揭秘 MIUI 视频从 0 到千万用户的研发迭代

DAU 几千万，总用户：亿级

对于研发来讲，10 万用户和千万用户产品的研发过程是不一样的。千万用户的研发迭代需要良好的技术架构支撑，同时需要简单可靠的版本发布流程，良好的质量控制，以及对产品的不断改进。

本文从 MIUI 手机视频研发角度，分享了一个 DAU 超过 2500 万用户资源型应用的研发迭代。从以下几点详情展开：

一、故事背景

2014 年 12 月份，王川让我们介入 MIUI 手机视频的研发，我们并没有太多意思想做，不打算只是接受一个需要不断维护的工作。解了一个月 BUG，熟悉现有业务，中间隔着一个春节，就混沌的过了。

2015 年 2 月底我们组建研发，开始重新开发前后端。来了一名义上的封闭开发，主要是让大家精神上高度集中，不被外界干扰，高效率集中研发。经过一个半月，清明前结束了封闭开发，基础架构搭建完毕，再进过一个月的 BUG 修复，正式上线了，6 月 1 号进入 MIUI 的第一个稳定版。相当高效，两位同事负责播放器，一位负责后端，一位负责 CMS，我兼职视频的主框架，虽然我们封闭，但是只有两个星期六来公司加班，每天晚上工作不超过 8 点，我们只做不超过两天的计划，每天准时完成迭代。

随着新人的加入，迭代进一步完善，现在 MIUI 视频，DAU 超过 2500 万，总用户 1 亿，

每周对外发布一个稳定版，作为原生应用，发布 3 天后可以到达 75%日活用户，一个星期内 85%日活升级到最新版本，最后两个版本可以达到 90%，一个月内的版本可以到达 95%以上，完全可以满足用户运营的迭代需求，这是了不起的成就。收入从 8 月中旬开始接入，9 月开始投放，当月收入超过 500 万，后面会详细讲解变现接入，11 月收入超过 1000 万，12 月接近 2000 万，这还只是其中可变现的小部分。

事情回到半年前，当我们帮助电视和盒子的在线视频，游戏中心和应用商店时，发现一个有趣的事情， 很多展示和业务是交织在一起的，十分不利于运营。于是川总和 BSP 负责人出人力，组织了一个三人小分队，决心搭建一个 demo——展示和业务无关。我们花了两个星期改造了游戏中心的首页，游戏中心的原始版本实现了展示和业务的分离，我们的改造重新抽象首页数据结构，需要后端一起配合。加之涉及业务操作流程的更改，推广自然是有难度。如是，王川让我们和应用商店团队配合，把应用商店的启动页面改造一下。经过和应用商店沟通，得到鼎力支持。感谢应用商店开放的心态，最终该架构在应用商店落地，并以小米的名义开源（http://github.com/xiaomi/android_tv_metro）。

随后的一个月高荣欣和我决定在此基础上把在线视频也改造一遍，我们又花了一个月，导入视频数据，基本做出了一个电视上的在线视频(http://github.com/aiandroid/stream)。要改变现有的基于 C++框架构建的视频框架，而且现有业务运行良好，多年的业务积累，必然伤经动骨，风险巨大，暂时搁置。我们又回去协助音响团队，实现小米电视音响的互联网化，主要是做一个小米电视 soundbar 在手机上操作屏幕，以及数据收集的工作，构建音响团队快速迭代搭建基础架构。

小总结：野蛮生长对创业团队的初期是合适的，怎么快怎么来，一旦证明业务模式是可行，就需要重新整理架构。一刻也不要等，否则越做越痛苦。BSP 负责人茹忆，王川，付出人力，给予团队时间，从小点突破，流行词叫赋能。

