DOIS

2018 | 中国·北京站

DevOps 落地，从这里开始

DevOps 国际峰会

暨 DevOps 金融峰会

指导单位：云计算开源产业联盟 OpenSource Cloud Alliance for IndustRy OSCAR

主办单位：DevOps时代　高效运维社区 GreatOPS Community

2018年6月29日-30日

地址：北京悠唐皇冠假日酒店

DOIS

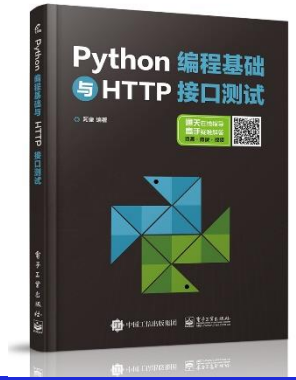
DevOps@BOC
器用之道 如琢如磨

阿奎

自我介绍

DOIS

- 在金融领域从事软件开发工作14年，拥有9年开发管理经验，带过70+人以上自有软件开发人员的团队，熟悉金融核心系统架构、业务分析。
- 最近5年来从事互联网金融软件开发管理和技术研究，同时，致力于组织级敏捷开发方法的推广和实施，对大型组织的敏捷转型有实际的经验和体会。
- 系统分析师、CCEP、CSM、DevOps Master



阿奎

目录

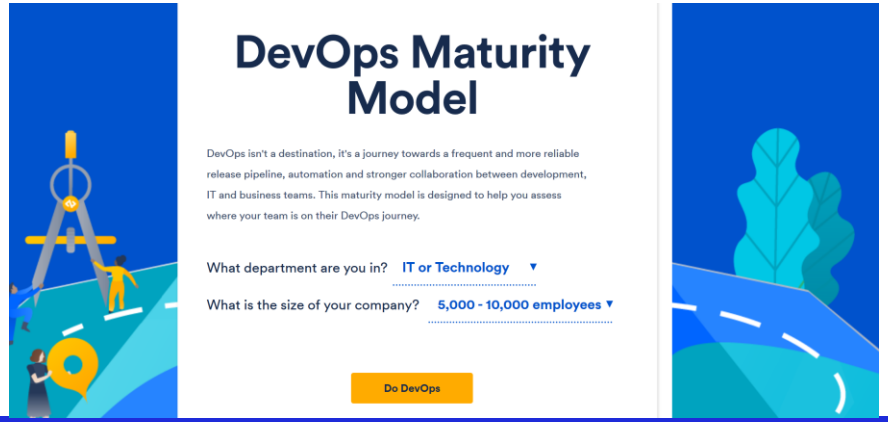
DevOps isn`t a destination, it`s a journey!

DOIS

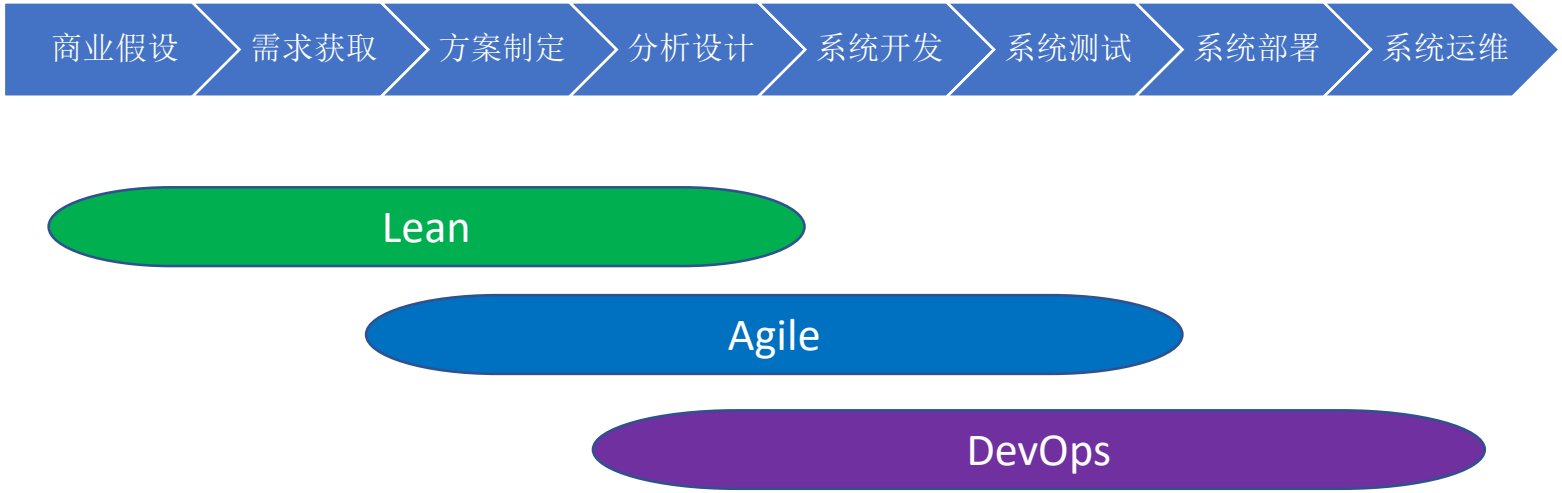
DevOps isn't a destination, it's a journey towards **a frequent and more reliable release pipeline**, **automation and stronger collaboration** between development, IT and business teams.

