



Build a better living



述职报告

部门：软件研发中心-云平台

报告人：Tadonis

日期：2020年11月28日









目录

01 个人简介

02 工作总结

03 工作规划

-  姓 名：Tadonis（唐辉丰）
-  入职时间：2018年11月26日
-  现任职位：产品研发中心-云计算开发架构师
-  工作职责：云平台开发、运维项目交付、平台搭建、团队管理
-  直属上级：Victor
-  教育背景：复旦大学（本科），复旦大学（硕士）

0
2

工作总结



团队建设-现状

- 组织架构:
- 能力分布:
- 应届生培养: 规范化、流程化应届生培养方案

业务成果

- IoT
- 商城后台
 - 完成商城基础架构设计与开发、完成4个微服务框架设计开发。
 - 持续迭代4个大版本，主要完成在线支付接入、运营优惠券、活动等内容
 - 完成erp系统对接，实现订单库存同步以及订单发货处理
- 运营后台
 - 上线菜谱评论/扩区/UGC菜谱/UGC大赛等版本功能，支撑菜谱模块相关业务持续迭代
 - 上线积分项目，包含积分签到、积分抽奖和兑换优惠券等功能，提供良好的扩展性来支持后续迭代
 - 完成Referral、组团抽奖等拉新活动
 - 上线KOL系统，为第三方带货提供可视化平台
 - 上线运营管理系统，提供可视化的营销平台
- 基础平台
 - 上线EDM邮件服务平台一期设计、开发，邮件功能与业务系统实现解耦
 - 完成消息链路追踪系统设计、开发、测试，优化内存后即可上线
 - 完成短信服务平台设计、开发
- 前端
 - (见下页)

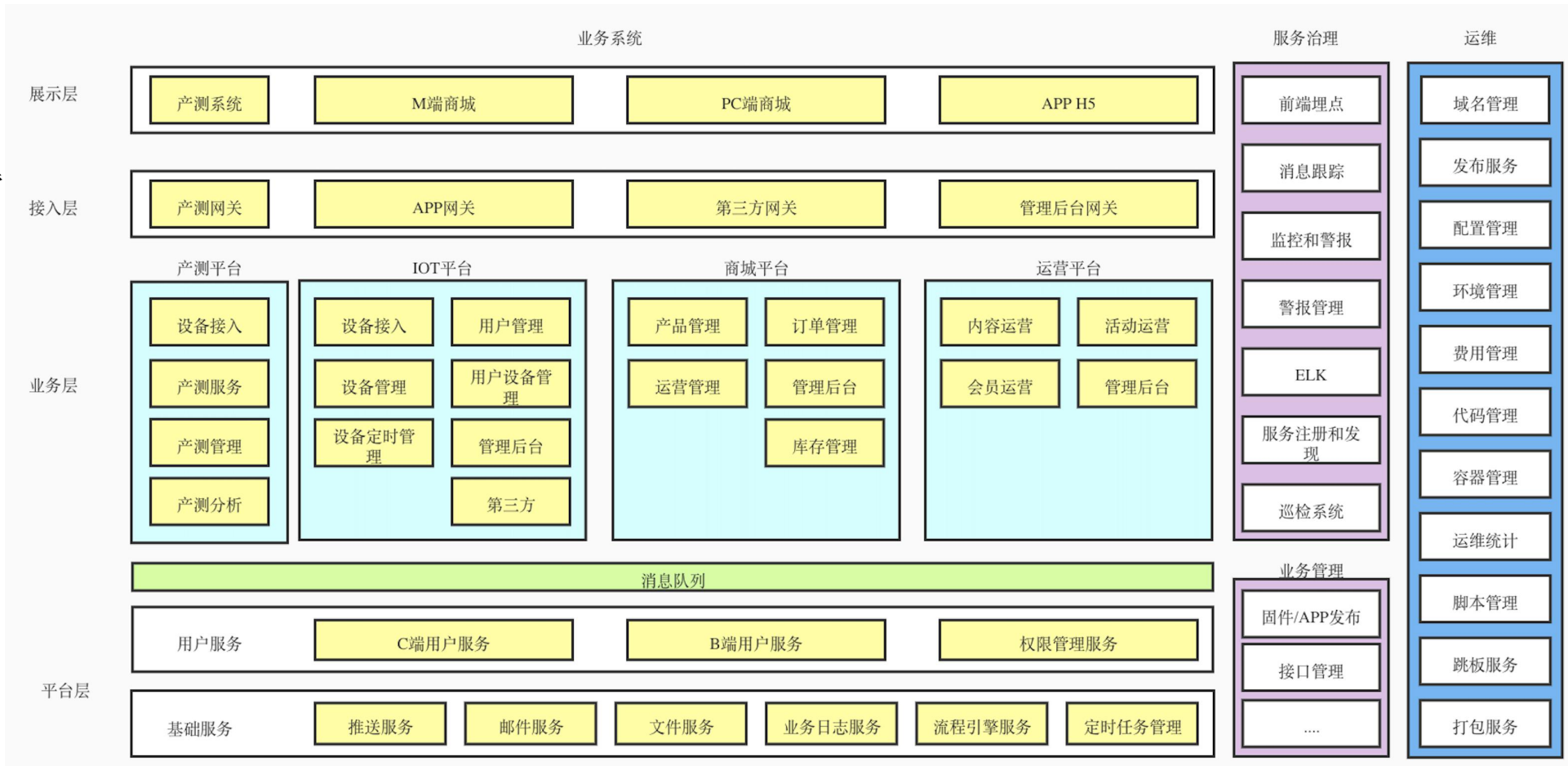
业务成果

► 前端

- 完成商城后台管理系统的搭建，支持了商品管理、订单管理、用户管理、优惠券管理等刚需功能的使用
- 高效完成PC端商城、Mobile端商城从0到1的实现，快速上线以支持运营团队的工作需要
- 快速响应需求，为山火项目实现了山火信息地图查看H5页面
- 支撑了运营后台管理系统、菜谱后台管理系统、积分项目、referral项目、组团抽奖、KOL系统、EDM系统、新森林重建、UGC菜谱、UGC大赛等项目的前端工作
- 安防项目实现了门锁、温控器等新需求的实现
- 实现了多个单品的产测需求
- 实现了多个单品的线上说明书需求
- 支持了包括菜谱分享页、帮助页、隐私政策页、用户协议页、延保落地页、VesyncFit迁移通知页等多个APP H5页面的需求实现和持续维护
- 整理了前端工作规范，从需求接入、排期、开发、提测和上线、问题处理等多个方面帮助大家更好更高效的开展工作
- 整理了前端简单H5页面模版、复杂H5项目模版，使对H5需求的响应更快速更便捷
- 持续整理前端公共代码组件，提高代码的复用性

总体架构

1. 新增了商城业务和运营业务
2. 为了顺应业务发展，独立出基础服务平台和用户管理平台
好处：
1. 团队分工更加明确，人员工作更加聚焦；
2. 减少工作之间的耦合
3. 将公共功能抽离，减少工作量



