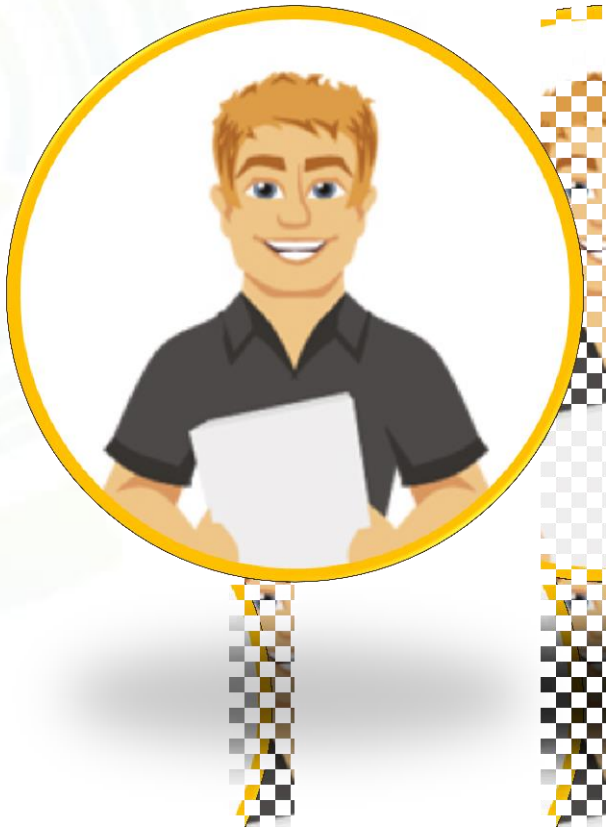
QAV自动埋点技术分享

自我介绍



- 姓名：周锦锋
- 部门：平台事业部
- 简要介绍：承担Qunar iOS基础组件和架构相关开发工作

目录

一、QAV自动埋点背景和优势

QAV自动埋点简介

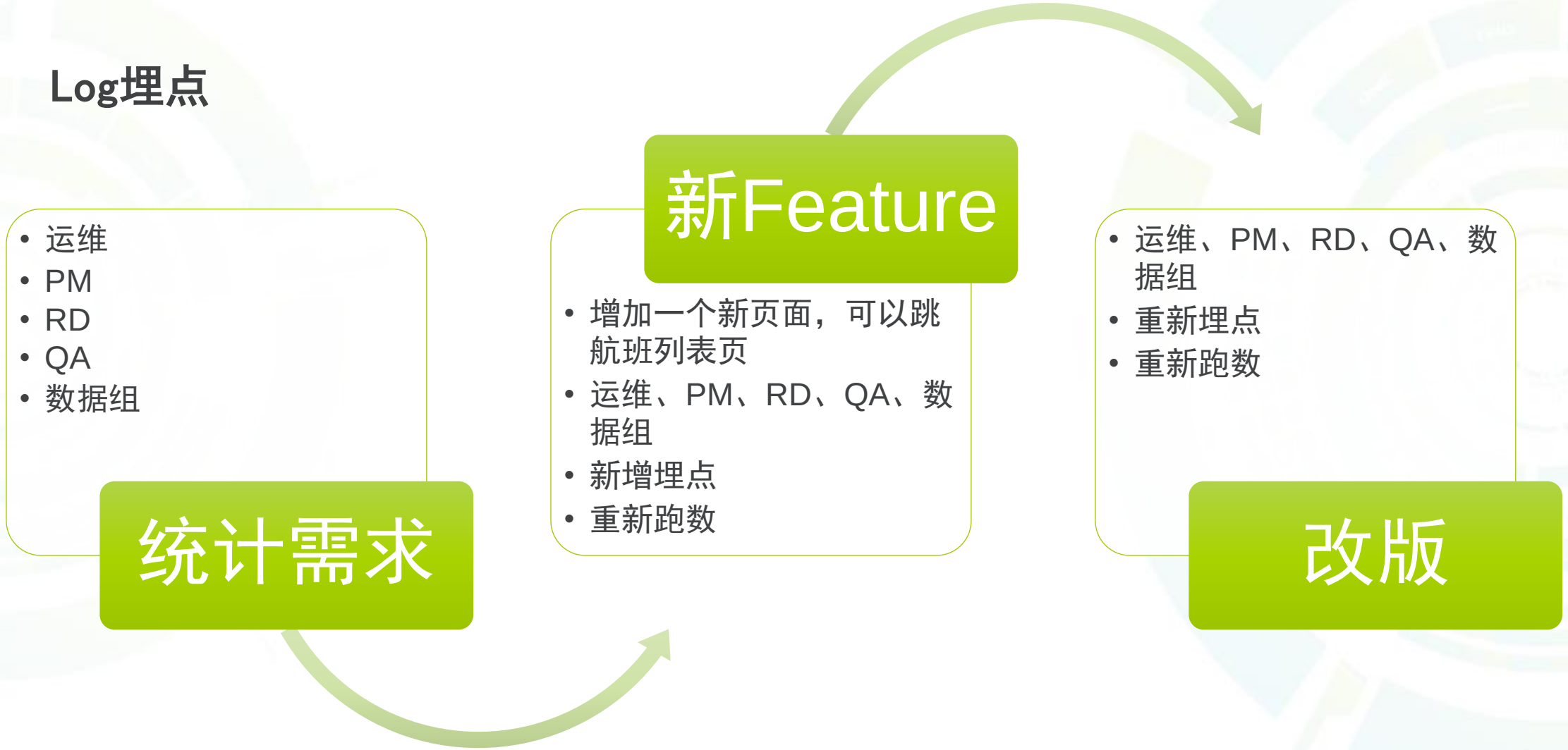
QAV是一个自动可视化统计系统

是一个手机客户端的用户行为操作和路径信息的采集、处理、存储、统计、管理的一整套解决方案。

