OASIS 系统项目设计文档

0.版本更新

修改人	日期	变更原因	版本号
丁玲燕	2020.02.17	创建草稿	V0.1
陆放明	2020.02.27	目录更改, 更新服务端架构设计说明	V0.2
丁玲燕	2020.02.29	更新前端架构设计说明	V0.3
陆放明	2020.03.05	api 更新; 部署部分	V0.4

1. 引言

1.1 编写目的

本文档详细完成对学术关系图谱系统 OASIS 的设计，达到指导详细设计和开发的目的，同时实现测试人员及用户的沟通。

本报告面向开发人员、测试人员及最终用户编写，是了解系统的导航。

1.2 定义

词汇名称	词汇含义	备注
OASIS	学术关系图谱系统	

1.3 参考资料

1. 项目启动文档
2. 需求规格说明书
3. 测试文档
4. 计划文档

2. 任务概述

2.1 目标

将不同数据源的学术数据集成到数据库，抽取其中的实体以及实体与实体之间的关系，构建一个学术关系图谱系统（OASIS）并提供用户搜索与相应展示。

2.2 运行环境

前端运行在主流浏览器上，包括 Chrome、Firefox、Edge 等；后端运行在阿里云服务器上。

2.3 需求概述

OASIS 需要构建学术关系，提供高效的论文查询、学术关系查询，构建学者画像及学术机构画像，提供学术同行评价、专家推荐系统、学术机构评价等，并将其可视化。

2.4 条件与限制

CON1：采用 Java 语言及其它相关的 Web 开发

CON2：系统使用的是基于 Web 的数据库应用系统

CON3：项目需要完整的单元测试、集成测试、系统级测试

CON4：项目后期会增加需求及开放式功能

CON5：将个人工程行为尽可能地记录在 Gitlab 上

CON6：每次迭代产品均必须完成部署（使用 Jenkins 实现一键部署）

3. 总体设计

- 系统主要以信息分发的方式进行架构的构建，根据已有的系统边界和参与者，限定使用者仅拥有相关的数据访问权限。参与者可以直接以游客访问的身份获取到平台的信息资源，从而不需要相关的权限认证模块搭建。
- 系统采取前后端完全解耦的方式进行架构设计，以 Tire 架构作为系统的主要架构风格，按照**数据源、中间件、服务、静态站点**进行集群的搭建与分配，从而尽可能降低各个 Tire 之间的相互影响性。
- 系统采用 DDD 进行领域模型设计

