2020年度工作总结

## 一、CWAP组工作总结：1-4月

### 1、具体负责内容：

1. 基础数据相关。包括：交易账户、债券、参数，场务维护、基础数据同步功能。计算、试算服务功能。RDI校验包及报表生成。API接口数据补全、验证。
2. 成交相关功能。原成交线所有功能。报价、成交、计费、监控、非核心交易功能。

### 2、年初计划改造完成情况：

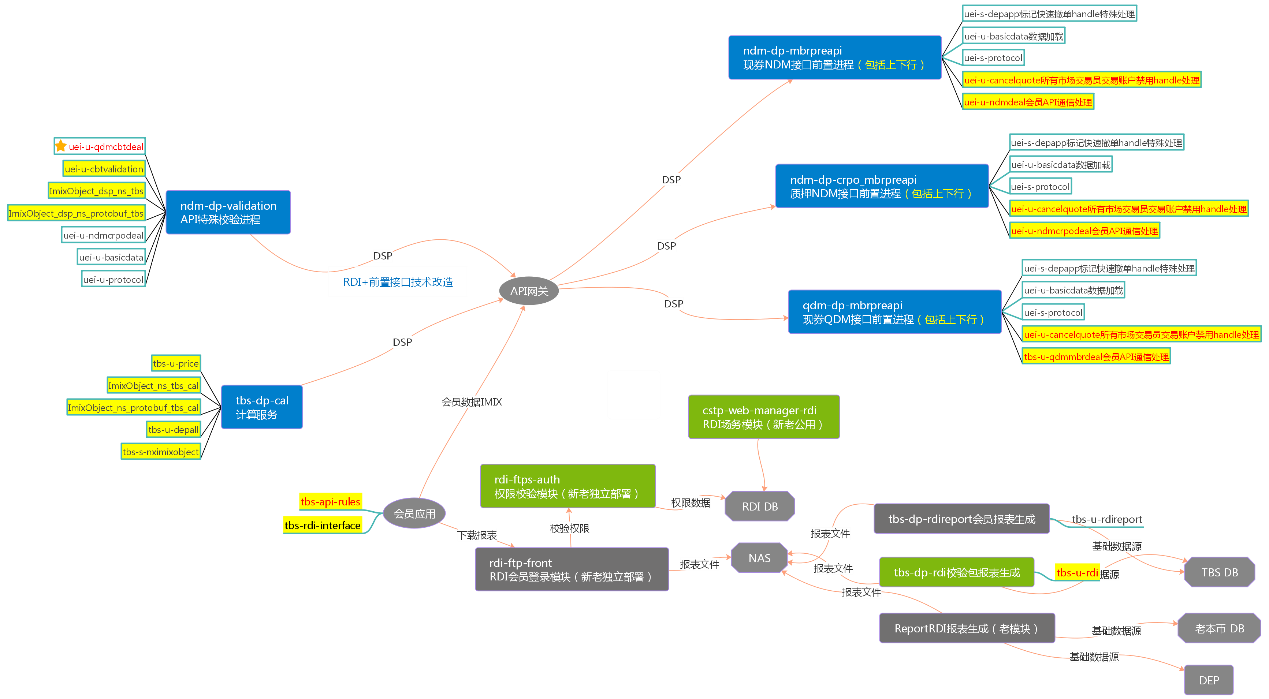
截至2020年5月底，涉及CWAP小组计划改造细化目标14项。其中已完成并上线6项，已完成未上线1项，部分完成4项，未完成2项。

|  |  |
| --- | --- |
| **小项目名称** | **完成情况说明** |
| 数据同步配置化改造 | 已完成未上线 |
| 补齐现券市场NDM、RFQ、RDI校验包机制 | 已完成并上线 |
| 前移现券市场验证与精简基础数据 | 已完成并上线 |
| 清除数据库中冗余的数据，建立定期清理机制 | 已完成并上线 |
| 清理数据库中的无效数据表 | 已完成并上线 |
| 产品中心、机构用户中心服务完善改造 | 部分完成 |
| 预处理基础数据优化-NDM | 已完成并上线 |
| 预处理基础数据优化-ODM | 已完成并上线 |
| 优化生成成交单性能 | 未完成 |
| 流控机制优化，根据实际性能情况动态调整流控 | 未完成 |
| 成交主体共享服务封装 | 部分完成 |
| 识别共享中心聚合逻辑单独实现的模块内容 | 部分完成 |
| 提升增加市场便利性 | 部分完成 |