高频、稳定

https://www.atlassian.com/devops/devops-maturity-model

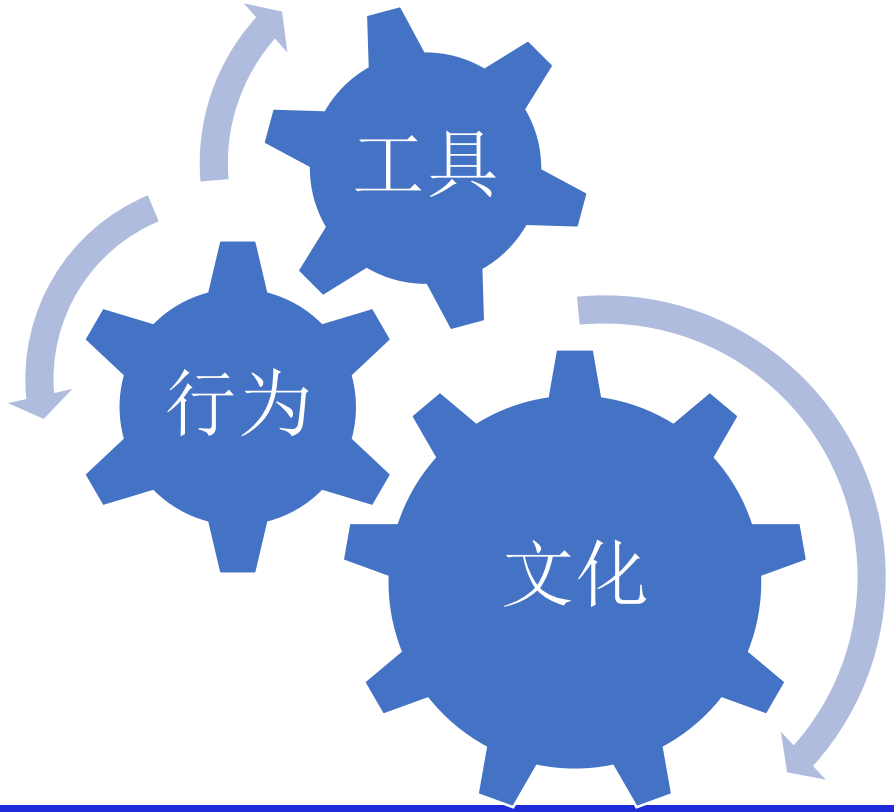


精益、敏捷和DevOps



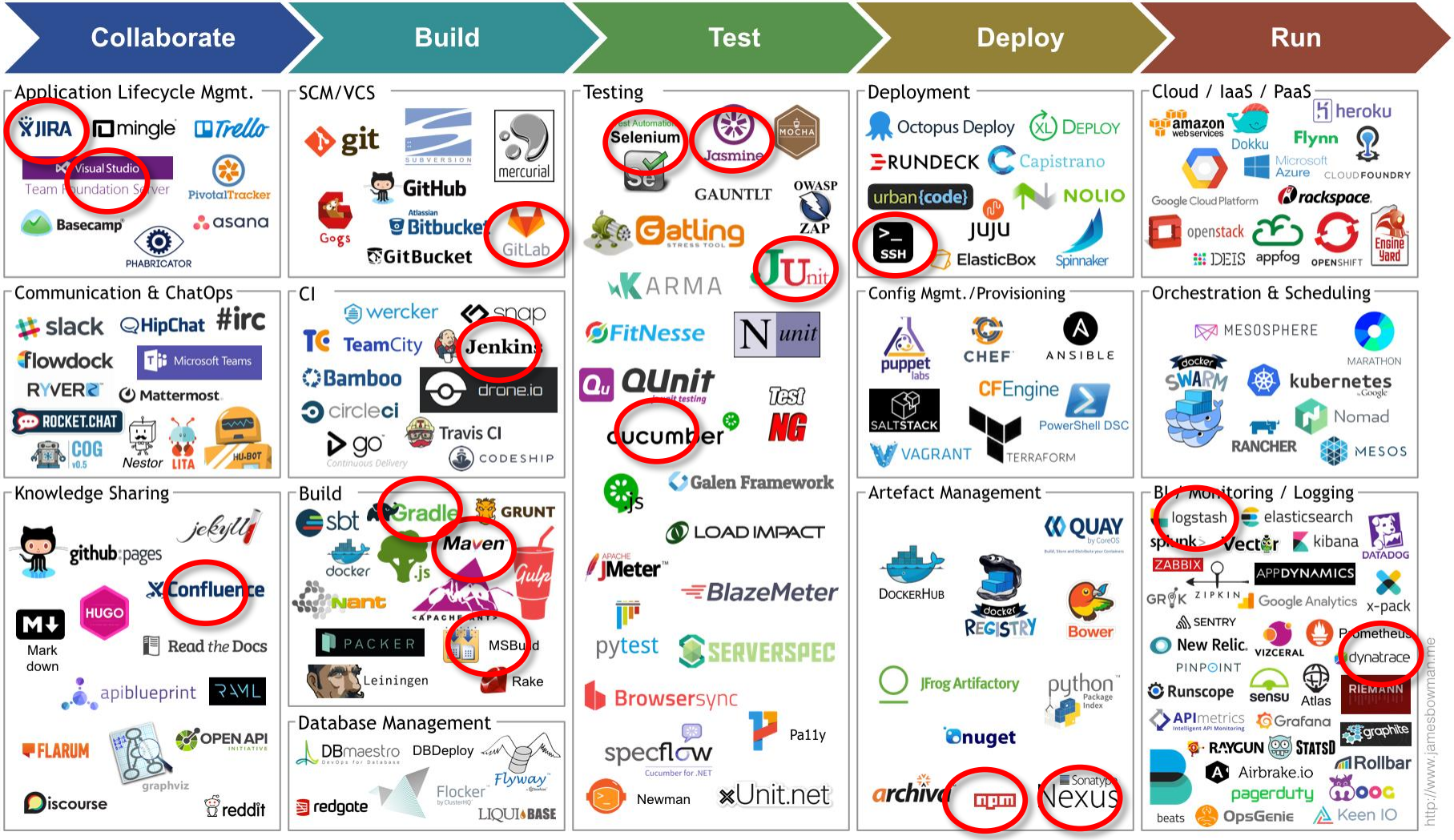
器用与DevOps

DOIS



https://www.felixglobal.com/culture-drives-behavior-behavior-drives-culture/

目录



自主仪表盘

DOIS

Jenkins

持续集成

需求/知识 → 源代码 → 构建 → 静态检查 → 单元测试 → 功能测试 → 制品 → 部署 → 监控

JIRA

SUBVERSION

Maven

sonarqube

JUnit

Selenium

