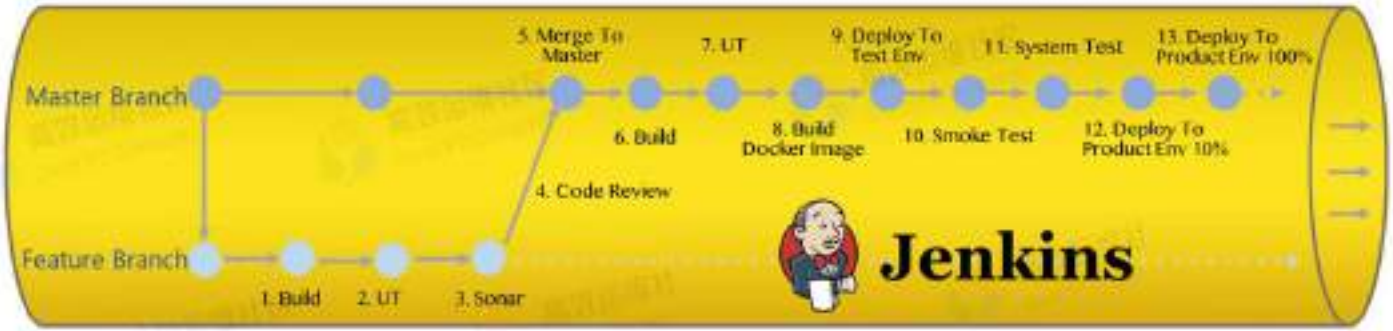
可视化流水线可以增强跨组织间的协同效率，提供有效的信息共享平台，从而统一组织目标，并且不断识别流水线中的约束点和瓶颈，以及潜在的自动化及协作场景，通过持续改进来不断提升软件交付效率。

级别	技术标准	备注
1	• 软件开发完成后再进行测试部署，整个软件交付过程存在复杂的跨门团队手工相关的等待浪费 • 交付过程大部分工作通过手工方式完成 • 交付过程不可见，过程信息是封闭的，交付状态不可追踪	
2	• 通过定义完整流畅的交付过程和清晰的交付规范，保证团队之间交付的有序 • 在软件各个交付环节内部建立自动化能力，提升各个环节的处理效率 • 交付过程团队内部可见，过程信息在团队间共享，交付状态团队内部可追踪	
3	• 部门间交付检查的定义由相关团队协同完成，发布必要环节进行手工确认 • 打通软件代码变更到部署之间的各个环节，建立全流程的自动化能力，并根据自动化测试结果控制软件交付质量 • 交付过程组织内部可见，过程信息有效整合，团队共享度量指标	
4	• 团队内部紧密协调，可实现标准交付完全前自主部署交付 • 建立可视化部署流水线，覆盖整个软件交付过程，每次变更都会触发完整的自动化部署流水线 • 交付过程全员可见，过程信息进行聚合分析展示	
5	• 持续部署流水线驱动持续改进 • 部署流水线过程信息进行有效的整合分析，通过数据价值挖掘推动业务改进	

【术】DevOps 三十六计

- DevOps 三十六计
  - 精益敏捷篇
    - 何勉 精益产品设计
    - 何勉 精益产品开发
    - 杨晓俊 敏捷项目管理
    - 用户故事
    - 敏捷Scrum
    - 精益看板
    - 规模化敏捷
    - 工具
      - 用友-何英华/宁锐 JIRA应用
  - 持续交付篇
    - 工程实践
      - 张乐 持续集成
      - 持续交付
      - 张乐 持续部署
    - 工具
      - 乐视-石雪峰 Git应用
      - 景韵 Jenkins应用
      - Ansible应用
      - 谭用 Docker应用
      - 赵舜东/姚炫伟 SaltStack应用
      - SonarQube应用
  - 安全篇
    - 安全开发
    - 安全测试
    - 韩方 安全运维
  - 性能优化篇
  - 开发架构篇
    - 微服务设计
      - 王磊 实践
      - Spring Cloud 工具
      - Dubbo
    - 高可用架构设计
    - 云原生应用
    - 重构
  - 测试技术篇
    - 实践
      - 徐琦琛 测试方法
      - 汪珺 自动化测试
      - 王磊 契约测试
    - 工具
      - Selenium
      - Jmeter
      - Appium
      - RobotFramework
  - 技术运营篇
    - 运维管理
      - 运维管理 涂彦
      - 运维规范 老王/韩晓光
      - 轻量级ITSM 闫林
      - 运维职业生涯规划 萧帮主
    - CMDB
    - 日常运维
      - 自动化运维 胥峰
      - 日常运维 大梁
      - 网络运维 张永福
      - 应用运维
        - 负载均衡
        - Nginx技巧 赵班长
      - 存储运营 高向冉
    - 监控与质量
      - 基础监控 赵班长
      - 应用监控
      - 容量管理 大梁
      - 工具篇
        - Zabbix应用 赵班长
        - OpenFalcon应用
    - 运维开发
      - Python 开发技巧（Python 运维必懂技巧） 郭宏泽
      - Shell 开发技巧
    - 日志管理
      - 日志治理 赵班长
      - ELK应用 赵班长
    - 数据库运维
      - 数据库运维（DBA生存三十六计） 周小军
      - MySQL运维 叶金荣
      - Oracle运维 盖国强
      - PostgreSQL运维 周正中
    - 容器篇
      - Mesos
      - Kubernetes
    - 数据中心运维
      - CDN运营 三十六计 高向冉
      - 数据中心节能运维 三十六计 闫林
      - IDC运维 王莹
    - 高级篇
      - 大数据质量
      - 智能运维
      - 硅谷SRE三十六计
      - ChatOps 熊昌伟
      - 大数据运维 范伦挺

【器】全开源端到端部署流水线V1.0



git Maven JUnit docker Selenium kubernetes

GitLab Nexus sonarqube HARBOR JMeter elastic

高效运维社区根据业界领先的 DevOps 实施模型以及来自BAT等一线互联网公司的实战经验，构建了基于全开源工具集、相互打通的整套持续交付流水线，旨在帮助企业快速搭建起 DevOps 工具链平台，快速拥有主流互联网公司的持续交付能力。