* 已完成并上线内容主要涉及：

（1）数据库流水数据、冗余数据定时清理。详见《数据分析代办事项-CWAP》。

（2）简化API交易接口链路，取消接口前置服务。参考基础数据为核心前后端提供数据模式，将校验和补全内容，保持API接口上行与客户端前端功能尽可能统一。

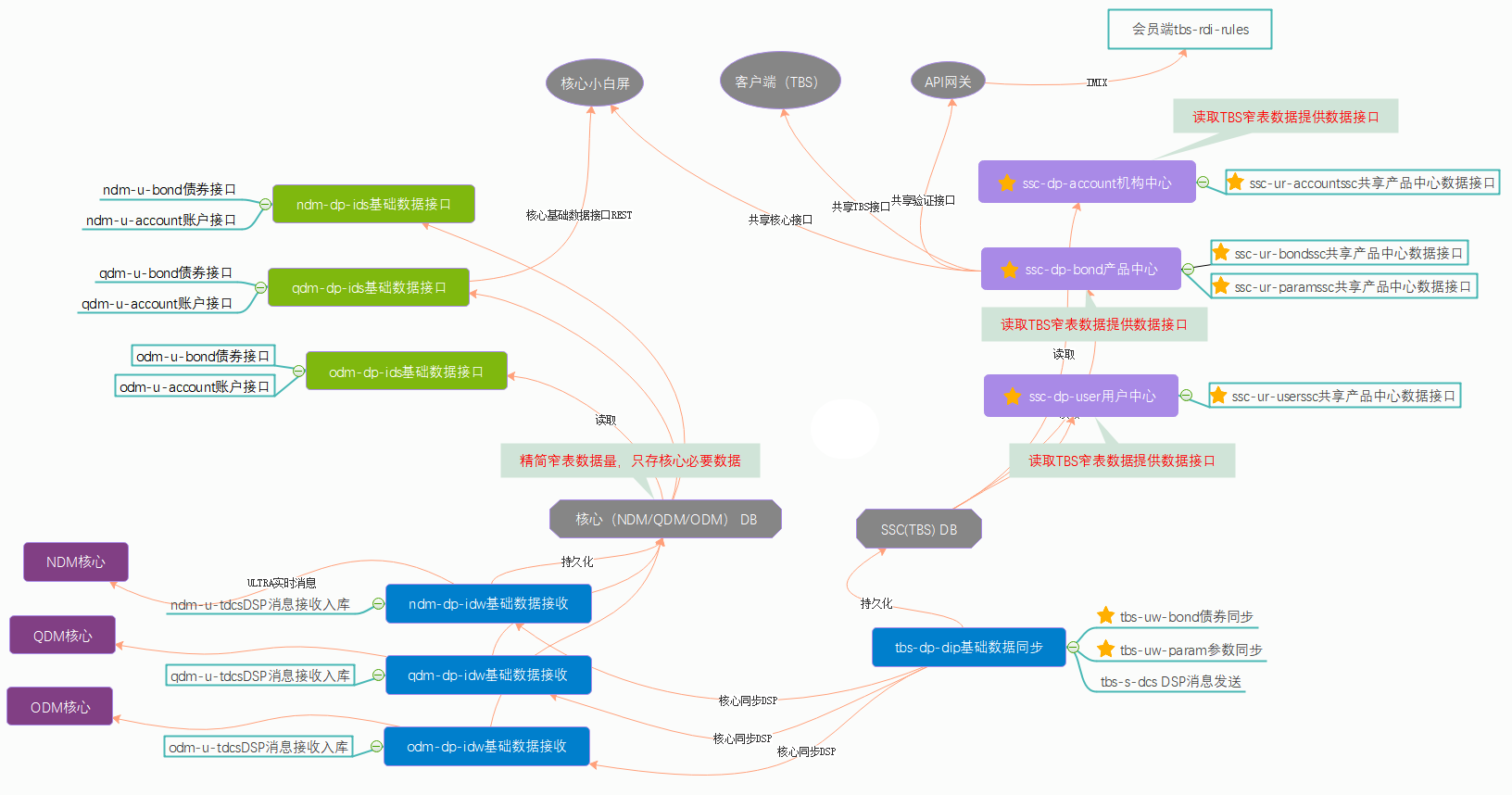


已完成未上线内容主要为“数据同步配置化改造”。通过DATAX引入使得宽窄表同步配置化提高，该方案目前应用于买断式回购市场。

* 部分完成内容主要涉及：

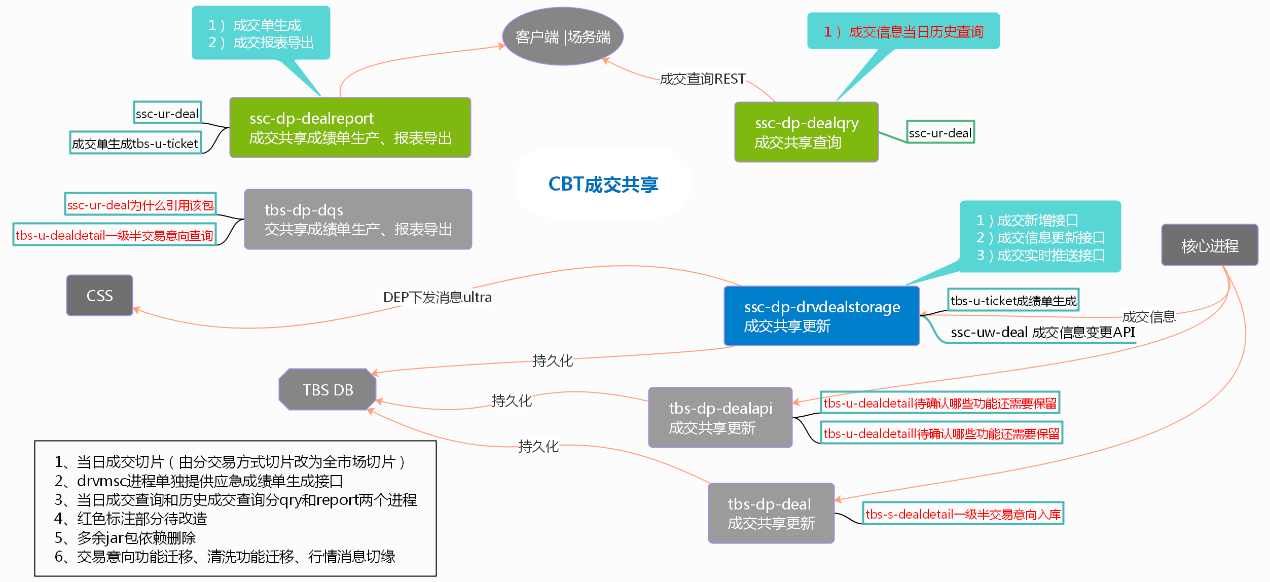
（1）产品中心改造。截至5月底，完成现券市场一期改造内容，将核心后端不涉及基础数据，迁移至共享中心为前端提供数据服务，同时将宽窄表同步逻辑由全量改为增量。目的：简化不必要数据同步。

**未完成部分包括规范原TBS内部基础数据接口调用。将类似债券范围筛选，行情内部数据补全，基础数据搜索逻辑以API接口方式提供其他模块服务。**



（2）成交共享中心改造。截至目前，完成现券市场一期改造内容，将现有成交查询、成绩单、成交持久化、成交状态变更等业务逻辑封装数据服务API。并优化查询、入库性能。回购、IRS、债券借贷已全部按共享模式改造或新建完成。

**未完成部分，原计划成交共享作为TBS内部唯一成交数据源提供数据服务，逐笔行情逻辑、汇总行情数据源还未切换改造。该部分改造重难点：数据清洗逻辑对改造的影响。**



（3）提升增加市场便利性。完成对基础数据、成交组 55个进程，160个jar包的梳理工作，结合共享服务改造计划，罗列需要合并、废弃的组件，已初步形成基础数据、成交相关业务功能改动涉及的组件清单。

**未完成部分，代码模板、雏形代码自动化生成设想未实现。**

* 未完成内容主要涉及：

（1）成绩单性能优化。原计划针对批量成绩单如成绩单未生成，则for循环多次读库逻辑进行优化。经过成交共享改造，成绩单生成在持久化之后进行，因此原循环读库逻辑不会被执行。该优化未安排实现。

（2）流控机制优化。原计划希望丰富API校验包功能，通过校验包推算API报价频次，结合当前全市场报价频率情况，实现对API报价的实时流控。该内容方案细节后续还未考虑全，针对API校验包改造主要放在简化改造。该项目未按计划推进。

