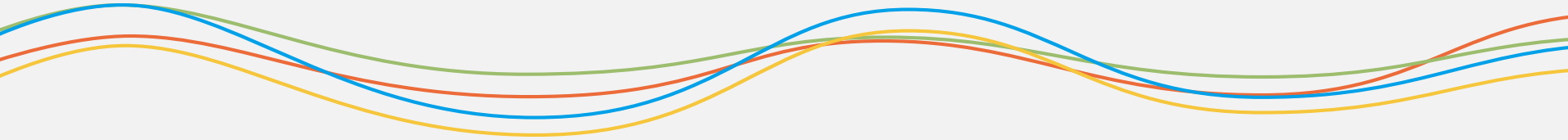
APMCon

小红书移动端
自动化数据采集实践

朱智伟



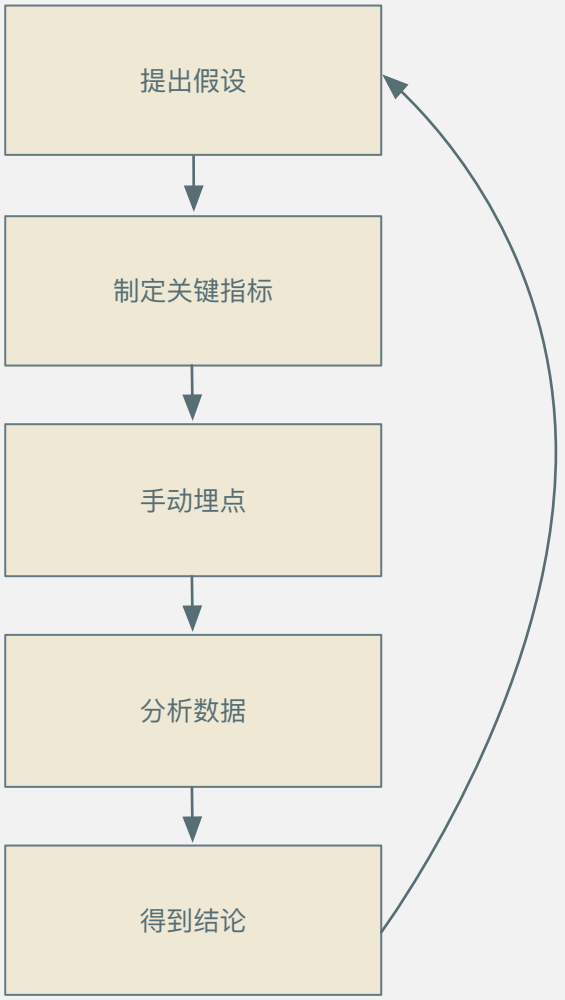
小红书数据采集的过去

初期我们在自建自己数据平台的时候，移动端主要是手动埋点采集用户行为数据，功能还没有现在那么复杂。

基本是采集PV UV，和一些分析师觉得有意义的行为。

通过某个具体指标的涨跌来定义这个页面或者功能是否达到预期。

得到每一个结论之前的链路太长，步骤过多，需要对每一个场景做分析，具体到那些点需要统计。



小红书以前是如何管理这些点

- 维护一个打点Key的表
- 每个点需要手动写Code
- 分析师通过Key的表来找到对应事件

391个打点事件需要手工通过excel维护

APMCon

	模块Code	页面	页面Code	功能点	点击	Type	Id	index_conte	埋点版本	去除版本	移动版本
首页	Home_Tab_View	首页	Home_Tab_View	点击头像	User_Clicked		Y				
				点击地区标签(品牌)	Brand_Tag_Clicked		Y				
				点击价格标签	Price_Tag_Clicked						
				点击地点标签	Location_Tag_Clicked						
				滑动笔记图片	Note_ImagesScroll		Y				
				点赞	Note_Like		Y				
				取消点赞	Note_Unlike		Y				
				收藏	Note_Collect		Y				
				取消收藏	Note_Uncollect		Y				
				点击分享	Note_Share		Y				
				相关商品（购物车icon）4.1改成点击商品cell	Note_Buy		Y				
				推荐标签点击	Recommendtag_Click				4.2		
				展开笔记内容	Note_SeeAllDesc		Y		4		
				收起笔记内容	Note_HideDesc		Y		4		
				发送评论			Y				
				添加评论	Note_AddComment						
				查看所有评论	Note_AllComment		Y				
				发布笔记	Post_Button_Clicked						
				关注作者	Follow_User		Y				
				取消关注	Unfollow_User		Y				
				停留查看	End_Scrolling	note	Y	offset:300	4.1		
				好友动态选项	Recommend_Selector		Y		4.1		
				不在首页显示该专辑或标签或ta收藏的笔记	Recommend_Hideall	Board/tag/collect	y		4.1.1		
				不在首页显示ta的笔记	Recommend_Hidethis	Board/tag/collect	y		4.1.1		