请扫码以直接欣赏

目录

1 DevOps的道法术器

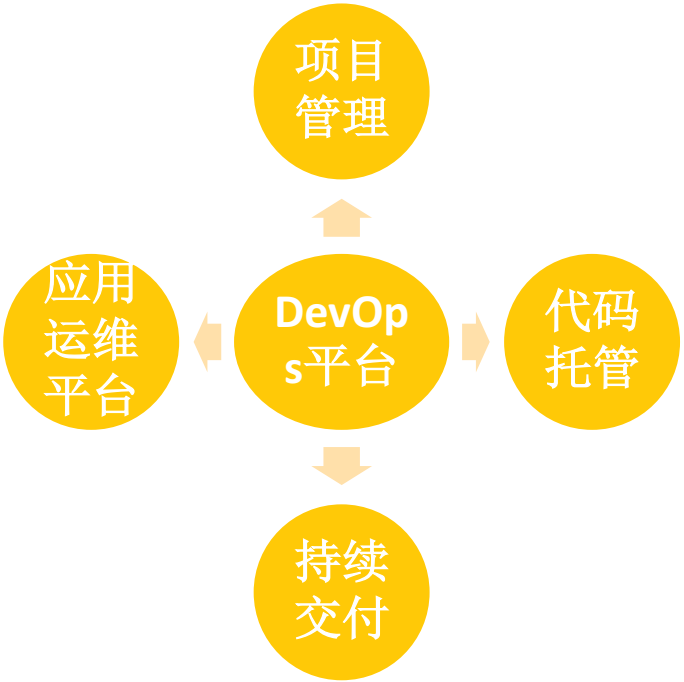
2 如何构建一个真正的DevOps平台

3 全开源端到端部署流水线

4 全开源全链路自动化运维体系

5 你来问，我来答

DevOps平台四大模块



DevOps平台之项目管理

JIRA

Redmine

DevOps平台之代码托管

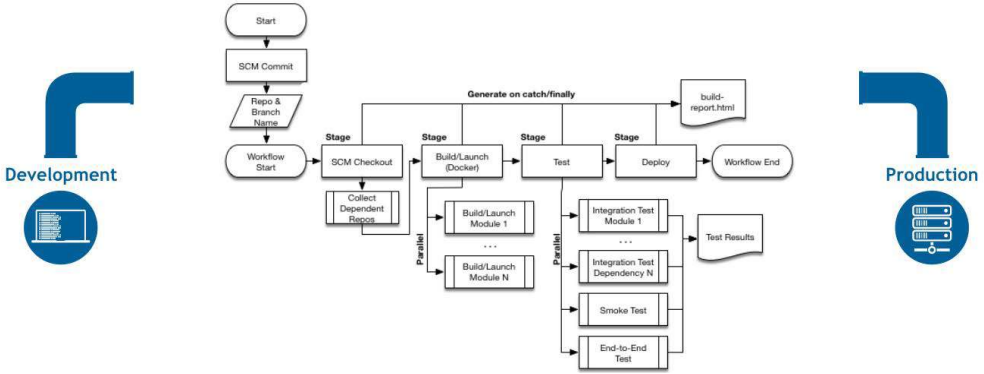
GOPS2017
Shanghai

SVN

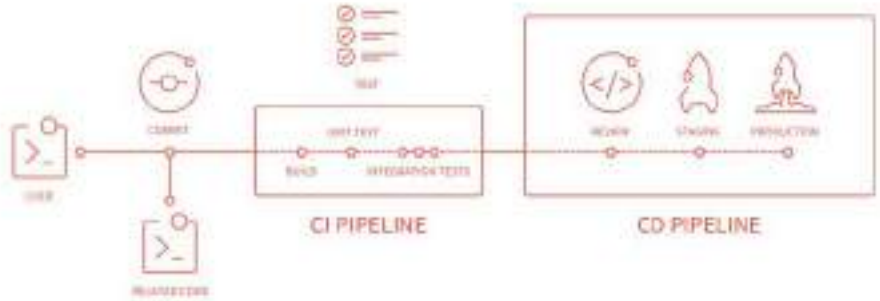
Git

DevOps平台之持续交付

Jenkins



GitLab



DevOps平台之统一运维平台

目录

1 DevOps的道法术器

2 如何构建一个真正的DevOps平台

3 全开源端到端部署流水线

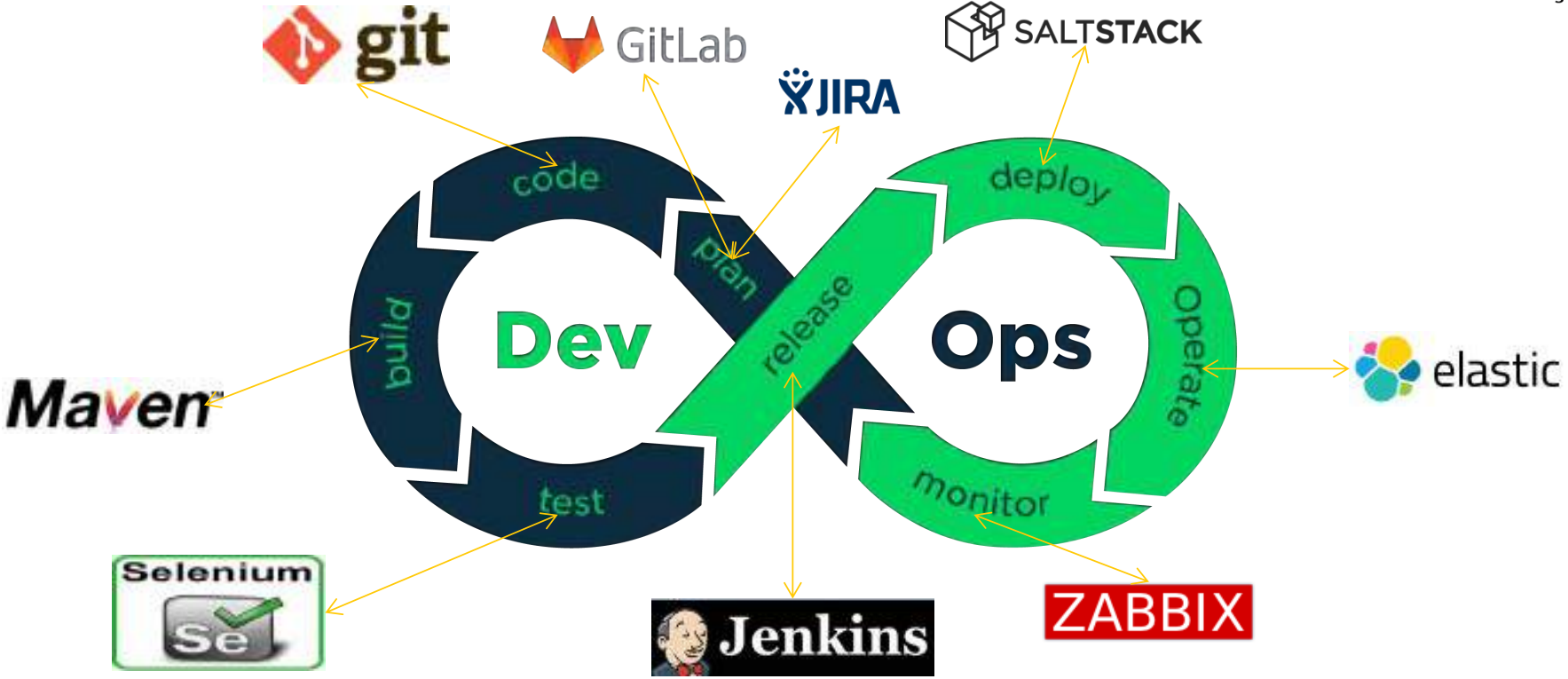
4 全开源全链路自动化运维体系

5 你来问，我来答

软件交付的原则

- ✓ 为软件的发布创建一个可重复且可靠的过程
- ✓ 将几乎所有的事情自动化
- ✓ 把所有的东西都纳入版本控制
- ✓ 提前并频繁地做让你感到痛苦的事
- ✓ 内建质量
- ✓ “DONE”意味着“已发布”
- ✓ 交付过程是每个成员的责任
- ✓ 持续改进

持续交付工具链


git
GitLab
JIRA
SALTSTACK
code
plan
build
test
release
deploy
operate
monitor
Dev
Ops
Maven
elastic
Selenium
Se
Jenkins
ZABBIX

什么是流水线

WIKIPEDIA
The Free Encyclopedia

Ford Model T

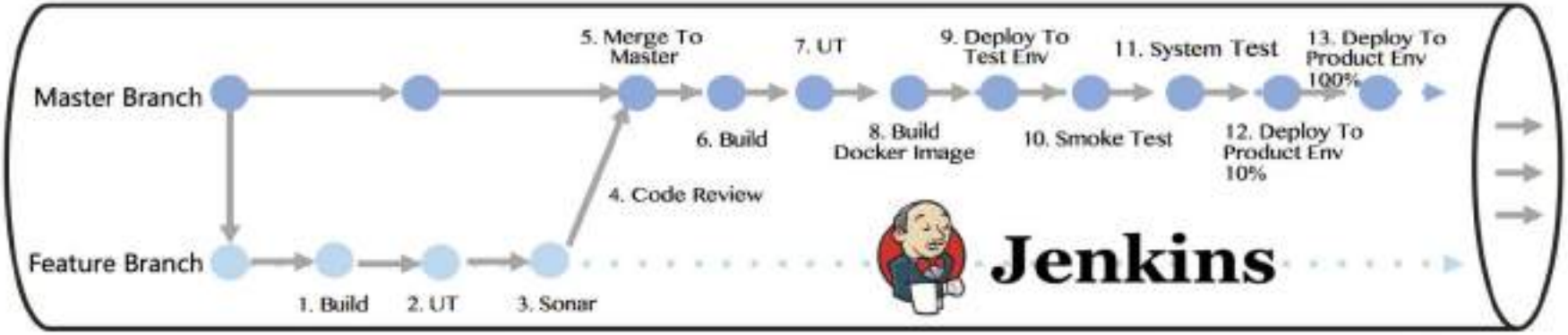
Mass production

In 1910, after assembling nearly 12,000 Model Ts, Henry Ford moved the company to the new Highland Park complex.
Ford's cars came off the line in three-minute intervals, much faster than previous methods, reducing production time by a factor of eight (requiring 12.5 hours before, 93 minutes afterwards), while using less manpower.

软件交付流水线

- 软件交付流水线是指软件变更从提交到版本控制库，到发布给用户的整个过程
- 软件的每次变更都会经历一个复杂的流程才能发布
- 这个流程包括构建软件、一系列不同阶段的测试与部署等，需要多团队协作完成
- 交付流水线对交付流程进行了建模，并支持查看、控制整个交付流程