平台建设-商城

► 难点

- ERP系统对接

ERP 存在时延、性能、可靠性等问题

- 支付功能接入

- 1、需要保证支付流程高可用、低延时；

- 2、需要保证支付结果不掉单；

- 营销业务优惠券

- 1、商城优惠券种类较多，维护比较困哪；

- 2、优惠券应用规则复杂，存在多种设定规则；

- 3、订单使用优惠券规则复杂，分别有门槛和无门槛概念；

- 4、优惠券对订单中商品最终销售价计算规则复杂；

平台建设-IoT

► 难点

平台建设-运营后台

► 新运营管理系统

◇ 产出:

搭建新的运营系统，为公司运营和市场用户建立起桥梁，让公司的运营活动更好地通知到用户。同时提供可视化页面用于观察活动通知的效果，方便后续进行活动复盘。

◇ 优化:

相较于老的推送系统，首先在多方优化推送服务链路的IO和线程池使用，加快推送效率(从每秒推送约100用户到每秒推送约1w用户);其次，在新的运营系统中引入审核机制，分角色权限使用推送系统，各司其职，提高推送质量和安全性。

◇ 后续规划:

1、将banner的UV和PV统计存储方式从MySQL迁移成redis缓存，再定时落库到MySQL。此举大大降低了新banner发布时的对MySQL的压力。

2、运营系统直接对接神策筛选用户，减少从神策导出数据的工作，提高运营效率。

平台建设-运营后台

► 积分模块

◇ 产出：经过三期的功能迭代，已对用户开放积分签到、抽奖、兑换优惠券等功能，即将上线积分抵扣商城订单功能。通过上线养成类项目，提高用户对app的粘性，引导用户在商城下单。

◇ 方案亮点：

- 1、使用redis的bitmap结构存储用户签到信息，仅需46k就能保存用户一年的签到数据；
- 2、以奖池的角度去设计抽奖活动，将抽奖活动抽象成几个功能板块。通过配置奖品参数的方式，大大降低了抽奖活动开发的复杂性。本方案目前在编写技术交底书，准备申请专利；
- 3、使用零存零取的方式管理积分，预留好积分管理的扩展方式，以此应对多变的运营规则；

◇ 后续规划：

- 1、日常的功能迭代
- 2、新增监控服务，每天监测异常的用户数据，对异常数据进行半自动化修补功能
- 3、优化数据最终一致性实现方案，降低业务代码开发量

平台建设-运营后台

► 活动任务模块

◇使用场景： 免费获取积分抽奖任务、集卡牌获取抽奖次数任务

◇ 方案亮点：

- 1、设计出通用的任务表、实现通用的任务统计方式等，抽象运营任务，实现模板化的运营活动开发；
 - 2、定义预触发机制，解决用户使用app时的中断行为导致客户端无法追踪用户任务完成情况的问题。
- 本方案目前在编写技术交底书，准备申请专利；

◇ 后续规划：

- 1、补充周期任务、定期任务的统计处理
- 2、设计kafka动态上下线groupId的方案，自动清理下线活动订阅信息

平台建设-运营后台

➤ 菜谱模块

◇ 产出：逐步开放菜谱评论、菜谱扩区、UGC菜谱、UGC大赛等功能迭代版本，增加用户在运营环节的粘性，促进用户使用、购买设备。

◇ 优化：在业务、技术方案上进行架构整改，解决历史遗留问题造成的当前业务、技术扩展能力受限的束缚，更加合理的设计支撑业务快速更新迭代，研发效率大大提高。

◇ 历史问题和优化方案：

问题	优化方案
菜谱内容分发区域不灵活，不能实现同区域不同国家内容数据共享，且难以支持扩展新的区域	在业务和技术上迭代了菜谱扩区功能，实现了同区域不同国家的菜谱内容共享
菜谱、设备关系通过同名菜谱关联不同设备的方式实现菜谱多设备的业务逻辑，但此方案存在多个不同的菜谱，致使其关联信息不能统一管理，且新的业务需求为此有太多的受限	业务和技术方案进行整改，将菜谱与设备的关系合理化关联，重构菜谱系统业务和技术架构
菜谱数据在各个流程环节使用多套数据库表格进行存储，状态变更时来回拷贝或依赖删除，逻辑变更也必须多套数据库表格及代码实现同时修改，扩展性极差	采用归并数据库表格方案进行整改，让数据库表结构更加接近业务设计，提高其扩展性，便于灵活适应业务需求的扩张和变化
官方菜谱预览使用单独的环境进行预览，浪费服务资源，且扩展性极差，为了节省资源部分环境处于不完整状态，致使测试、升级等环节易于出现差错	采用在后台预览的方案，释放预览对环境的依赖，降低环境成本和维护成本

平台建设-运营后台

► 日常管理安排

- ◇每周安排专员监控线上服务运行情况，对于异常信息通知负责人分析、定位和解决问题
- ◇每周进行一次组内的技术分享，提高小组整体专业水平
- ◇通过分配项目版本号、使用快照版本测试等规定，解决运营多个并行项目开发问题
- ◇遇到线上问题，首先安排后端开发查看线上日志，然后结合前端分析神策数据，定位出具体问题，根据前后端得出的结论，迅速讨论出解决方案
- ◇以极高的标准定义服务警报条件：所有接口内部错误有一个就报警；业务错误结合预期的访问量，适当调整业务错误的警报

平台建设-前端

► 难点

- PC商城支持SEO

经过充分的技术预研，选用Nuxt框架，通过在服务端渲染页面，以及对搜索关键字的设置，很好的支持了SEO需求

- 商城接入第三方统计（联盟推广统计）

平台建设-公共平台

► EDM邮件服务一期

◇ 独立出邮件体系，将邮件功能服务化；将邮件功能从业务系统脱离，解除模块耦合性；邮件运营标准化，提升邮件相关需求响应能力、降低人力资源成本；

◇ 支持通过模板、参数定制化邮件展示；支持邮件人工触发和基于事件触发功能；统计邮件送达率、打开率、点击率、退订率等，反馈重要运营信息；促进邮件运营能力更加灵活多样。

► 消息链路追踪平台

◇ 建设消息链路追踪平台，解决分布式调用链路中发生的异常、高延迟请求；提高研发人员定位问题的响应速度，从原来的平均3分钟以上的时间定位故障发生位置提高到一键搜索定位；

◇ 实现各环境异常、高延迟请求快速精准故障定位、系统各个环节的瓶颈和性能分析、服务链路监控以及调用链路可视化呈现。

业务成果-大数据

► 商城项目

- 完成6个商城项目数据需求，包括App商城一期到四期、pc商城和m端商城一期、新客中心

► 运营项目

- 完成10个运营项目数据需求，包括referral项目、延保项目、UGC大赛、组团抽奖、积分一二三期等

► 大数据平台项目

- 完成6个大数据平台项目数据需求，包括用户标签、用户基础指标、神策数据埋点等

► 数据查询类需求

- 完成82个数据查询类需求，包括商城运营18个，用户运营12个，用户研究5个，商城产品4个，内容运营7个，技术支持6个，IoT产品18个，其他类型12个

业务成果-大数据

► 重点项目-大数据平台

背景

1. 现有数据分散在各个数据系统中，无法集中管理，使用时效率低下
2. 现有架构无法支持对大量数据的可靠存储与快速分析处理

工作内容

1. 搭建基于Hadoop的大数据开发和运行环境，完善平台基础建设，实现对海量数据的可靠存储
2. 实现基于Hive的数据仓库及数据集市建设分析，实现各系统数据的统一规范化管理
3. 搭建基于Spark的快速分析框架，支撑海量数据的快速分析计算