4. 逻辑视角

OASIS 系统中，采用前后端解耦的基本架构，其中 UI 展示部分采用单 Tire 进行架构构建，服务端架构采用多 Tire 逻辑协同进行

图 1：UI 展示层架构视图

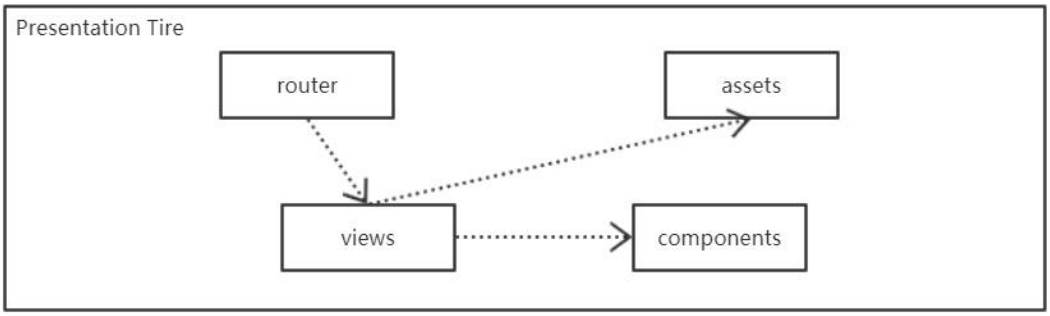


图 2：服务端架构视图

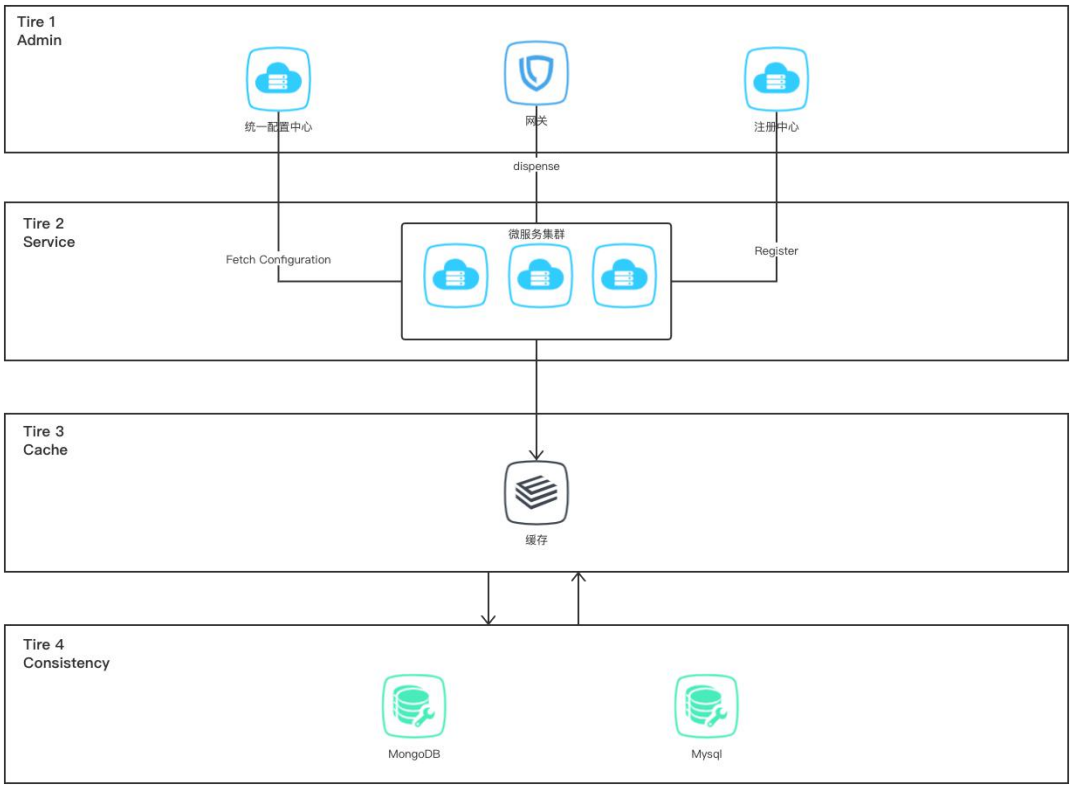
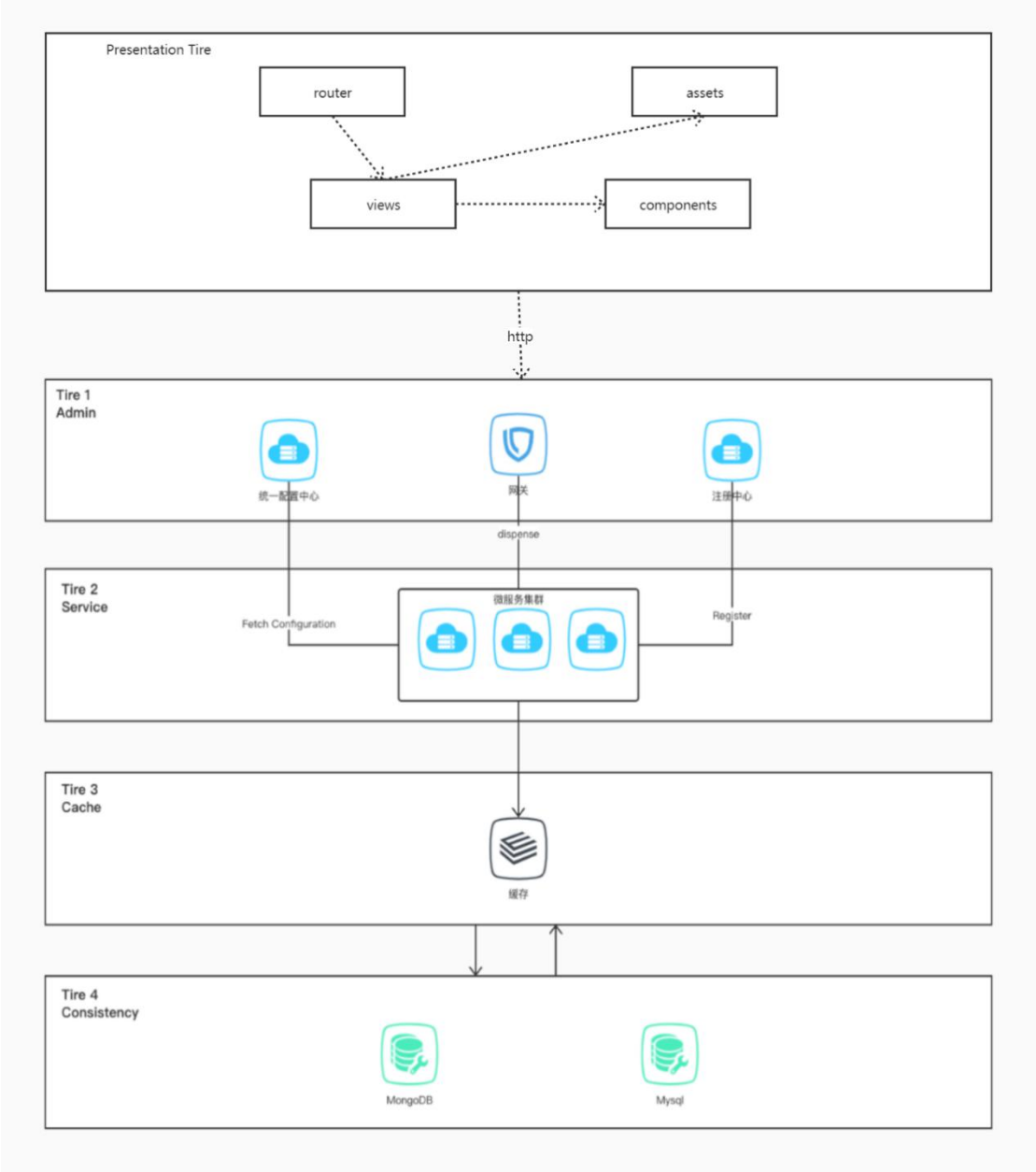


