上海久誉软件系统有限公司
Shanghai Jiuyu Software Systems CO.,Ltd.

上海久誉软件系统有限公司

市域铁路开发约定

2023年4月

技术管理基础

- 构建基础 MAVEN ， NPM
- IDE 随意
- 版本控制 SVN
- 持续构建 Jenkins
- 构建库(私服）Nexus Repository Manager 3
- 静态代码扫描及覆盖率 sonarqube，sonarLint
- 数据库设计工具 pd
- 数据库开发工具 PDManer
- 单元测试 junit5，mockit， database-rider
- 覆盖工具 jacoco
- 项目管理 jira

技术架构

- 以jdk1.8为编译目标,JDK11编译
- JDK17运行
- 部署方式 fat-jar
- 内嵌容器 tomcat
- 数据库 开发 oracle11g，生产ocean base，单元测试 h2
- 字符集　UTF8
- 时区 +0800
- 时间格式（带时区）ISO格式 yyyy-MM-dd ‘T’ HH:mm:ss.zzzZ 如（2017-08-09T07:00:00.000+08:00）
- 密码 国密（SM2，SM4）
- 摘要 SM3

开发框架

- 服务框架Spring Cloud 2021.0.5 （ eureka ，config server）
- ORM框架 MYBATIS（no plus，no ek）
- Socket框架 Netty4.1
- Sftp服务器框架 mina sshd
- 安全框架 Spring Security
- 服务状态监控 Spring boot Admin
- 业务实时监控 Prometheus+Grafana
- 底层节点监控 Zabbix
- 消息队列（rabbitmq）
- Cache 优先顺序 caffine>apache ignite>redis（redis需要项目组讨论批准才可使用）
- Json序列化 Jackson
- 数据库连接池 HikariCP
- 日志框架 slf4j+logback（no loombok）
- 日志收集框架 ELK
- 一级负载均衡 Nginx
- 前置负载均衡 Spring Cloud gateway

开发框架（续）

- 前端框架VUE3
- 浏览器定制 electron 开发以chrome 浏览器为调试标准
- 接口描述 沟通使用doc，开发使用pom依赖管理，固化版本

设计原则

- 全生命周期管理，设计中考虑运维与考虑实施同样重要
- 先设计数据库和接口，后实现业务代码
- 业务流程有补偿机制
- 业务操作的服务间接口有幂等
- 业务流程考虑各服务间的自动核对机制，并有核对异常的处理流程。
- 提供修正异常情况或重做的接口，尽量避免手工修改数据库业务数据的情况。
- 操作流水尽量记入数据库，建议业务请求及应答流水记在一行上并独立事务（insert then update），便于运维及追踪。

insert-(do,update) 两段事务（不调用他人或非关键业务）

insert-do-insert- update-(do, update) 5段事务（调用他人的关键业务）

设计原则（续）

- 多线程运行安全
- 多节点运行安全
- 异常情况多考虑（混沌工程，应急演练）
- 一个事务里不要有微服务调用，防止数据库连接被挂着，事务尽快返回，连接尽快归还
- 业务逻辑不在数据库里
- sql能少join就少join
- Mybatis的sql在xml里，不能在代码里或注解里
  （Mybatis plugin插件和Mybatis tools插件)
- 程序sql定制化，减少动态拼接，只修改必要的字段
- 只访问属于本微服务的数据库对象
- 接口API的javadoc描述尽量清晰
- 少用bean copy，尽量使用插件生成相关代码 如GenerateO2O，GenerateAllSetter

可读可维护性要求

- 代码必有javadoc，准确描述类，函数用途
- 代码注释请用中文描述清楚，当然英文很溜也行，
- Log的输出建议使用英文，防止部署环境和日志收集工具的字符集配置问题。
- Svn上提交日志请用中文描述清楚本次提交内容，后期将jira上bug号或甲方工单号放入