				添加微博好友	Header_Clickweibo				4.1		
				添加通讯录好友	Header_Clickcontacts				4.1		
				点击全部	Header_All						
				点击我关注的人	Header_OnlyFollow						
				无笔记占位图点击follow	EmptyNotes_Placeholder_Clicked						
				首页搜索	Search_Button_Clicked						
				推荐笔记点击小箭头 点击前往XX标签或专辑	Recommend_LinkDetail	Board/tag/collect	Y				
				笔记tag 点击	Tag_Clicked		Y				
				点击推荐标签	Recommendtag_Click		Y		4.2		
				推荐标签End Scrolling	End_Scrolling		Y				
				特殊事件Banner 点击	Home_Banner_Click	link	Y		4.2		
		首页搜索页面	Home_Search_View	事件同Explore_Search_View							
		只看关注的人		无笔记占位图点击follow							
		评论页面	Note_AllComments_View	评论点赞	Note_Comment_Like						

我们碰到了那些问题

APMCon

① 埋点代码遍地都是
② 复用性基本为零
③ 需求改动太快，埋点也要跟着一起改
④ 我们无法预知到分析师需要哪些数据
⑤ 维护一套打点事件的表成本很高
⑥ 错打漏打经常发生
⑦ 花费了大量时间进行数据清洗

自动化数据采集的推出

数据采集与上报是构建数据平台过程中最重要的环节

如果在采集与上报上花费了过多的精力，那么我们花费在得出结论的精力就会变少。

在数据采集和清洗上我们也许花费了80%的精力，但是只产出20%的价值。真正有价值的是这些数据告诉我的一些结论，可以帮助我们产品做决策。

所以我们决定做一个产品来解决这个问题

就是自动化数据采集

当时有2种解决方案，一种是可视化埋点，一种是无埋点

手动埋点虽然使用起来灵活，但是开发成本较高，每次页面改动都要修改大量打点代码。

可视化埋点开发成本较高，需要开发一个后台系统供分析师和产品经理去操作。

既然无法预知打点需求就全部采集
既然需求改动太快那就自动采集

无埋点 是最适合我们的解决方案，它所带来流量的压力和数据量的增加都是我们可以接受。

所以，小红书最后才用了无埋点的方案。

移动端有哪些有价值的数据

页面：

整个页面出现和消失

行为：

交互式事件打点

用户 赞笔记 评论笔记 收藏笔记

曝光：

每个笔记的出现

核心 ： AOP

面向切面编程（AOP是Aspect Oriented Program的首字母缩写），我们知道，面向对象的特点是继承、多态和封装。而封装就要求将功能分散到不同的对象中去，这在软件设计中往往称为职责分配。实际上也就是说，让不同的类设计不同的方法。这样代码就分散到一个个的类中去了。这样做的好处是降低了代码的复杂程度，使类可重用。

