

《开源软件通识基础》

开源的 DevOps 流程

华东师范大学 张琰彬

DevOps

从敏捷到DevOps

DevOps 和开源的碰撞

软件开发流程的发展看敏捷

1960年
软件开
发流程
定义

1969年，
结构化
编程

1980年
结构化
分析与
设计方
法

1990年
面向对
象程序
设计，
起步于
1960年

1991年
快速应
用程序
开发

1995年
Scrum

1998年
团队软
件过程
IBM维
护的统
一软件
开发过
程

1999年
极限编
程

2001年
敏捷宣
言

2005年
敏捷统
一过程
(Agile
Unified
Process,
AUP)

2010年
基于大
规模的
敏捷框
架
(SAFe)

轻量化的软件开发方法

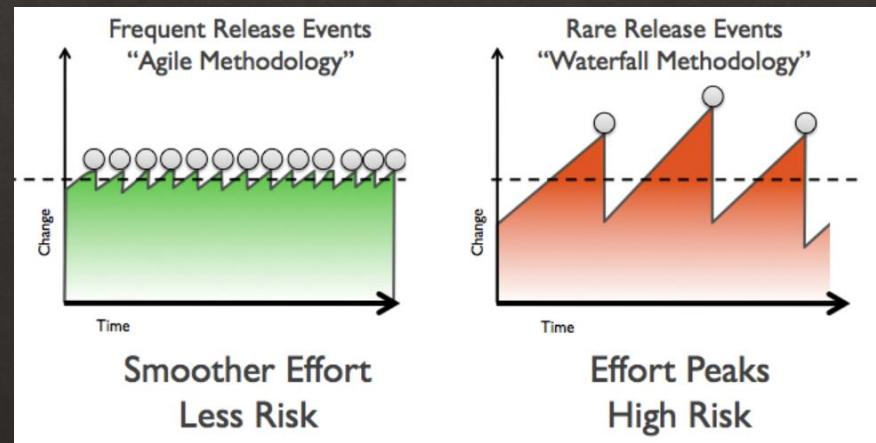
敏捷开发

价值

个体和互动：高于流程和工具。
工作的软件：高于详尽的文档。
客户合作：高于合同谈判。
响应变化：高于遵循计划。

敏捷开发框架

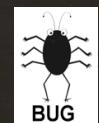
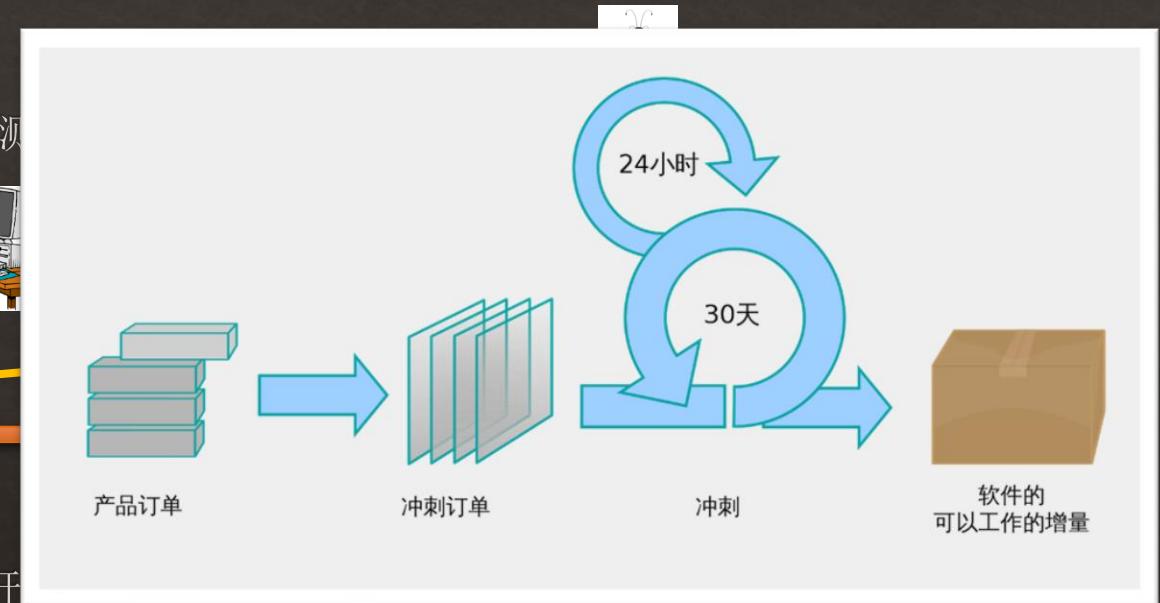
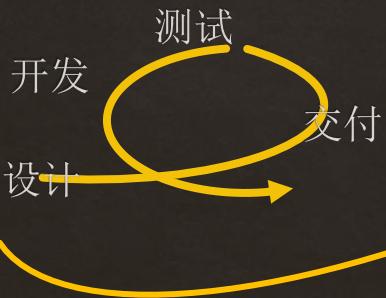
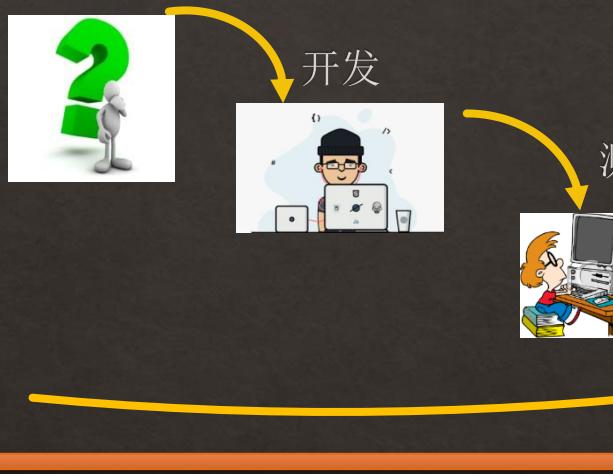
极限编程，精益软件开发（实践）
Scrum Scrum Kanban (管理工作流程)
FDD 功能驱动开发（需求规范和开发）



<https://commons.wikimedia.org/wiki/File:Agile-vs-iterative-flow.jpg>

瀑布WaterFall模型 VS 敏捷开发

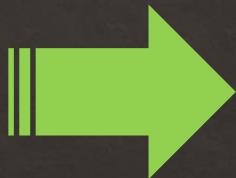
设计



敏捷开发

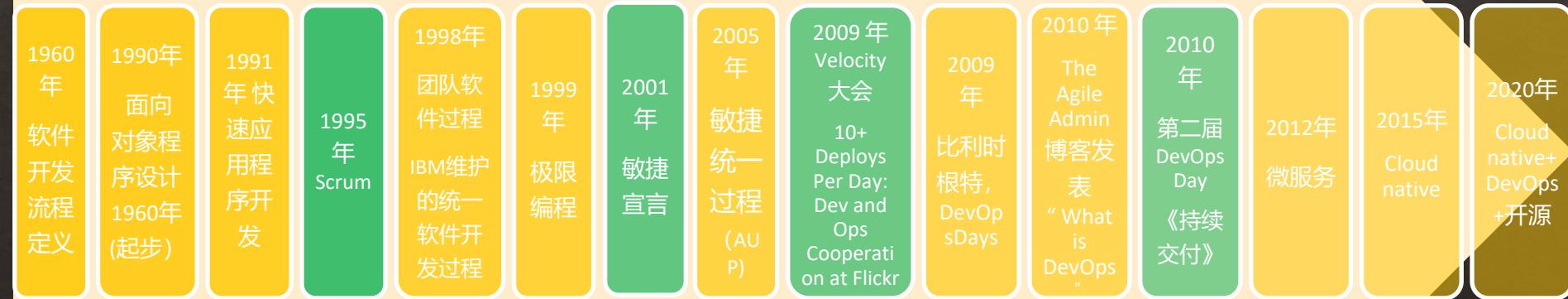
是解决需求方，产品方和开发团队合作的文化和方法
是一种应对快速变化的需求的软件开发能力

敏捷开发



DevOps

敏捷开发 → DevOps



轻量化与细微化开发管理，项目管理

敏捷开发



DevOps

用户+需求方
+产品方



敏捷
开发

开发团队：
开发+测试



DevOps

运维团队：
运维+IT

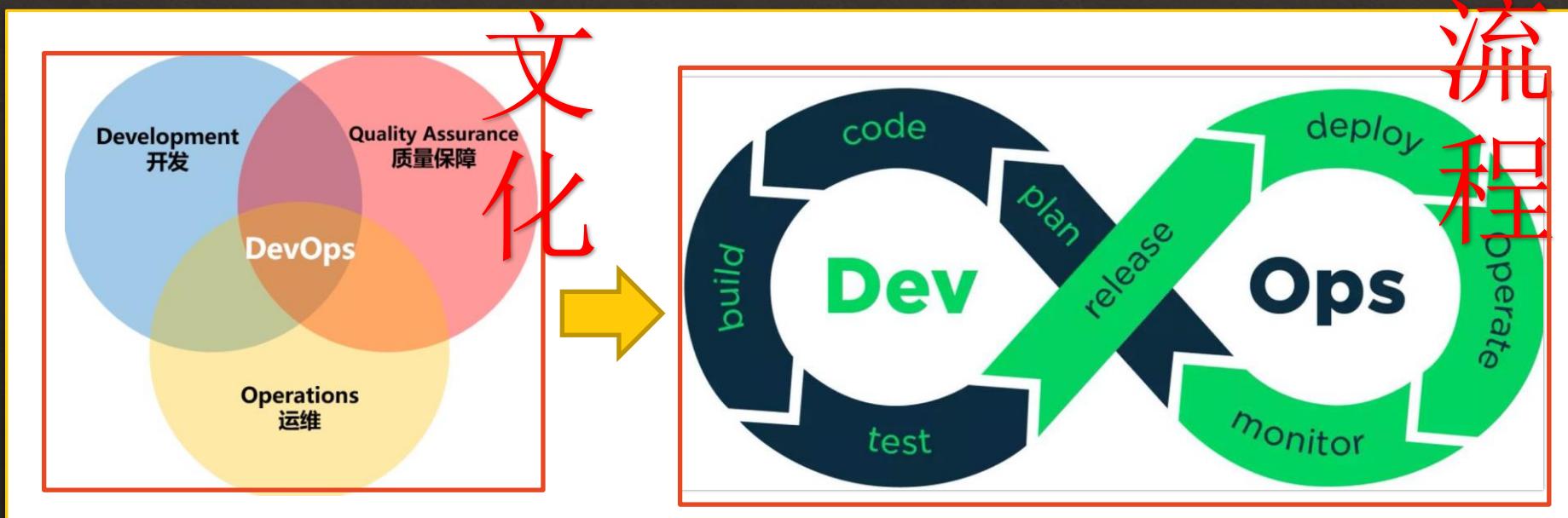


DevOps

DevOps

Development +Operations 重视“软件开发人员（Dev）”和“IT运维技术人员（Ops）”之间沟通合作的文化、运动或惯例。透过自动化“软件交付”和“架构变更”的流程，来使得构建、测试、发布软件能够更加地快捷、频繁和可靠。

<https://zh.wikipedia.org/wiki/DevOps>



DevOps

从敏捷到DevOps

DevOps 和开源的碰撞

DevOps 与 开源 (Open Source) 的碰撞

开源加速DevOps的成功

A periodic table of DevOps tools, similar to the periodic table of elements, showing the relationship between various tools and their categories. The table is color-coded by tool type and includes columns for license (Open-source, Free, Freemium, Paid, Enterprise) and row numbers.