工作方式的变化

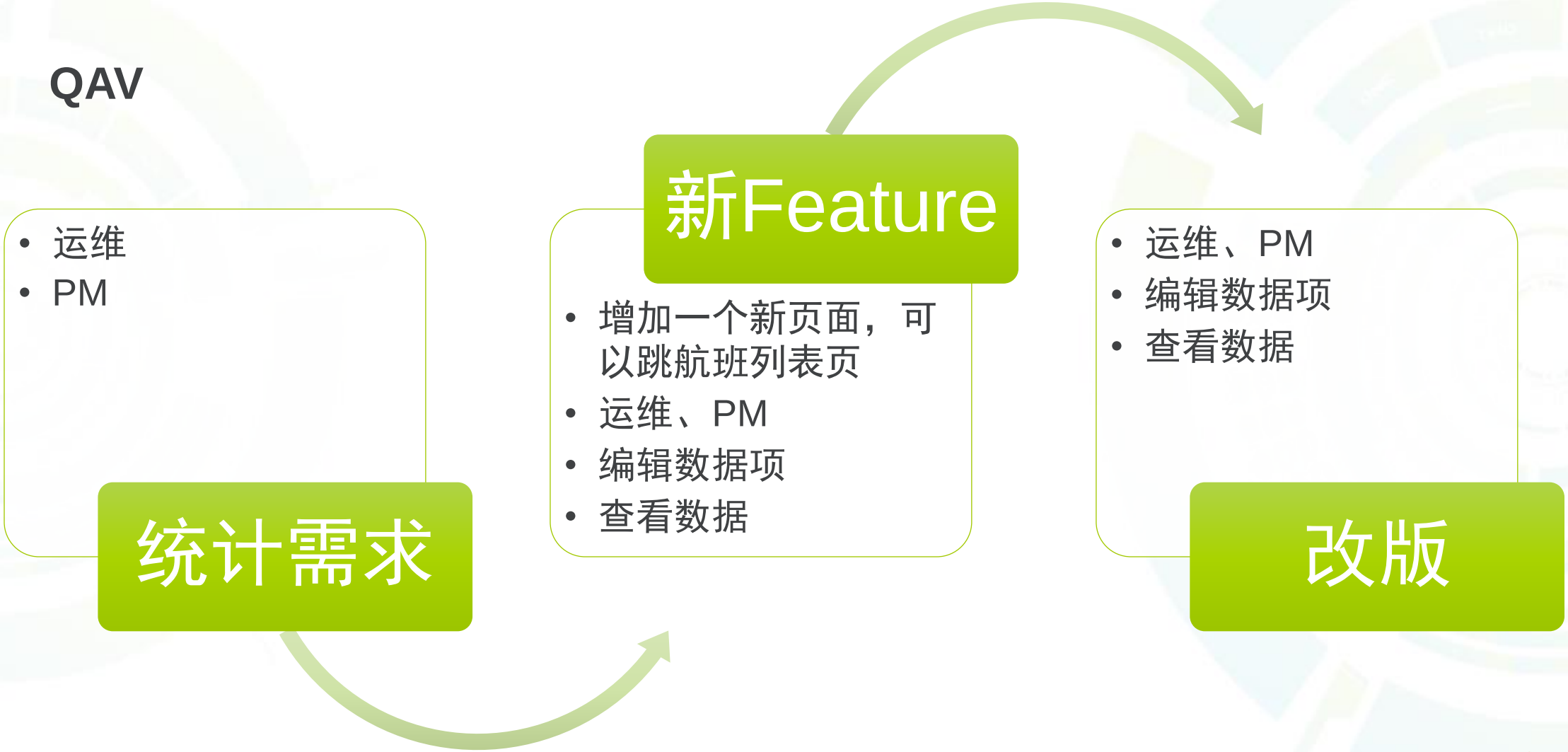
举个□ -进入机票航班列表页面的流量统计

Log埋点



举个□ -进入机票航班列表页面的流量统计

QAV



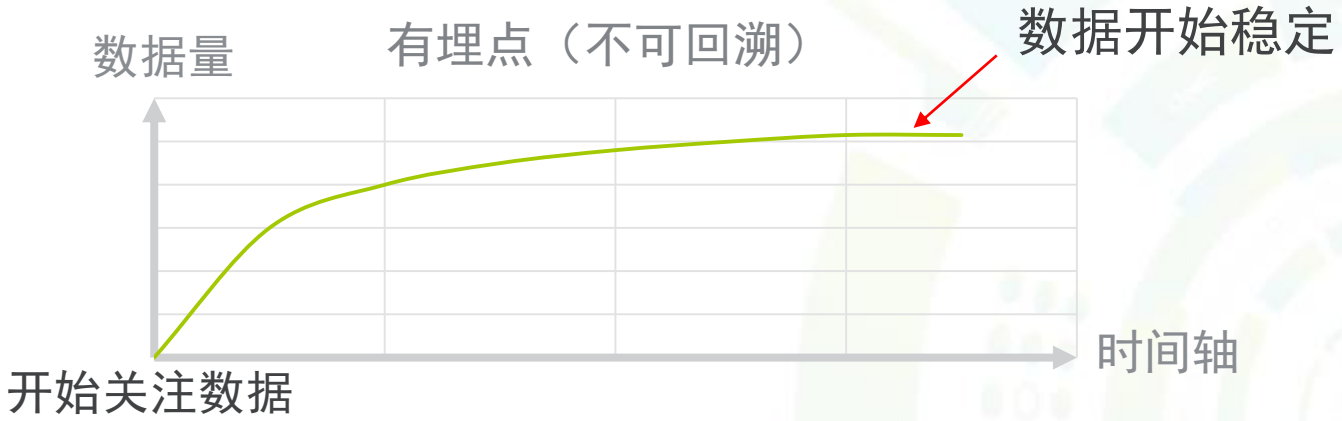
Log埋点统计遇到的问题

- 1、人工埋点，在业务代码中加入埋点代码
- 2、容易出错和遗漏，数据很难回溯
- 3、统计需求参与链条长，效率低
- 4、不能适应需求变化
- 5、迭代维护成本高
- 6、无法所见即所得，不直观

QAV的优势

- 1、自动埋点，无需人工埋点
- 2、全量采集，数据可回溯
- 3、效率高
- 4、很好的满足需求变化
- 5、迭代维护成本低
- 6、所见即所得，看图说话

全量采集解决数据『 回溯 』问题



发现不合适的交互


登录12306账号
首次登录立送30元火车票立减券
账号 手机号/用户名/邮箱
密码 输入密码
登 录
注册账号
忘记密码
12306账号购票专享特权
极速退改
无需核验
专享VIP抢票