概括的说，通过预编译方式和运行期动态代理实现在不修改源代码的情况下给程序动态统一添加功能的一种技术。

在数据采集这件事上

其实就是当用户发生行为

我是谁

我从哪里来

我要到哪里去

但在现实世界中，并不是所有问题都能完美得划分到模块中，有些功能是横跨并嵌入众多模块里的。

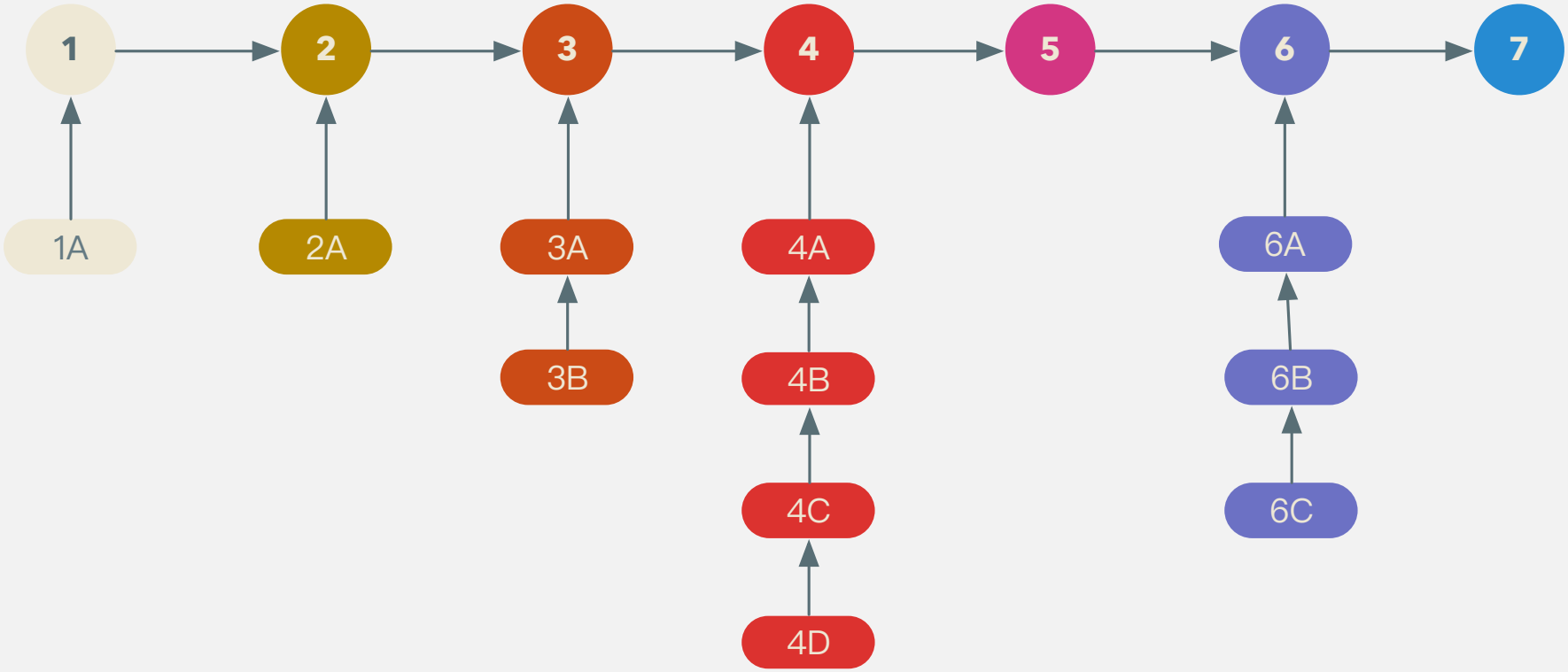
核心2 ： View树结构

APMCon

无论是 Web 上的 DOM 结点结构，还是 App 上的 UI 控件结构，都是构建好的一颗完整的树形结构渲染在页面或者屏幕上。
所以我们给View 和 Page增加了一些附加属性，来和业务产生数据绑定。