Category	Tool	License	Row
Continuous Integration	Aja (Atlassian Jira Align)	En	1
	Daa (Docker Automate)	Fr	2
	In (Instana)	Pd	3
	Dd (Datadog)	Fr	4
	Ja (JFrog Artifactory)	En	5
	Aws (AWS)	En	6
	Sl (Slack)	En	7
	Mt (Microsoft Teams)	En	8
	Rha (Red Hat Ansible)	Os	9
	Ht (HashiCorp Terraform)	Os	10
Deployment	Dk (Docker)	Os	11
	Rho (Red Hat OpenShift)	Os	12
	Lb (Liquibase)	Os	13
	Dp (Delphix)	Os	14
	Ud (UrbanCode Deploy)	Fm	15
	Ck (CyberArk Conjur)	En	16
	Hv (HashiCorp Vault)	En	17
	Ur (UrbanCode Release)	En	18
	Al (AWS Lambda)	En	19
	Abb (Atlassian Bitbucket)	En	20
Database Management	Sp (Splunk)	Fr	21
	Ad (AppDynamics)	En	22
	Snx (Sonatype Nexus)	En	23
	Az (Azure)	Os	24
	Gc (Google Cloud)	En	25
	Ac (Atlassian Confluence)	En	26
	Ch (Chef)	Os	27
	Acf (AWS CloudFormation)	Os	28
	Ku (Kubernetes)	Os	29
	Ak (Amazon EKS)	En	30
Source Control Management	De (Docker Enterprise)	En	31
	Do (Docker Enterprise)	En	32
	Ha (Harness)	En	33
	Vc (Veracode)	En	34
	Sr (SonarQube)	En	35
	Ff (Micro Focus Fortify SCA)	En	36
	Azf (Azure Functions)	En	37
	Ci (Compuware ISPIW)	En	38
	Gr (Grafana)	Os	39
	EI (Elastic ELK Stack)	Os	40
Testing	Yn (Yarn)	Os	41
	Nu (NuGet)	Os	42
	Os (OpenStack)	Os	43
	Mm (Mattermost)	Os	44
	Sa (Salt)	Os	45
	Hg (HashiCorp Vagrant)	Os	46
	Hp (HashiCorp Packer)	Os	47
	Gk (Google GKE)	Os	48
	Hm (Helm)	Os	49
	Db (Dockerflow)	Os	50
Value Stream Management	Cfd (Cloudbees CD)	En	51
	Acd (AWS CodeDeploy)	En	52
	Sn (Smart)	En	53
	Pbs (PortSwigger Burp Suite)	Fm	54
	Gf (Google Firebase)	En	55
	Cf (Cloud Foundry)	Os	56
	Jn (Jenkins)	En	57
	Azc (Azure DevOps CI/CD)	En	58
	Glc (GitLab CI)	En	59
	Tr (Travis CI)	En	60
Release Management	Cc (CircleCI)	Os	61
	Mv (Meven)	Fm	62
	Ab (Atlassian Bamboo)	Os	63
	Gd (Gradle)	Os	64
	Acb (AWS CodeBuild)	Os	65
	Aj (Atlassian Jira)	En	66
	Bi (BMC Helix ITSM)	En	67
	At (Atlassian Trello)	En	68
	Sw (ServiceNow)	En	69
	Td (TOPdesk)	En	70
Configuration Automation	Tt (Tricentis Tosca)	Fr	71
	Nn (Neotys NeoLoad)	Fr	72
	Se (Selenium)	En	73
	Ju (JUnit)	Fr	74
	Sl (Sauce Labs)	Pd	75
	Ct (Compuware Topaz)	En	76
	Ap (Appium)	Pd	77
	Sq (Squash TM)	En	78
	Cu (Cucumber)	En	79
	Jm (JMeter)	En	80
Containerization	Pa (Parosoft)	En	81
	Dai (Digital.ai)	En	82
	Tp (Tasktop)	En	83
	Pr (Plutora)	En	84
	Gl (GitLab)	En	85
	Pd (Papercut)	Os	86
	Os (OpenShift)	Os	87
	Bi (BMC Helix ITSM)	Os	88
	At (Atlassian Trello)	Os	89
	Sw (ServiceNow)	Os	90
Collaboration	Td (TOPdesk)	Os	91
	Nn (Neotys NeoLoad)	Fr	92
	Se (Selenium)	En	93
	Ju (JUnit)	En	94
	Sl (Sauce Labs)	En	95
	Ct (Compuware Topaz)	En	96
	Ap (Appium)	Pd	97
	Sq (Squash TM)	En	98
	Cu (Cucumber)	En	99
	Jm (JMeter)	En	100
Deployment Automation	Pa (Parosoft)	En	101
	Dai (Digital.ai)	En	102
	Tp (Tasktop)	En	103
	Pr (Plutora)	En	104
	Gl (GitLab)	En	105
	Pd (Papercut)	Os	106
	Os (OpenShift)	Fr	107
	Bi (BMC Helix ITSM)	Fr	108
	At (Atlassian Trello)	En	109
	Sw (ServiceNow)	En	110
Issue Tracking/ITSM	Td (TOPdesk)	En	111
	Nn (Neotys NeoLoad)	En	112
	Se (Selenium)	En	113
	Ju (JUnit)	En	114
	Sl (Sauce Labs)	Pd	115
	Ct (Compuware Topaz)	En	116
	Ap (Appium)	En	117
	Sq (Squash TM)	En	118
	Cu (Cucumber)	En	119
	Jm (JMeter)	En	120
Serverless/PaaS	Pa (Parosoft)	En	121
	Dai (Digital.ai)	En	122
	Tp (Tasktop)	En	123
	Pr (Plutora)	En	124
	Gl (GitLab)	En	125
	Pd (Papercut)	Os	126
	Os (OpenShift)	Fr	127
	Bi (BMC Helix ITSM)	Fr	128
	At (Atlassian Trello)	En	129
	Sw (ServiceNow)	En	130
Testing	Td (TOPdesk)	En	131
	Nn (Neotys NeoLoad)	En	132
	Se (Selenium)	En	133
	Ju (JUnit)	En	134
	Sl (Sauce Labs)	Pd	135
	Ct (Compuware Topaz)	En	136
	Ap (Appium)	En	137
	Sq (Squash TM)	En	138
	Cu (Cucumber)	En	139
	Jm (JMeter)	En	140
Deployment	Pa (Parosoft)	En	141
	Dai (Digital.ai)	En	142
	Tp (Tasktop)	En	143
	Pr (Plutora)	En	144
	Gl (GitLab)	En	145
	Pd (Papercut)	Os	146
	Os (OpenShift)	Fr	147
	Bi (BMC Helix ITSM)	Fr	148
	At (Atlassian Trello)	En	149
	Sw (ServiceNow)	En	150
Continuous Integration	Td (TOPdesk)	En	151
	Nn (Neotys NeoLoad)	En	152
	Se (Selenium)	En	153
	Ju (JUnit)	En	154
	Sl (Sauce Labs)	Pd	155
	Ct (Compuware Topaz)	En	156
	Ap (Appium)	En	157
	Sq (Squash TM)	En	158
	Cu (Cucumber)	En	159
	Jm (JMeter)	En	160

DevOps 与 开源（Open Source）的碰撞

开源需要DevOps

DevOps

方法：自动化流程

目标：
快捷、频繁和可靠，
缩短开发周期，
业务目标保持一致

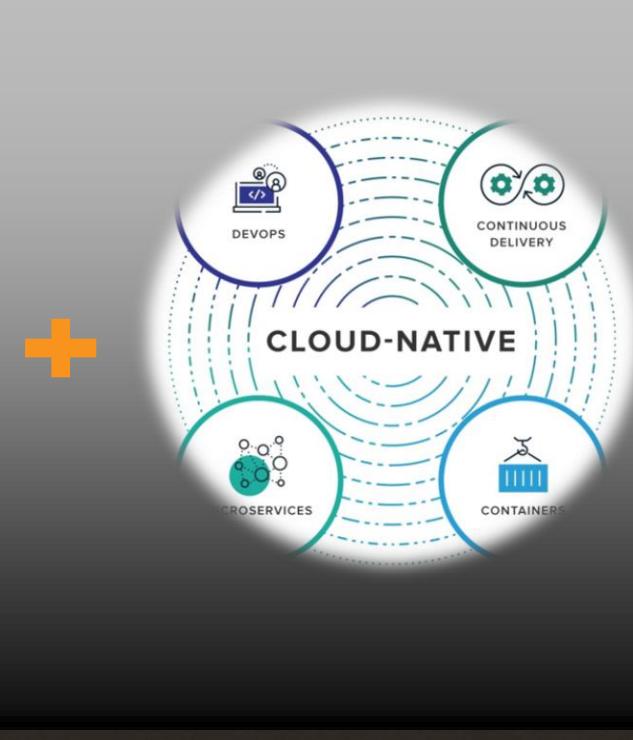
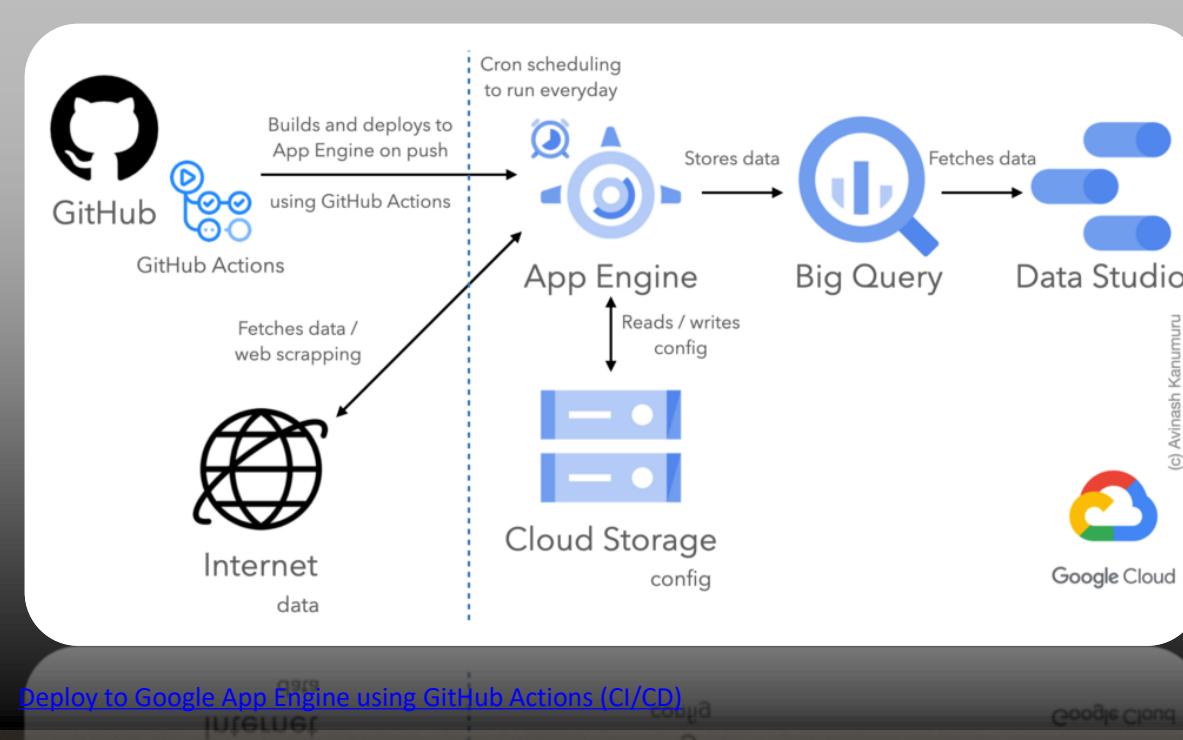
Open Source