图 3：系统整体架构视图



5. 架构设计

5.1 静态站点架构分解

技术栈： Vue.js + Vue-router + ElementUI + axios

主要实现本系统的前端页面展示。在基于 vue-cli 2.5 的模板基础上加

入了 vue-router 等配套设施, 由 vue-router 来控制 views 中各页面的跳转。开发时采用组件化策略，降低与页面之间的耦合，并使用 ElementUI 组件库进行美化，views 中各页面则复用已开发好的组件。使用 axios 与后端进行数据通信。

5.2 服务端架构分解

1. 主从式架构设计

技术选型：Spring Cloud eureka , zuul

主要实现服务注册与服务发现。系统的主要业务由一组独立的微服务组成，Worker 启动之后，将会以心跳机制注册到 Admin 注册中心，并且从配置中心获取服务相对应的配置内容, 目前我们通过服务注册与发现来让微服务可以感知彼此，微服务框架在启动的时候，将自己的信息注册到注册中心，同时从注册中心订阅自己需要引用的服务。此外，对外采用统一的路由接入机制，使用 Zuul 进行统一的路由分发和熔断。

通过统一配置管理来实现一个中心化的外部配置。

2. 缓存架构设计

当前数据源为 MongoDB 与 Mysql ，对于耗时较大的搜索查询服务，使用 Redis 集群进行中间件配置，起到缓存的作用。具体缓存策略如下：

服务类别	缓存变更行为

搜索资源	根据业务请求查询缓存，若存在则直接返回，否则加入缓存之后再进行返回
更新资源	实行饿汉式加载，若缓存中存在相应的内容，先进行缓存更新，后进行底层冷数据更新
删除资源	若缓存中存在该资源，则需要先进行缓存删除，后进行低层冷数据的删除
添加资源	若缓存中不存在该资源，则需要先进行缓存添加，保证下一次查询可以命中，后进行低层冷数据的添加

3. 整体部署架构设计

多个 Tire 之间采用 docker swarm 来进行一体化 CI/CD，使用 overlay 网络进行容器间通信，具体网络协议为 UDP / TCP 协议。