Referrer & Parent

简单的说就是你从哪来，和你是谁生的问题



页面数据如何自动采集

APMCon

页面数据其实就是记录每个页面的生命周期
在iOS中就是ViewController的生命周期
在Android中就是Activity/Fragment 的生命周期

这样其实非常简单了，我们只要根据我们的需求去hook一些生命周期的方法就行了。

在iOS 中我们可以使用runtime的method swizzle 去hook的viewController的viewWillAppear和viewWillDisappear方法。
Android通过继承在Activity的onResume和onPause方法中处理。

行为数据如何采集

APMCon

不管是web还是iOS Android 所有点击行为一定会有回调，那么我们只要找出所有的回调方法，那么就可以采集所有的点击行为。

在iOS中

UIControl

iOS大多数可点击UI控件都是基于UIControl，而所有的事件也都要通过

- (void)sendAction:(SEL)action to:(nullable id)target forEvent:(nullable UIEvent *)event

转发，通过该方法，我们可以监听到继承自UIControl等控件的Action事件。

Delegate

在+ (void)load方法里 swizzling tableview或者collectionview 的select事件好像不太可行。

所以我们在setDelegate的时候，去swizzling select事件。

Android 中行为数据如何采集

APMCon

通过hook Activity#getSystemService()，Fragment#getLayoutInflater()，
获取页面视图树。遍历视图树中View，设置自定义AccessibilityDelegate。

用户对控件的操作，会回调自定义AccessibilityDelegate#sendAccessibilityEvent()，

附带行为和被操作的View，通过前面添加的视图信息，组装埋点数据并发送。

曝光如何采集

如何更加精确的统计用户是否浏览过某些内容，以往的事件打点已经不能满足算法组的需求。

来自于广告CPM，因为广告主要精准的知道每条广告投放给了多少人，转化率多少，才能计算出成本。

在小红书，每一篇UGC的内容的展现也可以认为是一种广告投放，也会有投入和产出。投入就是占用多少曝光次数，产出是用户对这一篇UGC的内容的点击，赞，评论，收藏。我们肯定是希望用尽量少的投入获得尽量多的产出。

曝光在小红书的解释就是单个基础资源在用户手机屏幕上展现的次数，需要计算曝光的基本单元都是在可滚动试图中子视图，还需要去除快速滑动带来的无效曝光。

说人话：用户目光停留过的东西

如何判断用户目光是否停留呢

APMCon

变换思路

- 用户手指离开屏幕时
- 滚动结束时
- 页面离开时

iOS 通过method swizzling
1.离开页面的时候
viewwillapear
2.手指离开屏幕的时候
scrollViewDidEndDragging
3.控件结束滚动的时候
scrollViewDidEndDecelerating

Android：
三种情况进行曝光埋点
1.首次进入页面，显示的子元素；
2.拖动控件过程中显示的子元素（fling操作，即快速滑过导致显示的子元素不会曝光）；
3.控件滚动结束后，显示的子元素；

发送策略：
4个曝光合并成一个事件发送

上线之后结果如何

APMCon

3次大版本迭代

打点1.0 2015年中全量上线

打点2.0（自动打点第一个版本）2016年中旬全量上线

打点3.0（自动打点第二个版本）2017年5月全量上线

2016年刚上线自动打点后的数据

每日数据量 306,754,313 条

环比增加 300%

开发节奏的改变，加快了开发节奏，80%的打点被自动打点替代，每个版本只需要花费少量的时间去手动埋点。

小结

可以说是从2个角度去思考自动化数据

第一 深度

在一个行为发生时，尽可能得获得和它相关的数据

我们增加了referr，impression，UUID等，都是为了数据分析和实时分析系统更能精确的知道它的上下文。

第二 广度

通过iOS和Android的一些系统特性，定制化得hook一些系统方法，在不需要额外的情况下，尽可能多的覆盖用户交互场景。“漏打永远比错打更严重。”

我们hook 按钮点击，页面出现，滑动等系统事件去自动捕获用户行为。

THANK YOU