参与者：
任何人、任何地方、
任何背景水平的人

合作
提升
反馈
交付

DevOps 与 开源 (Open Source) 的碰撞

开源促使DevOps新形态



Q&A

《开源软件通识基础》

开源项目代码管理

华东师范大学 张琰彬

开源项目代码管理

开源项目面临质量问题

代码的持续集成测试管理体系

传统软件测试体系的变革和发展

基于开源的代码质量管理体系

开源项目代码管理

开源项目面临质量问题

代码的持续集成测试管理体系

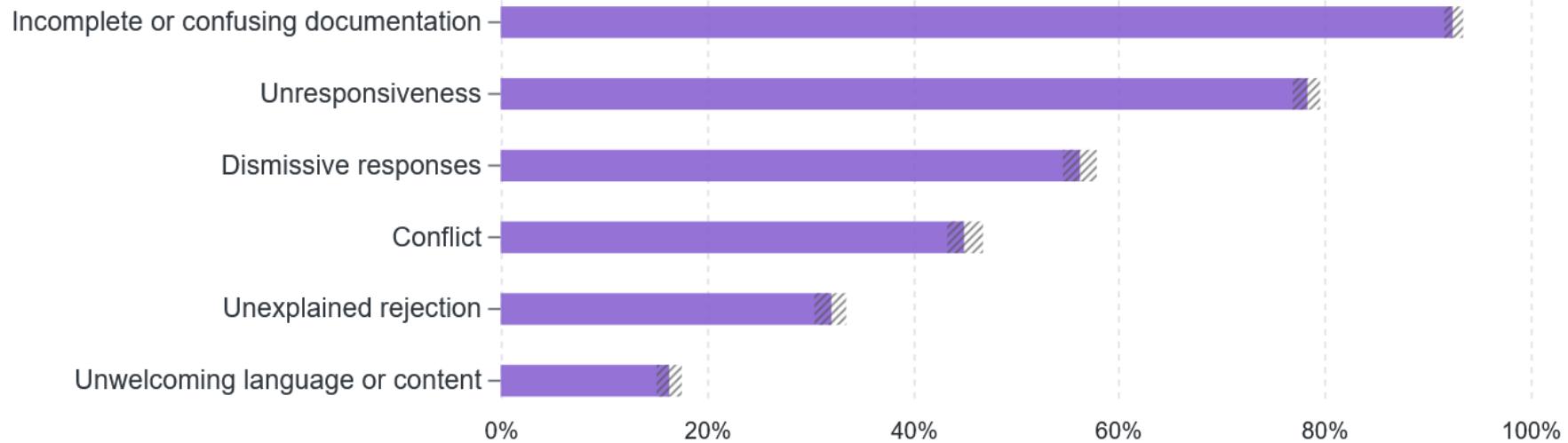
传统软件测试体系的变革和发展

基于开源的代码质量管理体系

什么是一个好的开源项目

Fig1. - Problems encountered in open source

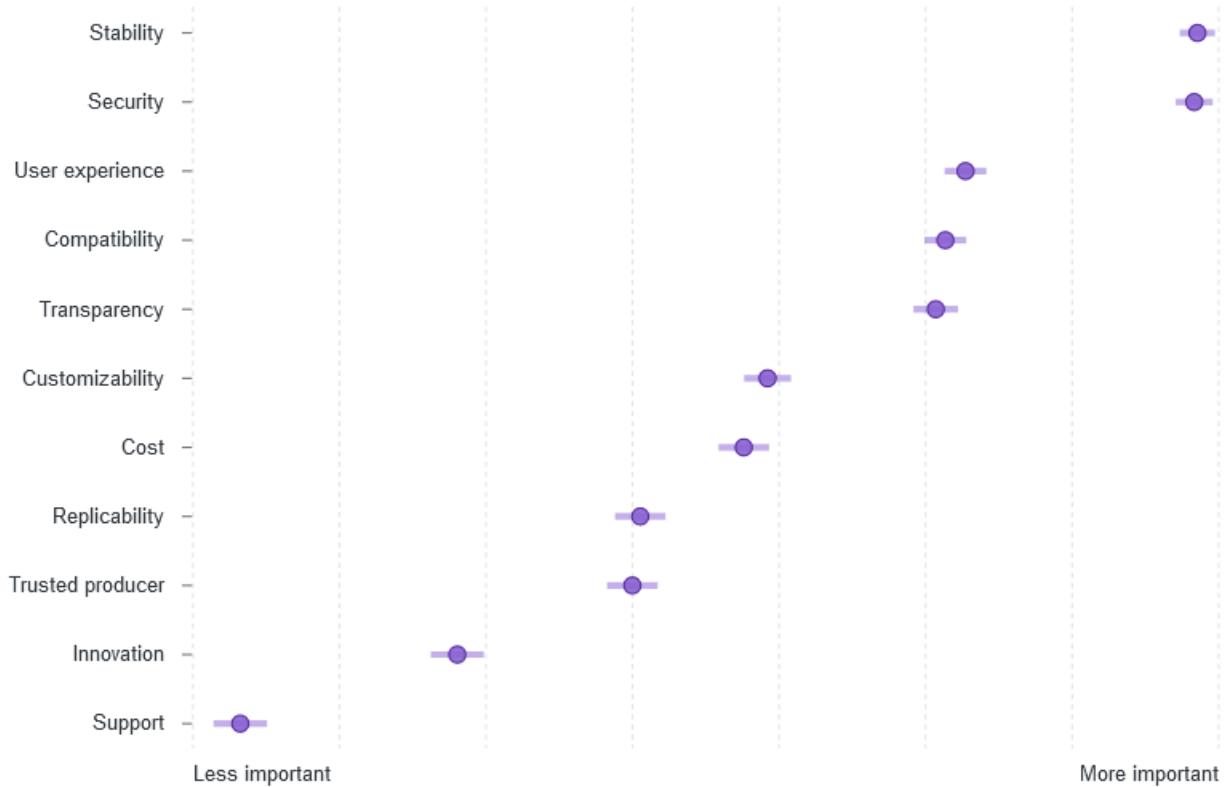
Source: opensourcesurvey.org



<https://opensourcesurvey.org/2017/#overview>

Fig5. - What open source users value in software

Source: opensourcesurvey.org



什么是一个好的开源项目

什么是一个好的开源项目

文档管理

介绍文档， 用户安装文档， 用户使用/开发文档
言简意赅， 清晰易懂， 及时更新

代码管理

高质量的代码管理
清晰的代码迭代管理

社区管理

积极的响应
活跃的社区建设
社区文化建设， License 选择



Where the world builds software

Millions of developers and companies build, ship, and maintain their software on GitHub—the largest and most advanced development platform in the world.

开源项目代码平台

56+ million
Developers

3+ million
Organizations

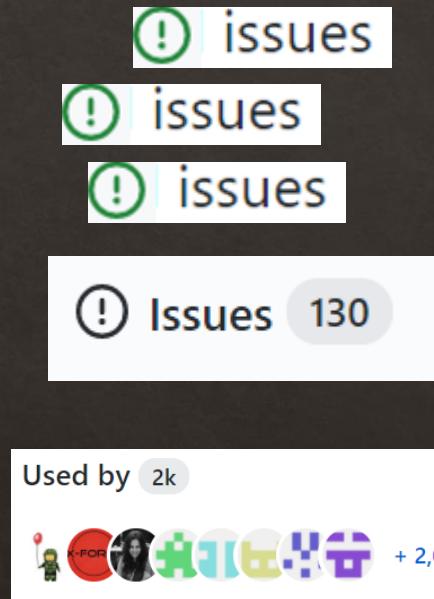
100+ million
Repositories

72%
Fortune 50

开源项目面临的问题

开源项目面临的问题

Contributors 303



⌚ 7,260 commits

Pull requests

Pull requests

Pull requests

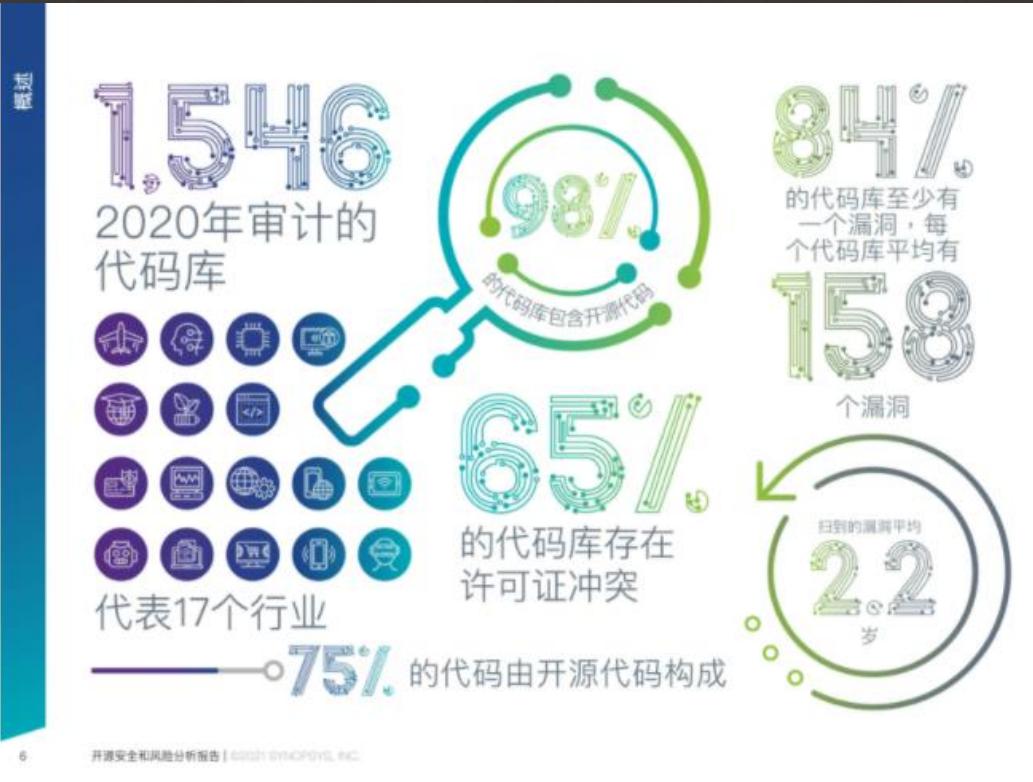
master ▾

94 branches

257 tags

如何解决这些问题和代码（pull request）的管理？

开源项目面临的问题



新思科技发布《2021年开源安全和风险分析》报告

- 98%的医疗保健行业代码库包含开源，其中有67%的代码库存存在漏洞。
- 97%的金融服务/金融科技行业代码库包含开源，其中超过60%的代码库存存在漏洞。
- 92%的零售和电子商务行业代码库包含开源，其中71%的代码库存存在漏洞。