成果：

1. 可满足云平台近两年所有用户和设备数据的存储，且支持快速扩容
2. 为商城、运营等业务部门提供统一的、规范化的数据来源
3. 提供数据的快速分析能力，将常规数据分析的时效性从平均两天提升到一小时

业务成果-大数据



► 项目背景

- 原有数据采集效率不高 (app启动/退出、浏览页面、点击按钮无法精确统计)
- 原有数据计算效率不高 (每次提取数据时, 都需要从原始数据库中导出数据, 进行复杂计算)
- 原有数据分析效率不高 (业务方所需数据散落在各个业务系统和邮件日报中, 需要自己建模)

► 工作内容

- 调研第三方数据分析系统, 经过各方综合考量后, 决定外采神策系统
- 对用户在App内的主要行为以及用户操作IoT设备的行为, 设计完整的数据采集方案
- 开发并实现

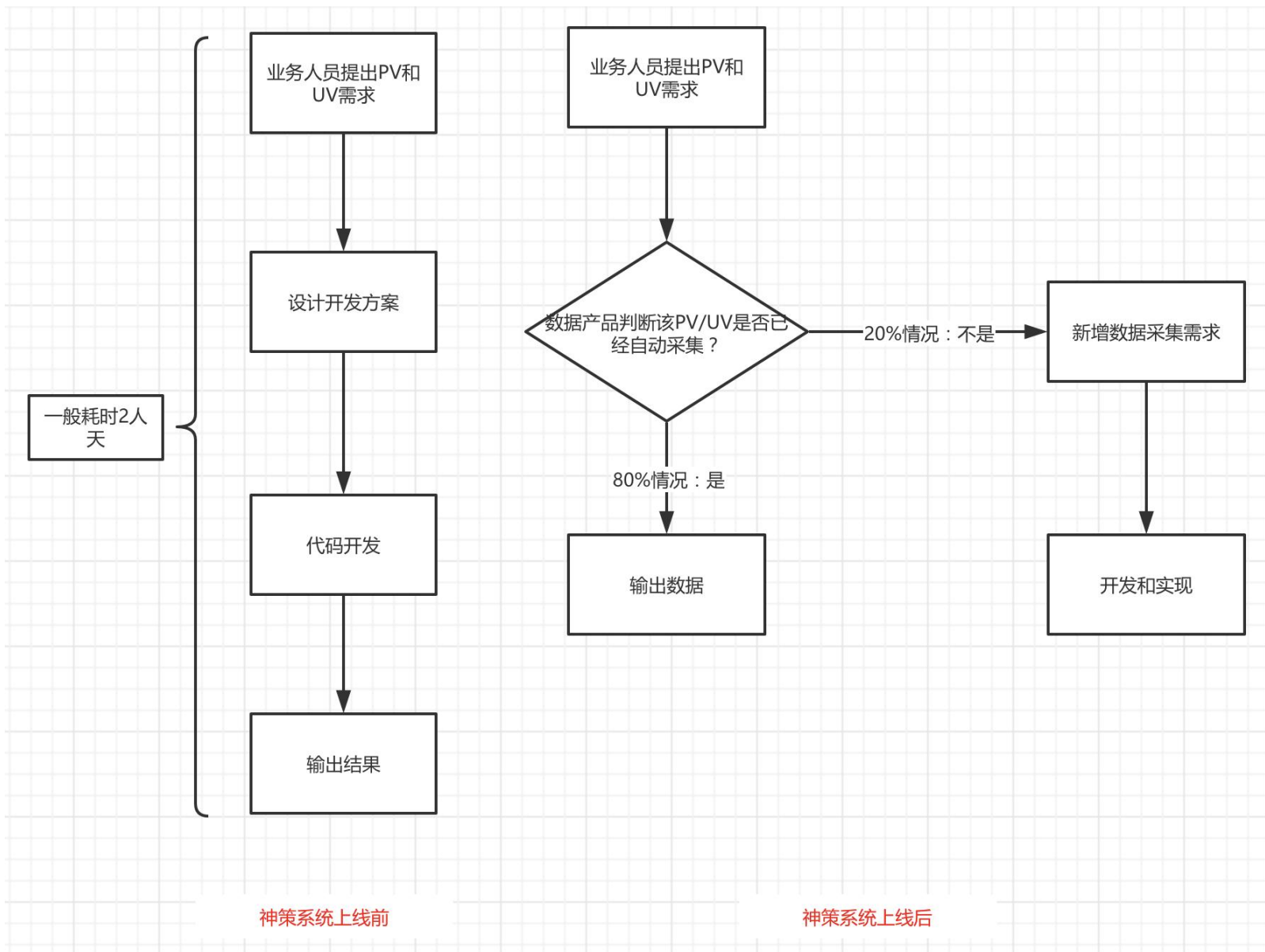
业务成果-大数据

▶ 产出和价值

- 完成了用户在App内的大部分行为数据采集、用户操作IoT设备的行为数据采集，每天以T+1的方式将数据同步到神策系统（每天采集约300w条数据）

- 神策系统上线后，来自业务人员的PV/UV类的数据统计需求减少80%

- 业务人员可在神策系统中使用各种分析模型，将结果保存为看板，方便下次直接查看（当前上线了38个看板，8个报表）



业务成果-大数据

► 项目背景

- 业务方同事对于数据分析工具的使用、数据驱动运营策略不是很熟悉，没有很好的把数据利用起来

► 工作内容

- 开展神策数据分析系统的使用培训，完善各项使用说明
- 从数据分析的角度，给业务方提供一些可落地的具体建议
- 在神策系统中，配置用户标签和用户画像，业务方可直接在各项分析模型中使用

► 产出和价值

- 通过数据分析，提出活动周期调整建议（周期从10日调整为7日），结果：对于活动参与用户，他们10日后留存的百分比从10%提升到35%
- 通过数据分析，支撑用户运营进行决策（App上线了一个每日抽奖送积分的功能后，通过分析用户消耗积分和获取积分的数据后，决定降低抽奖中积分的概率，减少公司成本）
- 通过数据分析，增加商城转化（App积分签到页面的PV在持续增加，在突破1000时，立即在积分签到主页增加商品位推荐，截止目前带来13笔订单的转化）

平台建设-大数据

► 项目背景

- 当前每天都需要发送20+邮件数据日报，不支持手动选择时间段查看数据；
- 神策的看板和概览功能不能满足业务方的全部需求；

► 预期工作内容

- 整理收集各业务方当前和潜在的报表及数据可视化需求；
- 根据数据需求调研第三方BI产品、开源框架等，评估和确定落地方案；
- 开发和落地系统；
- 将当前每天邮件发送的报表逐渐迁移到BI系统

► 预期工作价值

- 提供给到业务方随时查看指定时间段的数据报表功能；
- BI系统支持业务人员自定义报表，且不需要开发人员写代码；
- 补充神策报表功能的不足，方便业务人员更好的做数据洞察和数据可视化；
- BI系统支持各种可视化报表，为管理层做业务决策提供数据支撑；

平台建设-大数据

► 项目背景

- 产品人员在设计产品功能时，预期达到的效果和上线后的效果可能存在差异；
- 对于不同的业务方案，如何挑选出对于业务ROI最大化的方案有难度

► 预期工作内容

- 整理收集各业务方当前和潜在的A/B Test需求；
- 根据需求调研第三方产品、开源框架等，评估和确定落地方案；
- 开发和落地系统；
- 上线系统后，通过系统测试不同的运营方案。