Nexus

自主部署与监控

Visual Studio Team Foundation Server

MSBuild

checkstyle

Jasmine

QuickTest Professional

npm

Confluence

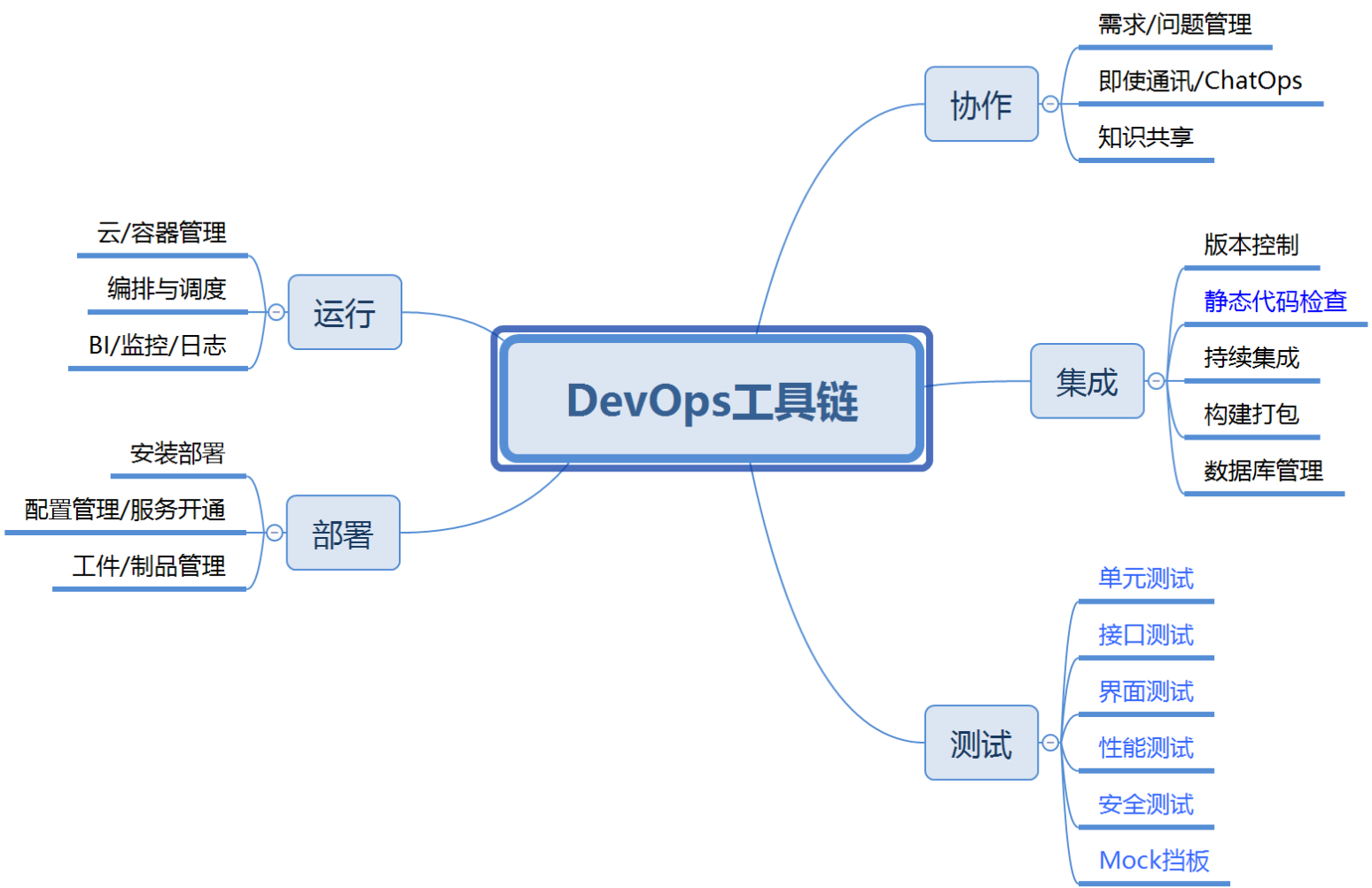
GitLab

Gradle

cucumber

Build Forge Rational. software

dynatrace


DevOps工具链
协作
需求/问题管理
即使通讯/ChatOps
知识共享
集成
版本控制
静态代码检查
持续集成
构建打包
数据库管理
测试
单元测试
接口测试
界面测试
性能测试
安全测试
Mock挡板
部署
安装部署
配置管理/服务开通
工件/制品管理
运行
云/容器管理
编排与调度
BI/监控/日志