如何解决这些问题？

代码的持续集成测试管理体系

开源项目代码管理

开源项目面临质量问题

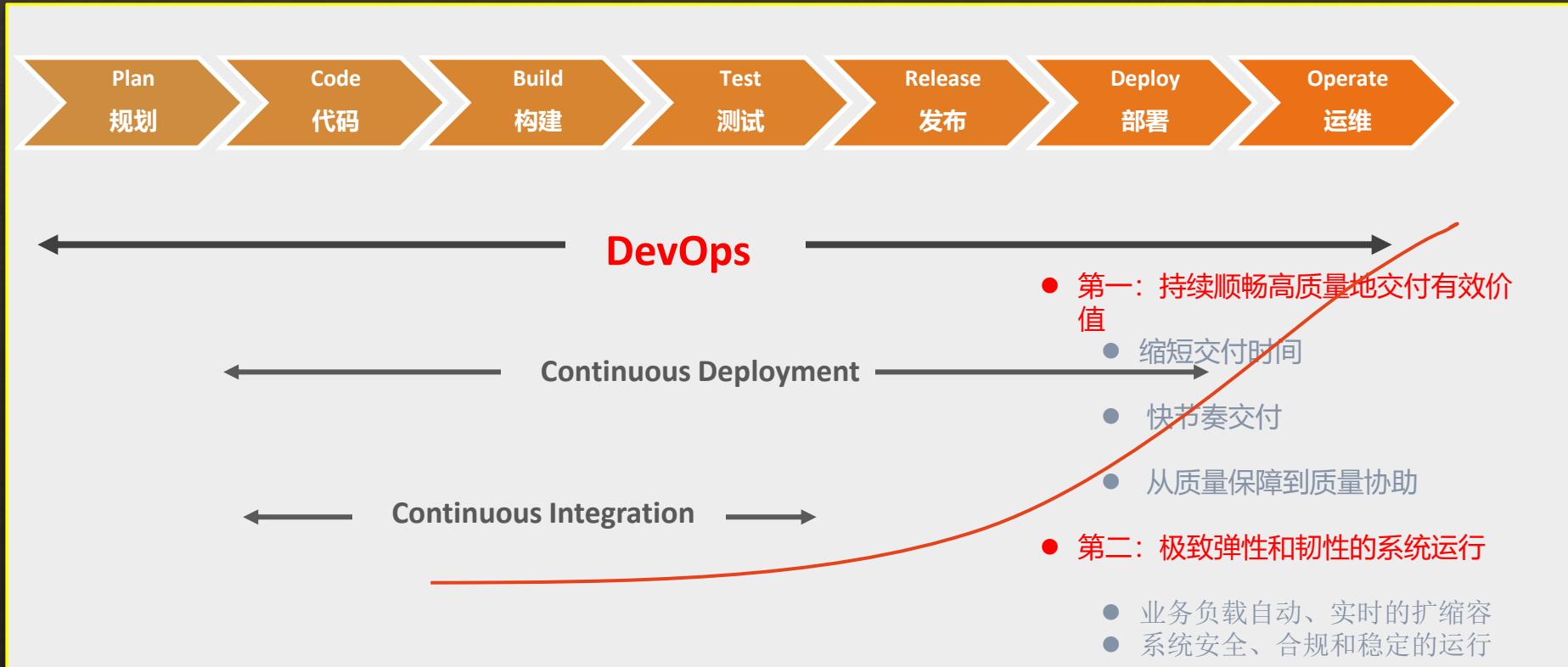
代码的持续集成测试管理体系

传统软件测试体系的变革和发展

基于开源的代码质量管理体系

什么是持续集成(Continuous Integration CI)

DevOps,CI,CD



持续集成(Continuous Integration CI) 操作流程

持续集成(Continuous Integration CI) 操作流程

Github: Continuous integration (CI) is a software practice that requires frequently committing code to a shared repository

“a development practice that requires developers to integrate code into a shared repository several times a day. Each check-in is then verified by an automated build, allowing teams to detect problems early” (Fowler, 2006) [Martin Fowler](#)

<https://martinfowler.com/>

频繁的多次的将本地的代码修改合并到公共代码仓库对应的分支

开发提交合并请求-PR (Pull Request)



主代码仓库repo



Forked repo

Git clone

本地代码

修改代码

提交代码变动

Git add
Git commit

Git push

Diff → Pull Request

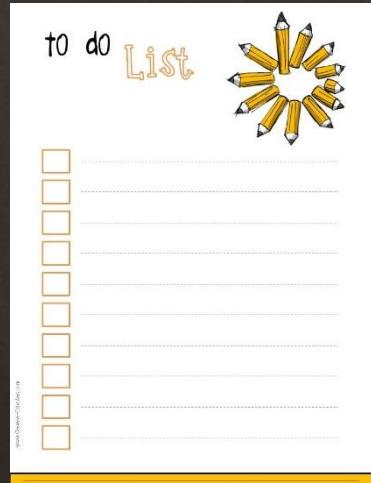
提交PR 到代码最终合并，中间发生了什么？



➤ 提交pull request

Pull requests

➤ 新的PR产生



✓ 触发集成测试 Continuous Integration Test(CI test)

✓ Comments

✓ Rebase/Update

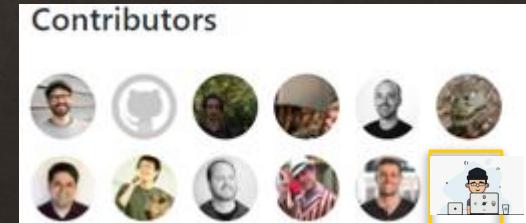
✓ 再次提交

✓ CI test ...

✓ Merge 合并代码

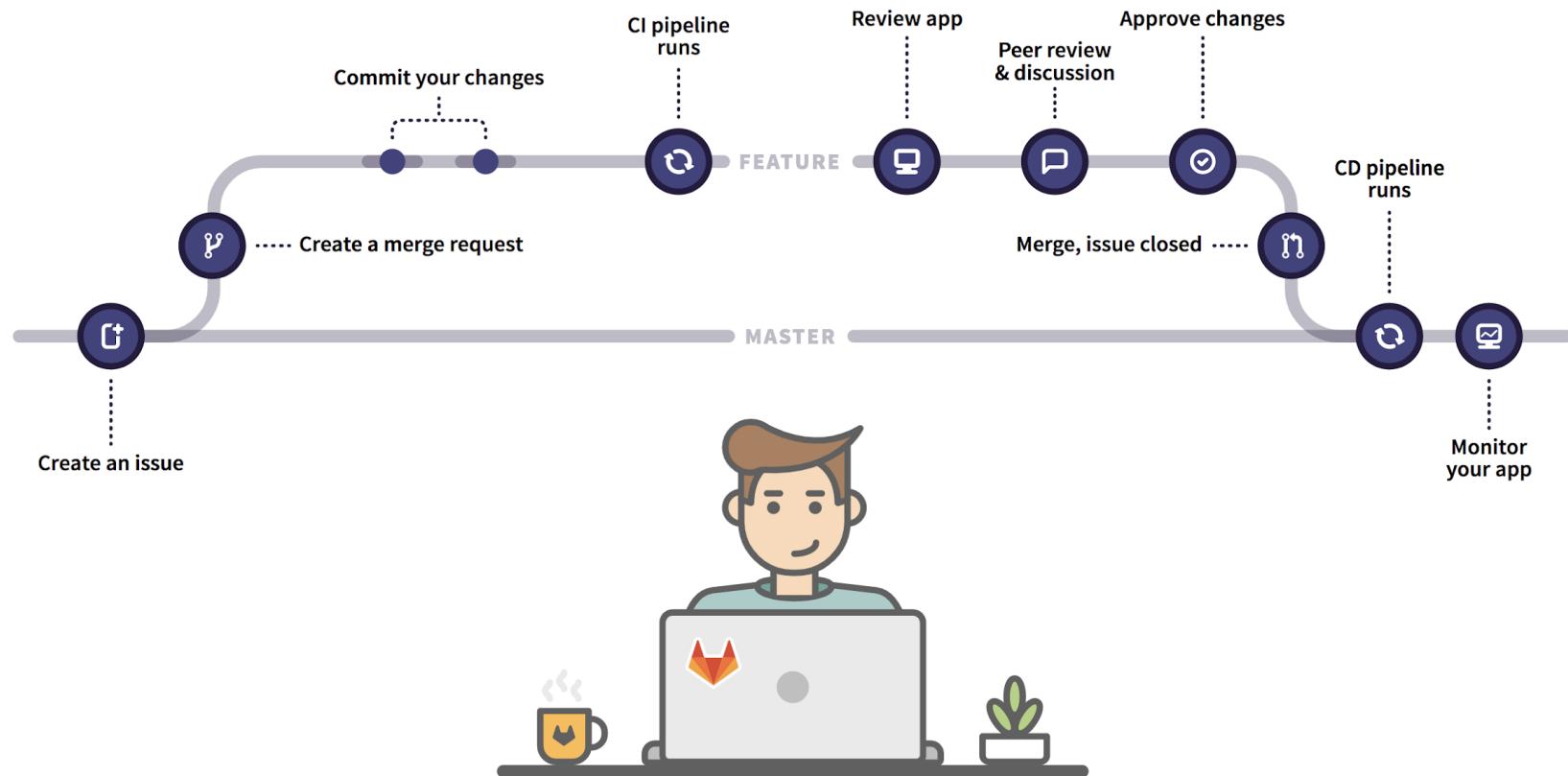
✓ 触发集成测试 (CI test)

✓ CD(持续集成部署)



➤ 成为Contributors

Gitlab推荐工作流



持续集成带来的优点

Continuous Integration doesn't get rid of bugs, but it does make them dramatically easier to find and remove.

Martin Fowler, Chief Scientist, ThoughtWorks

快速发现错误:

每完成一点更新，就集成到对应的分支，可以快速发现错误，定位错误也比较容易。

防止分支大幅偏离主干:

如果不是经常集成，分支或者主干又在不断更新，会导致以后集成的难度变大，甚至难以集成。

持续集成发布流程-CI/CD



通知

1. 检测代码变动
2. 自动构建编译
3. 自动测试
4. 自动打包



持续集成服务器



编译服务器



测试服务器

打包发布



持续集成测试流程管理工具平台

Actions -- Github

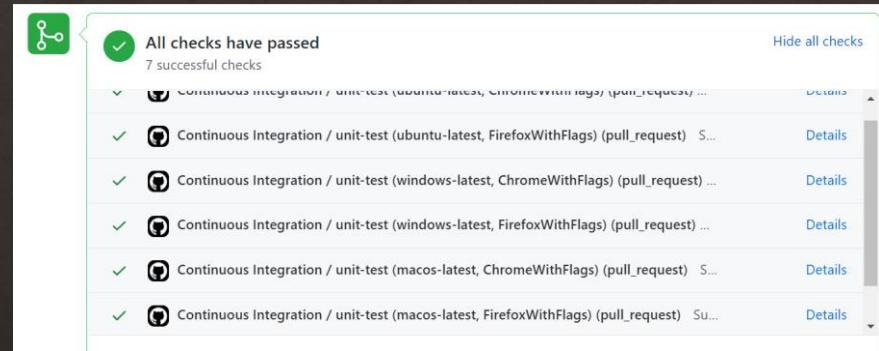
Travis CI – 3rd party public server

Jenkins -- Self host CI server

GitHub Actions

GitHub Actions

Automate, customize, and execute your software development workflows right in your repository with GitHub Actions. You can discover, create, and share actions to perform any job you'd like, including CI/CD, and combine actions in a completely customized workflow.