► 预期工作价值

- 通过A/B Test，可以在付出少量开发成本的情况下，过滤掉低价值的产品运营方案，集中力量提升核心业务指标。

平台建设-大数据

► 项目背景

- 在当前的运营过程中，对于运营动作产生的价值和作用进行精准的评估和迭代比较困难；
- 当前商城的产品排序规则比较简单，没有实现价值最大化。

► 预期工作内容

- 搭建精细化商城运营体系，对商城核心指标做进一步的拆解，并关联到各业务方项目落地后的效果评估及对商城整体目标的贡献；
- 通过商品标签和算法规则优化商品排序，提升总体转化率；
- 通过A/B Test系统对不同的方案进行测试，结合智能推送实现用户精细化运营。

► 预期工作价值

- 提升商城总体转化率
- 提升商城业绩
- 提升用户在App内的活跃度

数据安全

- ▶ 数据安全和合规工作文档：
<https://wiki.vesync.cn/pages/viewpage.action?pageId=209813594>
- 1. 建立账号和权限的OA申请流程，并严格执行
 - * 各个系统当前用户统一进行了权限合规修改，统一走OA进行了权限申请
 - * 各个系统管理员减少到1-2人，统一走OA申请权限，签署授权书
 - * 新用户和权限变更均走OA流程申请
- 2. 系统和数据安全
 - * 各个系统均建立了权限管理规范
 - * 各个系统地址统一采用https协议，登录验证，且只能在公司内部访问
 - * 取消线上数据导出和下载功能
 - * 数据脱敏：查询系统和业务日志对用户隐私关键字脱敏
 - * 数据备份：线上数据每天进行备份

业务成果-IOT

► 产品开发:

- 26款新产品开发, 全新品类: 加湿器、水壶、烤箱、网关和血压计

► 41次需求迭代

- VesyncFit迁移: 迁移4款单品, 包括ESF24, ESF28, ESF17, ESF18, 36万的设备, 3360万条用户数据
- 产品联动: 支持加湿器、净化器、插座、灯泡和开关等品类, 毫秒级的设备的联动
- 菜谱重构: 菜谱分布、视频菜谱
- Amazon快速配网 FFS: 打通产测、设备、Vesync云、Alexa云, 实现用户无感的快速配网

► 安防云开发

► 中国区云开发

业务成果-IOT-开放平台项目

背景:

1. 云需要接入第三方厂家产品

内容和难点:

1. 提供新的设备接入层和接入域名, 第三方接入既可以单独域名也能和vesync共用, 增强了扩展性
2. 新接入层tls放在ELB, 释放接入层的加解密压力, 接入能力更强
3. 设备接入协议废弃pid概念, 三方统一使用configModel标识
4. 后期vesync和第三方设备都是用新的接入层

成果:

1. 用新协议成功接入SK50和DS03单品并上线

业务成果-IOT-用户设备关系缓存项目

背景:

1. 用户和设备关系存储在mysql, 每秒查询达到2000次

内容和难点:

将操作用户设备关系的业务统一到一个微服务后, 在这个微服务上做缓存。

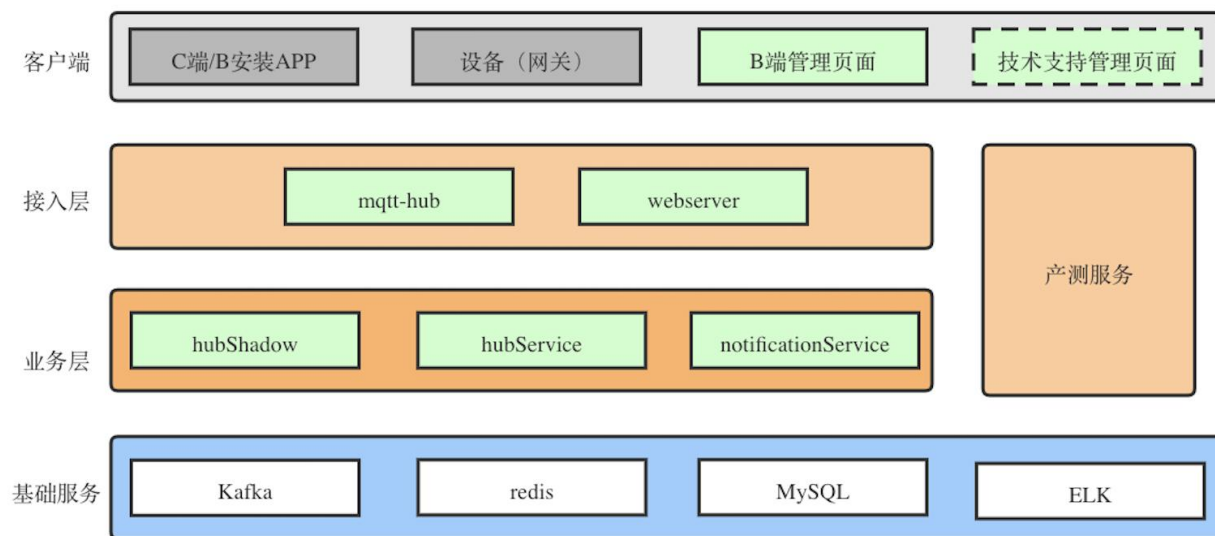
成果:

1. 用户和设备关系存储在mysql, 每秒查询由2500降到50, 降低mysql压力
2. 用户和设备关系查询平均时延由3ms降低到1.3ms

业务成果-

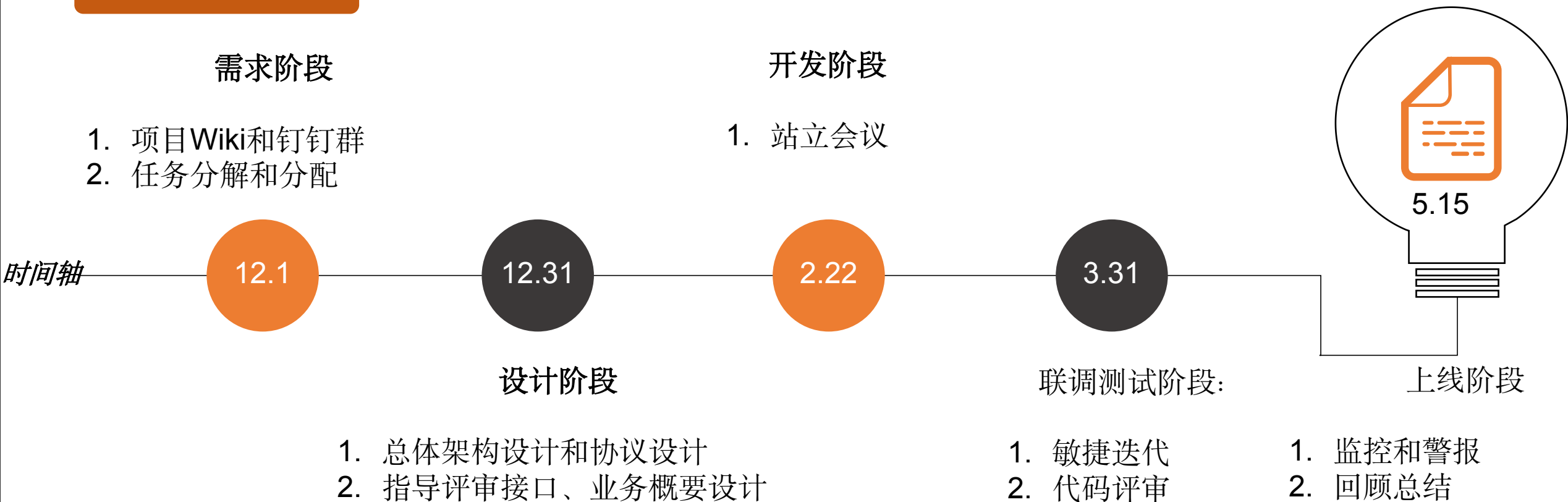
业务成果-安防云

- 架构设计：根据开发周期和业务量，采用简化的三层架构，提高开发效率，减少部署难度
- 调研并使用前端先进技术，如Echarts、datatable、Vue、webpack，ajax，采用前后端分离技术，提高开发和协同效率



网关架构图

业务成果-安防云



业务支撑-业务能力

- ▶ Vesync云业务平台扩展：湿度数据，设备群组，多固件，菜谱，设备联动，家庭和房间、FFS、手机验证码、QQ、微信第三方认证、天气获取等
- ▶ 安防云业务平台扩展：安防警报、B端社区，租客管理、Sim卡管理、网关和子设备管理等
- ▶ 前端开发业务能力：前后端分离，登入登出、表格、图表等
- ▶ 发明专利申请：专利局3项发明专利（2019年1项）

平台演进-思路

► 演进目标：不断提高性能、扩展性、可用性、可维护性，降低成本等

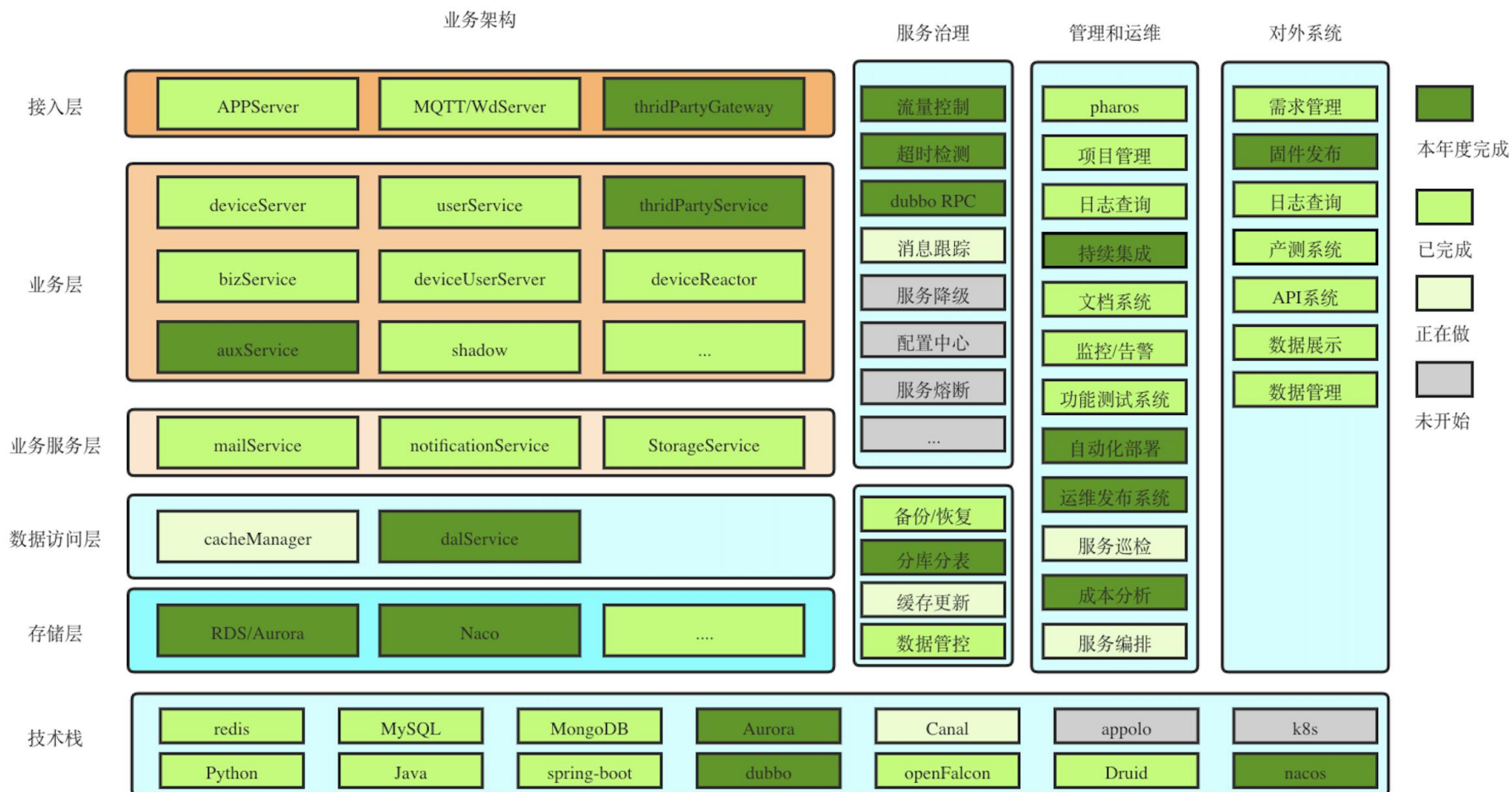
► 设计思路

- 业务需求驱动：比如设备增长、复杂业务实现等
- 通用设计理念：分层、解耦、分布式、高可用、缓存等
- 前沿技术：业界新技术的调研和跟进
- 数据驱动：时延、调用频率等
- 问题驱动：错误日志、警报、用户反馈等
- 从生活类比和思考解决方案

► 成效

- 4个9 的可用性（平均每个用户的不可用时间小于1小时）
- 单品开发效率，从16人日减少到9人日
- 单设备费用从2019年的到0.23美元到目前的0.14美元，降低了39%

平台演进



平台演进

- ▶ 扩展性：降低开发迭代难度，提高开发效率
 - 模块化：第三方接入、鉴权、数据服务等，进行进一步按功能抽象，
 - 代码生成：将菜谱内容和参数解耦，内容不变、参数跟随单品类型改变，开发人日从5到2
 - 代码生成：实现《基于组件化和配置化的产测开发方案》（已申请专利），开发人日从4到2

- ▶ 性能：降低成本、提高用户体验
 - 设备重连流程优化：优化了设备重连的流程，减少云端和设备的交互，减小业务复杂度
 - 设备重连存储优化：将设备认证信息保存到Redis，设备重连不再查询MySQL数据库，性能提高10倍
 - 电量存储优化：从按小时存储到按天存储，数量从100亿下降到4亿；平均时延从400ms下降到100ms
 - 静态资源管理优化：迁移CDN，减少OTA时固件下载时间