针对系统可用性问题进行了 Docker 镜像构建上的优化，主要使用 JIB 插件来进行构建时的镜像同步更新，借助阿里云镜像平台进行快速镜像构建与部署。

5.3 接口定义

5.3.1 paper 导入

接口信息

接口名称:paper 导入

接口路径:/api/query/paper

请求协议:HTTP

请求方法:POST

请求参数

参数类型：Json

根类型: Object

参数名	说明	必填	类型	限制	示例
id	论文 id	是	[number]		
title	论文题目	是	[string]		title
abstract	摘要	是	[string]		
conference	会议名称	是	[string]		
affiliation	隶属机构名,以分号隔开	是	[string]		
authors	作者名,以分号隔开	是	[string]		
terms	术语	是	[string]		
keywords		是	[string]		

5.3.2 论文搜索 - 模糊搜索

接口信息

接口名称:论文搜索 - 模糊搜索

接口路径:/api/query/paper/list?query=&returnFacets=

请求协议:HTTP

请求方法:GET

接口使用状态:正常启用

GET 参数

参数名	说明	必填	类型	限制	示例
query	搜索关键字，支持【paper 名，作者、机构、会议名、研究方向名】的查找。多关键词之间空格隔开，其中空格需要进行转义处理（%20）	是	[string]		
returnFacets	查询范围	是	[string]		
pageSize	每一页的大小，若实际数据不够则只返回部分【默认值为 10】	否	[number]		10
pageNum	页号 start from 0【默认值为 0】	否	[number]		0

返回参数

参数类型：Json

根类型: Object

参数名	说明	必填	类型	限制	示例
papers		是	[array]		
papers>>keywords		是	[object]		
papers>>terms	术语	是	[string]		
papers>>authors	作者名,以分号隔开	是	[string]		
papers>>affiliation	隶属机构名,以分号隔开	是	[string]		
papers>>conference	会议名称	是	[string]		
papers>>abstract	摘要	是	[string]		
papers>>title	论文题目	是	[string]		title
papers>>id	论文 id	是	[number]		
itemCnt	条目总数	是	[number]		

5.3.3 论文查询 - summary 获取

接口信息

接口名称:论文查询 - summary 获取

接口路径:/api/query/paper/summary

请求协议:HTTP

请求方法:GET

接口使用状态:正常启用

返回参数

参数类型：Json

根类型: Object

参数名	说明	必填	类型	限制	示例
conference	会议 summary	是	[object]		
affiliation	机构 summary	是	[object]		
author	作者 summary	是	[object]		

term	术语 summary	是	[object]		

5.3.4 论文查询 - 二次搜索接口

接口信息

接口名称:论文查询 - 二次搜索接口

接口路径:/api/query/paper/refine?refinements=可以的取值 conference，term，author，affiliation，year。其中 year 需要为 year:2017_2018，也就是下划线分隔

请求协议:HTTP

请求方法:GET

接口使用状态:正常启用

GET 参数

参数名	说明	必填	类型	限制	示例
refinements	额外添加的条件限定。	是	[array]		可以的取值 conference，term，author，affiliation，year。其中 year 需要为 year:2017_2018，也就是下划线分隔
pageSize	页大小，默认 10	否	[number]		
pageNum	页号，默认 1	否	[number]		

返回参数

参数类型：Json

根类型: Object

参数名	说明	必填	类型	限制	示例
papers		是	[array]		
papers>>abstract	摘要	是	[string]		

papers>>conference	会议名称	是	[string]		
papers>>title	论文题目	是	[string]		title
papers>>affiliation	隶属机构名,以分号隔开	是	[string]		
papers>>authors	作者名,以分号隔开	是	[string]		
papers>>terms	术语	是	[string]		
papers>>keywords		是	[string]		
papers>>id	论文 id	是	[number]		
itemCnt	条目总数	是	[number]		

5.3.5 年度热门方向（词及热度）的词云数据查询

接口信息

接口名称:年度热门方向（词及热度）的词云数据查询

接口路径:/api/report/wdcld/year?year=

请求协议:HTTP

请求方法:GET

接口使用状态:正常启用

GET 参数

参数名	说明	必填	类型	限制	示例
year	年份	是	[number]		

返回参数

参数类型：Json

根类型: Object

参数名	说明	必填	类型	限制	示例
term	术语	是	[string]		

count	计数	是	[number]		