二、架构重建

开始说说架构吧，MIUI 小米视频的架构核心就是展示和业务分离，为什么要做展示和业务分离，这里啰嗦几句。做 MIUI 视频前，我们一直在帮在线视频，游戏中心，应用商店。从逻辑分类上讲，这三家的逻辑没有什么不同，不就是展示嘛，如是乎，我们把三家的程序和数据结构过了一遍，发现大不相同，共同点很少。我们就决定把所有的展示抽象出来，业务往二级放，做了一个 all in one, 把视频，游戏和应用做成了一个应用，做到了和现有三个市场接近一致的用户操作体验。故事中说到在应用商店落地了，现在的在线视频也在用安卓做展示和业务分离，我们也参与其中。

1) UI 表达抽象化

MIUI 手机视频，从一开始就以展示和业务分离为原则开展，现阶段我们的互联网 web 应用就是最好的展示和业务逻辑分离的例子。视频应用中任何展示 UI 都被定义成一个展示单元(DisplayItem)的数据结构，展示的文字，图片海报，图标都被统一抽象出来。UI 展示是一个树结构，我们把集合定义成展示块(Block), 展示块(Block)也是一个展示单元(DisplayItem), 展示块(Block)可以嵌套多个展示块(Block), 和展示单元(DisplayItem)（程序员

应该能看懂我说的）。通过展示块(Block)我们把所有的展示逻辑框住了。

下图每一个位置都是一个数据结构定义，不涉及业务，只做展示用。黄色部分框住的是一个展示块的示例，“排行”中“电视剧”的“家和万事兴”是一个展示单元示例。

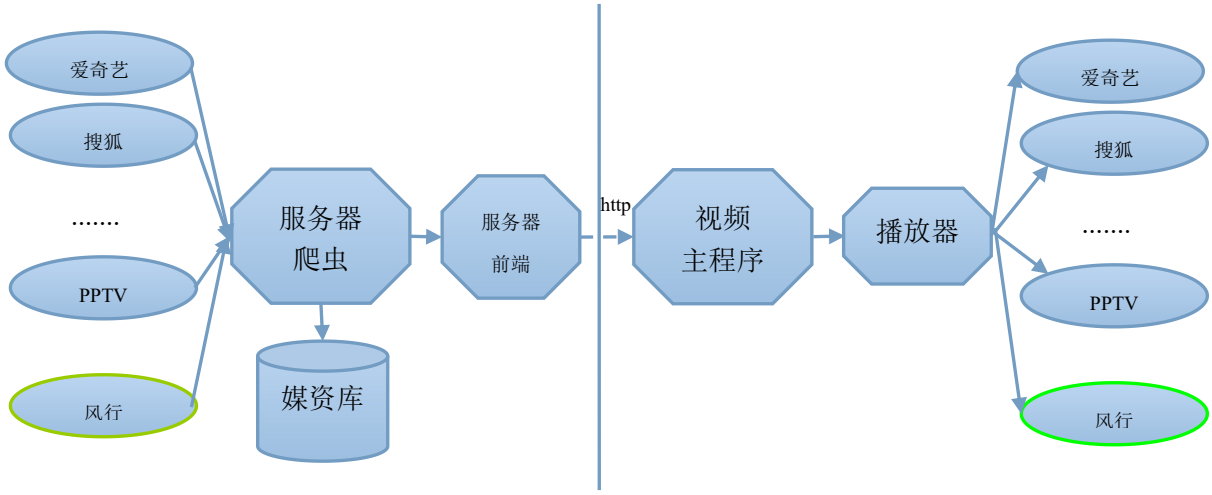
更多信息参考 http://github.com/aiandroid/stream，以及 Google 最新 Android TV 的展示和业务分离。

2) CP 集成插件化

MIUI 手机视频的业务特征是做视频流量分发，我们集成了国内除腾讯外的稍微大视频资源，甚至包括直播类的 YY，作为内容提供方 CP，他们给出了 VV(Video View)诉求，同时要保护版权。CP 方都会提供自己的 SDK 来保护内容，同时适配自己的服务器接口。这种模式就决定了 CP 方内容的接入和下架，要解除依赖(解耦)，不能和主程序的研发迭代，交织耦合太深。