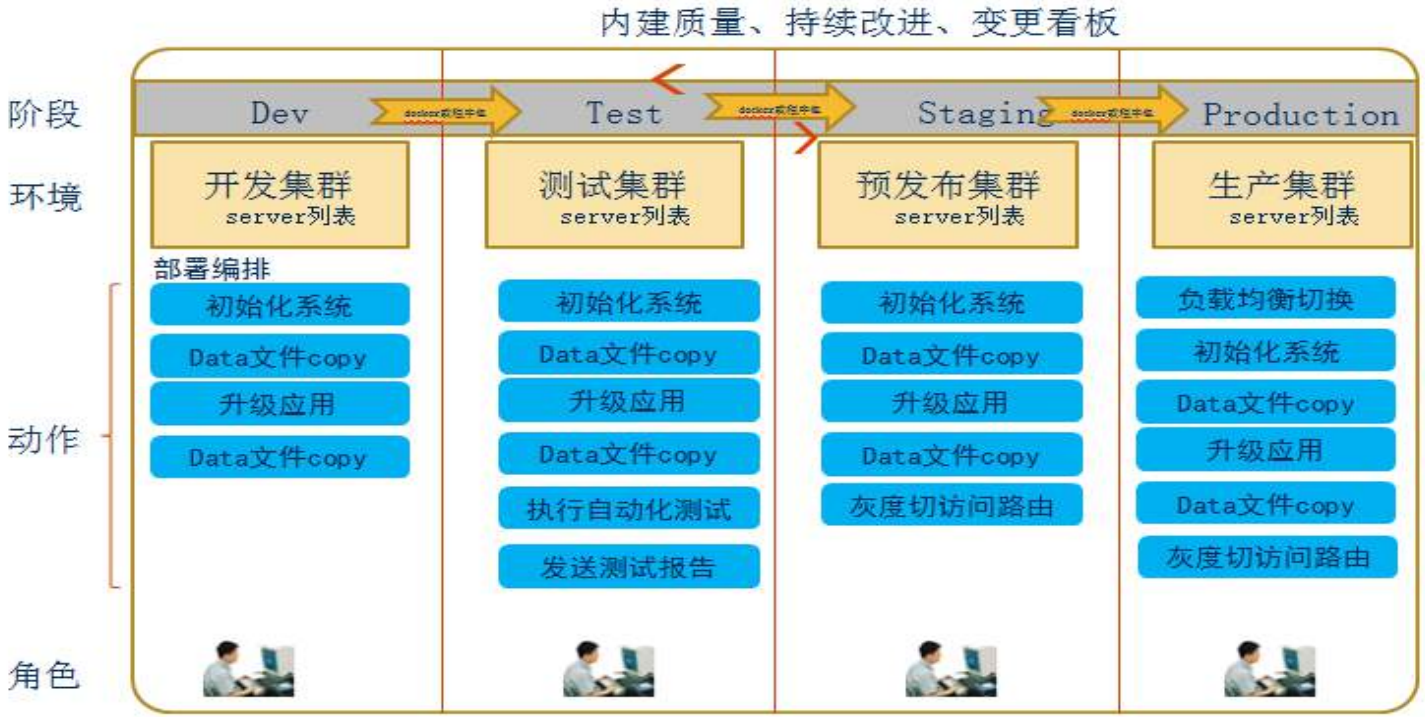
全开源端到端部署流水线总图



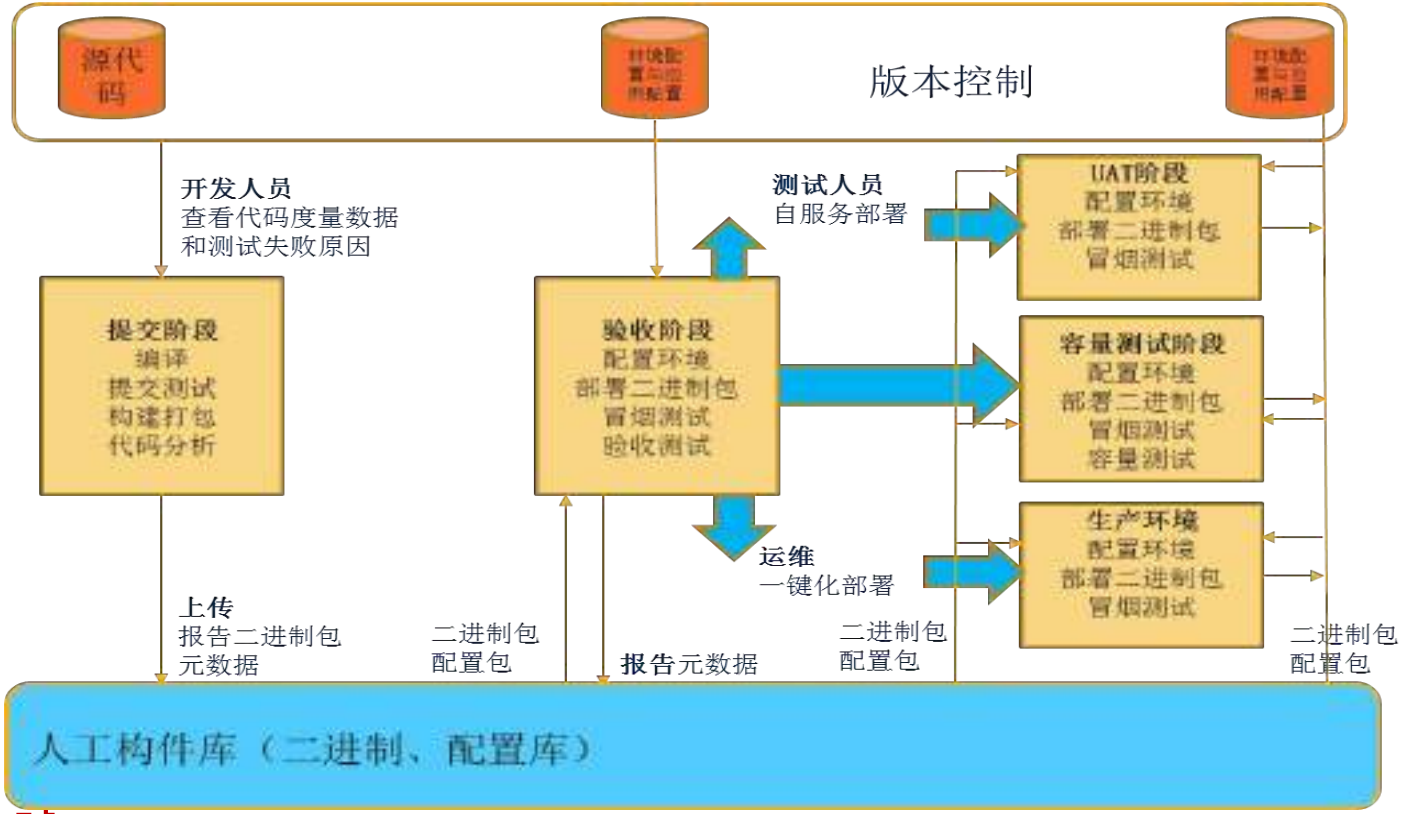
git Maven JUnit docker Se kubernetes

GitLab Nexus sonarqube HARBOR JMeter elastic

持续交付流水线


内建质量、持续改进、变更看板
阶段
环境
动作
角色
Dev
Test
Staging
Production
开发集群
server列表
测试集群
server列表
预发布集群
server列表
生产集群
server列表
部署编排
初始化系统
Data文件copy
升级应用
Data文件copy
初始化系统
Data文件copy
升级应用
Data文件copy
执行自动化测试
发送测试报告
初始化系统
Data文件copy
升级应用
Data文件copy
灰度切访问路由
负载均衡切换
初始化系统
Data文件copy
升级应用
Data文件copy
灰度切访问路由

部署流水线分阶段实施


源代码
环境配置与应用配置
版本控制
环境配置与应用配置
开发人员
查看代码度量数据
和测试失败原因
测试人员
自服务部署
UAT阶段
配置环境
部署二进制包
冒烟测试
提交阶段
编译
提交测试
构建打包
代码分析
验收阶段
配置环境
部署二进制包
冒烟测试
验收测试
容量测试阶段
配置环境
部署二进制包
冒烟测试
容量测试
运维
一键化部署
生产环境
配置环境
部署二进制包
冒烟测试
上传
报告二进制包
元数据
二进制包
配置包
报告元数据
二进制包
配置包
二进制包
配置包
人工构件库（二进制、配置库）

案例：可靠可重复的流水线

通过流水线阶段晋级，平衡测试反馈速度与覆盖度

通过流水线分析瓶颈、识别自动化改造点和协作点

划分阶段（Stage）：

- 多个Stages是串行执行
- 前一个Stage成功完成后自动触发
- 也可以通过手工触发

执行作业（Job ）：

- Stage里的多个Jobs串/并行执行
- Stage里的多个Jobs定时执行
- 自动判断或人工标记Pass/Fail
- Job Fails → Stage Fails