5.3.6 被引用数最多的论文 TOP K 查询

接口信息

接口名称:被引用数最多的论文 TOP K 查询

接口路径:/api/report/paper/rank/citation

请求协议:HTTP

请求方法:GET

接口使用状态:正常启用

GET 参数

参数名	说明	必填	类型	限制	示例
rank	top k，默认为 10	否	[number]		

返回参数

参数类型：Json

根类型: Object

参数名	说明	必填	类型	限制	示例
id	论文 id	是	[number]		
title	论文题目	是	[string]		title
abstract	摘要	是	[string]		
conference	会议名称	是	[string]		
affiliation	隶属机构名,以分号隔开	是	[string]		
authors	作者名,以分号隔开	是	[string]		
terms	术语	是	[string]		
keywords		是	[string]		

5.3.7 被引用论文数最多作者 top k 查询

接口信息

接口名称:被引用论文数最多作者 top k 查询

接口路径:/api/report/author/rank/paper_cnt

请求协议:HTTP

请求方法:GET

接口使用状态:正常启用

GET 参数

参数名	说明	必填	类型	限制	示例
rank	top 【rank】 默认为 10	否	[number]		

5.3.8 论文总数折线图数据获取

接口信息

接口名称:论文总数折线图 ，按照年份排

接口路径:/api/report/paper/trend/year

请求协议:HTTP

请求方法:GET

接口使用状态:已完成

返回参数

参数类型：Json

根类型: Object

参数名	说明	必填	类型	限制	示例
year	年份	是	[number]		

count	论文数量	是	[number]		

5.3.9 查询论文可见性

接口信息

接口名称:查询论文可见性接口路径:/api/permission/paper

请求协议:HTTP

请求方法:GET

接口使用状态:正常启用

5.3.10 论文初始化

接口信息

接口名称:论文初始化

接口路径:/api/permission/paper

请求协议:HTTP

请求方法:POST

接口使用状态:正常启用

5.3.11 论文清空

接口信息

接口名称:论文清空

接口路径:/api/permission/paper

请求协议:HTTP

请求方法:DELETE

接口使用状态:正常启用

6. 信息视角

6.1 信息持久化对象

Raw Data PO

PO	内容
PaperPO	包含唯一标识 id，论文名称，概要等

6.2 数据源

同时采用 MongoDB 的文档化存储和 Mysql 的数据表进行模型建立

6.3 领域建模设计

领域建模设计如下

名称	内容
Paper	基本论文内容，需要关联其对应的作者、会议
Author	关注于单个作者的信息，作者可以和多个论文相互对应，需要关联到作者对应的机构
Affiliation	机构的相关内容，和作者模型相互关联
Conference	会议实体，是前三个实体的中间联结点

7. 部署设计

7.1 技术选型

CI 工具：jenkins pipeline
测试可视化工具：JaCoCo Coverage Report
微服务集成：Docker , Docker swarm
DNS 自动解析: let's encrypt docker 镜像，自动 CA 证书获取
静态页面部署：webpack , nginx 容器代理
服务端部署: JIB , spring cloud

如图所示，本系统的 Jenkins 采用两个独立的 CI 流水线，分别负责静态页面和服务的部署工作。此外，系统使用组员的四台阿里云作为云端集成的服务器载体。

7.2 静态页面部署流程

通过 jenkins 设置 gitlab webhook 之后，监听 master 分支的 push 事件，随后进入配置的流水线过程。这里的流水线 Jenkinsfile 如下

```
pipeline {
    agent any
    stages {
        stage('Checkout') {
            steps {
                echo 'Checkout'
                git credentialsId: 'jenkinstoken', url: 'http://212.129.149.40/171250027_expelliarmus/frontend-cold4'
                sh 'npm i npm@latest -g'
            }
        }
        stage('Build') {
            steps {
                echo 'Building'
            }
        }
        stage('Deploy') {
            steps {
                echo 'Deploying'
                sh 'cd oasis-frontend && make resource deploy'
            }
        }
    }
}
```

由于前端暂时不进行测试，所以在流水线内部，直接进行静态文件的构建、资源 scp 拷贝、远端 Docker 镜像更新和服务重启过程。

系统静态文件的远端构建，通过 nginx docker 容器来做正向代理，并且通过 docker bridge 网桥来进行服务端内部局域网的并入，由 Let's Encrypt 自动监听 'proxy' 局域网下的启动事