2014 年 6 月，搜狐视频资源突然不能播放，就搞的在线视频运营团队很被动。吸取前人教训，手机视频播放器利用 dexload 的方式，定义一套 CP SDK/APK 需要 Follow 的一套接口，同时把每一个视频需要播控信息，透传到每一个对应的 SDK/APK，主视频应用和播放器并不关心每一个 CP 的播控数据要求，让 SDK/APK 提供方自己去解析。每一个 CP 播放视频需要的播控数据，会在服务器端组装完成，并透传给 CP 方提供的 SDK/APK。这意味着，播控和展示也分离了。

下图中风行 CP 的视频资源上线时间不依赖服务器前端，视频主程序，和播放器的研发，只要爬虫搞定， SDK 搞定就可以随时上线。极大的降低了对内容 CP 方的耦合解除。



CP 播放能力插件化衍生出一套 CP 的管理逻辑，这些逻辑都放在服务器管理，一旦某 CP SDK 有 BUG fix， 服务器发布后，主程序和播放器会及时获取最新的 SDK 来播放 CP 方提供的视频内容。比如最近支持奇艺的会员业务。

三、迭代

迭代的思维，大家都已经接受了，SCRUM 确实是好，研发初期，我们只做最多一个星期的计划，然后每天执行，早上严格高效同步，所有的问题暴露出来，大家一起想办法把问题解决。程序完成差不多的时候，把后续需要不断修改的地方解除耦合，做成可以多次分级发布，能力上包括：1) 视频主程序自升级能力 2) CP 插件管理和升级能力 3)功能灰度发布能力。

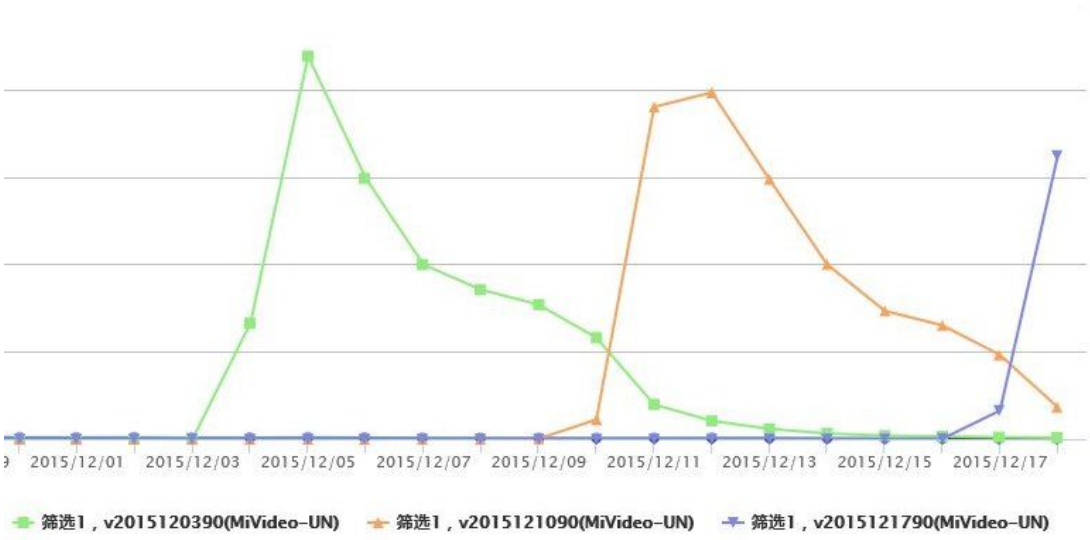
迭代的核心是良好的技术架构支撑，解除耦合是核心。展示和业务做分离，CP 视频内容和播控做分离，实现了这两个能力，就靠升级发布版本来不断推动前进了。视频应用版本的发布我们选择了外部全网络分发渠道，加小米应用商店，加视频的自升级。

