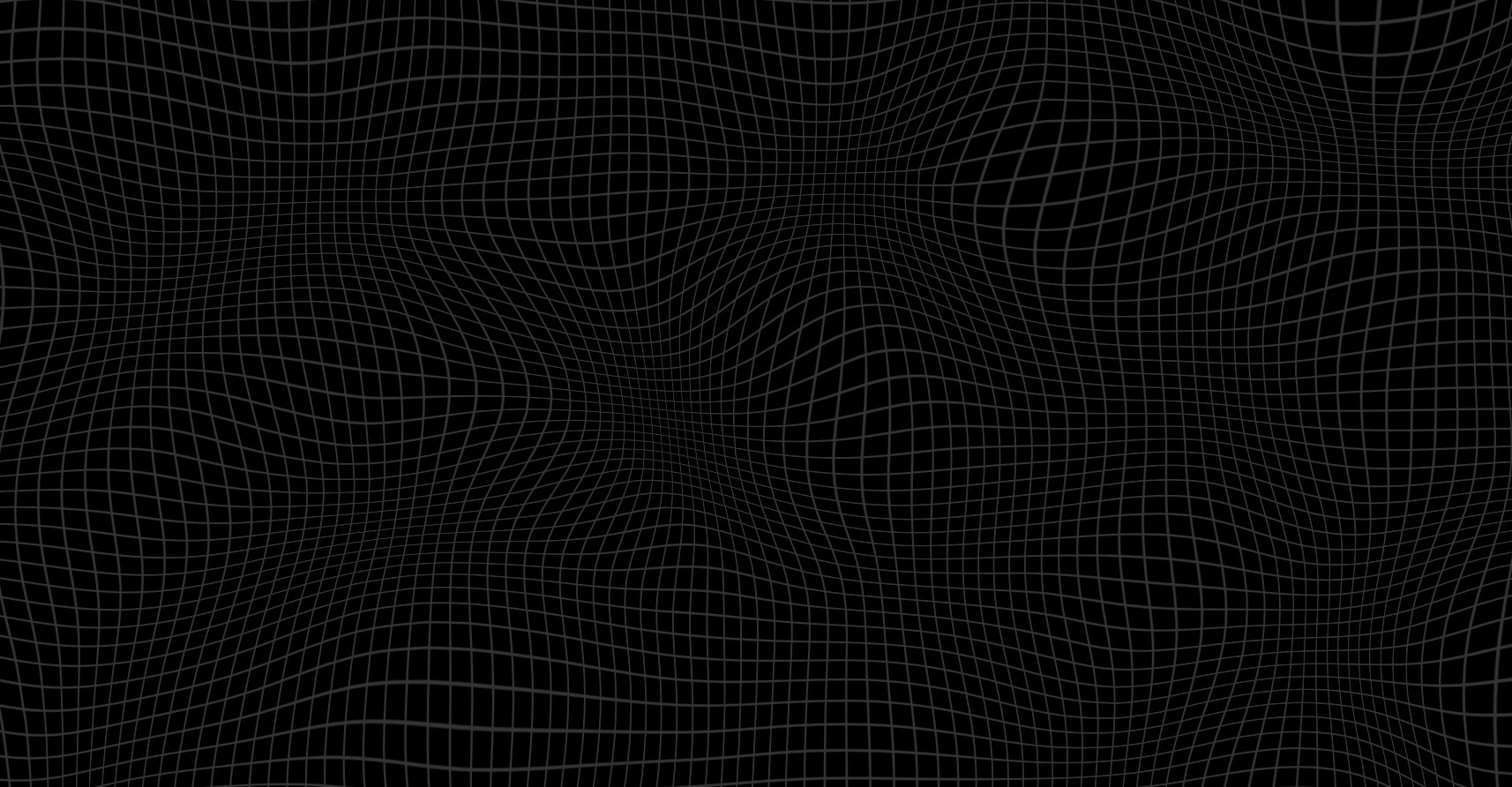


****

**项目后台通用设计文档**

**适用于JAVA开发人员丶前端**

****

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 序号 | 版本 | 作者 | 更新日期 | 备注 |
| 1 | 1.0.0 | 汪志 | 2018/06/07 | 初稿 |