<https://lab.github.com/githubtraining/github-actions:-hello-world?overlay=register-box-overlay>

GitHub Actions

通过*.yml 文件描述整个CI/CD 的workflow, 在 GitHub 提供的服务器上或者自己维护的服务器上进行CI/CD 的流程自动构建

```
4 name: Python package  
5  
6 on: push  
7     branches: [ master ]  
8 pull_request:  
9     branches: [ master ]  
10  
11  
12 jobs:  
13   build:  
14  
15     runs-on: ubuntu-latest  
16     strategy:  
17       matrix:  
18         python-version: [2.7]  
19  
20     steps:  
21     - uses: actions/checkout@v2  
22     - name: Set up Python ${{ matrix.python-version }}  
23       uses: actions/setup-python@v2  
24       with:  
25         python-version: ${{ matrix.python-version }}  
26     - name: Install dependencies  
27       run: |  
28         python -m pip install --upgrade pip  
29         python -m pip install flake8 pytest  
30         if [ -f requirements.txt ]; then pip install -r requirements.txt; fi  
31     - name: Lint with flake8  
32       run: |  
33         # stop the build if there are Python syntax errors or undefined names  
34         flake8 . --count --select=E9,F63,F7,F82 --show-source --statistics
```

监听事件

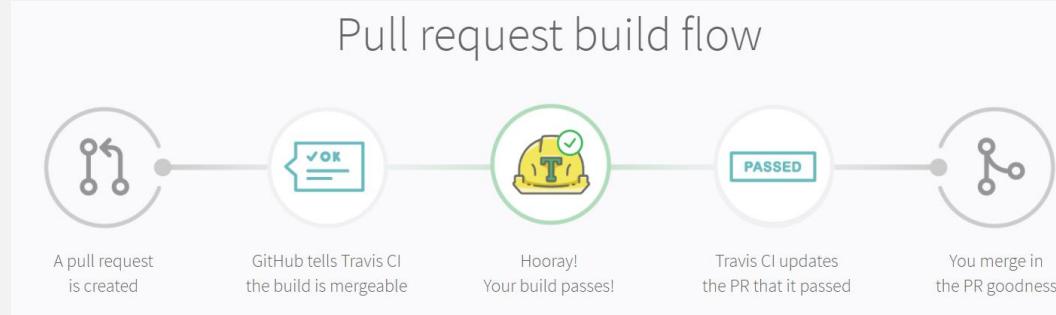
名字

编译环境设置

编译
运行

name
on
on.<event_name>.types
on.<push|pull_request>.<branches|tags>
on.<push|pull_request>.paths
on.schedule
permissions
env
defaults
defaults.run
concurrency
jobs
jobs.<job_id>
jobs.<job_id>.name
jobs.<job_id>.needs
jobs.<job_id>.runs-on
jobs.<job_id>.permissions
jobs.<job_id>.environment
jobs.<job_id>.concurrency
jobs.<job_id>.outputs
jobs.<job_id>.env

Travis CI –free for open source project



Travis CI is a hosted^[2] continuous integration service used to build and test software projects hosted on GitHub^[3] and Bitbucket.^[4]

Travis CI是在软件开发领域中的一个在线的，分布式的^[1]持续集成服务，用来构建及测试在GitHub^[2]托管的代码。这个软件的代码同时也是开源的，可以在GitHub上下载到^[3]，尽管开发者当前并不推荐在闭源项目中单独使用它。^[4]

Travis CI

通过*.travis.yml 文件描述整个CI/CD 的workflow,
在Travis 提供的服务器上进行CI/CD 的流程自动
构建

5 lines (4 sloc) | 45 Bytes

```
1 language: node_js  
2  
3 node_js:  
4 - node  
5 - 'lts/*'
```

语言

版本和环
境参数

Default Build Script

The default build script for projects using nodejs is:

npm test

Bash

✓ master Merge pull request #1 from yanbin621/test

凸 #1 passed

Modify test

⌚ Ran for 32 sec

-○ Commit ed7f519 ↗

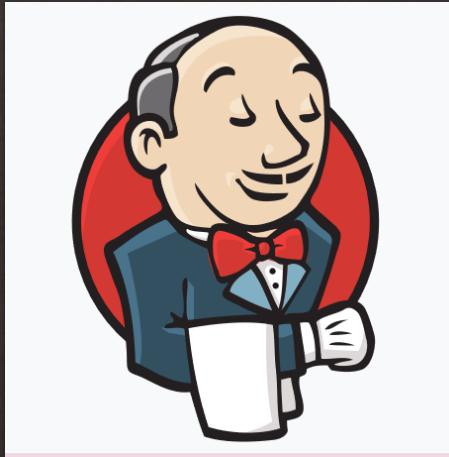
⌚ Total time 58 sec

↳ Branch master ↗

⌚ 20 hours ago

AliceCodeZhang

持续集成 (CI) 管理工具



Availability: **Free**

Platform: **Cross-platform**

<https://jenkins.io/>



Jenkins

Build great things at any scale

The leading open source automation server, Jenkins provides hundreds of plugins to support building, deploying and automating any project.

Jenkins 的优势

- 25万+的用户, 600+ contributors
- 强大的开源社区文化和运营, 持续交付基金会 (CDF)成员
- 超丰富的插件支持 (>1800+ Plugins)
- 支持几乎所有的source control management 和 Version control management tools
- 可定制化部署

Jenkins 的优势

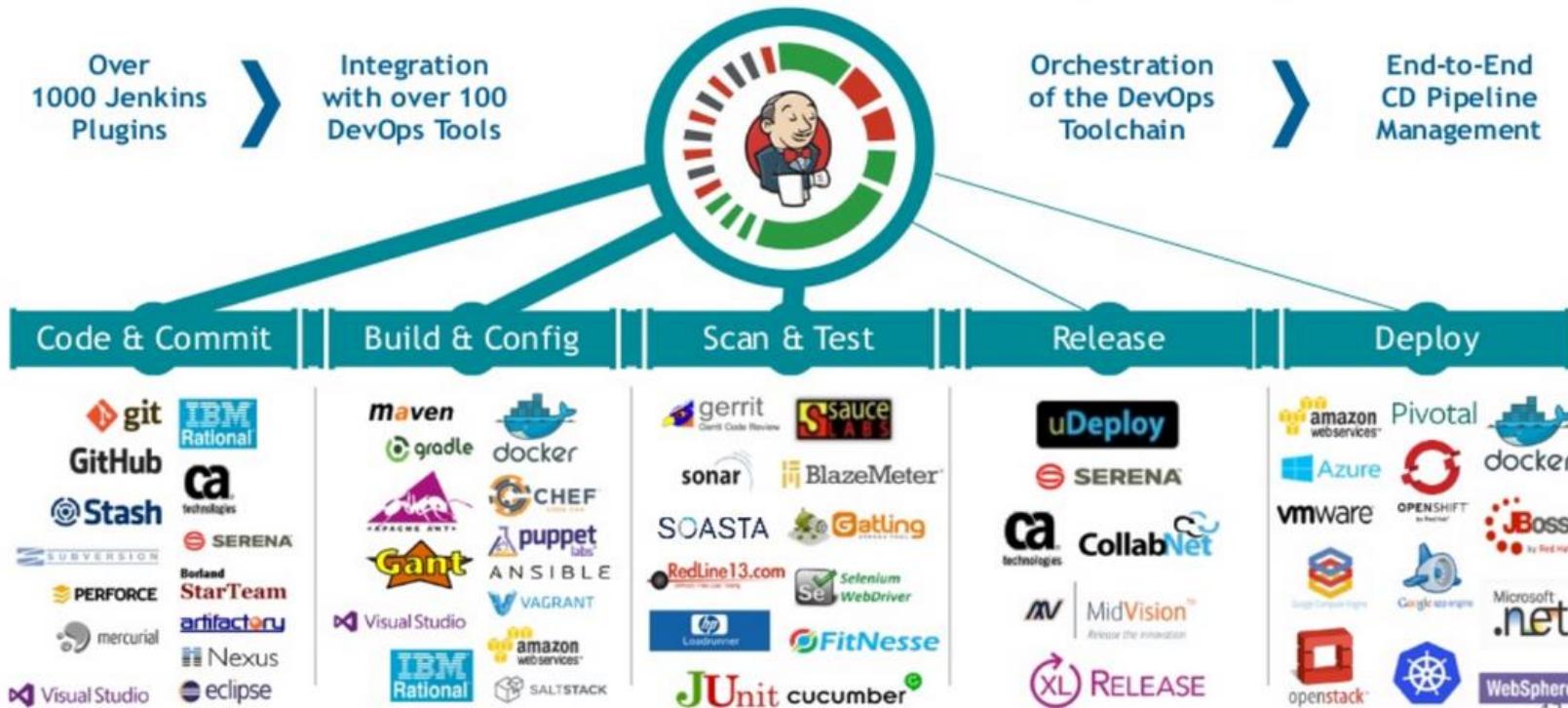
Source Control Management or the Version Control Management

https://en.wikipedia.org/wiki/Comparison_of_continuous_integration_software

Name	AccuRev	BitKeeper	CA Harvest	ClearCase	CVS	Darcs	Git	GNU Bazaar	Integrity	Mercurial	Perforce	Plastic	PVCS	StarTeam	Subversion	Surround	Synergy	Team Concert	Team Foundation Server
Vexor	No	No	No	No	No	No	Yes	No	No	No	No	No	No	No	No	No	No	No	No
TeamCity	Yes	No	No	Yes	Yes	No	Yes	Yes ^[24]	No	Yes	Yes	No	No	Yes	Yes	No	No	No	Yes
Team Foundation Server	No	No	No	No	No	No	Yes	No	No	No	No	No	No	No	Yes	No	No	No	Yes
OpenMake Software Meister	Yes	No	Yes	Yes	Yes	No	Yes	No	Yes	No	Yes	No	Yes	Yes	Yes	No	Yes	Yes	Yes
Jenkins-Hudson	Yes	Yes	Yes	Yes	Yes	Yes ^[17]	Yes	Yes	Yes ^[18]	Yes	Yes	Yes	Yes ^[19]	Yes	Yes	Yes	Yes ^[20]	Yes ^[21]	Yes ^[22]
Distelli	No	No	No	No	No	No	Yes	No	No	Yes	No	No	No	No	No	No	No	No	No
CruiseControl.NET	Yes	Yes	No	Yes	Yes	No	Yes	No	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	No	Yes
CruiseControl	No	No	No	Yes	Yes	Yes	Yes	No	Yes	Yes	Yes	No	No	Yes	Yes	Yes	No	No	Yes
CABIE	No	No	No	No	Yes	No	No	No	No	No	Yes	No	No	No	Yes	No	No	No	No
BuildMaster	Yes	No	No	Yes	Yes	No	Yes	No	Yes	Yes	Yes	Yes	Yes	Yes	No	Yes	Yes	No	Yes
BuildBot	No	No	No	No	Yes	Yes	Yes	Yes	No	Yes	Yes	No	No	No	Yes	No	No	No	No
Buddy	No	No	No	No	No	No	Yes	No	No	No	No	No	No	No	No	No	No	No	No
Bamboo	Yes ^[15]	No	No	Yes	Yes	No	Yes	No	No	Yes	Yes	No	No	No	Yes	No	No	No	Yes ^[16]
AppVeyor	No	No	No	No	No	No	Yes	No	No	Yes	No	No	No	No	Yes ^[14]	No	No	No	No
Apache Gump	No	No	No	No	Yes	No	No	No	No	No	No	No	No	No	Yes	No	No	No	No
Apache Continuum	No	No	No	Yes ^[2]	Yes ^[2]	No	Yes ^[2]	Yes ^[2]	No	Yes ^[2]	Yes ^[2]	No	No	Yes ^[2]	Yes ^[2]	No	Yes ^[2]	No	No
AntHillPro	Yes	No	Yes	Yes	Yes	No	Yes	No	Yes	Yes	Yes	No	Yes	Yes	Yes	No	Yes	Yes	Yes