发布一天后，最新版本日活用户就可以达到接近 60%。

详细数据 2015/12/18

版本	新增用户	新增用户占比	日活跃用户	日活跃用户占比	升级用户	启动次数
v2015121790 (MiVideo-UN)	205,511	50.57%	9,570,467	57.11%	8,100,835	32,742,709

下图展示的是版本迭代节奏，一个星期一个稳定版本



测试作为迭代的一部分，怎么融合起来，怎么保证一个星期一个版本。面对几千万用户，有一个问题是，怎么做测试？测试章节会详细讲解，这里不展开。

1) 视频主程序自升级发布，

应用程序的包越小，越有利于快速到达用户手里。

视频和播放器是打包一个应用，各个 CP 插件并不包含在主程序(视频和播放器)中，我们尽量控制主程序的 size，在 MIUI ROM 中只有不到 5M，即使加上 codec 能力，外发版本也只有不到 11M，而各个 CP 加起来就有 30M 之多。主版本发布不包含各种 CP 的发布。

视频发布有一个独特的地方，测试版本和稳定版可以一起发布，并能控制测试版本的受众用户群，这套机制，解决了我们对多小米机型，多安卓，多芯片平台的测试，在测试章节将详细展开。

稳定版发布质量 OK 后，我们会先发布到我们的自升级平台，一般灰度 30%，同时会发布到小米应用商店，Hold 住，等半天用户的反馈时间，这半天我们会密切监控用户反馈，小米论坛，视频自由的 VIP 用户群（这是我们的核心用户群），如果没有 FC，不能播放的严重问题，就放全量，同时同步小米应用商店和外部第三方应用商店。一般的小问题，就不管了，下个版本修复就行了。

最开始做升级，自升级并没有什么威力。用户进主页时，用户一天会有一次机会看到弹出升级提醒，我们发现大部分升级是来自小米应用商店后台自动灰度升级的，用户主动升级的很少，如是认为小米应用商店最重要，每周升级率也很高。后来发现，那时是刚刚开始，用户量小，等用户量到达千万后，发现周升级率下降了至少 10 个百分点，这意味着迭代变慢，我们的修改到达用户变慢。这样不行啊，如是想有什么版本能提高迭代速度，即升级率。找到充电+有 WIFI 网络，自动升级视频应用，这是一个神器啊，上线一个月升级率大幅度攀升，满足一个星期一个稳定版本。听说某 ROM 应用升级到 50%，就再也难往上走了。

目前升级渠道，自升级占 70%，小米应用商店 20%，其他第三方填空。

理论上现在一旦发布版本，可以一天升级 3000 万用户，下图是统计到充电+WIFI 的用户和次数，为了不影响系统，4 小时内视频只有一次机会升级机会，也只有一次统计。

日期	次数	人数	人均次数	取值总和
2015-12-18	80,937,543	30,236,217	2.7	80,937,543

2) CP 插件发布

一套插件管理，加一堆 CP 插件，播放器定义了一套 interface，提供给 CP 方，播放器通过回调控制 UI 和下一步动作。主程序会提前把插件下载，同时如果播放时发现 CP 插件需要更新，或者没有下载，会主动下载一次，如下面的右图；实现插件实体的下载和使用的分离。视频也借助 MIUI ROM 帮助集成插件，提高用户播放视频时，需要下载插件等待时间过长的不良体验。

3) 功能集成和发布

服务器 feature 和客户端 feature 的发布是分开进行的，有的功能，需要分版本（我们尽量避免，实在不行就用代码搞定，而不是放到 CMS 中去选择版本，否则运营要疯），有的功能需要灰度发布。对于不确性的功能，需要通过云控进行 AB 测试，和灰度发布。

一、服务器发布

服务器两条简单的代码线，release，dev，每个星期一个大版本，小版本随时发布，总体来说比较灵活，遇到只有新的客户端版本支持的 feature，做版本判断，简单粗暴，让内容不出现就 OK 了，因为，我们是展示和业务分离，只要和业务相关的展示没有，不同版本的客户端会看到不同的内容，只是内容的有和没有关系，尽量不做不同的 UI 展示。同时有的功能需要控制灰度，比如视频的 VIP 会员，就是先 10%，30%，65%，最后 100%发布，要能够控制 feature 的受众范围。这些都是通过服务器来控制。