平台演进

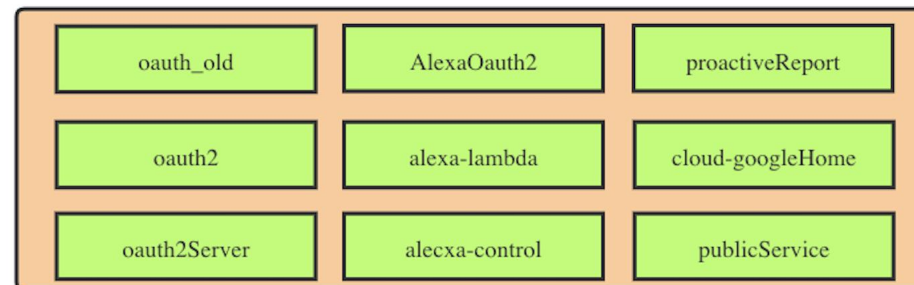
- 可维护性：减小运维难度，减少问题定位难度
 - Dubbo微服务通信：增加服务注册和发现，模块上下线不需要修改ELB
 - 自动化发布系统：优化了自动化发布系统，实现测试环境全自动化发布
- 可用性：降低云平台故障的时间
 - 监控细化：对所有模块的内部错误、错误日志进行监控，报表
 - 警报细化：增加异常流量、异常时延警报；对警报进行分级；
 - 警报通知优化：增加钉钉，和工作更紧密的结合
- 成本：降低服务器的成本
 - 模块化：将非核心、低访问量的模块进行合并，下架了超过10个模块
 - 存储优化：调研和引进业界领先的数据库Aurora，按需付费（中国区）
 - 设备接入重构：通过Netty技术，对非7A设备接入进行重构，大幅度提高性能，节约4台服务器