质量门（通过标准）：

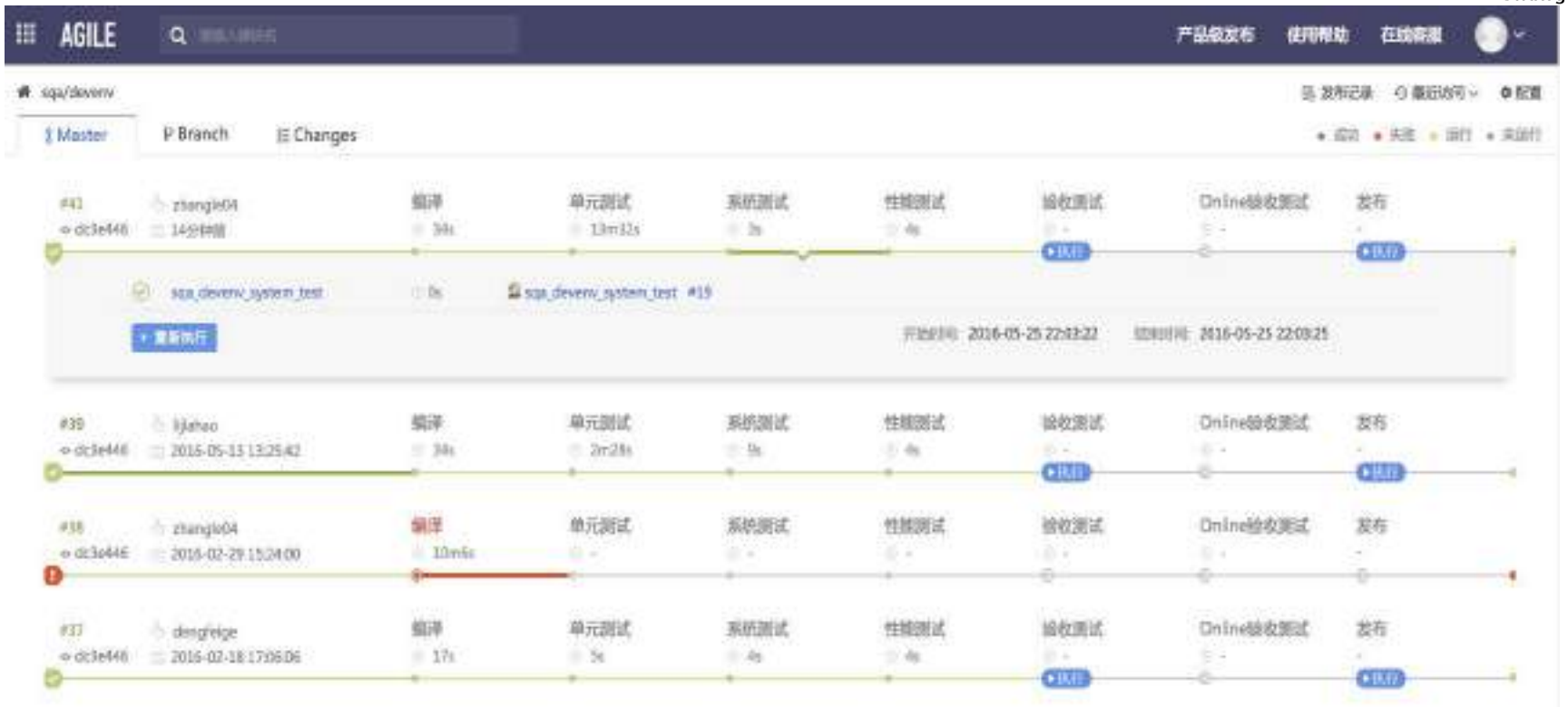
- Pass/Fail判定标准
- 测试通过率>xx%
- 代码覆盖率>xx%

决策点（人工干预）：

- 可配置人工决策，一键Approve后流水线自动执行
- 也可配置前序Stage成功后自动触发执行

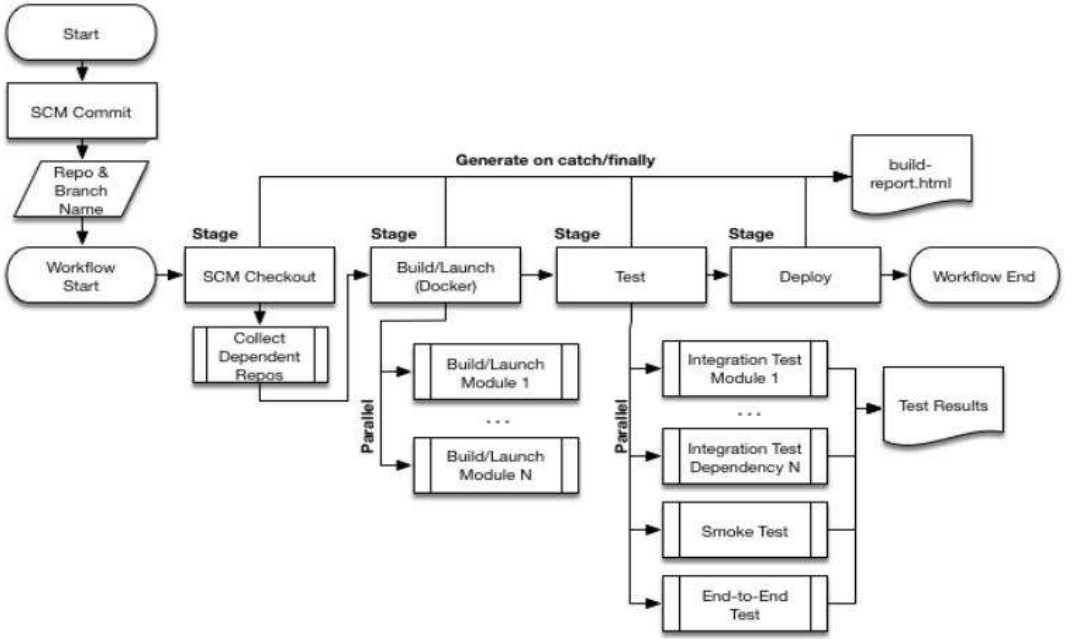
案例：百度交付流水线的实现

GOPS2017
Shanghai



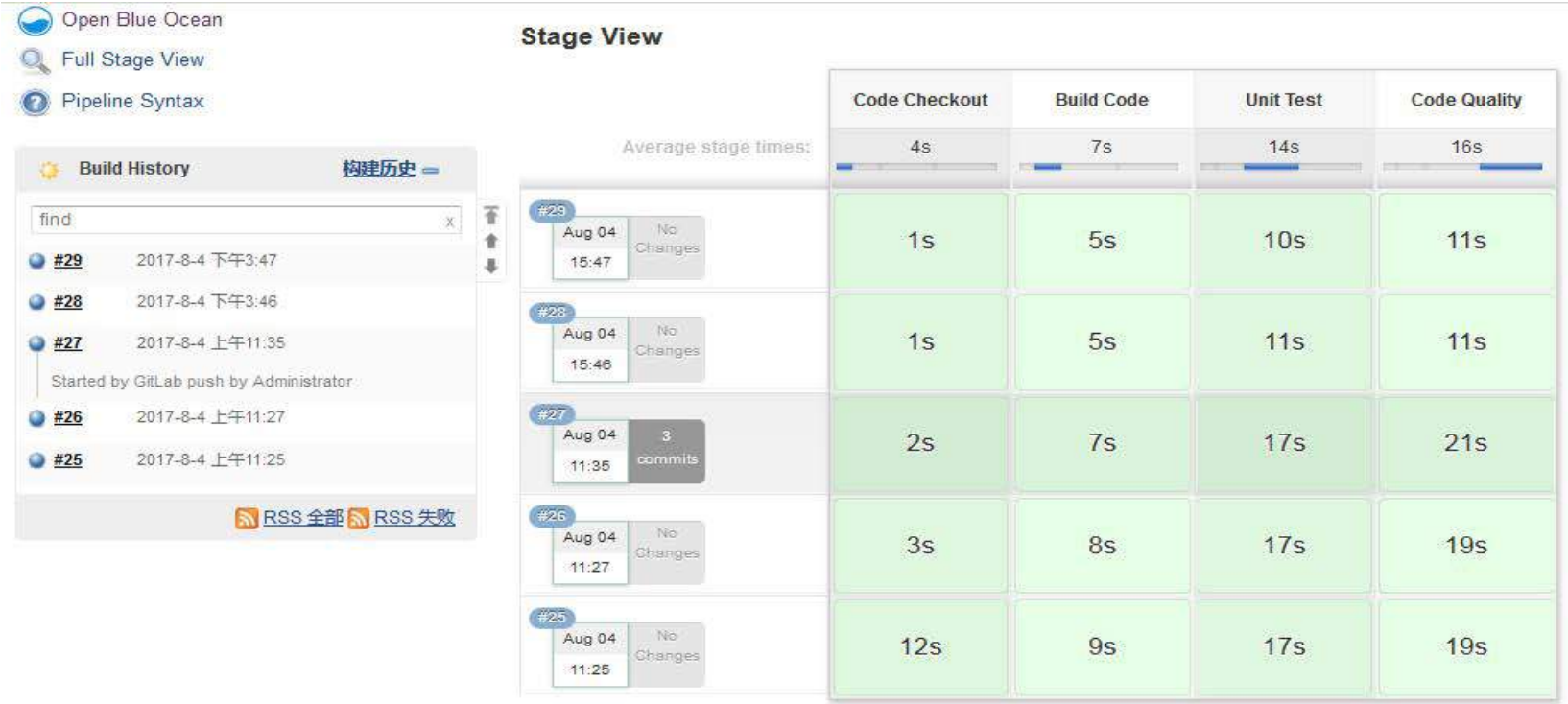
基于Jenkins的Pipeline as Code

GOPS2017
Shanghai



Jenkins Pipeline

GOPS2017
Shanghai



Pipeline as Code

GOPS2017
Shanghai

```
node {
   gitlabCommitStatus(builds: ["Commit Build"]) {
      stage('Code Checkout') { // for display purposes
         // Get some code from a GitLab repository
         git credentialsId: '844d014b-3f81-4557-b4a2-32761eba6008', url: 'git@code.greatops.net:devops/account.git'
      }
      stage('Build Code') {
          echo "Build Code"
          sh "mvn clean compile"
      }

      stage('Unit Test') {
          echo "Unit Test"
          sh "mvn test"
      }
       stage('Code Quality') {
          echo "Code Quality"
          sh "mvn sonar:sonar"
      }
   }
}
```

自动化部署设计目标

GOPS2017
Shanghai

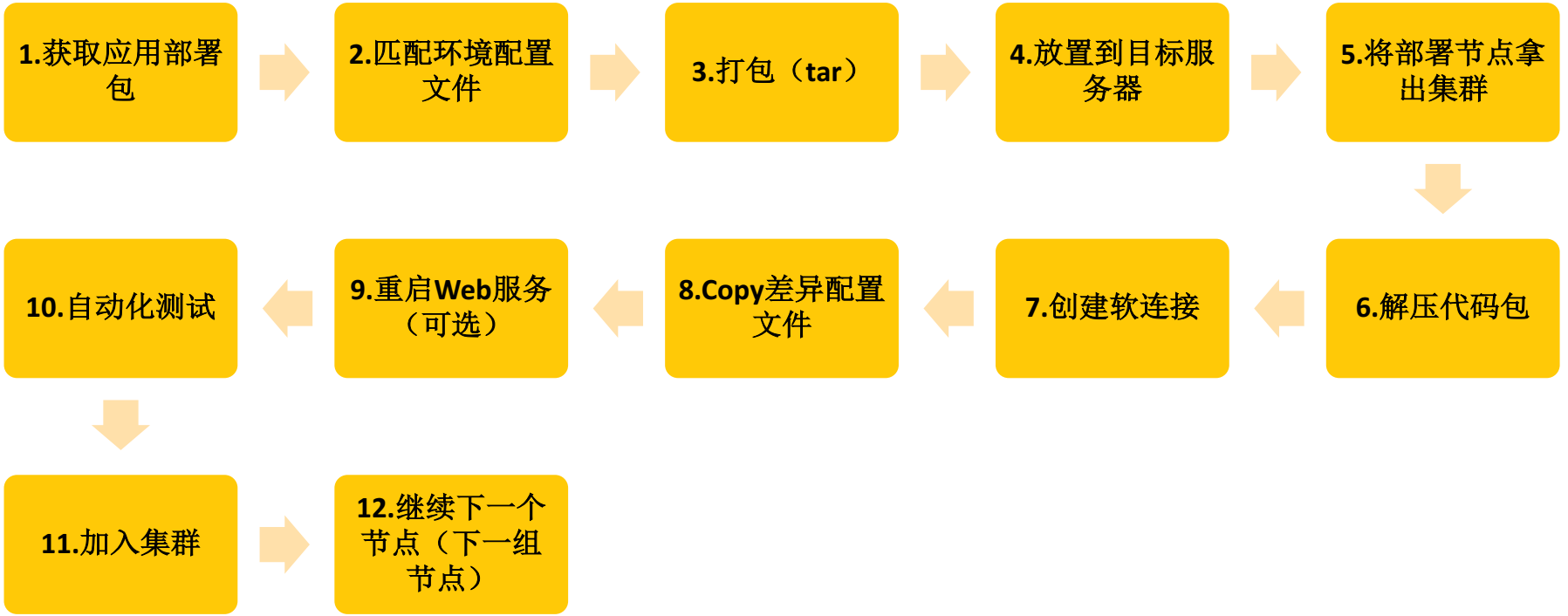
一键部署

- 无需运维参与
- 用户无感知

秒级回滚

- 故障处理原则
- 正常回滚和紧急回滚

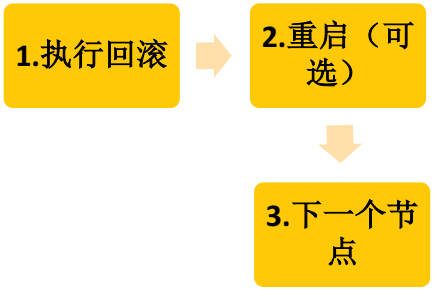
自动化部署流程设计


1.获取应用部署包
2.匹配环境配置文件
3.打包（tar）
4.放置到目标服务器
5.将部署节点拿出集群
6.解压代码包
7.创建软连接
8.Copy差异配置文件
9.重启Web服务（可选）
10.自动化测试
11.加入集群
12.继续下一个节点（下一组节点）