二、客户端代码集成和功能发布

客户端的版本管理相对复杂些，视频有三条线，MIUI alpha, MIUI dev, 和视频的外发 standard，进入稳定开发阶段，我们的每天开发的代码会很快的提交到 alpha 线，每天 alpha 体验版会自动 OTA 升级，帮助及时发现问题。这个版本及时发现了问题，也不是那么严重，改动相对频繁；相对稳定的会及时进 dev，这些 feature 会进每周 5 的传统发布，理论上到达 500 万开发版用户。开发版用户报上来的问题，需要及时修复，否则影响团队在 MIUI 的声誉。具体的规则不叙述，涉及小米研发的内部流程。

那为什么需要一个外发的 standard（名字而已）线呢，我们的外发迭代都在这条线上，视频的外发稳定版本会每两个星期，或者一个星期发布一个版本，走的比 MIUI 稳定版更快（MIUI ROM 更复杂，用户量太，稳定性是第一位），这条线解决了对 MIUI 系统的依赖，用一个版本去适配所有的小米手机和安卓版本，视频的 codec 团队做出了卓越成绩。

解决了代码管理问题后，就是功能发布了。对于老功能提升的部分，需要服务器和客户端用云控来解决，比如嗅探，离线下载，服务器会传给客户端一个配置，客户端根据配置来决定功能分支的选择。比如是否打开下载能力，甚至不同的版本客户端可以传不同的配置。

三、客户端启动时触发自动升级

为了让用户能体验最新的业务，视频保留了最后一招的能力，客户端自动升级，至今没有用过。服务器如果发现客户端版本不能支撑业务，会返回一个 418 的 code，客户端收到该返回，会触发自动升级。做这个功能时，内心有点像当英雄，最后来次大拯救行动。

对于有的业务，我们希望客户端升级上来尽快体验，会把展示业务用一个升级条盖住，告诉用户去点击升级，这是对升级的补充，体验可能并不好。

四、云控

用控是基础架构能力，视频实现了一个简单的服务器和客户端的 key-value 同步映射，同步到客户端的本地数据库，客户端封装一个简单的 ORM 来访问数据配置。云控使用者不需要关心配置怎么来，简单读取就可以，也可以通过简单的本地配置来调试服务器云控。视频在用户能输入的搜索界面打了一个洞，能把配置写入本地，方便随时调试。