极速退改
无需核验
专享VIP抢票
180
150
120
90
60
2017-06-23
PV : 195
UV : 61

统计实现方案的比较

- **有代码埋点** Google Analytics、百度统计、友盟、TalkingData 等为代表

```
- (void)viewWillAppear:(BOOL)animated
{
    [super viewWillAppear:animated];
    [MobClick beginLogPageView:@"PageOne"];//("PageOne"为页面名称,
}
- (void)viewWillDisappear:(BOOL)animated
{
    [super viewWillDisappear:animated];
    [MobClick endLogPageView:@"PageOne"];
}
```

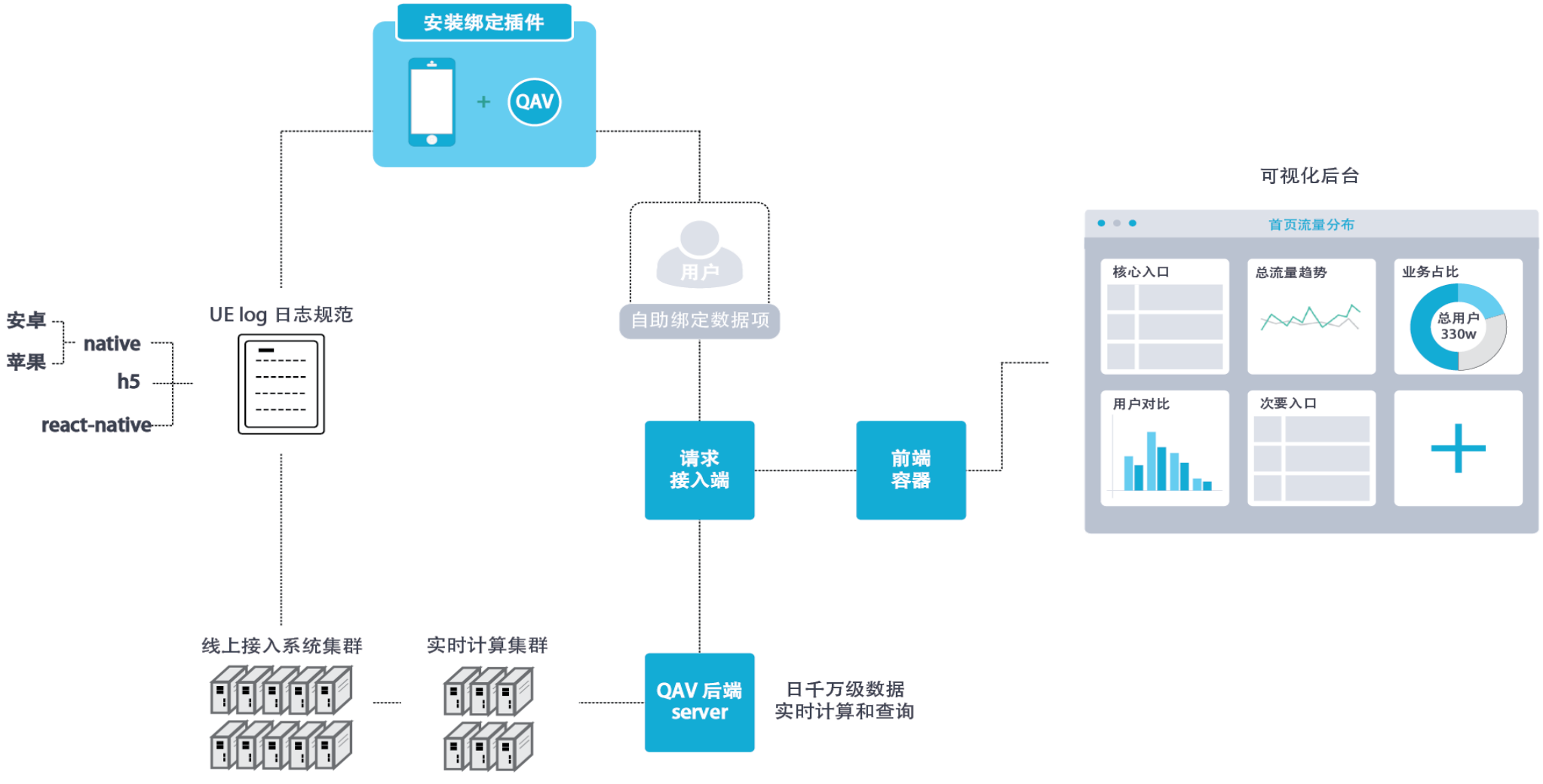
- **可视化埋点**

Mixpanel、TalkingData、
诸葛IO 等

二、QAV系统整体技术方案

1. QAV系统架构和组成
2. QAV的采集端
3. QAV的统计端
4. QAV的数据处理服务
5. QAV的统计管理系统

QAV整体架构


安装绑定插件
QAV
用户
自助绑定数据项
安卓
苹果
native
h5
react-native
UE log 日志规范
请求
接入端
前端
容器
可视化后台
首页流量分布
核心入口
总流量趋势
业务占比
总用户
330w
用户对比
次要入口
线上接入系统集群
实时计算集群
QAV 后端
server
日千万级数据
实时计算和查询