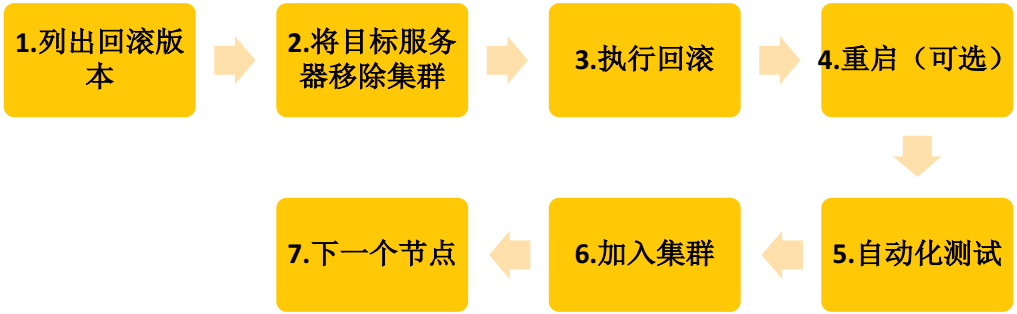
自动化回滚流程设计

GOPS2017
Shanghai

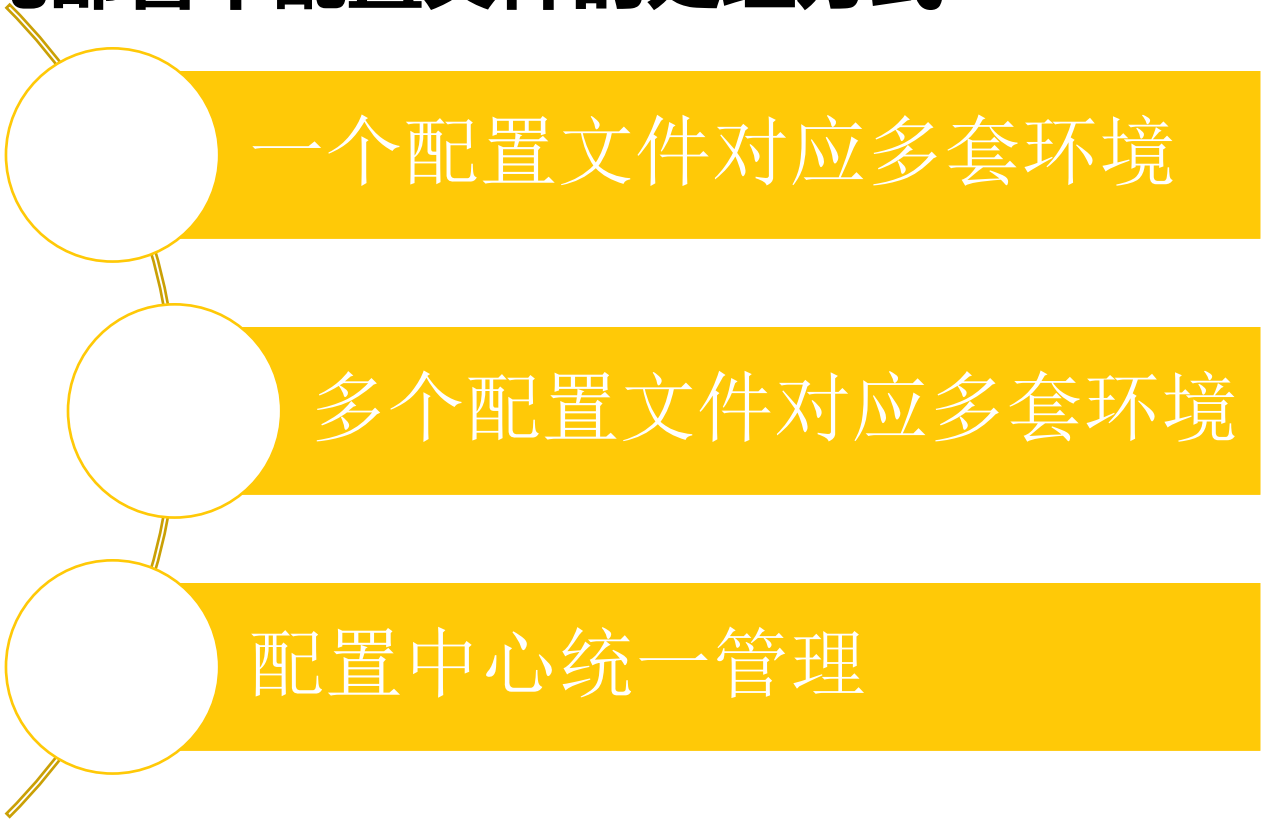
紧急回滚



正常回滚



自动化部署中配置文件的处理方式

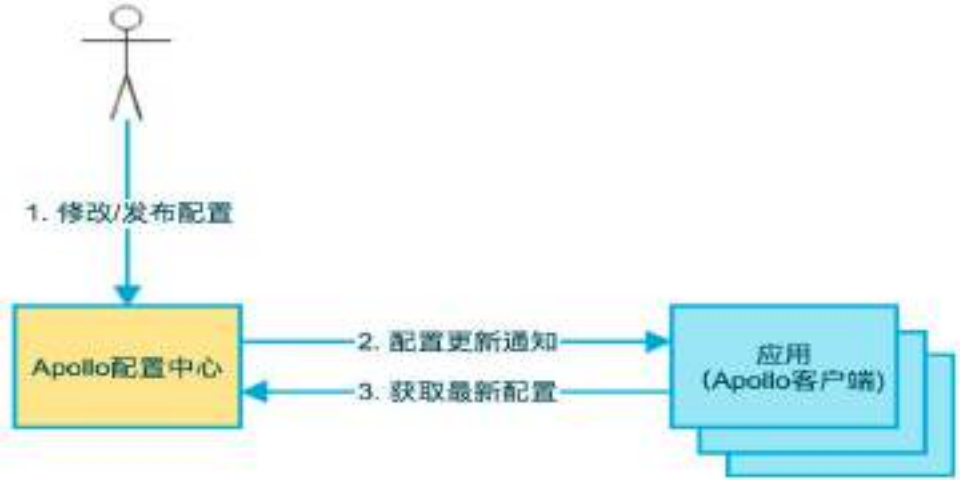


自动化部署中配置文件的处理方式-配置中心

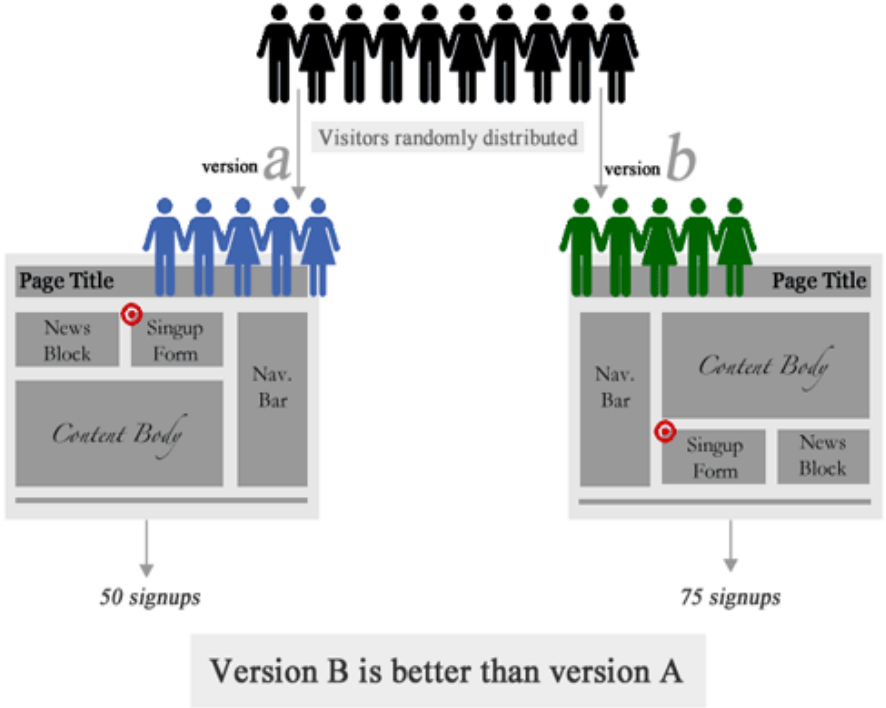
GOPS2017
Shanghai

https://github.com/ctripcorp/apollo

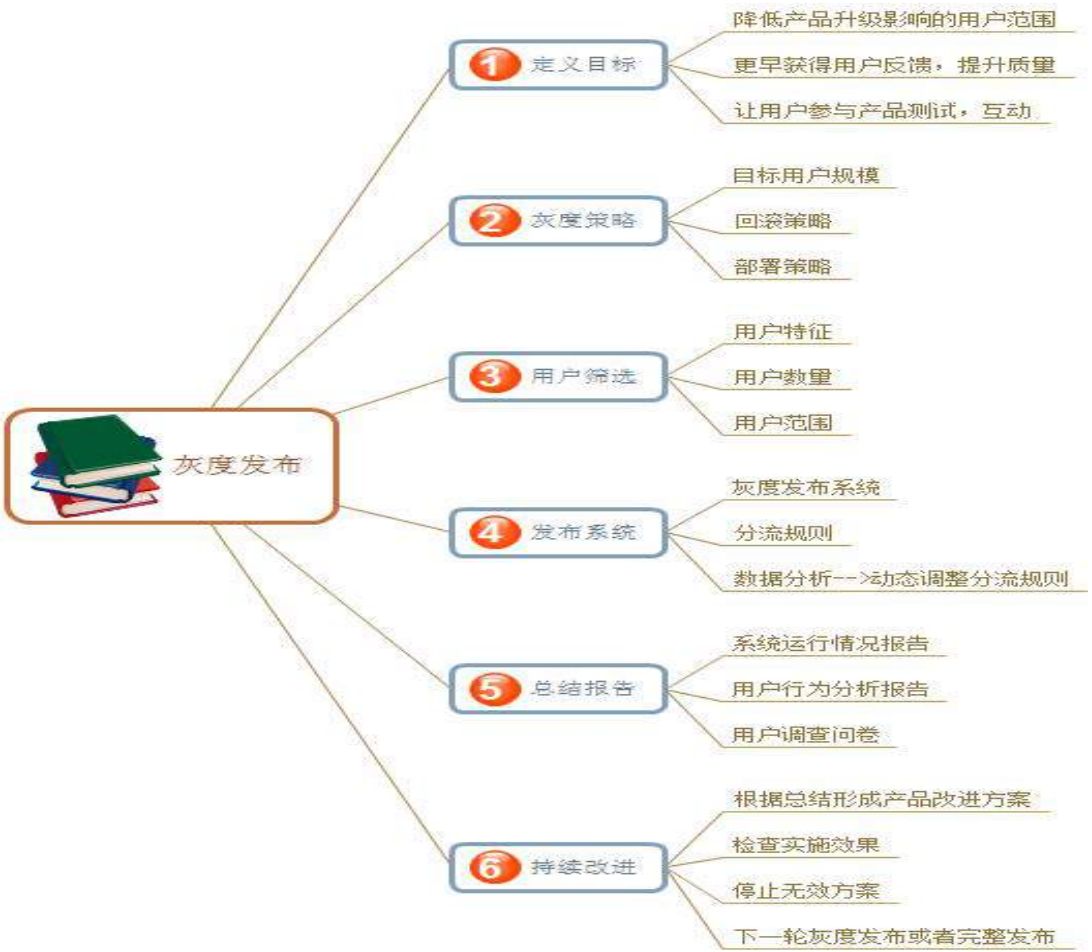
https://github.com/knightliao/disconf



A/B测试与灰度发布

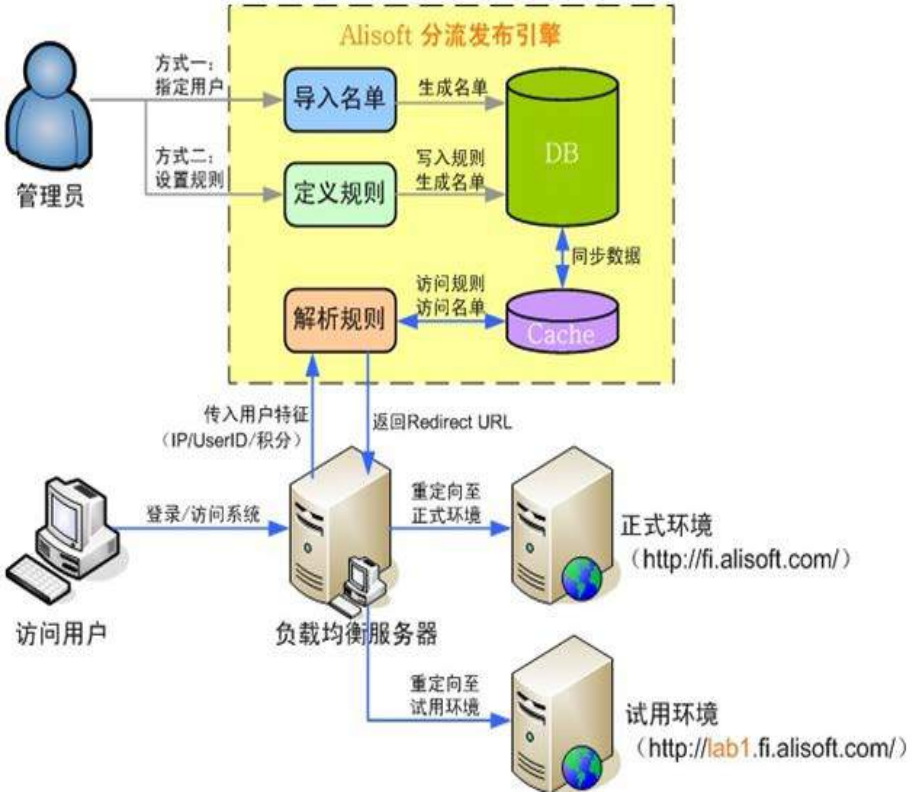