目录

自主仪表盘

DOIS

Jenkins

持续集成

需求/知识 | 源代码 | 构建 | 静态检查 | 单元测试 | 功能测试 | 制品 | 部署 | 监控

JIRA

SUBVERSION

Maven

sonarqube

JUnit

Selenium

Nexus

自主部署与监控

Visual Studio Team Foundation Server

Confluence

GitLab

MSBuild

checkstyle

Jasmine

QuickTest Professional

npm

Build Forge

Rational. software

dynatrace

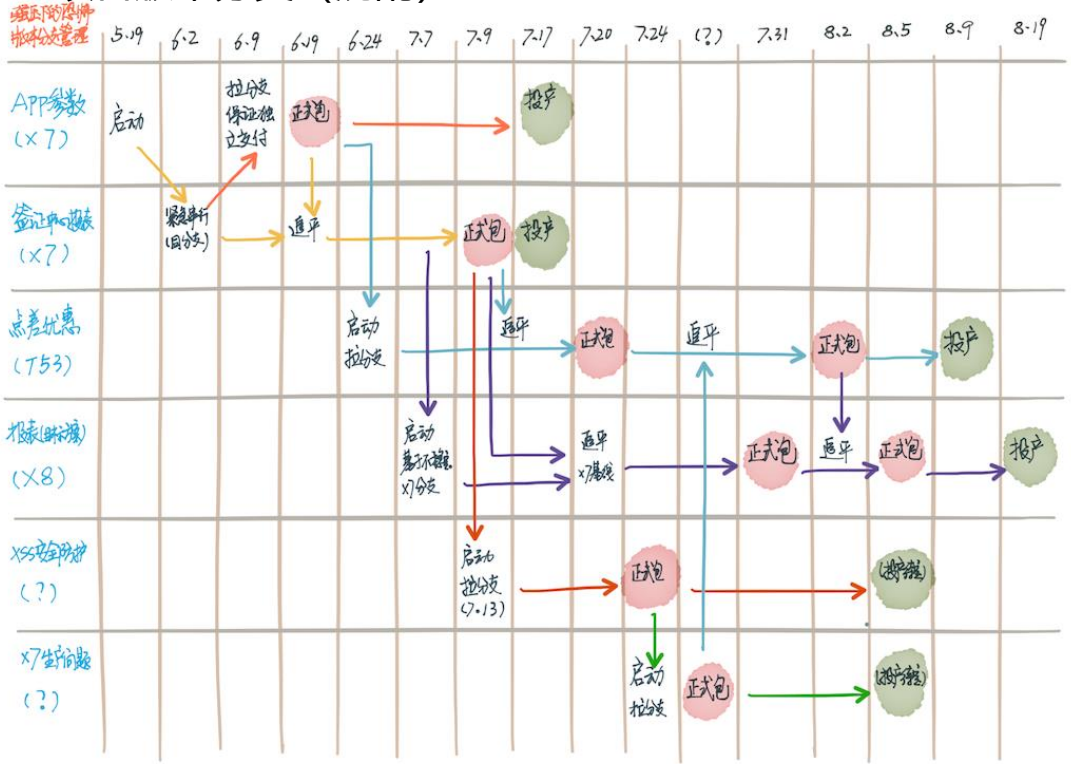
Gradle

cucumber

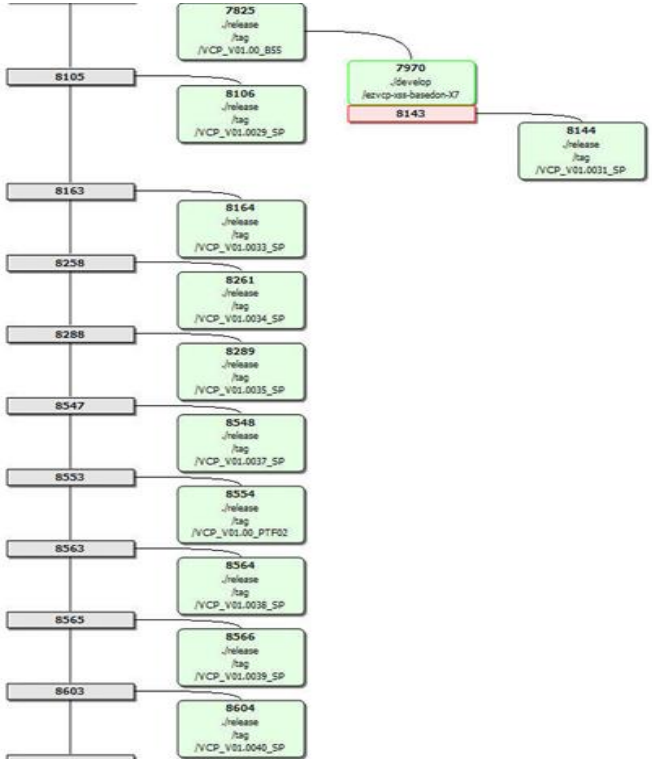
在软件开发中追平就是浪费

DOIS

项目版本分支（混乱）



项目版本分支（清爽）



何为主干开发(TBD)/Git or SVN

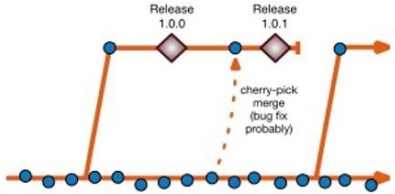
DOIS

理想的主干开发：

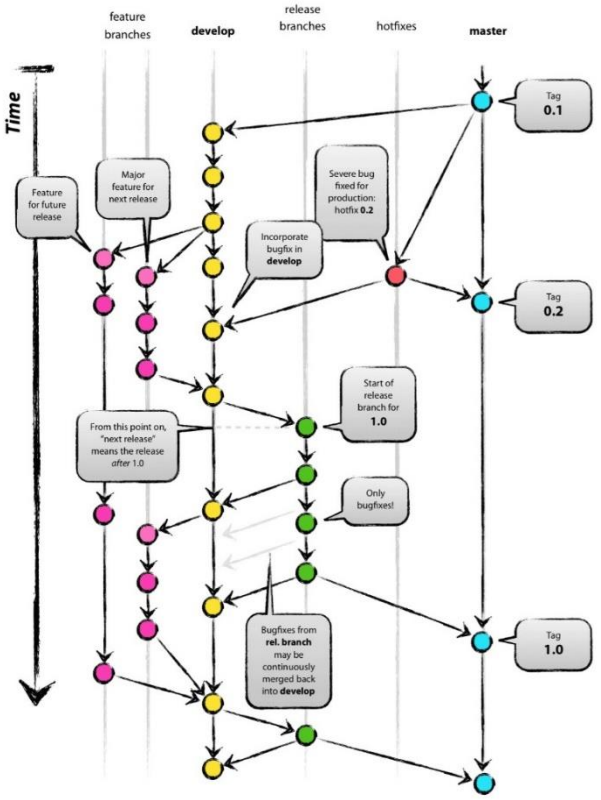
- 所有代码都直接提交到主干(pull request?)
- 主干构建稳定，随时可发布