QAV系统组成

QAV采集端

app集成sdk

任意app引入qav代码包后
分发给公司内部开发、产品、运营
即可使用

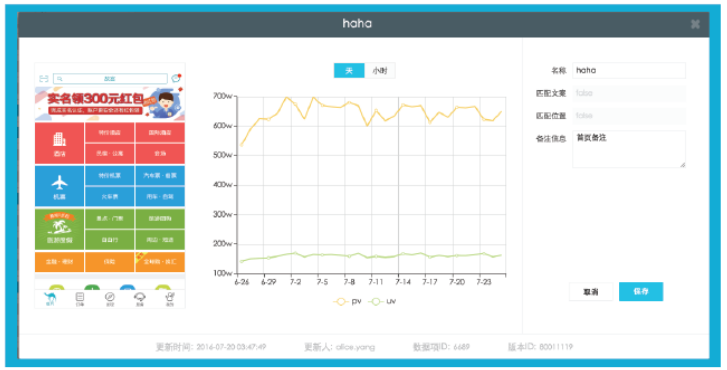
? + QAV

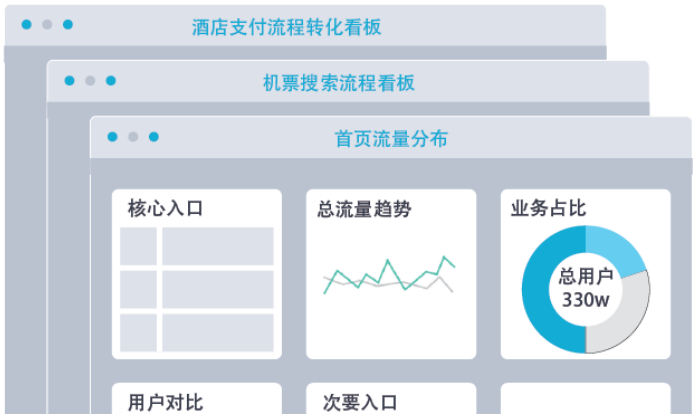
绑定

包括数据项的名称、备注、截图等

- 酒店首页入口
- 消息中心入口
- 机票首页banner
- 机票首页日历按钮
- 首页扫一扫
- 首页轮播图
- 个人中心-订单列表
- 门票搜索按钮

数据项截图 + 数据





数据后台-看板管理

按工作需要建立多个数据看板
自由组合任意数据项到一个看版中展示
自定义展现形式，按业务场景建立各种可视化图表
向组内同学分享自己的数据看板，提升工作效率

QAV系统组成

QAV采集端

app集成sdk

任意app引入qav代码包后
分发给公司内部开发、产品、运营
即可使用

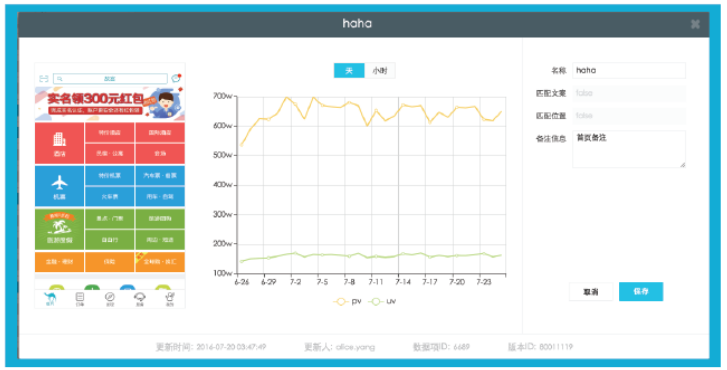
绑定

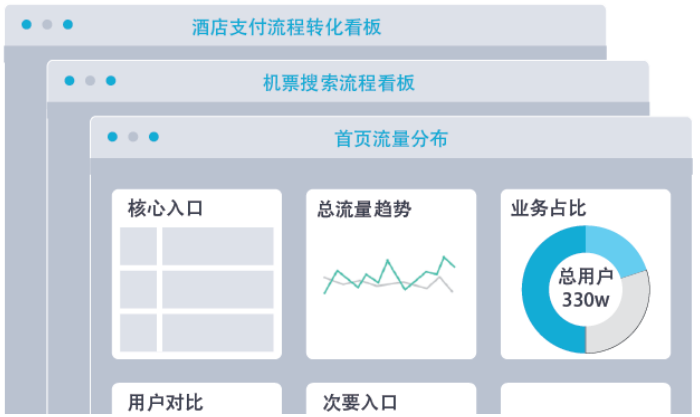
数据处理服务

包括数据项的名称、备注、截图等

- 酒店首页入口
- 消息中心入口
- 机票首页banner
- 机票首页日历按钮
- 首页扫一扫
- 首页轮播图
- 个人中心-订单列表
- 门票搜索按钮

数据项截图 + 数据





数据后台-看板管理

按工作需要建立多个数据看板
自由组合任意数据项到一个看版中展示
自定义展现形式，按业务场景建立各种可视化图表
向组内同学分享自己的数据看板，提升工作效率

QAV系统组成

app集成sdk

任意app引入qav代码包后
分发给公司内部开发、产品、运营
即可使用

QAV统计端

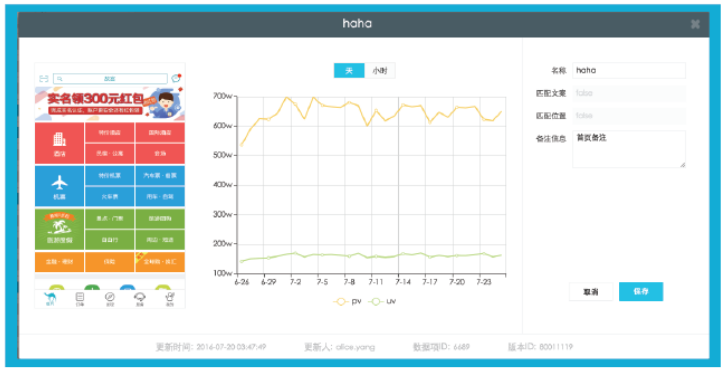
绑定

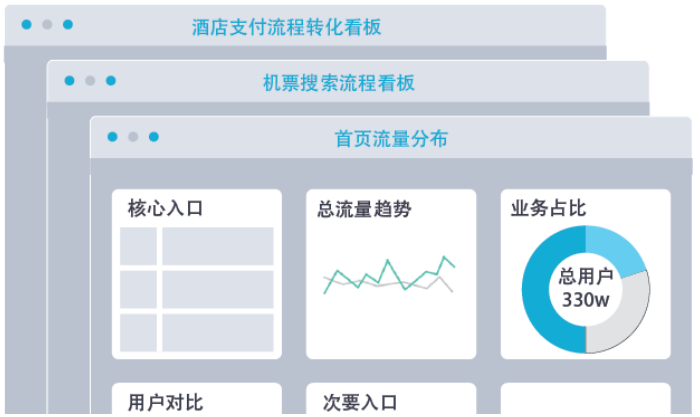
数据处理服务

包括数据项的名称、备注、截图等

- 酒店首页入口
- 消息中心入口
- 机票首页banner
- 机票首页日历按钮
- 首页扫一扫
- 首页轮播图
- 个人中心-订单列表
- 门票搜索按钮