Jenkins 与 CD 和 DevOps 关系

Jenkins is the Hub of the CD/DevOps Ecosystem



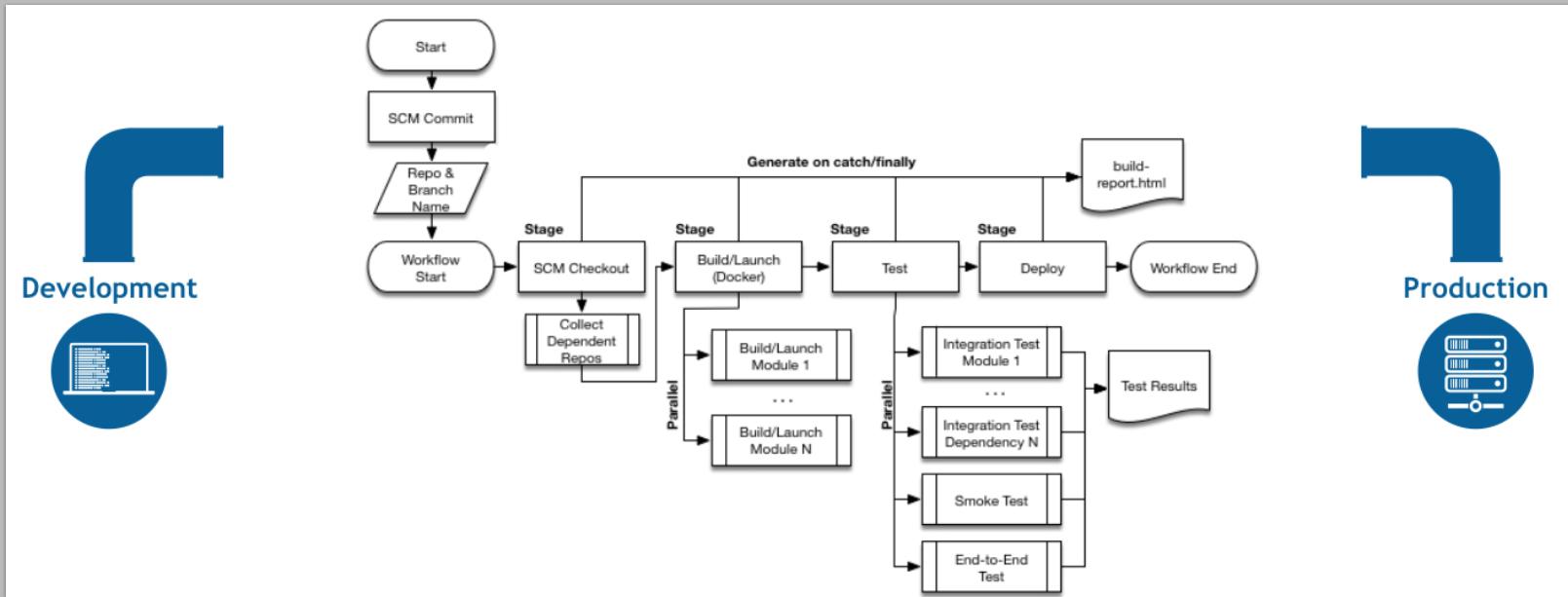
Jenkins Pipeline 机制

用解释性代码Jenkinsfile来描述，通过Jenkins服务器和一系列Jenkins 插件将整个CD(持续交付流程

Jenkins Pipeline 机制

Pipeline as code

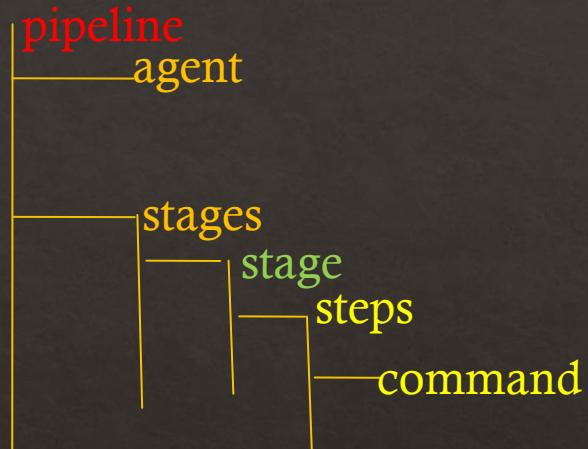
Design a whole pipeline with Jenkinsfile



<http://Jenkins.io>

Declarative Pipeline scripts 语法

```
1 pipeline { .....  
2   agent{ .....  
3     label 'mac'  
4   }  
5   environment{  
6     mvnHome = tool 'maven'  
7   }  
8   stages{ .....  
9     stage('Preparation') { // for display purposes  
10       // Get some code from a GitHub repository  
11       steps{ .....  
12         git 'https://github.com/AliceCodeZhang/androidSample.git'  
13       }  
14     }  
15     stage('Build') {  
16       // Run the maven build  
17       steps{  
18         sh "pwd ; cd ./SeleniumTest; '${mvnHome}/bin/mvn' -Dmaven.test.failure.ignore clean package"  
19       }  
20     }  
21     stage('Results') {  
22       steps{  
23         junit '**/target/surefire-reports/*.xml'  
24         archive 'target/*.jar'  
25       }  
26     }  
27   }
```



Declarative Pipeline scripts 语法

DECLARATIVE PIPELINE SYNTAX - REQUIRED

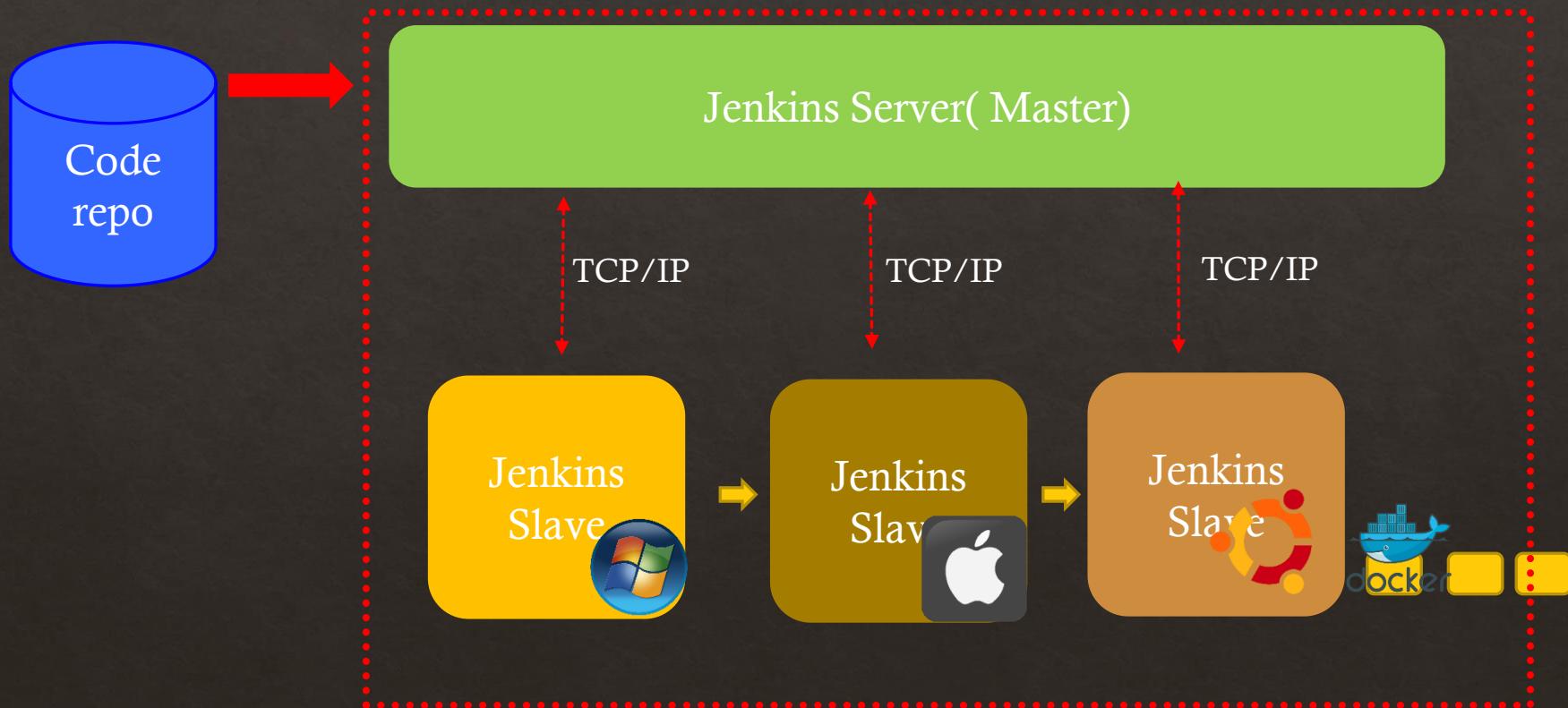
- » **pipeline** - Contains the entire Jenkins Pipeline definition
- » **agent** - Defines the agent used for the entire Pipeline or a stage
 - **label** - Existing Jenkins node label
 - **docker** - Requires Docker-enabled node
 - **image** - Run inside specified Docker image
 - **label** - Existing Jenkins node label
 - **args** - Arguments for Docker container
 - **dockerfile** - Use a local Dockerfile
 - **filename** - Name of local Dockerfile
 - **label** - Existing Jenkins node label
 - **args** - Arguments for Docker container
- » **stages** - Contains Pipeline stages and steps
- » **stage** - A specific named "stage" of the Pipeline
 - **steps** - One or more build steps that define the actions in the stage. Contains one or more of the following:
 - Any build step or build wrapper defined in Pipeline e.g. **sh**, **bat**, **timeout**, **echo**, **archive**, **junit**, etc.
 - **parallel (optional)** - Execute steps in parallel. May not be used with other steps
 - **script (optional)** - Execute Scripted Pipeline block
 - **when (optional)** - Runs stage conditionally
 - **branch** - Stage runs when branch name matches
 - **expression** - Boolean expression
 - **agent**, **environment**, **tools** and **post** may also optionally be defined in **stage**