现实的主干开发：

- 所有代码都直接提交到主干
- 主干构建稳定，随时可拉分支
- 分支存在周期不超过一个迭代



主干开发，分支发布

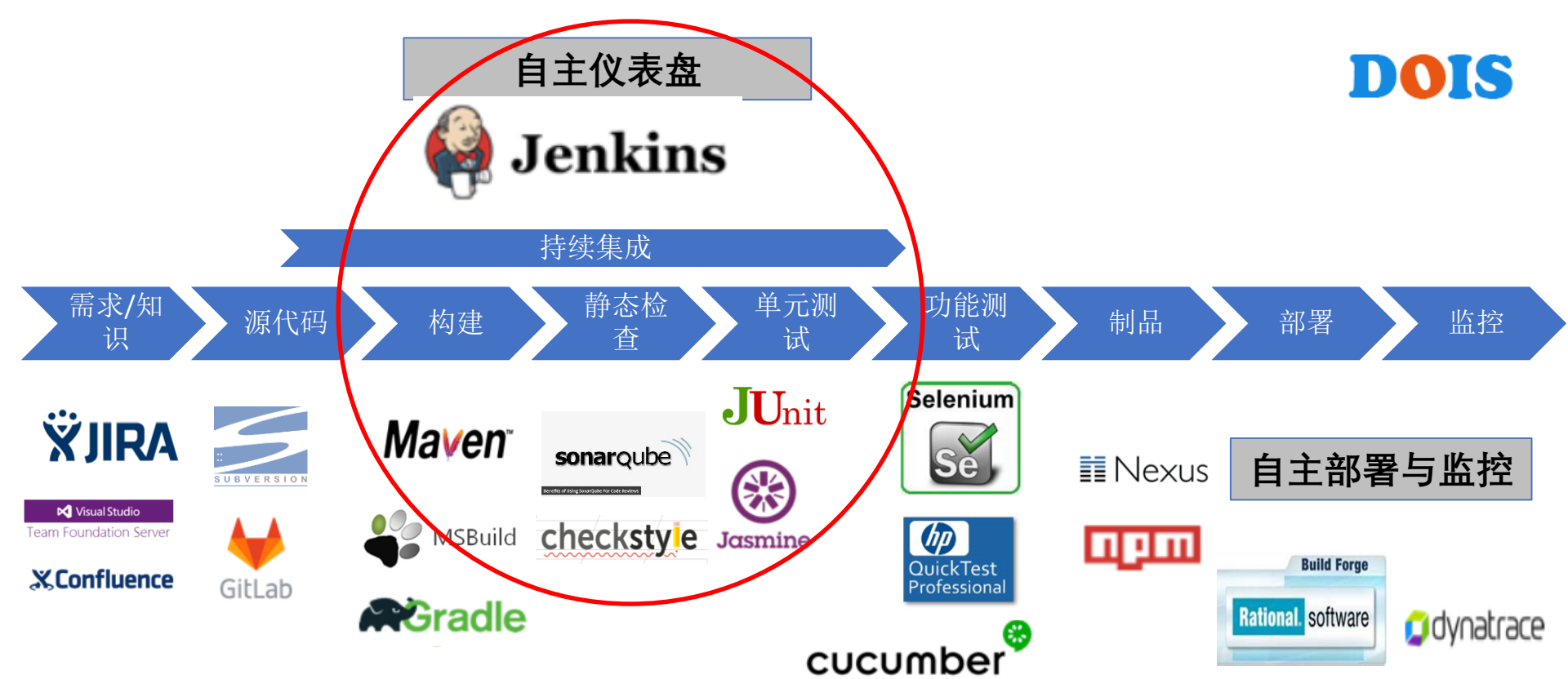


不良效应是什么？困难是什么？

主干开发(TBD)，想说爱你不容易：

DOIS

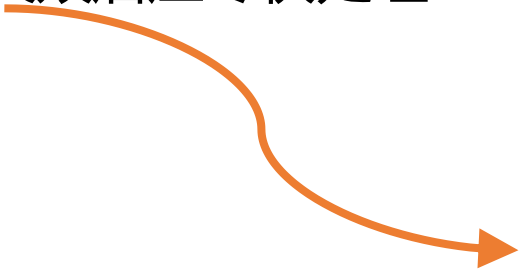
- **足够短的迭代周期（两周，可投产版本）**
- **足够端的投产前置周期（三个工作日）**
  - **无单独的功能测试阶段**
  - **投产轻量化管理**
- **不顾一切保证需求串行**
  - **按需申请生产任务**
  - **任务后排期（报备制）**


自主仪表盘
Jenkins
持续集成
需求/知识
源代码
构建
静态检查
单元测试
功能测试
制品
部署
监控
JIRA
SUBVERSION
Maven
sonarqube
JUnit
Selenium
Nexus
自主部署与监控
Visual Studio
Team Foundation Server
Confluence
GitLab
MSBuild
checkstyle
Jasmine
QuickTest Professional
npm
Build Forge
Rational. software
dynatrace
Gradle
cucumber