平台演进-第三方

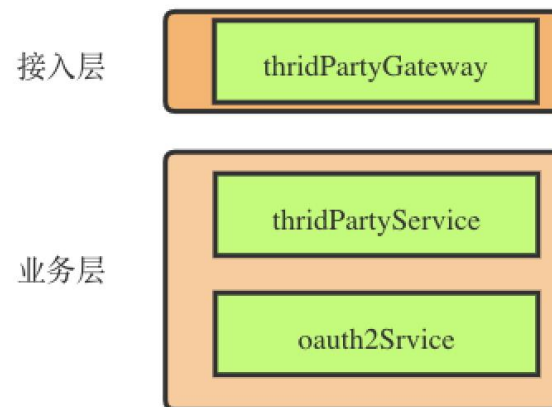
► 背景：模块多，代码冗余、交互复杂、开发迭代难度大，容易出错，问题定位难

► 举措1：从需求出发，彻底重构！

- 业务抽象
- 分层架构设计，按功能进行抽象和模块化
- 统一接入层的日志、监控



第三方服务（老）



第三方服务（新）

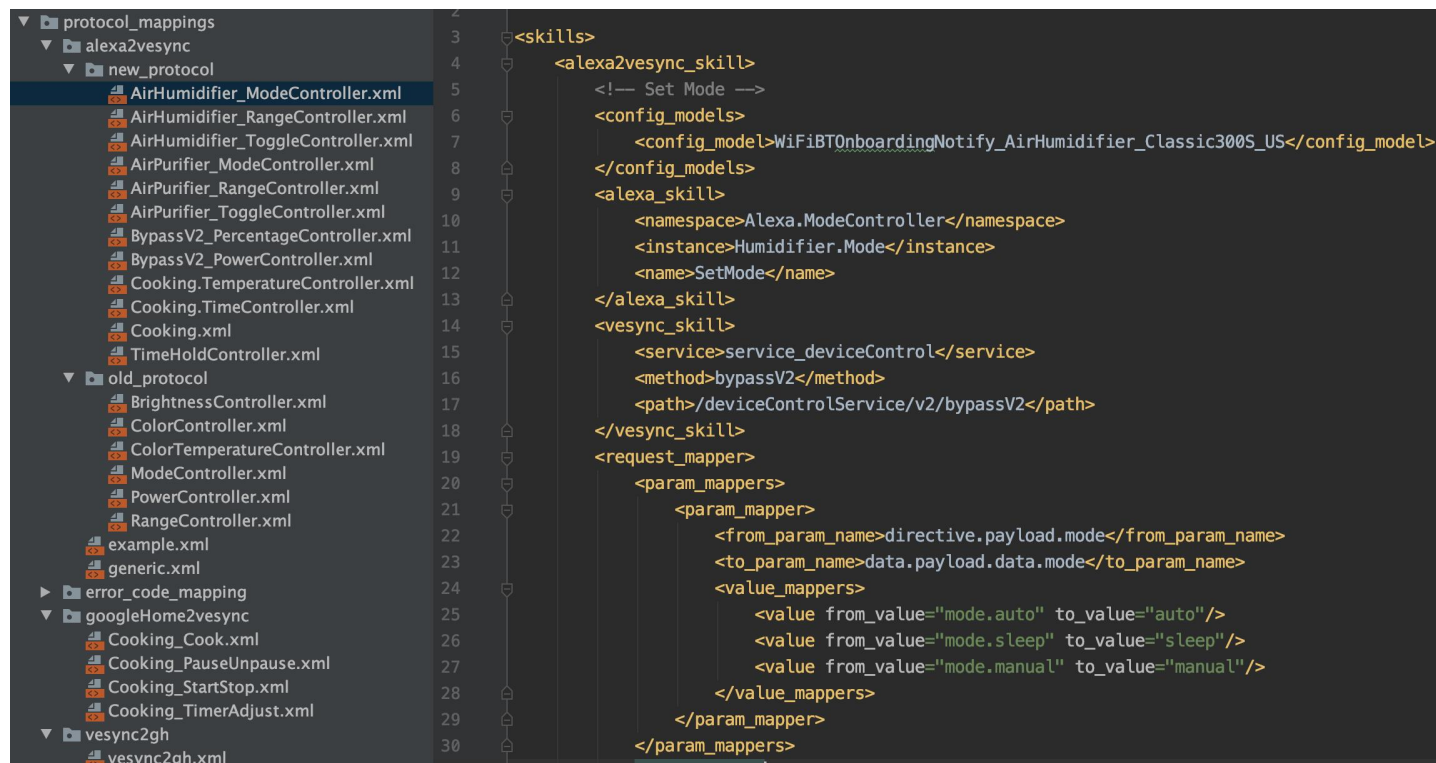
平台演进-第三方

► 举措2

- 将协议和功能进行解耦
- 通过配置文件实现协议转换, 实现无代码开发

► 成效

- 扩展性: 开发更加简单, 开发效率更高: 5个人日到0.5个人日
- 成本: 从9个模块到3个模块



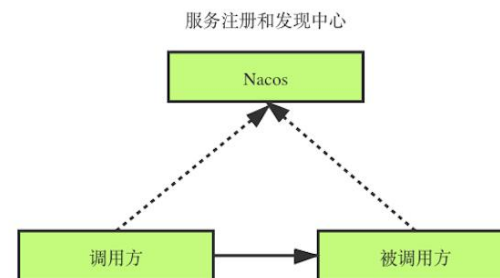
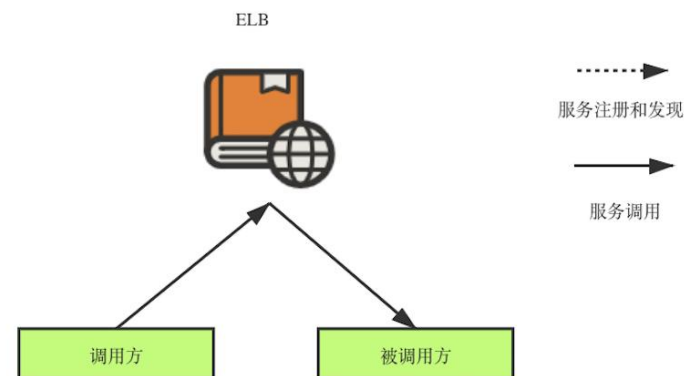
The screenshot displays a code editor with two panels. The left panel shows a file explorer with a directory structure for protocol mappings. The right panel shows the content of a selected XML file, which defines a skill and its request mappings.

```
protocol_mappings
├── alexa2vesync
│   └── new_protocol
│       ├── AirHumidifier_ModeController.xml
│       ├── AirHumidifier_RangeController.xml
│       ├── AirHumidifier_ToggleController.xml
│       ├── AirPurifier_ModeController.xml
│       ├── AirPurifier_RangeController.xml
│       ├── AirPurifier_ToggleController.xml
│       ├── BypassV2_PercentageController.xml
│       ├── BypassV2_PowerController.xml
│       ├── Cooking.TemperatureController.xml
│       ├── Cooking.TimeController.xml
│       ├── Cooking.xml
│       ├── TimeHoldController.xml
│   └── old_protocol
│       ├── BrightnessController.xml
│       ├── ColorController.xml
│       ├── ColorTemperatureController.xml
│       ├── ModeController.xml
│       ├── PowerController.xml
│       ├── RangeController.xml
│       ├── example.xml
│       └── generic.xml
├── error_code_mapping
├── googleHome2vesync
│   ├── Cooking_Cook.xml
│   ├── Cooking_PauseUnpause.xml
│   ├── Cooking_StartStop.xml
│   └── Cooking_TimerAdjust.xml
└── vesync2gh
    └── vesync2gh.xml
```

```
<skills>
  <alexa2vesync_skill>
    <!-- Set Mode -->
    <config_models>
      <config_model>WiFiBTOnboardingNotify_AirHumidifier_Classic300S_US</config_model>
    </config_models>
    <alexa_skill>
      <namespace>Alexa.ModeController</namespace>
      <instance>Humidifier.Mode</instance>
      <name>SetMode</name>
    </alexa_skill>
    <vesync_skill>
      <service>service_deviceControl</service>
      <method>bypassV2</method>
      <path>/deviceControlService/v2/bypassV2</path>
    </vesync_skill>
    <request_mapper>
      <param_mappers>
        <param_mapper>
          <from_param_name>directive.payload.mode</from_param_name>
          <to_param_name>data.payload.data.mode</to_param_name>
          <value_mappers>
            <value from_value="mode.auto" to_value="auto"/>
            <value from_value="mode.sleep" to_value="sleep"/>
            <value from_value="mode.manual" to_value="manual"/>
          </value_mappers>
        </param_mapper>
      </param_mappers>
    </request_mapper>
  </alexa2vesync_skill>
</skills>
```

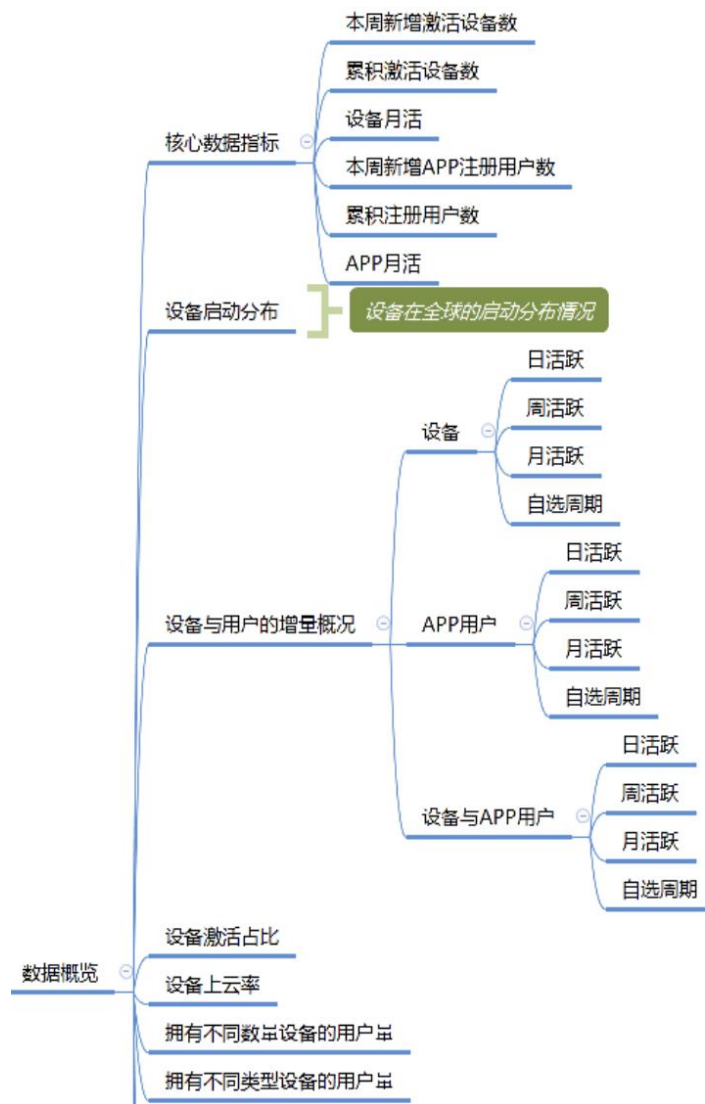
平台演进-dubbo

- 背景：模块通信通过ELB转发，每次部署，都需要对ELB进行配置，耗时耗力，比较容易出错；ELB本身有时候也有故障，导致服务不可用
- 举措：增加服务注册和发现，由统一路由改为直接调用
- 成效：
 - 可维护性：部署无需进行ELB路由更改，更加简单
 - 性能：消除中间调用，减少调用时延，约0.5ms
 - 可用性：消除ELB的使用，避免ELB不可用带来的故障
 - 可扩展性：方便微服务治理的扩展



数据分析

- 大数据平台构建
 - 数据清洗和数据存储
 - 数据分析
- 数据挖掘和分析：用户/设备的数量/活跃度、设备地区分布、用户拥有设备数量等
- 数据查询和展示系统
 - 日志查询：设备ID、用户ID、时间等条件进行查询
 - 展示：用户、设备活跃度、设备地区分布、用户拥有设备数量等
 - 产测数据：生产数量、生产日期、设备ID、固件版本号



