DevOps调研

# 相关工具列表

|  |  |  |  |
| --- | --- | --- | --- |
| **流程/阶段** | **工具** | **描述** | **Jenkins** |
| 流程，项目管理/缺陷，  问题追踪 | JIRA, 禅道, | JIRA, 禅道 | 不适用 |
| 缺陷追踪 | Bugzilla | 开源 | 不适用 |
| 知识管理/分享 | confluence | 不开源，知识管理的工具，通过它可以实现团队成员之间的协作和知识共享。 | 不适用 |
| 源代码控制管理（SCM） | git | 不支持私有仓库，如果搭建私有仓库需要付费 | Jenkins 可以集成 |
| 源代码控制管理（SCM） | SVN | 严格的权限管理控制和档案仓库，分支 | Jenkins 可以集成 |
| 构建工具 （build） | cmake | c/c++构建 | Jenkins 可以集成 |
| 构建工具 （build） | maven， gradle | java构建 | Jenkins 可以集成 |
| 构建工具 （build） | npm | nodejs构建 | Jenkins 可以集成 |
| 静态测试 | SonarQube | 开源, 可以检测代码量、复杂度、代码增量、测试覆盖率, 重复代码， 潜在bug， 代码规范，安全性漏洞等问题， 通过web UI展示 | Jenkins 可以集成 |
| 静态测试 | Checkmarx | 收费 | Jenkins 可以集成 |
| 动态测试 | Junit | 开源的单元测试框架，Jenkins pipeline可以通过MVN test在pre-build阶段集成 | Jenkins 可以集成 |
| 动态测试 | Jacoco | 开源，代码覆盖率, 单元测试中代码执行量与代码总量之间的比率。 | Jenkins 可以集成 |
| 动态测试 | Selenium | 开源自动化测试框架 | Jenkins 可以集成 |
| 仓库管理 | Nexus | 分为开源版和专业版，其中开源版足以满足大部分 Maven 用户的需求 | Jenkins 可以集成 |
| 自动化集成工具 | Jenkins | 开源，多操作系统支持，java环境，1300+插件，几乎可以集成任何工具和服务 | Jenkins |
| 持续部署 | Ansible | 开源，无需客户端，1000+插件 | Jenkins 可以集成 |
| 日志 | Splunk | 不开源 | 不适用 |
| 日志 | ELK | 开源，三个组件全部开源，Logstash收集日志，Elasticsearch日志处理，Kibana日志展示 | 不适用 |
| 监控 | AppDynamics | 有免费的轻量级版本 | 不适用 |
| 监控 | Zabbix | 开源，企业级监控平台，硬件监控，系统监控，java监控，网络，应用，数据库，web监控等 | 不适用 |
| 容器 | docker， docker compose | 开源 | Jenkins 可以集成 |
| 容器 | Kubernetes | 开源 | Jenkins 可以集成 |

# DevOps实现维度

### 流程改进 -

### 工具自动化 - 可扩展，减少错误发生

### 平台及环境 - 搭建灵活，可扩展，可配置，弹性的平台与环境

### 文化 - 信任，沟通，协作的文化氛围

# DevOps流程

## （一）现状 – 公司现状的充分了解

（空，修订中）

## （二）参考 - 流程参考

### 代码开发阶段/版本控制

1. 开发人员在他们的IDEA中使用代码检测插件者运行分析本地代码。
2. 开发人员A针对某次修改/功能实现发起代码提交请求。
3. 开发人员B评审检查此次代码修改, 检查业务逻辑和代码实现，更早的发现问题。
4. 如果B发现代码问题，则通知A修改重新提交，如果没有问题，merge到相应的开发分支。
5. 开发B相当于开发A的结对编程伙伴，对此开发修改的模块的业务和代码也能同样熟悉，在开发A请假缺席的情况下可以快速充当backup。

### 构建

1. 点击Jenkins pipeline触发流水线。（也可以通过hook设置每次提交自动触发构建，或者设置自动每天晚上构建。）
2. Jenkins pipeline调用Sonar Scanner对项目代码进行扫描分析，并且将报告持久化并且可以通过UI展示，可以设置一些rules。 （Sonar自集成了数据库做持久化。）
3. Jenkins pipeline调用动态测试，按需要集成Junit，Jacoco进行动态测试。
4. 前置的各项检测和测试如果符合要求，Jenkins pipeline开始使用对应的工具构建打包。 （c/c++ -> Cmake，Java -> maven， Node.js -> npm。）
5. 将打好的包打上tag标签并推送到包管理工具Nexus。（没有Nexus的话，指定文件路径也可以。）

### 部署

1. Jenkins pipeline将包部署到指定测试环境 。（也可以使用Ansible，视情况而定。）
2. Jenkins pipeline调用Selenium进行动态测试并生成测试报告。
3. 开发和QA在测试环境验证打好的包，如果需要，进行手动的回归验收测试。
4. Jenkins pipeline从Nexus拉取相应tag的release的包并推送到生产环境完成部署。 （也可以使用Ansible。）

### 日志和监控

1. 使用Zabbix，AppDynamic进行服务器级别和应用级别的监控。
2. 所有日志集成到Splunk或者ELK，方便业务追踪，问题定位和质量分析。

### 注意事项

1. 生产环境必须使用测试环境最终验证过的包，即相同tag的包。
2. 保证测试环境和生产环境的一致性。（Ansible的任务编排可以配置管理，实现不变的基础设施 Iac，也可以引入Docker容器化实现。）
3. 尽量避免开发和减少运维人员连接到测试和生产环境手动执行操作，即使连接，保障账号权限最小化。
4. 践行持续，向敏捷靠拢，保证每一次代码提交的质量和小颗粒度的持续集成部署，不要攒一波再进行某一项任务，摆脱瀑布模型的方式。
5. 保存每次release的分支，方便问题回溯和回滚。

## （三）目标 – 定义明确的DevOps目标

注：基于公司现状与其他公司的DevOps实践，设定目标，要实现什么样的DevOps目标

## （四）计划 - 如何实现目标

（空，修订中）