何为持续集成

DOIS

- 开发人员每天至少提交一次代码
- 代码提交后立即触发构建和测试
- 构建和测试失败后应尽快处理



我们的CI纪录

执行七部提交法

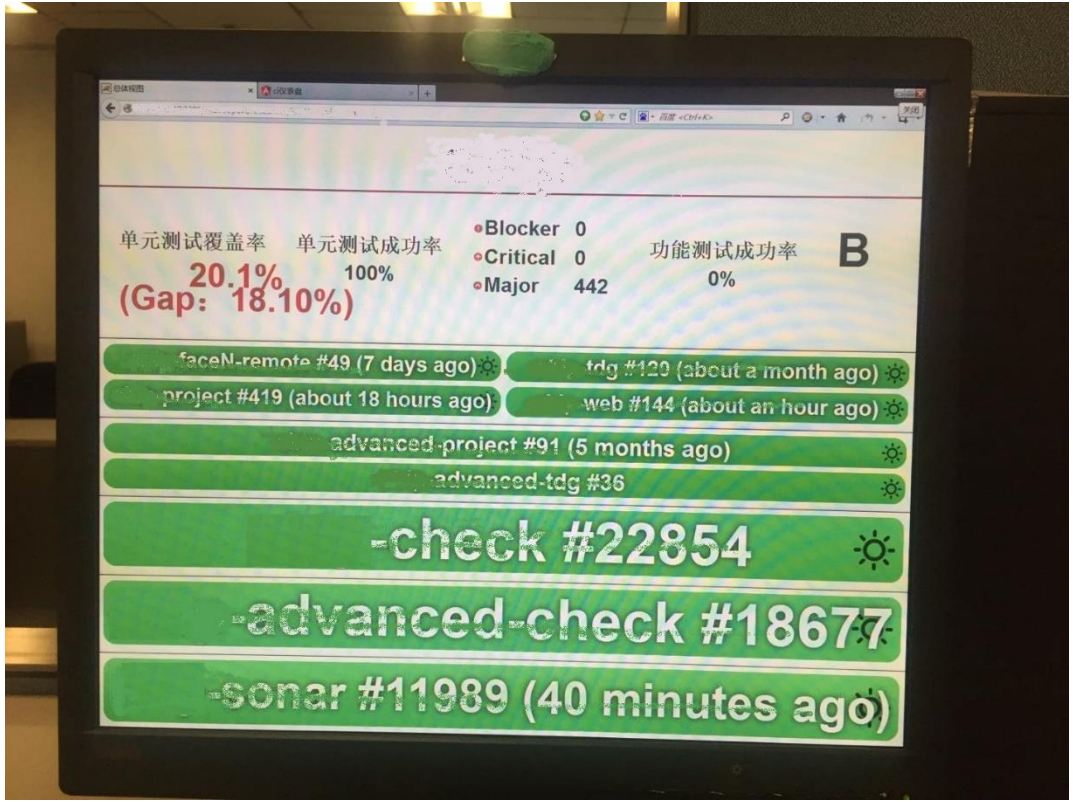
提交完整代码集

本模块 UT 和 CT 都通过

物理仪表盘不变红，变红不下班

代码中严禁大段注释的代码和@Ignore

我们的仪表盘 物理仪表盘



思考题：
物理仪表盘有什么用？

物理仪表盘变红不下班的故事

DOIS

2017.8.24 22:07pm
@西安

我们的仪表盘 电子仪表盘

DOIS

持续集成——软件质量改进和风险降低之道

CI目标：单元测试成功率100%，覆盖率50%+(现有指标不下降)。功能测试成功率100%，功能测试案例数15+个(覆盖主业务流程)；问题数 Blocker 0，Critical 0，Major 尽量少。

产品整体情况(后台) 产品整体情况(前台)

产品名称	单元测试成功率	单元测试覆盖率	代码复杂度	代码重复度	功能测试成功率	代码行数	Sonar问题数	安全问题数	性能波动	构建状态	产品状态	活跃分支
[illegible]	2376(100.00%)	57.63%	2.72	2%	-----	71520	1130(0, 0, 1130)	[illegible]	-----	SUCCESS	B	1/1
[illegible]	853(100.00%)	34.04%	1.32	3%	-----	93064	843(0, 18, 825)	[illegible]	-----	SUCCESS	B	2/2
[illegible]	686(100.00%)	19.61%	1.47	32%	-----	35497	451(0, 0, 451)	[illegible]	-----	SUCCESS	B	0/1
[illegible]	0(0.00%)	0.00%	1.70	17%	-----	10579	158(0, 6, 152)	[illegible]	-----	SUCCESS	C	0/1
[illegible]	233(100.00%)	21.98%	3.89	5%	-----	15899	206(0, 0, 206)	[illegible]	-----	FAILURE	C	0/6

产品状态说明

产品状态总分为100分，其中代码质量、单元测试、jenkins最近五次构建和自动化测试各占1/4的比重；A等级：[100, 70]；B等级：(70,50]；C等级:(50,0]

操作手册和支持文档（点击可直接下载，用户名为svn账号）

CI规范及要求 项目组测试环境列表 CI仪表盘计算方法 敏捷项目轻量级CI工具 CI仪表盘-代码覆盖率算法说明

我们的仪表盘 电子仪表盘

DOIS

sonar信息 | 自动化功能测试信息 | 版本提交信息 | 构建信息 | 问题数趋势图 | 代码覆盖率趋势图 | 安全扫描 | 性能诊断 | 版本信息

sonar服务器(22.11.140.80:9000) 代码质量报表

模块名称	阻断问题	严重问题	主要问题	次要问题	一般问题	测试成功率	代码行数	测试覆盖率	复杂度(方法)	重复度	技术债务(人天)
[illegible]	0	0	8	31	18	100%	1524	82.7%	5.8	0%	0
[illegible]usiness-service]	0	0	16	67	9	0%	546	0%	0.8	3.3%	0
[illegible]common	0	0	17	11	19	100%	903	39%	4.9	0%	0
[illegible]meeting-service	0	0	82	130	57	100%	11855	79.5%	4.5	2.1%	0
[illegible]-meeting-service]	0	0	136	243	121	0%	4254	0%	0.4	1%	0
[illegible]-online-service	0	0	22	42	17	100%	2326	77.6%	6.1	1%	0
[illegible]-online-service]	0	0	36	21	29	0%	809	0%	0.9	1.4%	0
[illegible]-platform-app	0	0	184	236	119	100%	11485	50.6%	4.9	3.6%	0
[illegible]-platform-dao	0	0	284	970	26	100%	4977	0%	0.1	0.2%	0
[illegible]-platform-portal	0	0	123	155	200	100%	14429	56.8%	3.8	2%	0
[illegible]-platform-report	0	0	17	25	7	100%	1411	67.2%	3	0%	0

我们的仪表盘 电子仪表盘

DOIS

sonar信息 | 自动化功能测试信息 | 版本提交信息 | 构建信息 | 问题数趋势图 | 代码覆盖率趋势图 | 安全扫描 | 性能诊断 | 版本信息

今日提交代码人员统计

账号	姓名	提交次数	代码分支	最后一次提交时间	历史记录统计
[illegible]	[illegible]	11	develop	2018-06-05 15:43:52	查看
[illegible]	[illegible]	7	develop	2018-06-05 15:24:45	查看
[illegible]	[illegible]	3	develop	2018-06-05 14:24:15	查看
[illegible]	[illegible]	2	develop	2018-06-05 14:15:30	查看
[illegible]	[illegible]	2	develop	2018-06-05 14:29:45	查看
[illegible]	[illegible]	2	develop	2018-06-05 14:47:51	查看
[illegible]	[illegible]	1	develop	2018-06-05 13:50:12	查看
[illegible]	[illegible]	1	develop	2018-06-05 15:29:09	查看
[illegible]	[illegible]	1	develop	2018-06-05 09:31:10	查看
[illegible]	[illegible]	1	develop	2018-06-05 10:16:32	查看
[illegible]	[illegible]	1	develop	2018-06-05 16:06:11	查看
[illegible]	[illegible]	1	develop	2018-06-05 14:56:05	查看