件，触发 CA 证书的更新和 DNS 解析工作。

最终可以在 https://oasi.top 下访问到项目的前端内容。

7.3 服务部署流程

这里的部署只涉及到架构图中 `微服务集群` 这一 Tire 的 CI / CD 。其他 Tire 的部署已经提前实现完成(包括数据的容错备份、恢复)，并且为了提高系统可用性，不因微服务集群的重启而影响其他的 Tire 集群状态。

Jenkins 流水线示意如下

```
pipeline {
    agent any
    stages {
        stage('Checkout') {
            steps {
                echo 'Checkout'
                git credentialsId: 'jenkinstoken', url: 'http://212.129.149.40/171250027_expelliarmus/backend-cold4'
            }
        }
        stage('Build') {
            steps {
                echo 'Building'
                sh 'mvn clean'
            }
        }
        stage('Test') {
            steps {
                echo 'Testing'
                echo "starting unitTest....."
                sh 'mvn org.jacoco:jacoco-maven-plugin:prepare-agent -f pom.xml clean test -Dautoconfig.skip=true -Dmaven.test.skip=false -Dmaven.test.failure.ignore=true'
            }
        }
        stage('JacocoPublisher') {
            steps {
                jacoco()
            }
        }
        stage('Deploy') {
            steps {
                echo 'Deploying'
                sh '''
                ssh root@39.96.75.119 'cd service && make deploy-app'
                '''
            }
        }
    }
}
```

7.3.1 Jacoco 测试报告生成

采用 Jenkins Jacoco 插件进行最终的代码测试覆盖率生成报告，最近一次的覆盖率报告如下：

JaCoCo Coverage Report

Download `jacoco.exec` binary coverage file

Overall Coverage Summary

name	instruction	branch	complexity	line	method	class
all classes	30% **M:** 1973 **C:** 833	21% **M:** 120 **C:** 32	27% **M:** 227 **C:** 84	31% **M:** 478 **C:** 211	33% **M:** 156 **C:** 76	61% **M:** 15 **C:** 23

Coverage Breakdown by Package

name	instruction	branch	complexity	line	method	class
com.example.oasisconfigserver	**M:** 8 **C:** 0 0%	**M:** 0 **C:** 0 100%	**M:** 2 **C:** 0 0%	**M:** 3 **C:** 0 0%	**M:** 2 **C:** 0 0%	**M:** 1 **C:** 0 0%
com.example.oasisdocument	**M:** 5 **C:** 3 38%	**M:** 0 **C:** 0 100%	**M:** 1 **C:** 1 50%	**M:** 2 **C:** 1 33%	**M:** 1 **C:** 1 50%	**M:** 0 **C:** 1 100%
com.example.oasisdocument.config	**M:** 0 **C:** 55 100%	**M:** 0 **C:** 0 100%	**M:** 0 **C:** 4 100%	**M:** 0 **C:** 18 100%	**M:** 0 **C:** 4 100%	**M:** 0 **C:** 1 100%
com.example.oasisdocument.controller	**M:** 277 **C:** 84 23%	**M:** 11 **C:** 3 21%	**M:** 18 **C:** 6 25%	**M:** 63 **C:** 17 21%	**M:** 11 **C:** 6 35%	**M:** 0 **C:** 2 100%
com.example.oasisdocument.docs	**M:** 0 **C:** 104 100%	**M:** 0 **C:** 0 100%	**M:** 0 **C:** 30 100%	**M:** 0 **C:** 45 100%	**M:** 0 **C:** 30 100%	**M:** 0 **C:** 2 100%
com.example.oasisdocument.exceptions	**M:** 16 **C:** 3 16%	**M:** 0 **C:** 0 100%	**M:** 5 **C:** 1 17%	**M:** 6 **C:** 1 14%	**M:** 5 **C:** 1 17%	**M:** 2 **C:** 1 33%

7.3.2 JIB 协作

项目中配置 JIB 插件，在 maven package 阶段自动进行 docker 镜像的搭建，基于 openjdk 基础镜像，将当前 spring boot 项目构建至阿里云公有镜像库。在 Jenkins 流水线中，进行 maven 的构建之后，镜像均会进行更新。

ssh 至远端后直接进行 docker-compose pull 更新服务端镜像，实现高可用性