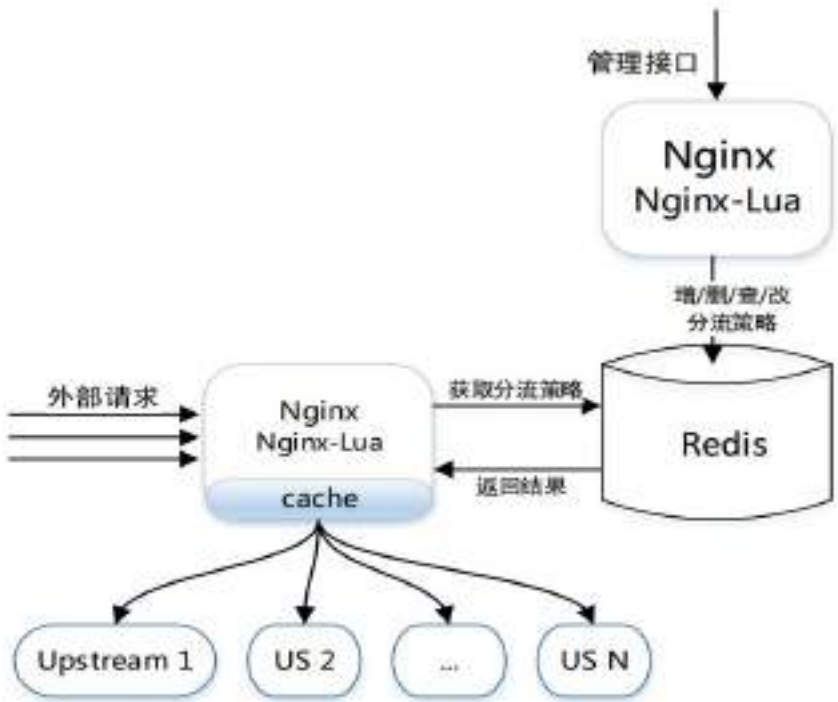
将真人照片放在网站上会获得一倍的转化。研究说明，我们潜意识被照 片吸引了。



- 灰度发布是对某一产品的发布逐步扩大使用群体范围。
- A/B测试重点是在几种方案中选择最优方案

灰度发布案例


Alisoft 分流发布引擎
方式一：
指定用户
方式二：
设置规则
管理员
导入名单
生成名单
定义规则
写入规则
生成名单
DB
同步数据
解析规则
访问规则
访问名单
Cache
传入用户特征
（IP/UserID/积分）
返回Redirect URL
登录/访问系统
访问用户
负载均衡服务器
重定向至
正式环境
正式环境
（http://fi.alisoft.com/）
重定向至
试用环境
试用环境
（http://lab1.fi.alisoft.com/）


管理接口
Nginx
Nginx-Lua
增/删/查/改
分流策略
外部请求
Nginx
Nginx-Lua
cache
获取分流策略
返回结果
Redis
Upstream 1
US 2
...
US N