安全合规

► 数据管理规范

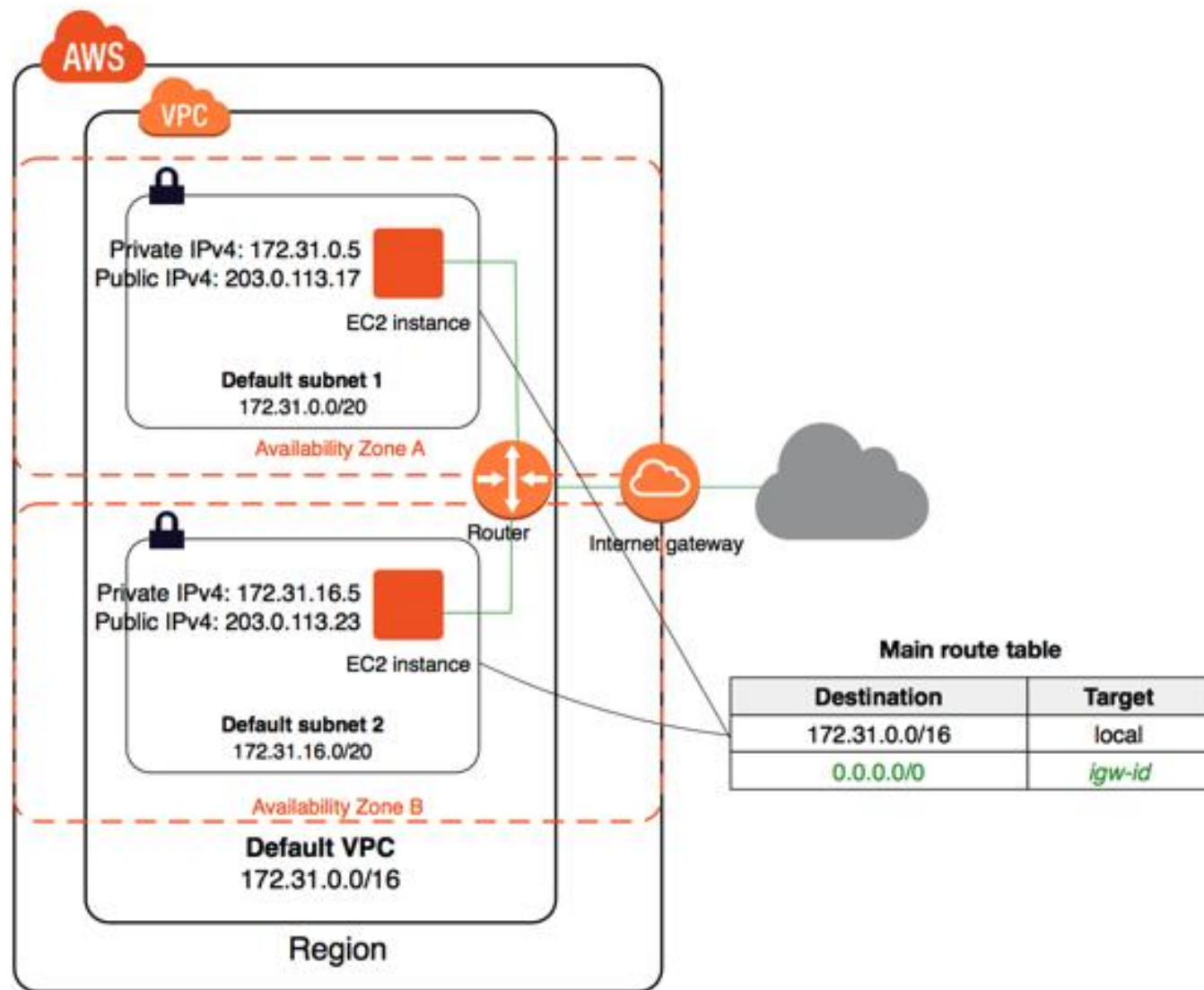
- 数据库管理规范（权限最小化）
- 日志管理规范

► 部署架构

- 业界领先的VPC技术
- 测试和线上隔离

► 其他:

- 日志掩盖用户敏感信息



不足和反思

- 团队建设有待加强
 - 未能完成招聘指标，入职率较低
 - 团队目前整体人才梯队高端人才不足，特别是架构师级人才太少
- 先进技术投入精力不够
- 产品思维不足：思考问题多从产品用户角度出发，而不是仅仅从技术出发

03 工作规划



业务规划

- ▶ 战略： 通过与用户的深度链接，科技的持续创新，硬件、软件、内容的一体化服务，为全球家庭提供健康智慧的生活方式
- ▶ 业务规划
 - 支持各产品线的产品开发和迭代（持续），实现成熟产品无代码开发（Q3）
 - 智能业务：商城、社区、用户运营等后台开发（持续）；
 - 开放平台：初步设计和完成开放平台（Q2）；
 - 大数据分析：参与大数据分析团队的团队建设，业务规划（Q1）；

大数据分析

大数据分析：通过数据推动业务发展

- 通过引进Hadoop、Spark、ClickHouse等大数据分析技术，持续提高云平台大数据处处理和分析能力（Q1）
- 对用户行为、设备行为等数据进行建模，然后将数据导入OLAP系统，支持实时的数据分析（Q2）
- 数据挖掘，协同产品、营销等部门，对用户、设备、review、反馈等数据进行挖掘，为产品定义和业务决策提供更多数据支撑，通过数据，推动产品持续提升(持续)
- 建设数据查询、展示和分析系统，方便其他团队进行数据获取和分析（持续）
- 数据权限管理，对不同团队、层级设置不同权限，确保数据的安全性

技术规划

- ▶ 调研并引进业界先进的缓存管理技术Canal，提高缓存的一致性（Q1）
- ▶ 基于通用的消息跟踪技术，结合我们已经申请的专利，《一种分布式慢请求和错误请求的跟踪和警报方法》，构建自研的消息跟踪系统，提高云平台问题发现和定位的能力（Q2）
- ▶ 调研并引进容器编排技术K8s，容器管理系统Rancher，实现微服务容器化，提高系统运维的能力，减少代码开发到发布的时间（Q2）
- ▶ 设计并实现灰度发布、服务降级、服务熔断和请求限时处理等分布式服务治理手段，其中请求超时的设计，我们已经申请了专利《一种分布式请求限时处理的方法》（Q4）

新技术研究

- AI和机器学习技术调研，实现用户精准运营、智能客服、运营推广预测等
- 进行更多第三方语音的调研，如apple HomeKit，阿里巴巴的天猫精灵，提高产品竞争力
- ServiceMesh技术的研究和应用，将服务治理和业务解耦，提高开发效率，平台演进效率
- 新产品支撑：
 - IPC相关技术：直播、视频流、图像识别、编解码等
 - 扫地机器人：地图识别算法、导航算法等

团队建设

- 人才梯队建设：重点招聘和培养3名架构师，以满足明年智能业务的发展（持续）
- 管理人才培养：优化分工和职责，提升个别小组长的管理范围和权限，培养中间管理梯队（Q1）
- 等级和晋升制度：确定人员等级和晋升制度，在团队内部宣导人才能力模型和晋升制度，提高平台战斗力（Q1）
- 流程规范
 - 对新的技术、业务制定流程规范（持续）
 - 对已有的流程规范持续跟进，确保流程规范的落地（持续）
- 拥抱变化：转变思维，积极拥抱组织和业务的变化

A top-down view of a desk with various items. In the upper left, a white laptop keyboard is visible. To its right is a row of colorful books standing upright. Below the laptop, a magazine titled 'iPod и iPod mini' is partially open. In the lower left, a spiral-bound notebook with a pencil and an eraser lies on it. In the center, a smartphone is placed on top of an open magazine. The magazine pages show logos for 'Quark', 'conEdison', and 'University Science Park'. The background is a light-colored wooden desk.

谢谢