## 二、成交&行情组工作总结：5月至今

### 1、具体负责内容：

1. 成交。原成交线所有功能。报价、成交、计费、消息盒子。交易类日终相关批处理。
2. 行情。首页概览行情，报价行情计算，逐笔成交，汇总行情计算。行情外围消息发送。日K线、走势图、加量图、活跃利率曲线、气泡图等图表。
3. 非核心交易功能及类似流程申报功能。匿名拍卖、存单路演、x-bargain、实物交割、债券借贷附加协议等。
4. 头寸风控比例功能。机构头寸、债券持仓、风控比例验证等。

### 2、年初计划改造完成情况：

自6月截至目前，涉及成交&行情小组计划改造细化目标8项。其中已完成2项，部分完成3项，未完成3项。

|  |  |
| --- | --- |
| **小项目名称** | **完成情况说明** |
| 降低XBond向TBS发送的活跃券报价行情消息数量 | 已完成 |
| 组播技术使用规范 | 已完成 |
| 报价主体共享服务封装 | 未完成 |
| 调研分布式框架基础技术解决方案 | 未完成 |
| 消息全链路跟踪及通信优化 | 未完成 |
| 支持基于交易时段消息触发跑批，延迟闭市（22:00前）无需手工调整跑批时间 | 部分完成 |
| 批处理性能优化和调度优化，影响次日交易的批处理在1小时内完成。 | 部分完成 |
| 应急跑批支持指定交易日跑批，且批处理可跨日。 | 部分完成 |

* 已完成内容主要涉及：

（1）活跃券行情订阅优化。将原活跃券每天上午7点全量订阅私有行情，优化为机构登录时才订阅。减少核心行情订阅量。

（2）组播技术使用规范。新增服务均配置动态扩容；标准服务走depall,非标服务走ns\_depall,内部只用非标服务。

* 部分完成内容主要涉及：

（1）批处理改造优化。已完成现券市场批处理支持跨日、支持重复跑改造、调度梳理重排改造、部分批处理性能优化改造。

**未完成内容，现券市场定时刷新逻辑取消croltab方式触发，目前方案已明确待排版本优化。衍生品、回购市场批处理延迟闭式、支持跨日、重复跑改造，债券借贷市场支持跨日改造，目前方案已明确待排版本优化。批处理时长性能优化，例如：数据清洗批处理，行情信息更新，行情清空，头寸计算等。目前部分批处理暂无优化方案，部分批处理待排版本优化详见《成交&行情生产数据分析汇总表》。**