下图是打开本地调试开关，其他 key-value，可以类似打开，20100408 是一个前缀码，

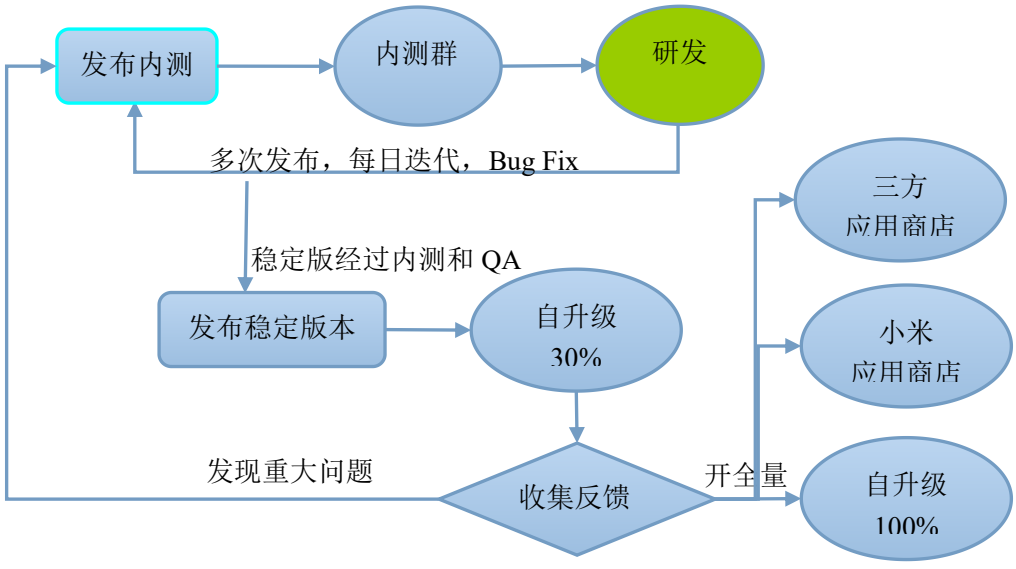
米粉节是 4 月 8 号。

题外话，为什么没有采用 H5 的发布能力，我们的迭代速度能做到周级，周能升级 85% 日活，能满足现有迭代；另外没有人做而已，应该需要尽快尝试。

四、客户端升级和用户反馈

升级能力也是迭代出来的，我们趟了不少坑，总体来说，快速跑过来了。要顺畅的升级，关键的一条是解决和系统(MIUI SDK)的依赖，和硬件编码(Codec)的依赖。在解决和系统的依赖上我们遇到了一些问题，把 V5 的几百万用户直接搞得用不了视频，最后没有办法，为了和主 ROM 风格一致，视频自己实现了状态栏控制，设置风格，以及对话框。所以至今也没有和 ROM 保持十分一致的风格。视频编码(Codec)一套代码解决了软硬编解码，多安卓版本，多芯片平台的依赖，赞雷威同学。技术问题解决后，就是开发出发布渠道，加速这些渠道升级速度的问题。

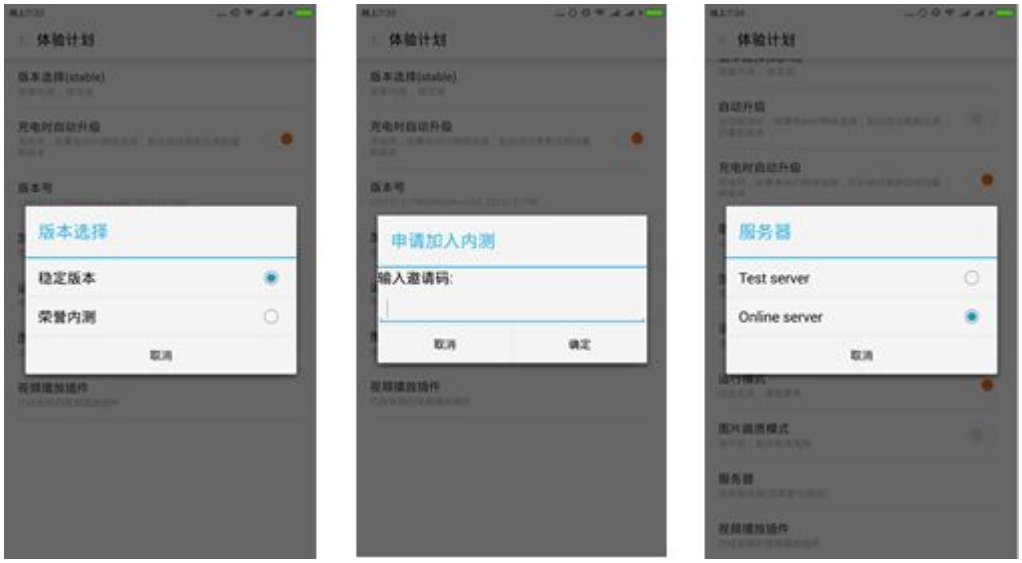
我们的升级策略一开始就和用户测试绑定在一起，同时维护了近千人规模的内测用户，他们可以很简单的加入我们的内测群，并获得最新的内测版本， 十足的参与感。 我们甚至会每天发内测版本，内测用户的反馈意见当天就能获得。日迭代+周迭代，日迭代用于日常开发，周迭代发布稳定版本。2015 年 4 月 20 号发布第一个稳定版本，到 12 月 19 号，一共发布了 181 个版本，31 个稳定版，151 个内测版本。