**特别说明：最新更新的规范一般为黄色底纹标示。另外，因还未真正启动项目，此文档可用作大方向构建架构，细节方面的正确性还有待实践，之后会不断更新。**

# 1 项目基础架构

## 主要技术组成

Spring Cloud（eureka，Hystrix，Ribbon，Feign，Config）/dubbo+zookeeper

Spring Boot/Spring+SpringMVC

Mybatis

Maven

Mysql

Redis

Solr

RabbitMQ

FastDFS/FTP

Tomcat/Undertow

log4j/logback

项目前后端分离，后台返回数据格式均为json，跨域问题待实施。

## 新建项目相关

1、一个父工程xxx-parent，其它工程均作为父工程的module，模块按业务规则划分。

2、非业务基础公共模块：

2.1 common项目公共模块，主要包含以下内容，之后每个模块均要依赖common：

①com.xxx.xxx.abstracts包，主要是基础的抽象类或接口，类似于BaseController，IBaseDao等等；

②com.xxx.xxx.annotation包，自定义注解包，自定义的注解主要用于AOP编程、PO类属性等等；

③com.xxx.xxx.constants包，常量包，包括日志记录描述信息常量、消息队列常量、redis缓存常量等等；

④com.xxx.xxx.enumerate包，枚举包，各种枚举类，如各业务的返回错误码；

⑤com.xxx.xxx.POJO包，包含DTO、PO、VO包，DTO为封装请求所带参数的数据传输对象，VO为返回数据的传输对象，PO为持久对象；

⑥com.xxx.xxx.exception包，RuntimeException的子类，自定义异常类。

⑦com.xxx.xxx.utils包，包括各种工具类，字符串、时间等等。

2.2 service-common 项目公共服务模块：

主要为项目内部非业务服务，包含保存日志服务、权限相关服务等等。

2.3 provider-protocol 与web层交互协议服务模块，主要作为暴露接口模块。

2.4 time-task 定时任务模块。

2.5编码中所遇到的有关常量的问题，只要常量有变动的可能，均应放在一个常量类中，其它地方进行引用，当业务有变动，可维护性高。

## 技术点概括

1、Redis

主要用来保存加密通信的秘钥信息（token），用户的登录信息，变动较少的菜单信息等等；

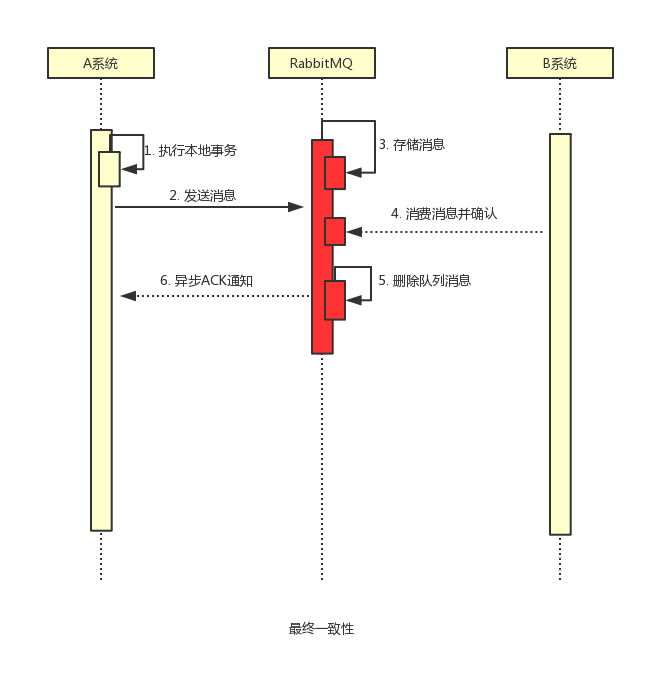
2、RabbitMQ

项目架构设计宗旨是要实现层间低耦合，利于功能的扩展和维护；

所以开发过程杜绝模块间的调用联系，但是又为了增加代码的复用性，使用消息队列作为一个服务层异步通信工具，同时预防了超时的可能性。

**分布式事务的控制：**

时序图：



流程说明：

A系统先本地事务，若失败执行事务回滚，成功则发送消息到队列。

队列会持久化该消息，并推送监听了该队列的B系统去消费。

消费完成后确认消息消费成功。

队列此时删除持久化了的消息。

异步ACK通知消费端是否成功消费。

补充：通常消费端可能会由于各种原因导致消费失败，导致整条线只成功了一半无法满足一致性要求。针对该现象系统引用了死信队列+重试机制来确保一致性，如消费端因消息被拒绝丶消息过期丶队列达到最大长度等原因导致的失败，统统都会被一个新的exchange路由到死信队列中去，此时会有定时任务将死信队列的消息重新消费一次，重试5次。

3、Solr

作为可选工具，如有全文检索。

4、FastDFS

相比于FTP，FastDFS性能更好，更易扩张。

# 2 数据库设计

## 2.1 概要

使用开(mian)源(fei)的MySQL，引擎使用默认的InnoDB，项目数据量如果大了可以集群，表与表之间不使用强制外键，影响写的速度，查询较多的条件字段，可适当添加索引。

## 2.2 基础必要字段

CREATE TABLE `xxx` (

`id` int(11) NOT NULL AUTO\_INCREMENT COMMENT '自增ID',

.

.

.

`status` int(1) DEFAULT '1' COMMENT '状态,默认1，0删除，1启用，2禁用',

`create\_time` datetime DEFAULT NULL COMMENT '创建时间',

`update\_time` datetime DEFAULT NULL COMMENT '更新时间',

PRIMARY KEY (`id`)

) ENGINE=InnoDB DEFAULT CHARSET=utf8 COMMENT='备注';

**业务中对于数据应尽量避免物理删除，应逻辑删除，将状态改为0即可，以免带来恢复数据的麻烦。**

## 2.3 一些基础必要的表

**日志记录表：**

因为数据会较多，应分开两张表，一张记录登录日志(user\_login\_log)，一张记录操作日志(operation\_log);

**数据字典表(sys\_code)：**用于侧边菜单、下拉菜单等等。

CREATE TABLE `sys\_code` (

`code\_id` int(11) NOT NULL AUTO\_INCREMENT COMMENT 'id',

`code\_parent\_id` int(11) NOT NULL COMMENT '父id',

`code\_type` varchar(100) NOT NULL COMMENT '字典类型',

`code\_value` varchar(100) DEFAULT NULL COMMENT '值',

`code\_name` varchar(100) NOT NULL COMMENT '名称',

`code\_alias` varchar(100) DEFAULT NULL COMMENT '别名',

`code\_order` varchar(100) DEFAULT NULL COMMENT '排序号',

`status` int(1) DEFAULT NULL COMMENT '状态 0删除 1正常 2禁用',

`create\_time` date DEFAULT NULL COMMENT '创建时间',

`update\_time` date DEFAULT NULL COMMENT '更新时间',

PRIMARY KEY (`code\_id`)

) ENGINE=InnoDB DEFAULT CHARSET=utf8 COMMENT='数据字典';

PS：开发过程中对于表的改动(新增或删除表、字段以及表中添加必要数据等等)，需列出来以作团队共知。