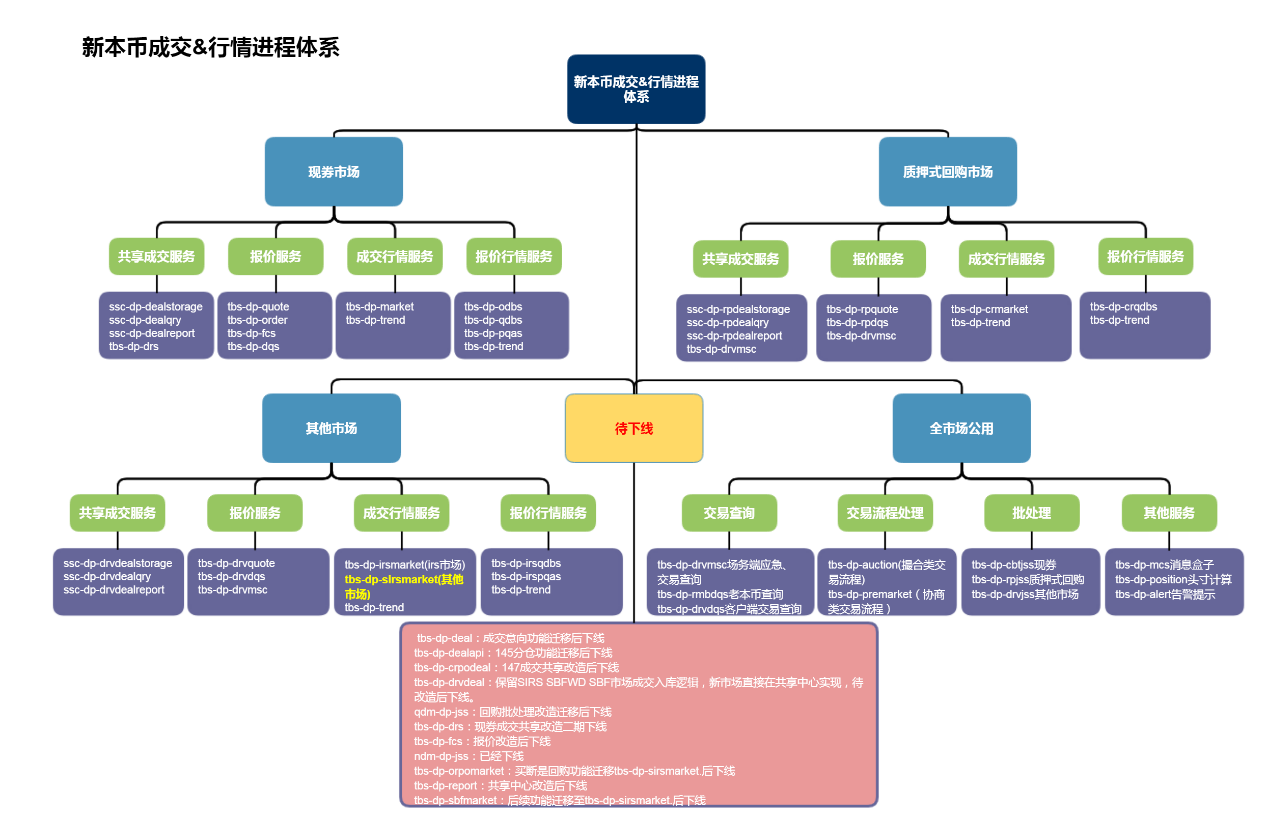
* 未完成内容主要涉及：

（1）报价共享服务。原计划参照成交共享模式，完成对报价数据统一管理并提供服务。目前该部分内容未进行具体设计。项目暂未启动。

（2）消息全链路跟踪及通信优化。原计划年内完成服务链路可视化功能，支持小项目、进程、服务等多粒度的服务链路展示。该部分内容，目前只是借助发版工具进行消息配置准确性检查。暂未实现目标功能。

（3）调研分布式框架基础技术解决方案。原计划年内参考业内主流分布式微服务等框架平台，尝试搭建雏形原型。目前事项暂未启动进行。

## 三、具体负责模块技术完善情况

目前经过三大市场的上线和一轮技术改造。本币交易系统，成交、行情模块体系已经基本定型。按生产业务实际量以及业务功能划分。涉及进程主要划分为现券市场、质押式回购市场、其他市场、全市场公用四大类，详见上图。伴随技术改造，已逐步下线或规划下线8个在线进程。另外，qdm-jss、ndm-jss只保留核心报价按有效时间变更逻辑，其他跑批逻辑迁移至批处理进程。具体各模块情况如下：

共享成交服务：\*\*\*dealstorage、\*\*\*dealqry、\*\*\*dealreport、drvmsc。成交持久化目前通过disruptor、多线程批量入库以提高数据补全效率和入库性能，所有基础数据补全通过基础数据接口调用（含缓存）；成交查询，主要是sql层面，本方对手方逻辑转换，权限条件拼接，基本形成固定模式。针对原系统查询性能有待进一步优化。Qry针对当日数据的查询，report针对历史数据的查询、成绩单、场务查询。Drvmsc场务端成绩单生成和查询。

报价服务：\*\*\*quote、\*\*\*order、\*\*\*dqs、drvmsc。Quote+order，报价与订单的持久化；查询现券dqs、rpdqs、drvdqs。场务查询与应急功能drvmsc。针对回购市场查询、导出已进行SQL优化。衍生品，现券做事报价导出还存在查询性能慢的问题。

成交行情：trend、\*\*\*market。Trend所有行情查询服务。\*\*\*market成交行情的计算持久化，实时推送等（行情数据源油直接接收核心数据改造为从共享中心中接收，逐笔成交表与成交明细表融合）。定时推送，数据清空跑批等技术点有待优化。

报价行情：\*\*\*odbs、\*\*\*qdbs、\*\*\*pqas、trend。目前基于akka架构相对稳定，但针对做市分层、私有行情计算量、Cassandra存储数据合理性上，还有待进一步分析和优化。

其他全市场公用：类交易流程处理，auction进程，撮合类交易流程处理。Premarket,协商类交易流程。批处理，cbtjss、rpjss、drvjss 三个批处理进程。头寸、分控比例，消息盒子，预警进程。