DECLARATIVE PIPELINE SYNTAX - OPTIONAL

- » **environment** - A sequence of "key = value" pairs to define environment variables
 - **credentials('<id>')** - Bind credentials to variable
- » **options** - Options for the entire Pipeline. For example:
 - **skipDefaultCheckout** - Disable auto checkout SCM
 - **timeout** - Sets timeout for entire Pipeline
 - **buildDiscarder** - Discard old builds
 - **disableConcurrentBuilds** - Disable concurrent Pipeline runs
- » **tools** - Installs pre-defined tools to be available on PATH
- » **triggers** - Triggers for Pipeline based on schedule, polling, etc.
- » **parameters** - Pipeline parameters that are prompted for at run time
- » **post** - Defines actions to be taken when pipeline or stage completes based on outcome. Conditions execute in order:
 - **always** - Run regardless of Pipeline status
 - **changed** - Run if the result of Pipeline has changed from last run
 - **success** - Run if Pipeline is successful
 - **unstable** - Run if Pipeline result is unstable
 - **failure** - Run if the Pipeline has failed

Jenkins 的Master 和 Slaves 架构



各个构建工具对比

□ GitHub Actions

配置简单

无需额外配置服务器

支持自维护服务器挂载

□ Jenkins

插件灵活

自定义，自维护，分布式部署

□ Travis CI

第三方CI/CD 平台，构建相对简单，对免费支持的限制，可能会取消免费支持

开源项目代码管理

开源项目面临质量问题

代码的持续集成测试管理体系

传统软件测试体系的变革和发展

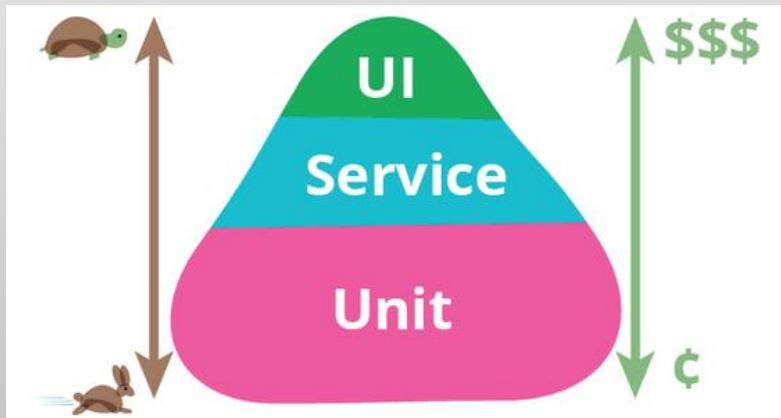
基于开源的代码质量管理体系

传统的测试体系

尽可能少的UI 测试

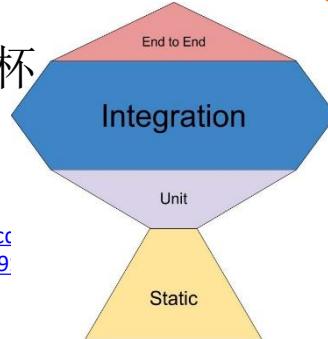
API, component , Integration test 多覆盖

Unit test 全覆盖



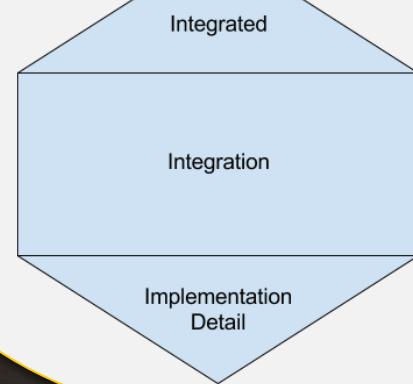
Testing Pyramid Rationale - Source: [Martin Fowler](#)

JS Web 的奖杯
模型



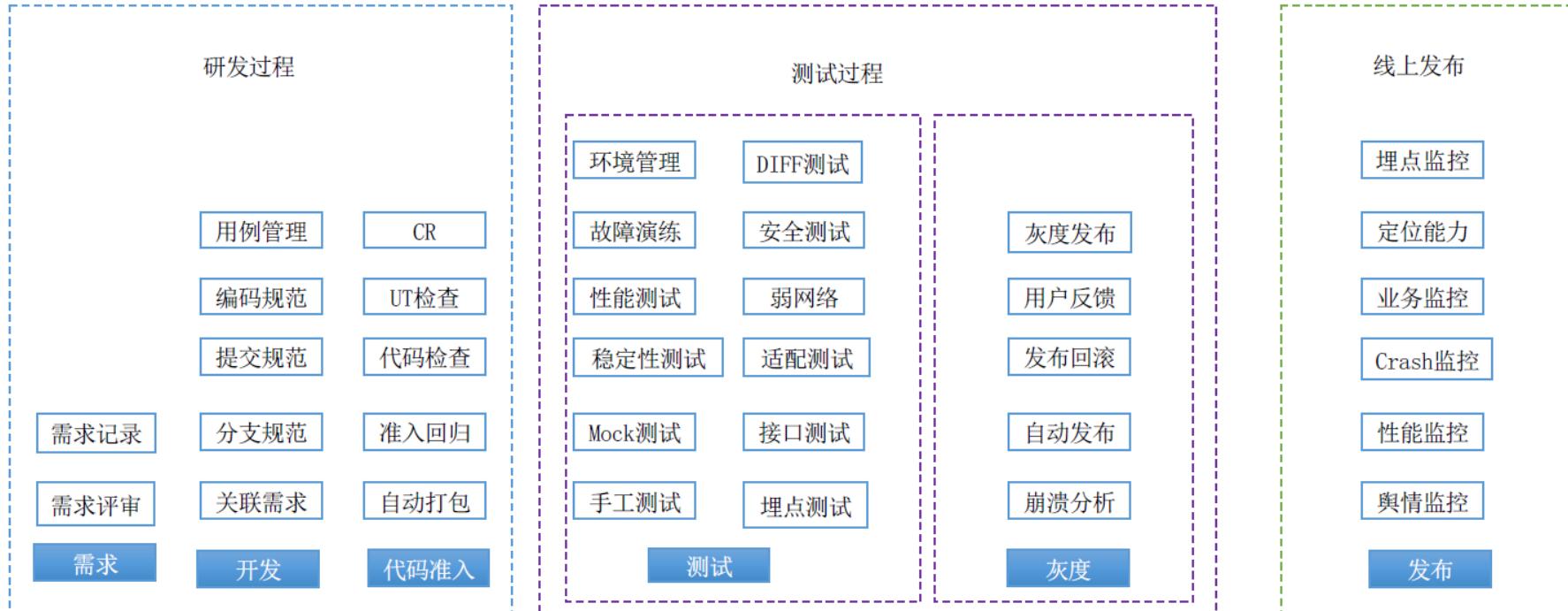
<https://twitter.com/kentcdodds/status/9607231725992832/photo/1>

微服务测试蜂窝



<https://engineering.atspotify.com/2018/01/11/testing-of-microservices/>

完善的质量管理体系



开源项目代码管理

开源项目面临质量问题

代码的持续集成测试管理体系

传统软件测试体系的变革和发展

基于开源的代码质量管理体系

完善的质量管理体系

开源需求、问题、评审、分支
管理机制

研发过程

用例管理	CR
编码规范	UT检查
提交规范	代码检查
需求记录	分支规范
需求评审	准入回归
需求	关联需求
开发	自动打包
代码准入	

测试过程

环境管理	DIFF 测试
故障演练	安全测试
性能测试	弱网络
稳定性测试	适配测试
Mock 测试	接口测试
手工测试	埋点测试

测试

灰度发布
用户反馈
发布回滚
自动发布
崩溃分析

灰度

线上发布

埋点监控
定位能力
业务监控
Crash 监控
性能监控
舆情监控

发布

开源测试体系

开源需求、问题、管理机制

- W3C Endpoint DELETE /session/{session id}/actions not supported Enhancement

#15572 opened on Jun 24 by Sam55555

- Appium 2.0 docs planning Documentation v2

#15566 opened on Jun 22 by jlippis 8 tasks

- update READMEs for each package in monorepo Documentation Enhancement

#15478 opened on Jun 9 by boneskull

- document new release workflow Documentation v2

#15432 opened on May 28 by boneskull

Issue

- Popup blocker still occurs even when setting 'safariAllowPopups' capability Bug P1 XCUItest

#8280 opened on Apr 19, 2017 by ilants

- Driver Session Details are not getting updated properly Bug P1

#7730 opened on Jan 21, 2017 by SrinivasanTarget

- Appium safari doesn't always return the active window location Bug Help Needed iOS Mobile Safari P1

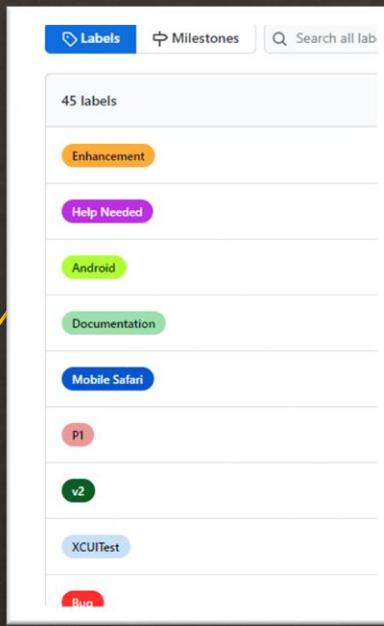
#4932 opened on Apr 16, 2015 by SBoudrias

- set_network_connection command causes fatal race condition with multiple devices Android Bug P1

#4288 opened on Dec 19, 2014 by yahman72

- implement nativeWebTap for Android Android Bug P1

#4075 opened on Nov 15, 2014 by 0x1mason



- How to inspect element if app disable screen sharing and screenshot? Question
#15865 by mostafa-raafat was closed 22 days ago
- Access from remote address to the mjpeg server (UiAutomator2) Question
#15854 by spesnil was closed 20 days ago
- Can I do a "longpress" action without release or cancel longpress actions Question
#15766 by focalism was closed on Aug 27
- Get currently running sessionId Question
#15701 by PandaPandaPanda was closed on Aug 7

开源评审管理机制

The screenshot shows a GitHub pull request page for a commit related to 'quic_agent'. The commit message is 'fixed building issues related to quic_agent #1027'. The pull request has 2 conversations, 1 commit, 4 checks, and 3 files changed. The commit history includes:

- tfrankow-intel commented on Jun 25: 'fixed issues related to quic_agent build' (commit 50d468d)
- tfrankow-intel requested review from qwu16 and starwafan as code owners 4 months ago
- qwu16 requested a review from jianjunz 4 months ago
- jianjunz commented on Jun 25: 'Conference server depends on QUIC SDK v5.0, not the latest version. The version number is specified [here](#). The library name of QUIC SDK v5.0 is libw遽_quic_transport.' (Collaborator)
- tfrankow-intel commented on Jun 25: 'ok, then I suppose the change in scripts/pack.js is the only relevant one. Of course modified to libw遽_quic_transport.so. Is that right?' (Author)

At the bottom, there's a 'Review required' section indicating 'At least 1 approving review is required by reviewers with write access' and a 'Reviewers' dropdown showing '3 pending reviewers'. The 'Checks' section shows all checks have passed with 7 successful checks, including results from centos-conference-chrome[apiTest], centos-mcu[build], ci/jenkins/server, and continuous-integration/jenkins/pr-head.