数据项截图 + 数据





数据后台-看板管理

按工作需要建立多个数据看板
自由组合任意数据项到一个看版中展示
自定义展现形式，按业务场景建立各种可视化图表
向组内同学分享自己的数据看板，提升工作效率

QAV系统组成

app集成sdk

任意app引入qav代码包后
分发给公司内部开发、产品、运营
即可使用

? + QAV

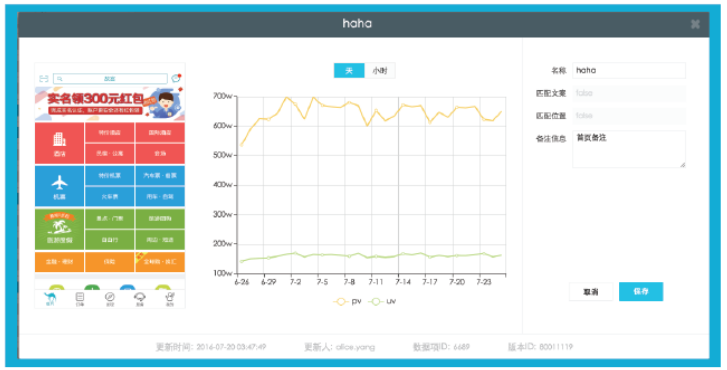
QAV统计端

绑定

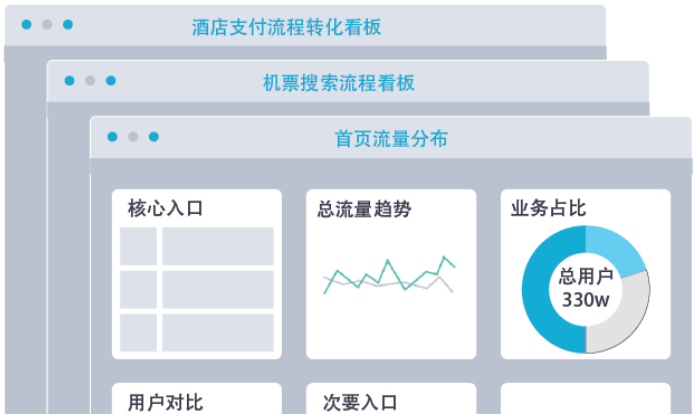
包括数据坝的名称、备注、截图等

- 酒店首页入口
- 消息中心入口
- 机票首页banner
- 机票首页日历按钮
- 首页扫一扫
- 首页轮播图
- 个人中心-订单列表
- 门票搜索按钮

数据项截图 + 数据



统计管理系统



数据后台-看板管理

按工作需要建立多个数据看板
自由组合任意数据项到一个看版中展示
自定义展现形式，按业务场景建立各种可视化图表
向组内同学分享自己的数据看板，提升工作效率

QAV系统组成

app集成sdk

任意app引入qav代码包后
分发给公司内部开发、产品、运营
即可使用

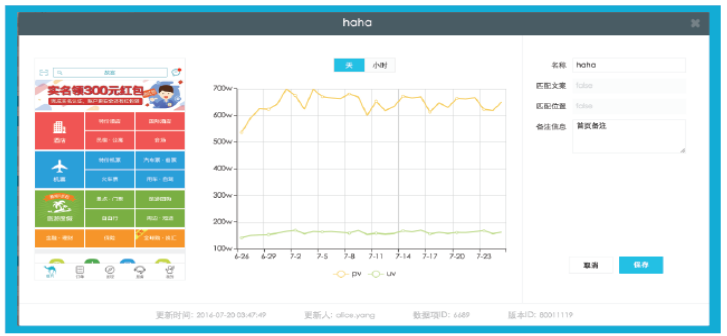
绑定

app功能内所有场景按钮均可使用
浮动小球实时捕获，手机上即可完成数据项命名
sdk也提供简单流量数据查看功能

数据后台-数据项管理

集中管理所有用户在手机上绑定过的数据项
包括数据项的名称、备注、截图等

数据项截图 + 数据



数据后台-看板管理

按工作需要建立多个数据看板
自由组合任意数据项到一个看版中展示
自定义展现形式，按业务场景建立各种可视化图表
向组内同学分享自己的数据看板，提升工作效率

QAV采集端

native

普通按钮click控件
iphone非click点击事件兼容
区分同类结构tab展示层中按钮事件

h5

从 hybrid 到适配所有 webview
多控件选取
list选取

react native

QAV的统计端

1

安装QAV客户端包

下载并安装对应版本的QAV包，登录后

2

小球选取数据项

小球选取控件后，会展示控件和页面对应的数据，可根据需求进行位置、文本埋点的匹配