目录

1 DevOps知识体系精讲

2 企业如何进行DevOps落地？

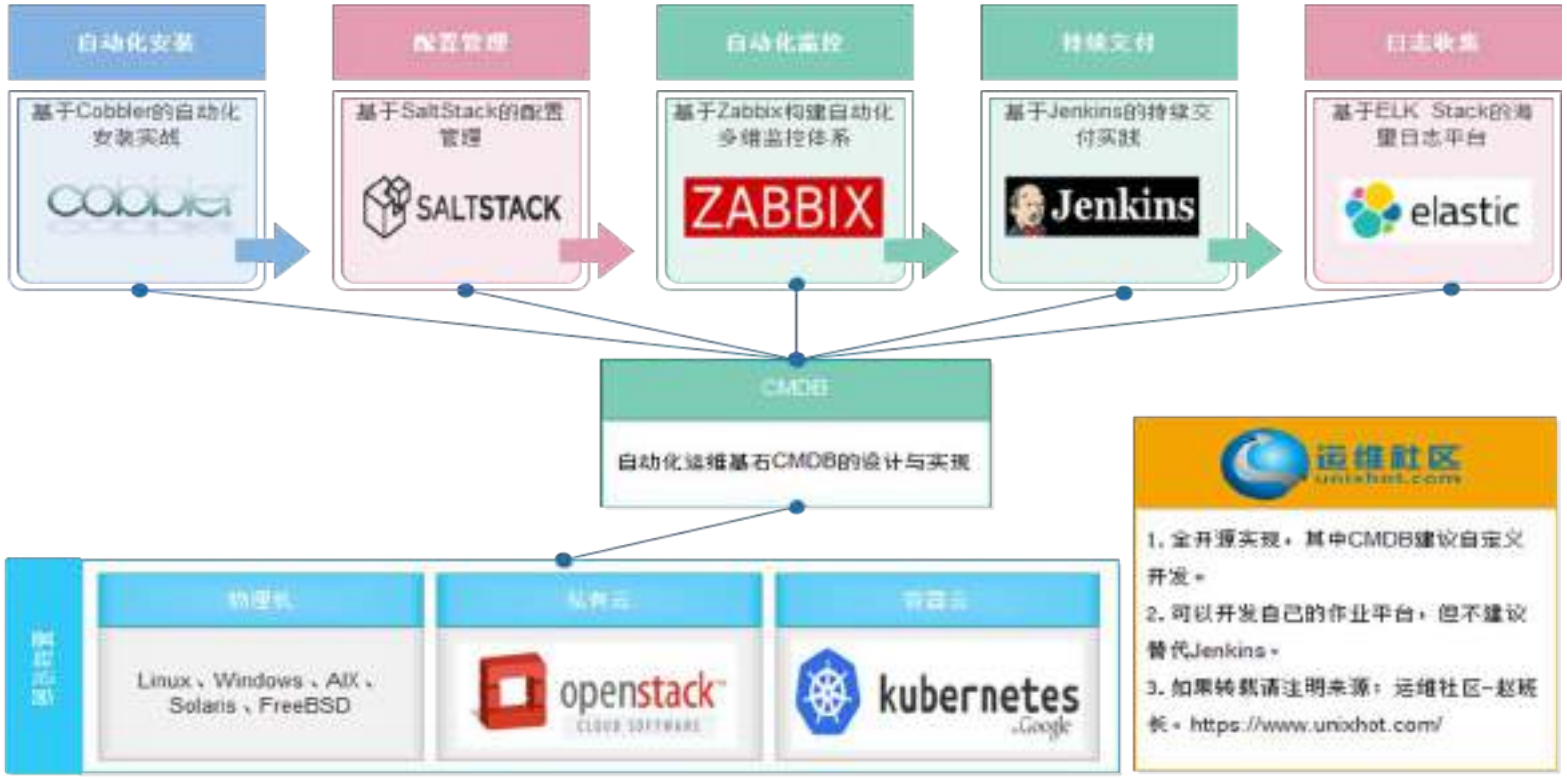
3 如何构建一个DevOps平台？

4 基础架构即代码和不可变基础设施

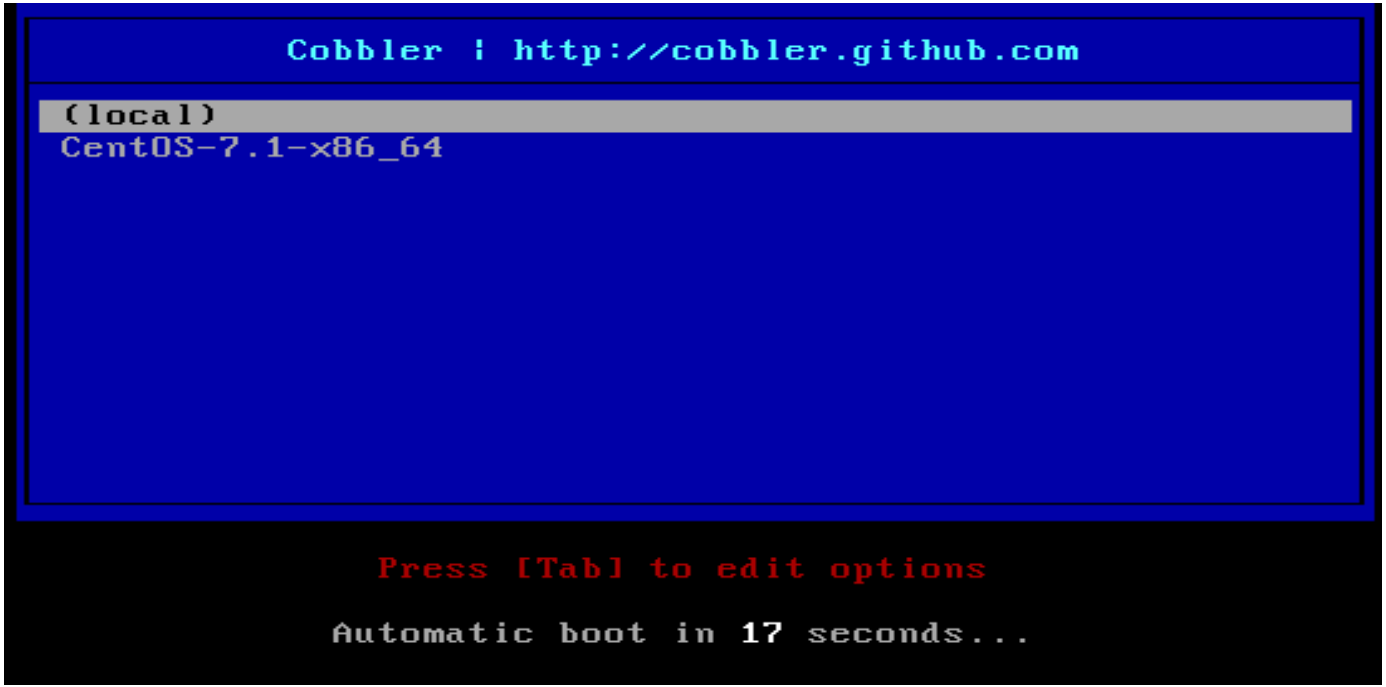
5 持续交付之端到端部署流水线

6 全链路自动化运维实践

基于开源的开源全链路自动化运维体系



基于Cobbler的自动化安装

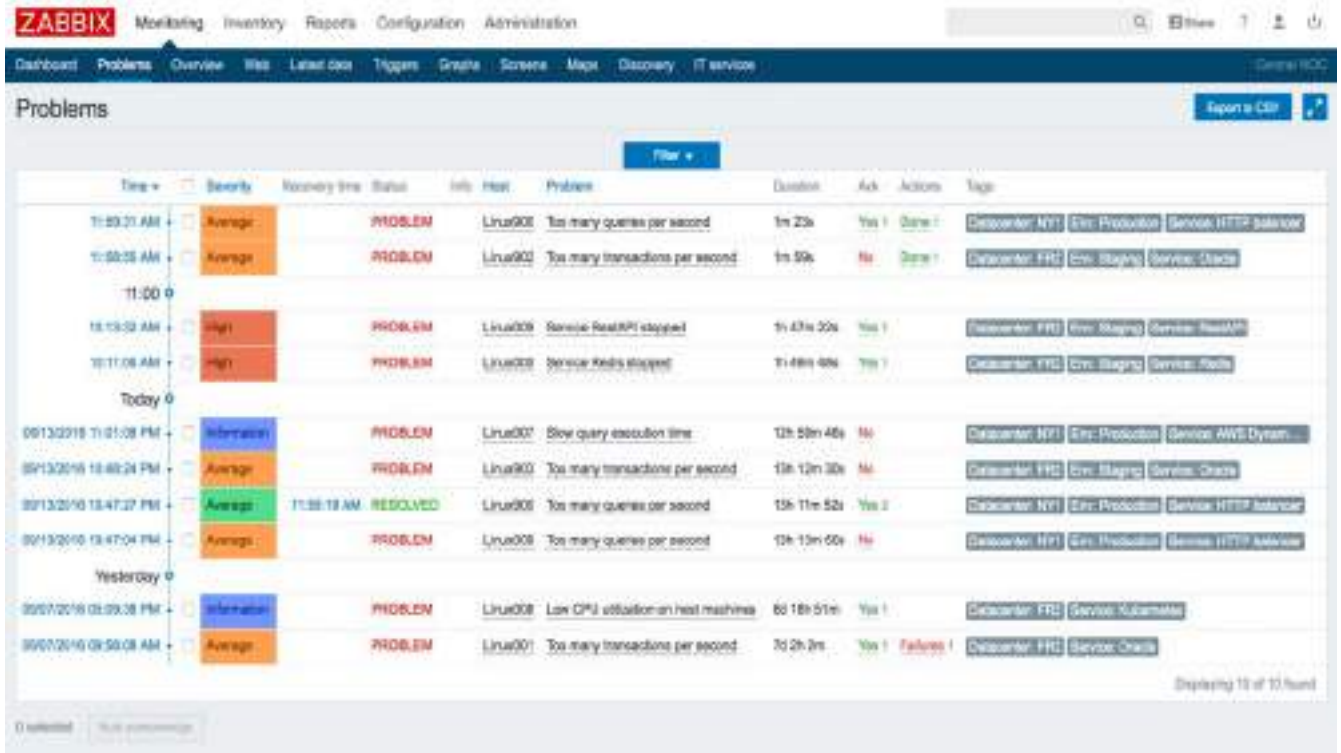


SaltStack自动化配置管理

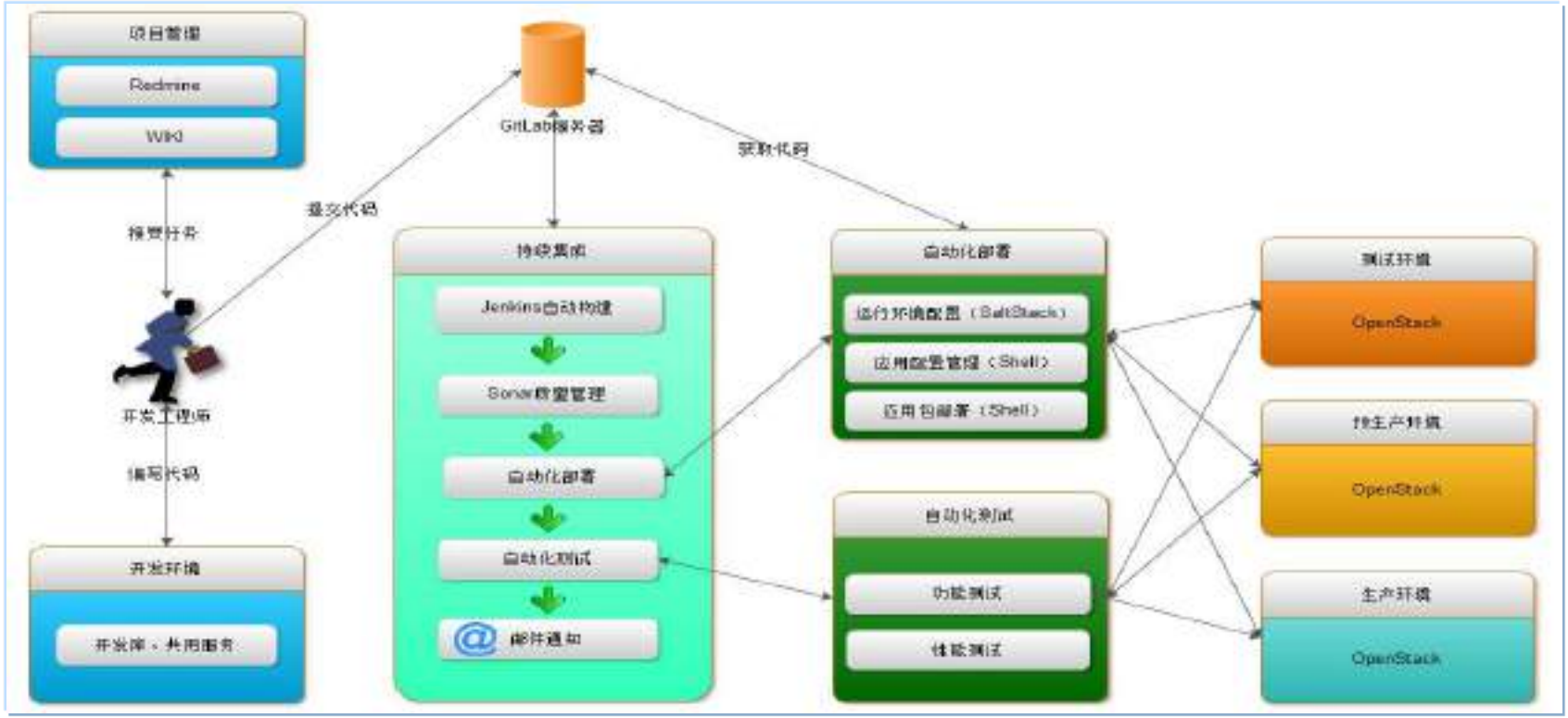
- 1.系统初始化
- 2.基础模块
- 3.应用模块

基于Zabbix构建企业级监控平台

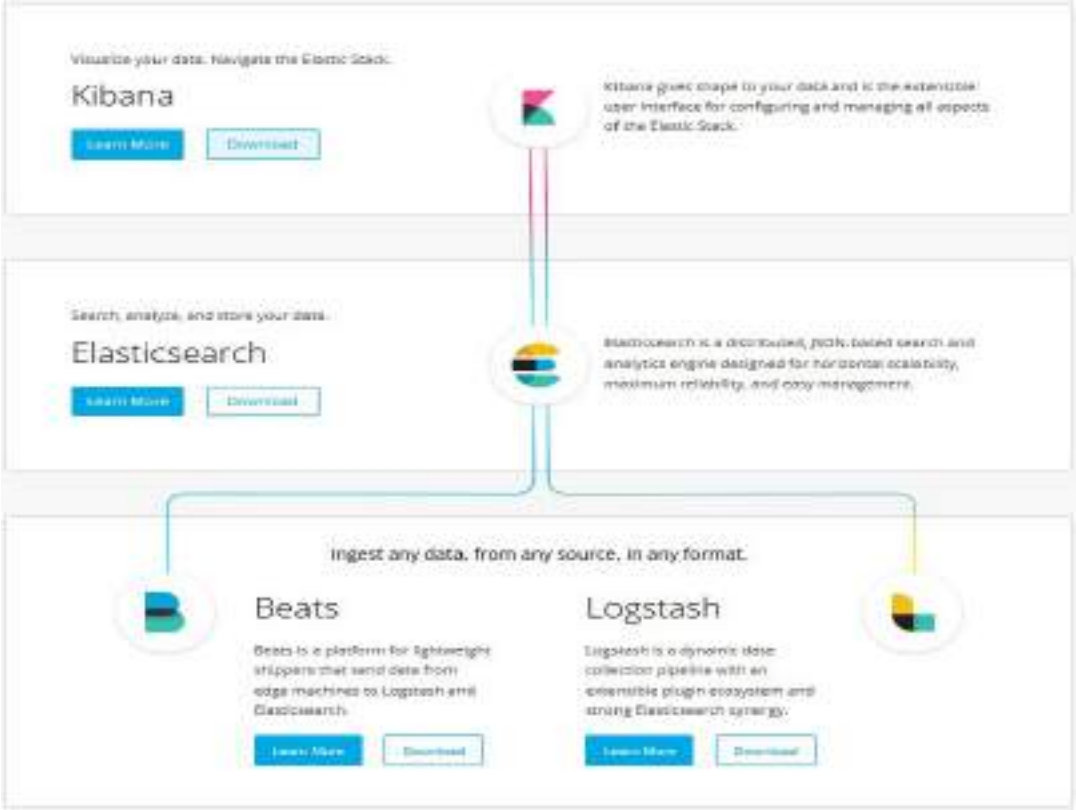
GOPS2017
Shanghai



基于Jenkins的持续交付



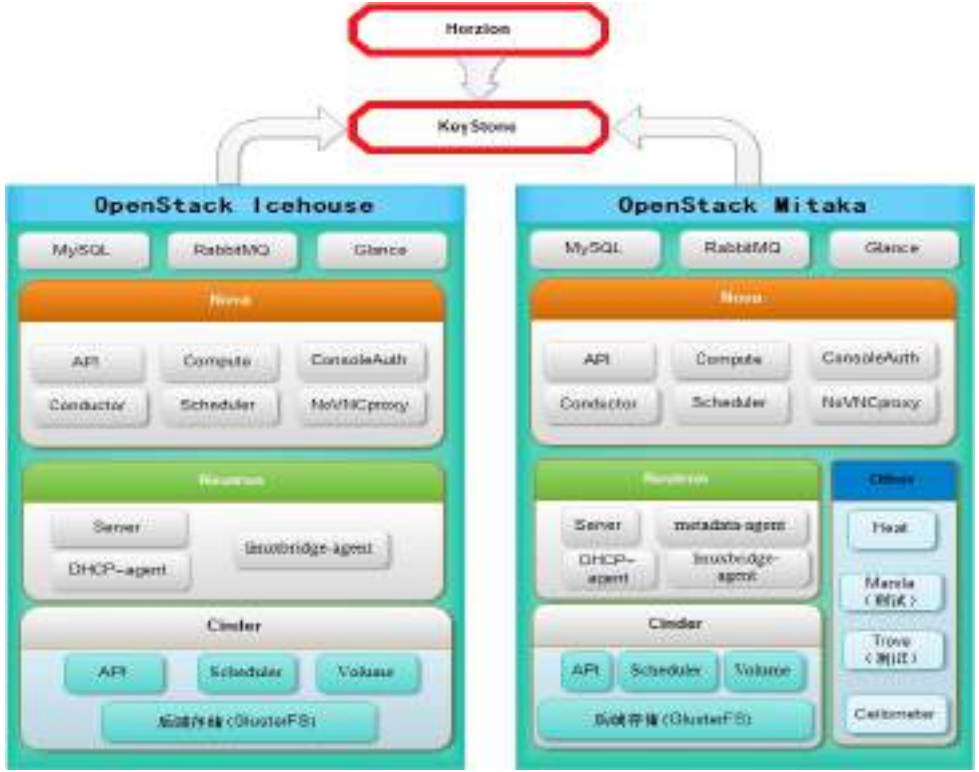
基于Elastic Stack的日志平台



The Open Source Elastic Stack

- Kibana
- Elasticsearch
- Beats
- Logstash

OpenStack私有云架构



- 1.中小规模私有云建议使用VLAN。
- 2.分布式存储GlusterFS，未来向Ceph迁移。
- 3.多集群建设，避免单点，独立开发作业平台。

运维架构层级/运维角度			内容描述/主要技术关键词
客户端层	浏览器		Cookie、浏览器缓存协商（Last-Modified、Expires、Etag）、组件分离、前端优化、运维检测工具
	DNS		浏览器DNS缓存、DNS缓存、自建DNS服务器、商业DNS产品、智能DNS、公共DNS（BGP anycast）、bind+DLZ/DPDK
	客户端/APP		HTTP-DNS、打点日志、加密传输、移动推送、各类SDK（监控SDK、推流SDK等）
外部层	第三方CDN		GSLB、反向代理缓存、分布式存储、流量调度、配置管理、用户端（各类API如：带宽监控、预缓存、缓存刷新）
	云计算		公有云服务、混合云、运维外包服务、APM（应用性能管理）、第三方安全解决方案（防DDOS、WAF）
网络层	互联层		多机房互联（VPN、专线）、异地灾备-->异地多活-->按SET部署
	核心层		防火墙、路由器、Ipsec VPN、链路负载均衡和高可用（CCNP级别）
	汇聚层		三层交换 动态路由（OSPF）、静态路由、EC（端口汇聚）、MSTP+VRRP等（CCNP级别）
	接入层		二层交换（VTP、SPF、Trunk、端口安全）等（CCNA级别）
接入层	负载均衡高可用	四层负载均衡	开源：LVS（IP负载均衡）+Keepalived、Haproxy 商业：F5、Netscaler
		七层负载均衡	反向代理：Haproxy、Nginx、Apache（根据HTTP协议支持的属性进行L7分发）、A/B Test Gateway、WAF
	反向代理缓存		ATS、Squid、Varnish、Nginx(缓存分级、预缓存、缓存刷新)