质量要求

- 核心代码必有单元测试
- 单元测试覆盖率 行覆盖70%
- 静态代码扫描 Sonarqube blocker（阻断）级 和 critical（严重）级 为 0
- 以上结果以Jenkins及sonarqube结果为准
- 项目依赖的第三方库尽量使用主流的，更新活跃的，有基金会支持的库。
- 第三方库有的功能，尽量避免使用网上copy的重新实现的代码。如国密算法，使用BC库，而不是网上手搓的代码。

可靠性要求

- 服务接口的输入参数的逻辑边界检查和业务检查一定要有，反正异常输入把服务挂掉的可能。相应边界检查的单元测试要完备、
- 服务尽量不依赖共同的基础服务。（DB是无法避免的）
- 服务都具备水平多节点多活能力
- 防止一个基础服务异常，导致所有业务都受影响的情况。（eurka，rabbitmq是无法避免的）但redis是可以避免不使用的。

部署要求

- 项目引用的jar包都以maven配置依赖，非公开私包需要项目组讨论许可后才放入项目源代码目录中。
- 部署的交付物以jenkins上svn上的代码mvn release出来产物为准，紧急情况下的分支补丁需要特批才能在裸环境拿svn版本用maven命令行进行release 。
- 数据库的建立，变更必须以各自工程的svn里的SQL语句为交付物，含DDL(数据定义语言)DML(数据操纵语言)，并附版本号。
- 配置分为基础，开发环境，测试环境，生产环境，做到与代码无关，项目的交付物可以在测试环境，生产环境无修改直接部署（现在采用spring active profile，启动脚本指定）
- 配置文件格式 yml

安全要求

- mTLS在微服务平台内使用，解决通讯安全及身份识别问题。
- DevSecOps工具链使用，安全左移，解决开发中的安全问题。
- 配置与代码分离，解决部署安全问题。
- 后期会使用spring security解决服务调用接口权限问题（哪些服务能调用我方哪个接口）
- 数据安全，敏感数据的log遮蔽
- 数据安全，敏感数据存储加密
- 数据操作安全，只修改必须修改的数据字段，不要全对象/行更新

性能要求

- 每句sql都要用工具查看执行计划，避免全表扫描
- @transactional 尽量短事务
- 先保证业务代码正确，再考虑优化
- 流水表分区设计，建议按入库时间进行分区，控制单分区大小

审计要求

- 关键业务数据变更必须流水入库
- 静态数据表必须有最后操作时间，请求模块，人员，流水号
- 静态数据变更可通过history模式入库进行，既一次 insert or update 都把变更后数据 copy 到对应history表中

功能交付的完整性标准

什么叫功能开发好了？

- 输入业务参数检查了
- 业务代码完成
- DML,DDL脚本完成
- 静态代码扫描完成
- 用mybatis 日志查看事务边界及输出的SQL（ SQL Params Setter SQL 插件）
- 查看过SQL执行计划并相应索引等sql建立
- 单元测试完成，行覆盖率70%
- 静态代码扫描双0
- 在Jenkins上build确认上述ok
- 可以在jenkins拿到build产出物

完成以上这些，我们的功能才算是完成。

测试环境测试集成测试通过，才是达到交付标准

命名规范

- 在svn上会建立一个csv文件记录公共字典，四列（数据类型，中文描述，英文缩写，数据库列名）
- 服务名随概要设计的模块缩写
- 包名com.jiuyv.shtps.服务名.xxxx
- 接口module　服务名-API
- 服务module 服务名Service
- 变量名 英文单词
- 数据库
  - 表空间 TBS_服务名
  - 表名 TBL_服务名_业务
  - 索引 IDX_表名去掉TBL前缀_xxx
  - 序列 SEQ_表名去掉TBL前缀_xxx
  - 列名 用_作为空格使用
  - 对于取值的列表请在对应列的comment里描写清楚 值，中文描述

谢谢大家！