3

绑定数据项

根据需求对数据项进行位置、文本、埋点的匹配后进行绑定

QAV的统计端高级功能

匹配位置和文本

热力图

QAV的数据处理服务

es 集群

按小时清洗日志建索引

经纬度转化、时间转化等预处理在hive阶段完成
尽量减少es的索引体积，对查询条件的字段数据要做优化

单条文档结构

```
city: 邢台市,
dt: 2016-02-15,
isLogin: false,
os: 9.2.1,
uid: 12BBF9DF-5742-4F7A-8D01-313674599AC5,
vidPrefix: 80
beh_actions: [ ..... 当天该uid所有行为 ..... ]

{
    action: 11169846,
    actionPos: -1,
    actionType: -1,
    cid: C1001,
    pid: 10010,
    timestamp: 1455544991744,
    timestampR: 79391744,
    vid: 80011109
}
```

es 查询

基础查询 & 用户分群

QAV的统计管理系统


去哪儿旅行-大
概览
数据管理
独立数据项
合并数据项
看板
图表
数据管理（旧）
用户
筛选
细查
页面流转
留存
漏斗
埋点管理
客户端发布
输入名称、ID或用户名搜索
搜索
全部
我的收藏
全部平台
Android
IOS
【Android】1 火车票首页
数据项ID：85107169
【Android】1.2 火车票_在线选座页
数据项ID：85099375
【Android】1.3 火车票_订单列表页
数据项ID：85106086
【Android】1.4 火车票_更多服务页
数据项ID：85108178
【Android】1.41 火车票_更多服务_...
数据项ID：85111399
数据箩筐

三、QAV自动埋点的技术实现

自动埋点的覆盖范围

自动埋点的原理

举个□ -获取IOS的UIControl的点击事件

1、添加UIControl扩展

```
@interface UIControl (Statistics)