开源分支管理机制

open-webrtc-toolkit / owt-server Public

Code

Issues 209

Pull requests 33

Actions

Projects

Wiki

Security

master

7 branches

10 tags

Switch branches/tags

Filter branches/tags

Branches

Tags

✓ master

default

4.2.x

4.3.x

5.0.x

360-video

gst-analytics

uhd

[View all branches](#)

(#1085)

Add a workflow for WebTrac

update certificate expiration

Allow data flow from QUIC

Update gst Dockerfile to 5.0

fix install deps for centos (#

Enable AV1 stream forward

Cl: Build&Pack package with

Add Chromium to Thirdpar

Added line ending normalization.

.gitignore

Don't keep third_party/webrtc

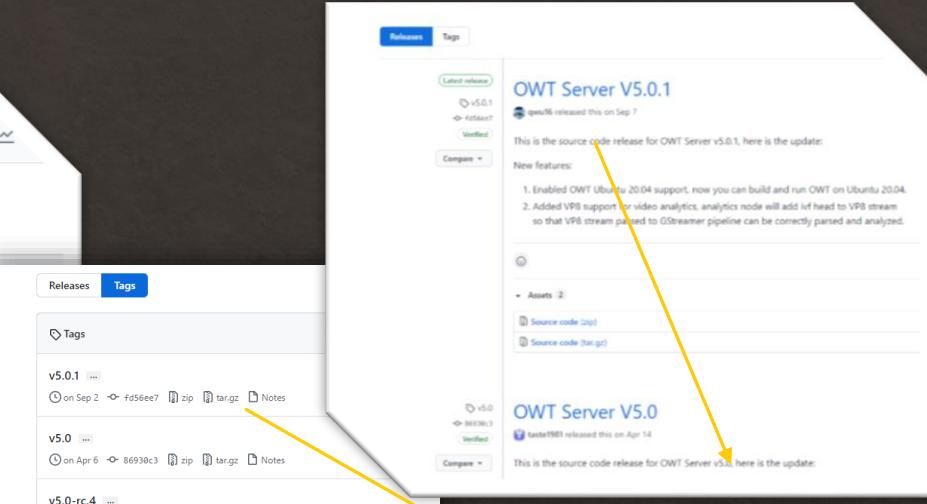
LICENSE

Add apache 2.0 license

README.md

Add eslint in packing process (#936)

branches



Releases Tags

Tags

v5.0.1

on Sep 2 · fd56ee7 · zip · tar.gz · Notes

v5.0

on Apr 6 · 86930c3 · zip · tar.gz · Notes

v5.0-rc.4

on Mar 17 · 7cc5cf5 · zip · tar.gz · Notes

v5.0-rc.3

on Jan 22 · ca50ff80 · zip · tar.gz · Notes

v5.0-rc.2

on Dec 10 · 3e6a5a2 · zip · tar.gz · Notes

Marketplace / Actions / Setup Node.js environment



GitHub Action

Setup Node.js environment

v2.4.1 Latest version

OWT Server V5.0.1

qux16 released this on Sep 7

This is the source code release for OWT Server v5.0.1, here is the update:

New features:

1. Enabled OWT Ubuntu 20.04 support, now you can build and run OWT on Ubuntu 20.04.
2. Added VP8 support for video analytics, analytics node will add ivf head to VP8 stream so that VP8 stream passed to GStreamer pipeline can be correctly parsed and analyzed.

Assets 2

Source code (zip)

Source code (tar.gz)

OWT Server V5.0

taste1981 released this on Apr 14

This is the source code release for OWT Server v5.0, here is the update:

Release

开源需求、问题、评审、分支 管理机制

- 1. 标题简洁规范
- 2. 内容清楚

bug 为例

- 1. 问题的大概原因
- 2. 对应的软件版本，硬件环境描述
- 3. 重现的步骤
- 4. 问题出现时候的log，截图信息等

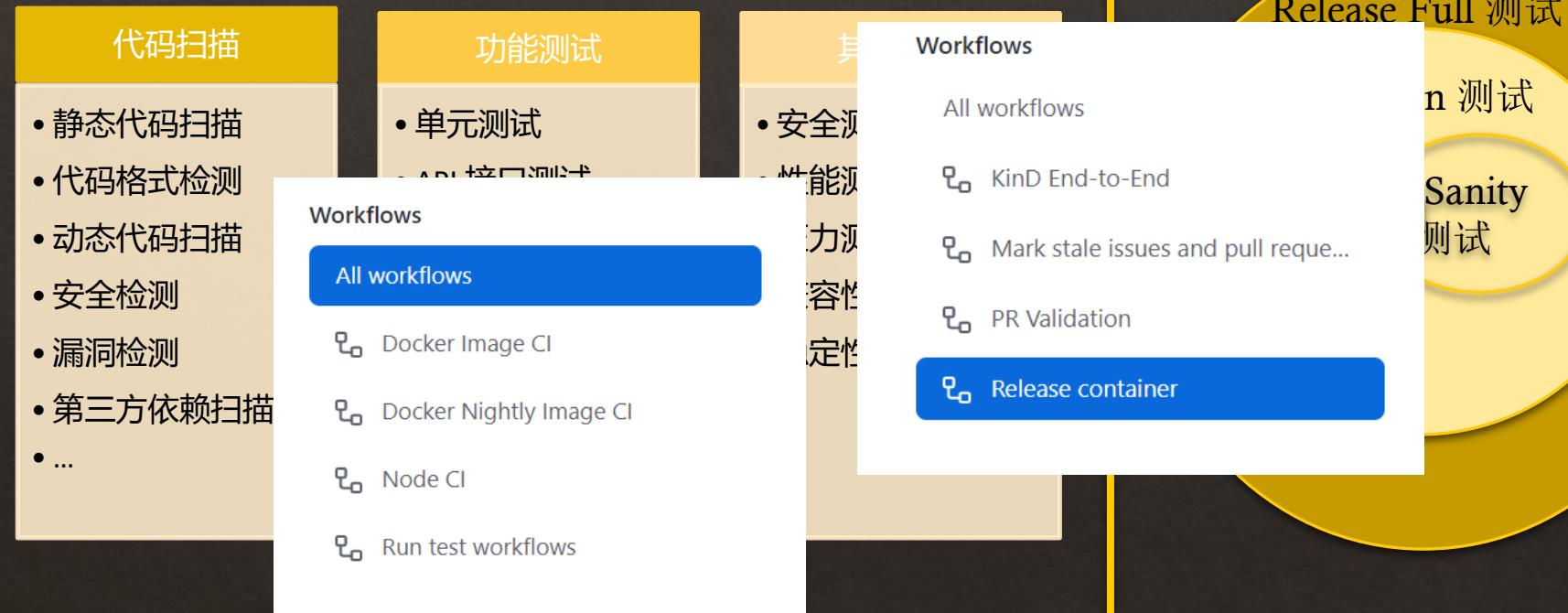
Contributors

Reviewers

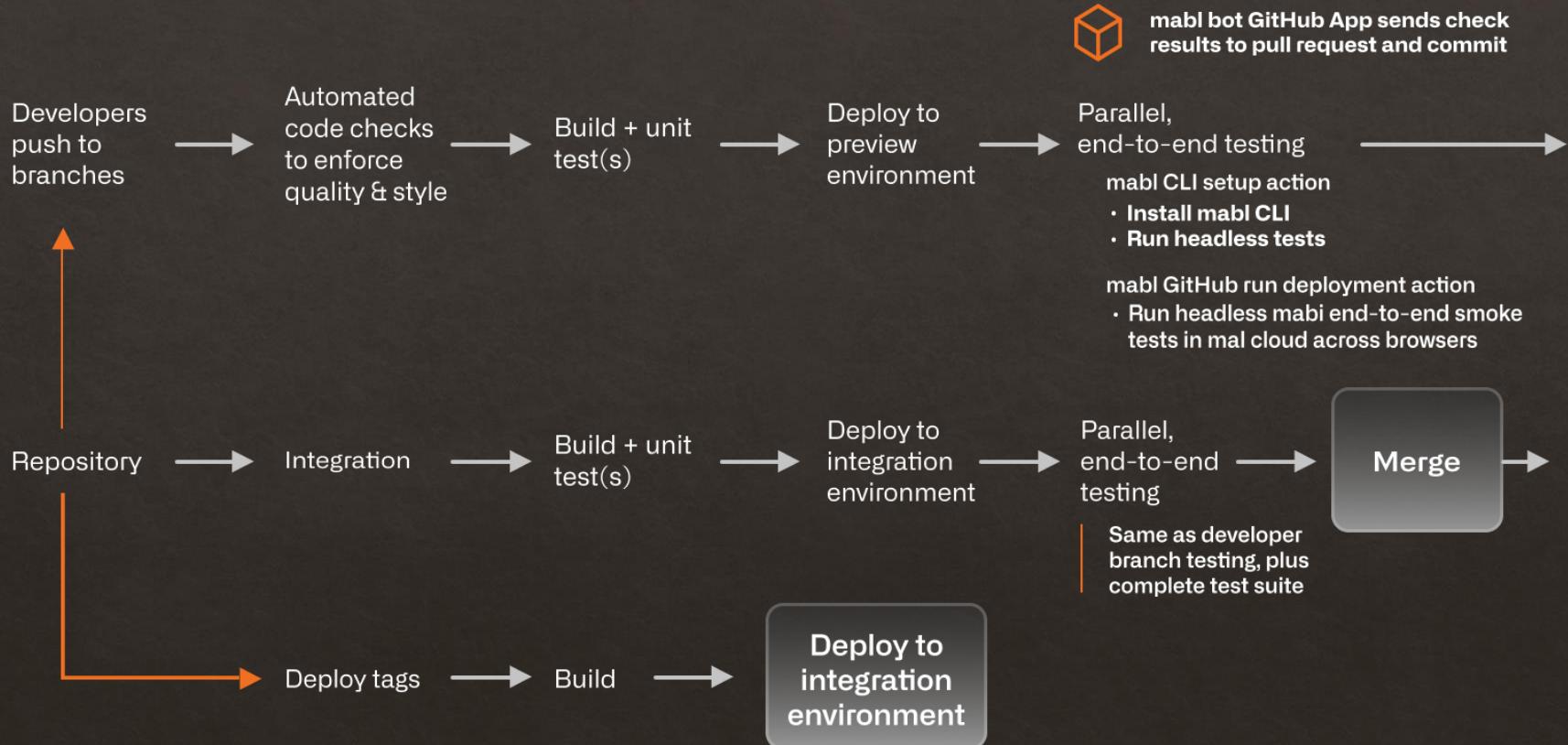
- 1. 及时查看Issues 内容，使用Tag进行分类
- 2. 分配到合理的模块管理人员
- 3. 设置milestones 设置解决期限
- 4. 对应的PR 如果解决之后要及时关闭bug

Issue ←→ Code

测试体系



开源Workflow



开源项目整体管理



Q&A