应用服务层	Web服务层		HTTP协议、Web服务器（Apache、Nginx/OpenResty、Tomcat、Resin、Jboss）安全设置、性能优化
	应用服务层		运行环境（PHP Python Java C C++）、性能优化、缓存（OPCache、LocalCache）、Session存储、代码部署
	业务层	业务实现	API网关、302调度、业务模块化（电商例：用户、商品、购物车、结算中心、价格等服务）、微服务
		服务层	SOA框架（Dubbo）、微服务框架（Spring Cloud）、协议（RPC、RESTful）、框架安全、应用性能监控
	分布式层	消息队列	ActiveMQ（成熟）、RabbitMQ（成熟、案例多）、RocketMQ（业务应用）、Kafka（日志传输）、ZeroMQ（快）

GOPS2017 Shanghai

运维知识体系（部分）

Web缓存知识体系-V1.0 赵舜东（赵班长）【转载请注明来自于-运维社区：https://www.unixhot.com/】

缓存分层	缓存分级	内容	内容简介/主要技术关键词
用户层	DNS	浏览器DNS缓存	Firefox默认60秒,HTML5的新特性：DNS Prefetching
		应用程序DNS缓存	Java（JVM）、PHP语言本身的DNS缓存
		操作系统DNS缓存	客户端操作系统DNS缓存
		DNS缓存服务器	专用的DNS缓存服务器、LocalDNS缓存
	浏览器	浏览器缓存	HMTL5新特性：Link Prefetching
			基于最后修改时间的HTTP缓存协商： Last-Modified
			基于打标签的HTTP缓存协商： Etag
			基于过期时间的HTTP缓存协商： Expires、cache-control
代理层	CDN	反向代理缓存	基于Nginx+（Squid、Varnish、ATS）等，一般有多级
Web层	解释器	Opcache	操作码缓存
	Web服务器	Web服务器缓存	Apache（mod_cache）、Nginx(FastCGI缓存、Proxy cache)
应用层	应用服务	动态内容缓存	缓存动态输出
		页面静态化	动态页面静态化，需要专门用于静态化的CMS
		Local Cache	应用本地缓存，PHP（Yac、Xcache） Java（ehcache）
数据层	分布式缓存	分布式缓存	Memcache、Redis等
	数据库	MySQL	MySQL自身缓存、innodb缓存、MYISAM缓存
系统层	操作系统	CPU Cache	L1（数据缓存、指令缓存） L2、L3
		内存Cache	内存高速缓存、Page Cache
物理层	Raid卡	Raid Cache	磁盘阵列缓存（Raid卡可以控制是否使用磁盘高速缓存）
	磁盘	Disk Cache	磁盘高速缓存
备注	1.此体系结构仅包含读缓存（Cache），不包含写缓冲（Buffer），所有很多缓冲区没有列举。 2.根据用户发起一个HTTP请求开始，持续更新中，欢迎大家添加更多的内容。		

缓存知识体系

关注我

GOPS2017
Shanghai

GOPS2017
Shanghai

Thanks

高效运维社区

开放运维联盟

荣誉出品

想第一时间看到
高效运维社区公众号
的好文章吗？

请打开高效运维社区公众号，点击右上角小人，如右侧所示设置就好