- (void)statisticsSendAction:(SEL)action to:(nullable id)target forEvent:(nullable UIEvent *)event;
- (void)statisticsSendActionsForControlEvents:(UIControlEvents)controlEvents;
@end
```

2、将需要的hook的方法和添加的扩展方法进行交互

```
//UIControl
[StatisticsUELog swizzInstanceOriginalSelector:@selector(sendAction:to:forEvent:) toNewSelector:@selector(statisticsSendAction:to:
    forEvent:) forClass:[UIControl class]];
[StatisticsUELog swizzInstanceOriginalSelector:@selector(sendActionsForControlEvents:) toNewSelector:@selector
    (statisticsSendActionsForControlEvents:) forClass:[UIControl class]];
```

3、当hook到点击事件时，添加统计信息后，回调原来的响应方法

```
- (void)statisticsSendActionsForControlEvents:(UIControlEvents)controlEvents
{
    [[StatisticsUELog getInstance] addStatisticsWithUIControl:self withTarget:self withAction:@selector
        (statisticsSendActionsForControlEvents:)];

    [self statisticsSendActionsForControlEvents:controlEvents];
}
```

去哪儿? Qunar.Com 聪明你的旅行

IT大咖说 知识分享平台

Q&A

去哪儿? Qunar.Com 聪明你的旅行