我们的仪表盘 电子仪表盘

自主仪表盘

DOIS

Jenkins

持续集成

需求/知识 → 源代码 → 构建 → 静态检查 → 单元测试 → 功能测试 → 制品 → 部署 → 监控

JIRA

SUBVERSION

Maven

sonarqube

JUnit

Selenium

Nexus

自主部署与监控

Visual Studio Team Foundation Server

MSBuild

checkstyle

Jasmine

QuickTest Professional

npm

Confluence

GitLab

Gradle

cucumber

Build Forge

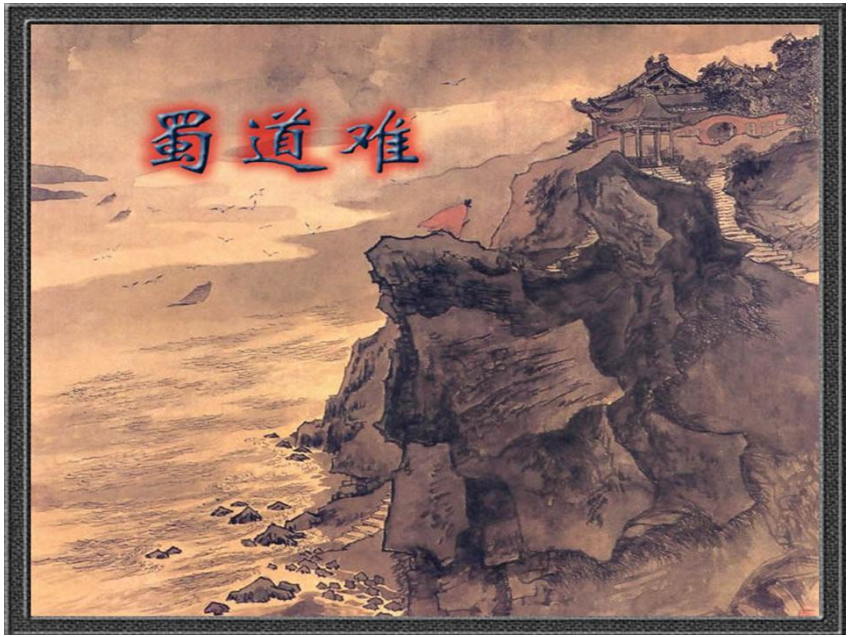
Rational. software

dynatrace

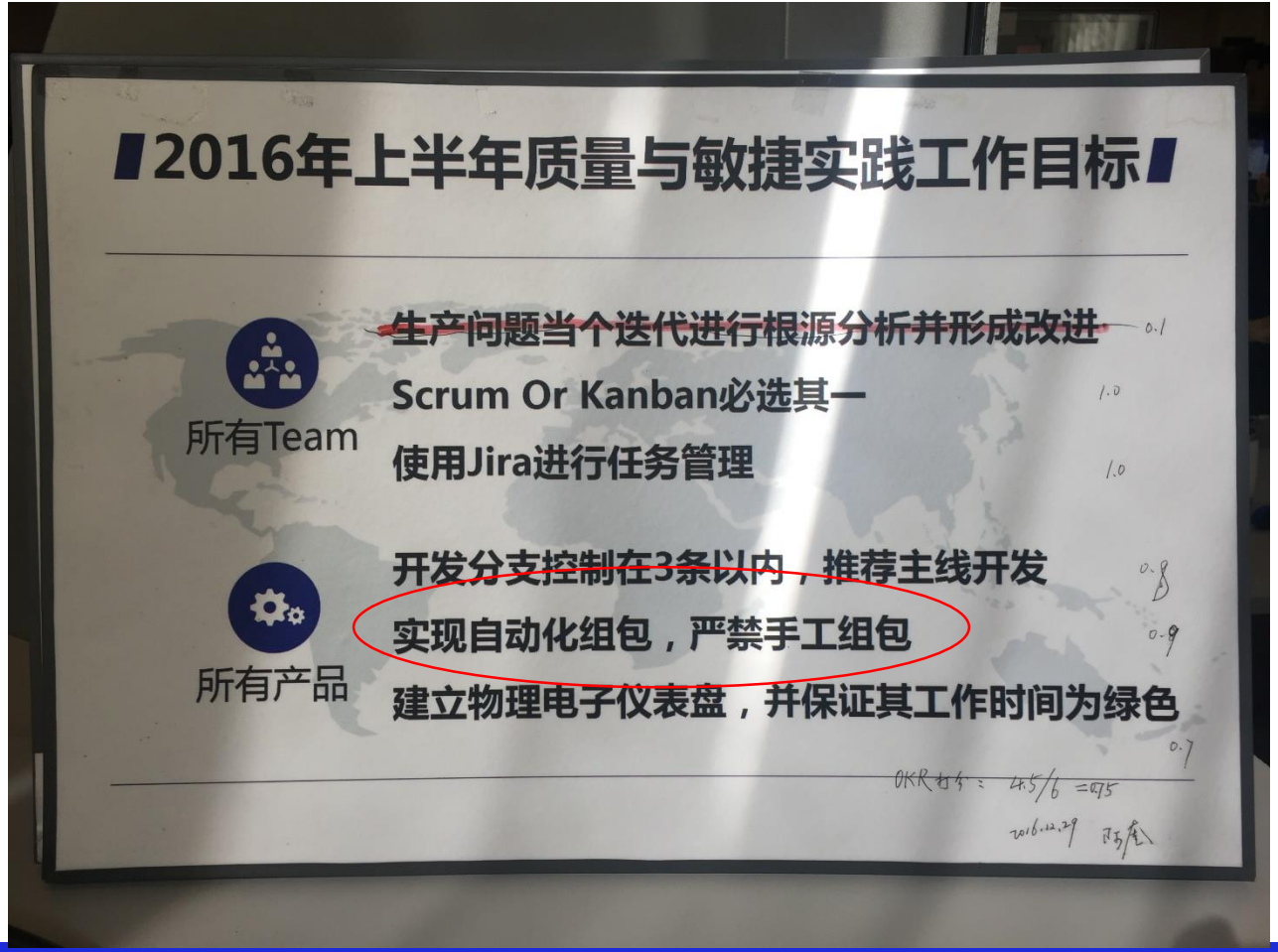
自动化部署之难

DOIS

- 生产环境权限管理融合难
- 生产环境配置参数收集难
- 生产环境资源变更联动难
- 增量、全量的支持统一难
- 自动组包标准贯彻落实难



图片来自网络：http://pic.kekenet.com/2015/0511/64971431307945.jpg



DOIS

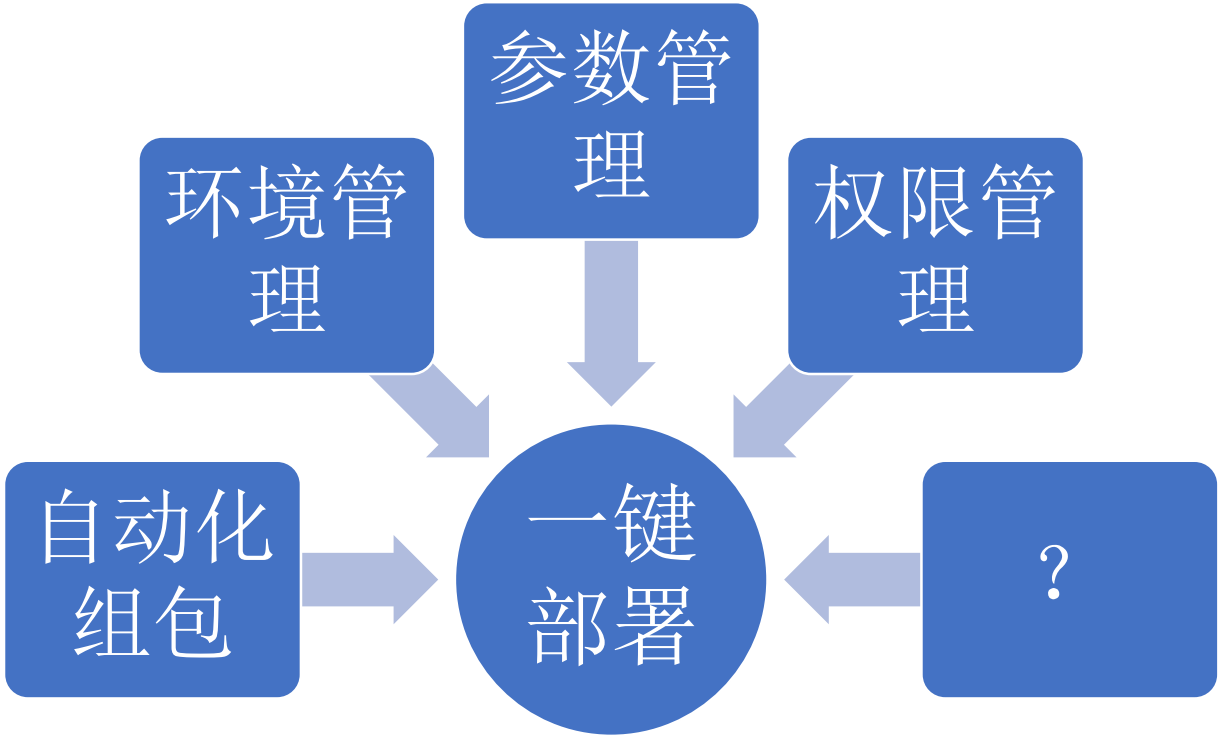
手工组包的问题：

- 重复性工作
  - 次数多
  - 频率高
- 过程繁琐
  - 步骤多
  - 细节多

易出错

即脆弱

自动化部署/一键部署



目录

自动化功能测试的苦难

DOIS

- 2016年多个产品集中编写了将近1000个面向UI的自动化功能测试案例（Robot Framework），半年后，**全部废弃**，没有再被运行过。