怎么把测试版本传递给内测用户？

我们想了一个办法，无需用户通过 ADB 安装，直接通过应用升级，尽量减少用户的安装成本。让用户直接能在视频切换到内测基线，当然我们也需要控制内测群用户数量，就做

了一个版本切换+输入邀请码的模式。用户甚至能切换到我们的内测服务器，来深度帮助测试。这套逻辑同时提供给研发和兄弟团队联合调试。尽量不要找人安装测试版本，每个人的时间都很宝贵，用程序去解决。下图是从客户端体验计划进入的操作过程。



视频自升级服务向服务器查询是否有新版时，服务器会检测最后两位数字，我们规定，最后两位数字为 00 到 09，就是内测用户，它将从内测的基线检测版本，尾号为 90 到 99 的认定为稳定版，这些用户就归为稳定版用户。视频客户端请求升级版本信息时，会根据是否切换了内测版本，把最后两位数字篡改，保证在 00~09，或者 90~99。之前还设计了 10~19 为开发版用户，发现不需要，就放弃了。

五、数据打点

初期视频是研发驱动，做出第一个稳定版后，由于视频是强运营的产品，必然过渡到运营驱动和数据驱动。每一个细小的改动都会对用户产生影响，并最终反映在数据上，打点是必须的。 通过打点，我们能看到巨量的用户行为信息，选择一个合适的统计平台可以帮大忙。之前做音响时，自己开发数据采集，后端也自己处理，累的要死，还要做数据分析， 成本不划算，工程师多贵。视频新版从一开始就选择用小米的统计平台，放弃了原来自有的打点服务。很多时候，争论不休，建议用数据说话，用云控来做 A/B 测试，看哪个数据好。

数据对全组成员开放，让大家养成看数据的习惯，强化数据分析，只需要一段时间，大家做 feature 时，就会很自觉的加上数据打点，程序发布后，立马看数据。本文很多图都是来自统计后台的数据。

别一下子一个功能做的太深，太深会成为程序员的负担，谁都不愿意放弃自己开发的“牛逼”功能，让数据说话。

服务器后端的数据分析，也是必不可少的，每天要发出来和统计平台对数据。甚至出一些实时的热区数据， 便于运营及时调整位置和内容。

六、QA

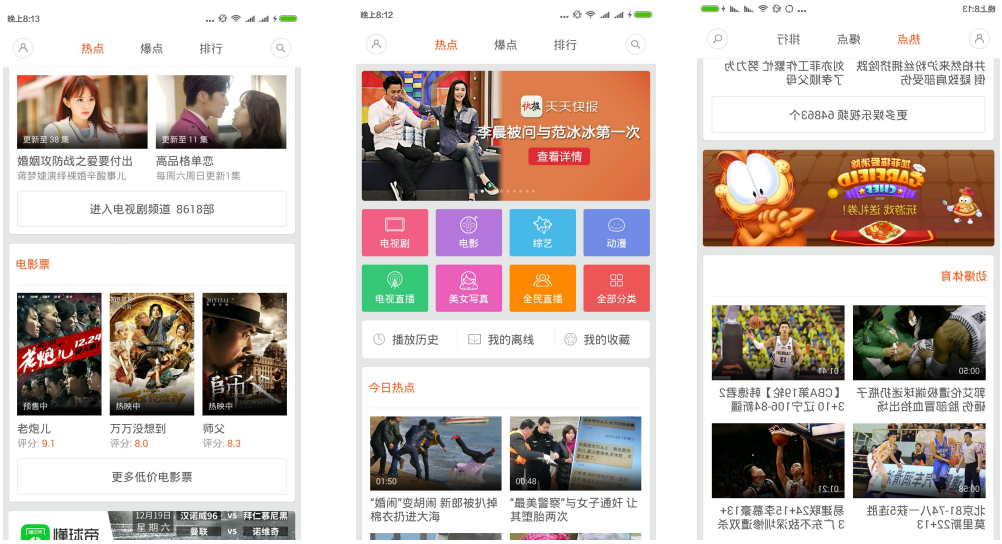
视频只有一个专职测试，负责对研发做日常验证，发版本最后的验证也由她完成， 这个工作还是必不可少的， 一个测试工程师支撑几千万 DAU 产品， 还一个星期发布一个版本， 但是我们的测试工程师并不是很忙，也不需要加班到深夜。 因为我们转移一部分测试工作量——依赖内测群做规模测试。 这个群帮最终稳定版用户尝新，帮研发发现问题，我们有一个很擅长用户交流的项目经理。这个用户群对我们的迭代研发起了关键的作用，加速了视频的迭代。实现了把用户纳入我们的研发环节， 十足的参与感。