## 四、典型经验教训

### 1、生产问题的教训

（1）越简单细节越容易出问题。

》修数问题

按照业务需求通过脚本插入了两笔成交到成交表。单表两个insert就可以搞定。但巧合的是现券成交共享改造在成交表加了用于逐笔行情清洗用的字段，但该功能又未完成未上线。脚本也没有退。开发按照UAT环境表结构写了脚本，模拟因为已经是142所以表结构是修改后的所以没有该问题。但是在升级生产的时候就报错了。

规避问题途径：

1. 成交共享改造的时候未实现的功能没有剥干净，下掉字段变更万事大吉。
2. Insert脚本只针对必须字段进行填值，该问题会解决。
3. “就是一个插数”，小变更重视程度不够。应该按流程在迁移环境跑一把。

》配置问题

匿名拍卖上线，成交明细下发服务sessionserial配置和msc进程冲突。配置应该是加上成交共享进程分别配置1-6。结果匿名拍卖进程和msc进程都配置的是1、2。原因，在进程提交release的时候就没有把配置变量修改为配置组要求的变量。而是写死了配置。但庆幸的是没有和成交入库进程冲突。所以实际对生产的影响还可控。

规避问题途径：

1. 守门员审核遗漏。
2. 点检工具检查。生产技术验证的遗漏。
3. 小项目过程中的再前进一步。

》其他：成绩单水印、清空数据表批处理、130基础数据上线PPN数据比对。等等

* 教训

巧合永远就是这么巧。

小概率事件发生在自己身上，那概率就是100%。

边界问题，应该都前进一步。

（2）上线第一天数据的依赖。

回想目前我们的生产问题，很大一部分都是上线第一天因为新老平台替换，第一天依赖数据的问题。初始化是一方面，批处理也是一方面。

》批处理执行的时间点

143批处理改造，行情、头寸批处理部分由T日上午迁移到T-1日晚上。而T日交易时段内是会依赖该批处理计算的数据。但评估过程中，应用代码迁移看似无什么影响，却忽略了批处理迁移之后，对于生产T-1日其实还是老版本，批处理不会执行。

规避问题途径：

1. 从项目启动开始守门员+小项目就开始梳理上线第一天依赖的数据项。
2. 尤其是针对线上功能的迁移，不管是迁移进程还是时间点上的迁移。都多想一想上线第一天是否OK。
3. 小项目集成第一天、总集成第一天。所有问题不能简单归结为环境问题、漏脚本、没跑批。针对第一天的问题需要追溯而且组织集中过一次。

* 教训

任何业务需求和技术变更，想想第一天的场景是否支持。

### 2、技术债

1、tbs-s-dealdetail包废弃的残留影响。

2、成交、报价、订单流水等历史表数据量越来越大的解决方案。

3、行情承载能力。

4、目前用户权限树每次全量更新的优化。

5、SQL优化。

6、安全问题、线程池配置的调整。

### 3、小项目过程中的感受

9月份我们组这边出了一个《守门员代码审核标准规范》。借贷小项目结项时，按照规范是不满足结项标准的。其中有一项就是提供所有SQL的Druid分析结果。做这件事情本身没什么额外工作量。外包开发同事一直来问这个能不能不做集成阶段再说。再三坚持后，小项目开发同事按照要求基于生产数据量级做了分析，并且对有问题的内容做了修改，时间不长2-3天时间。后续性能测试，至少这块没有出什么大问题。之前版本出现过很多验收最后时刻，出现类似SQL优化的简单性能问题，但在那个时段，修复和验证的困难程度都翻了好几倍。

技术守门员并不想一味的“卡”小项目结项，但**改一个问题不同阶段的成本区别很大**。这个成本不光是加加班，有修改的影响范围、还有很多不受控的“阻碍”。一些问题一开始发现了，后面可能就会被漏掉（这里因为这个原因还出过生产问题），当然这也反映出我们组对于审核结果修改情况追溯机制的问题。这方面结项的仪式感我觉得还是要有的。早发现问题，早解决。守门员补位小项目是职责，小项目补位守门员是负责。不要把集成定义为填坑。

我觉得针对小项目的成功的定义，延不延期应该算其中一个因素，应该还需要更综合的评价。包括小项目对守门员的评价。比如我自己觉得我交底的内容应该算及格。但实际情况小项目总有能做偏差的，然后等审核出问题后大家再去“赶工”修改。这到底是我交底的问题，还是小项目的问题，感觉现在是少这一层交流和质量的反馈。