- 某产品编写的大量面向UI的自动化功能测试案例，全部运行一次一般需要**一到两周**才能完成，有2-3名测试开发人员**专职负责**运行和编写，以及更新这些案例。

反思：


发布频率
每100天
发布一次
每10天
发布一次
每日发布
每天发布
10次
每天发布
100次
分支模型
主干开发
为发布创建分支
拉动提交
请求到发布分支
在分支上开发
合并到发布分支，然后发布
后再次创建分支
在主干开发
从主干发布
测试
由单独的质量保证部门
负责功能测试自动化
在部署流水线中组织快递、完备的
自动化单元测试和功能测试
由开发人员和测试人员共同维护自动化功能测试
架构
所有东西 一起部署
每个产品/每种web资源单独一个包
严格的SOA架构，向前和向后兼容
发布
灰度发布、蓝绿部署、金丝雀发布
开发自己将变更部署到生产
功能版本，版本火车
基础设施
可以通过收版本控制管理的脚本
来准备类生产环境
异构，运维部门
统一管理
标准化的PaaS或者IaaS、Caas
平台来交付
数据库
设计中考虑应用对数据库的前向、后向
兼容性（采用扩展或契约）
采用增量脚本变更数据库
并对回滚进行演练
手动迁移

有纪律的增加自动化功能测试案例：

DOIS

增加的自动化功能测试案例，必须可以

- 每天至少运行一次
- 运行预期应该是100%成功的
- 如果没有成功，一定是由于应用问题，而不是因为环境、数据等非应用问题

单元测试：FIRST原则 → 功能测试：SIR原则

寄语

DOIS

填空题：

工具很重要，

_____更重要？

。。。。。。

DOIS

Thanks

DevOps 时代社区 **荣誉出品**

AgileDoer

DOIS

想第一时间看到高效运维社区的新动态吗?