高水准的职业化程序员极其重要，他能很好的自测，会能一盯到底，会观察数据，甚至自己就及时调整了。

七、商业化

直到 2015 年 8 月底，我们还没有自己的商业化接入，当然这时候我们的版本也稳定了，是时候赚钱了。目前视频的商业化，分成四大块，品牌，应用分发，电商和视频前贴片。 除电商外，都已经接入了，目前重点在应用分发。分享一下我们接入变现平台的过程。从时间上 9 月中旬开始介入应用分发，9 月 10 号产生第一笔收入，10 月达到了超过 5 百万的收入，11 月超过 1000 万，12 月接近 2000 万，增长速度很快。

视频从一开始选择了服务器接入，而不是客户端接入，变现和内容用一套数据结构，一套 UI，客户端只需要增加处理打点。视频把广告当成一种展示，同时视频客户端从能力上把所有的位置既当内容位，又当成变现位来处理，这种模式极大的加快了变现的接入。 变现的接入变成了服务器的事情，开变现位置就行，理论上所有客户端支持的 UI 都可以是变现，只要视频客户端支持打点，就可以赚钱。这种模式的核心是展示和业务的分离，和我们的基础架构结合很好，客户端快速迭代支持打点，支持新变现 UI 形式也是关键。



原生广告，内容即广告

电影票 UI 形式完全和电影视频 UI 一样，实际上服务器端是没有区别的；轮播图第二

张和第一张也没有区别；通栏和正常的视频位置也是一样。不同的地方在于点击的跳转，视频支持安卓全能力跳转链接，通过视频客户端和服务器约定数据交换模式，只要服务器输入的数据，在客户端能组装出安卓跳转的 Intent，视频客户端就能跳转到正确的应用。同时为了商业化，视频加入了应用下载和安装的能力。有了这两个实现，就可以支持下载和安装应用，并跳转到需要的目的地。

```
"target": {
        "url": "yymobile://3g.yy.com:80?bundle_extra_type=0",
        "params": {
            "apk_url": "http://image.box.xiaomi.com/.../pFU92B.apk",
            ...
            "apk_version": "26",
            "present_url": "",
            "new_task": false,
            "action_url": "",
            "tick_url": "",
            "android_action": "com.duowan.mobile.mi.entry",
            "android_component": "com.duowan.mobile.mi"
        },
        "entity": "intent"
    }
```

present_url 是展示打点需求，可以是多个，能够集成第三方的打点需求，如秒钟

android_component 是应用的包名，如果应用已经存在系统中，会直接打开应用，如果不存在，会下载对于的应用。

八、研发体会

当别人反对，如果认为是对的，就做出来，用数据说话，用 make difference 勉励自己。
做不确定性的事情需要勇气，需要耐得住，坚持，再坚持。
别把时间浪费在无聊事情上，还不如去锻炼身体。
争吵总是让人进步，不要害怕争吵
相对于流程，人更重要
选择好的队友
允许犯错
勇气

刘华东，于 2015 年 12 月 19 号，北京，雾霾
liuhuadong@xiaomi.com, 13810139250
微信:liuhuadongweixin