天猫双11之前端

每年双11的第1秒和第1小时是并发量和流量最大的时候，安然度过这1秒这1小时，从技术角度讲就意味着双11成功了一半。按照每年双11的规模都有2-3倍的增长，我们就可以大致估算出双11所面对的峰值，这是所有技术方案面临的挑战底线。前端团队担负全站人机交互界面的实现和品质保证，每年的双11都会遇到如下挑战：1. 短时间内完成巨大的界面开发量（天猫全站各种双11特有的界面表达和数百个主分会场），并精确到秒地发布它们，同时要快速响应界面调整，实现全站实时发布2. 大并发量和大流量面前，为了减轻后端服务的压力和提升前端性能，需要对页面的部分内容进行分级降级（不同的情况下降级不同的内容），这需要小心设计业务的模块，部分前端代码的临时下线不影响整体，或者服务端压力过大对流量进行限制时，前端如何对用户进行友好的显示和持续的引导3. 数百个运营活动涉及到非常多的内容维护和策略调整，如何保证不出现错误的图片、错误的链接、性能保持一致高效

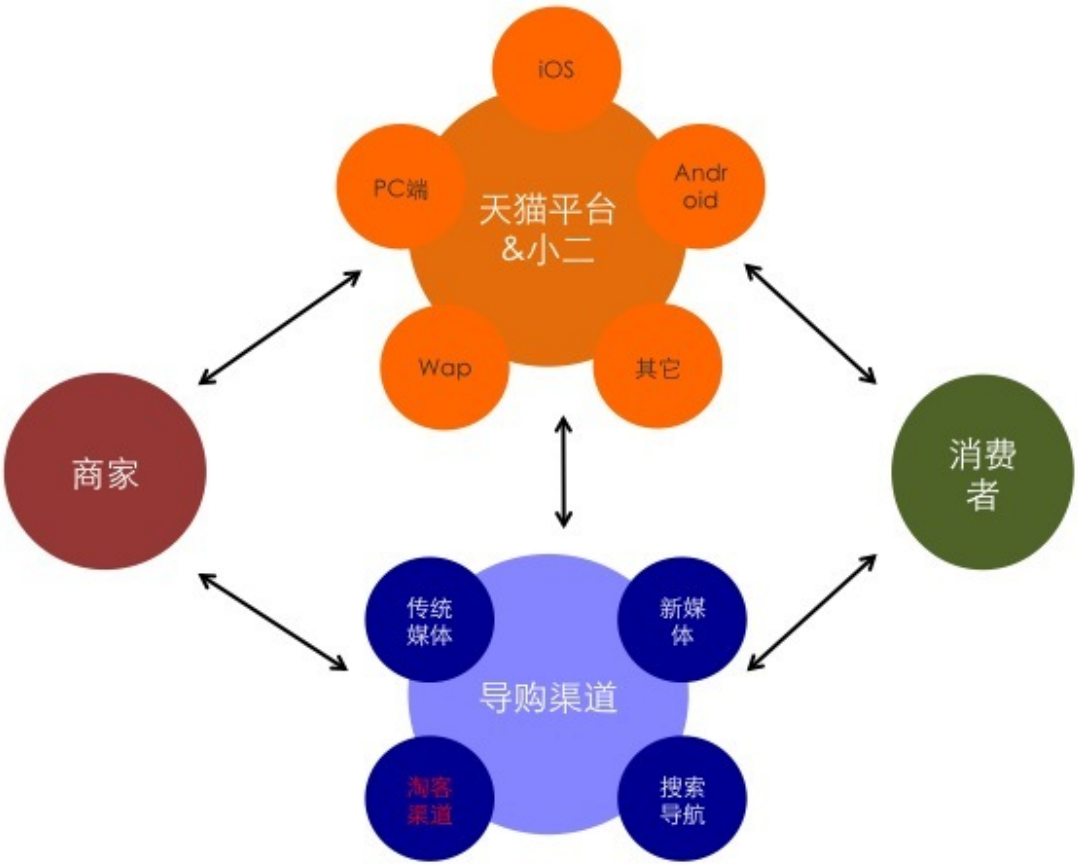
除了这些传统挑战，今年双11的用户环境发生了巨大的变化，用户终端的碎片化导致以往前端只需要解决多浏览器兼容，现在不仅仅面临多个终端（桌面、平板、手机）多个浏览器的兼容，还要应对终端和终端之间的互动。下面从五个方面来谈谈我们面临这些挑战的解决思路、方案和经验。

一、跨终端

移动智能终端的兴起，用户不再像以前一样只是停留在Desktop上，Pad、Phone的大量涌现使得用户终端碎片化。对于电商平台而言，直接导致了如下问题：

1. 导购渠道碎片化：
   a. 消费者在各个导购渠道上看到的核心业务不一致
   b. 相同业务的核心人机交互和业务逻辑差异较大
2. 商家和小二各导购渠道运营需求无法满足
3. 实施成本高，基础设施重复建设

图1-1 ： 导购渠道碎片化



针对这个问题，我们的结论是：

1. 用户流和信息流在跨终端流动，它穿越了传统终端（电视、广播、平面媒体...）、智能终端（PHONE、PAD、DESKTOP、TV...）和软件终端（BROWSER、SNS、IM、ALL APPS...），这些流动不是一维单向平面的，而是多维双向立体的。
2. 移动用户正在快速成长，正在或即将成为大多数，同时移动智能设备的人机交互形式代表未来，会引导桌面的人机交互形式向移动化变革
3. 人的本性、业务的本质和商业模式的本质不会随着终端的改变而改变，只是形式会随着终端而进化。

因此为各种碎片场景给用户提供本质相同的电商服务，技术上面临最大的挑战就是如何快速、高质量地开发跨终端的产品。我们的核心思想是利用Mobile First的理念，同一个团队实现同一个业务所有主流终端的需求，实现业务的跨终端。这样做的好处非常明显：保证了技术和业务上的一致性，提升了研发的效率和品质。

图1-2：同一个团队维护同一个业务的跨终端模型

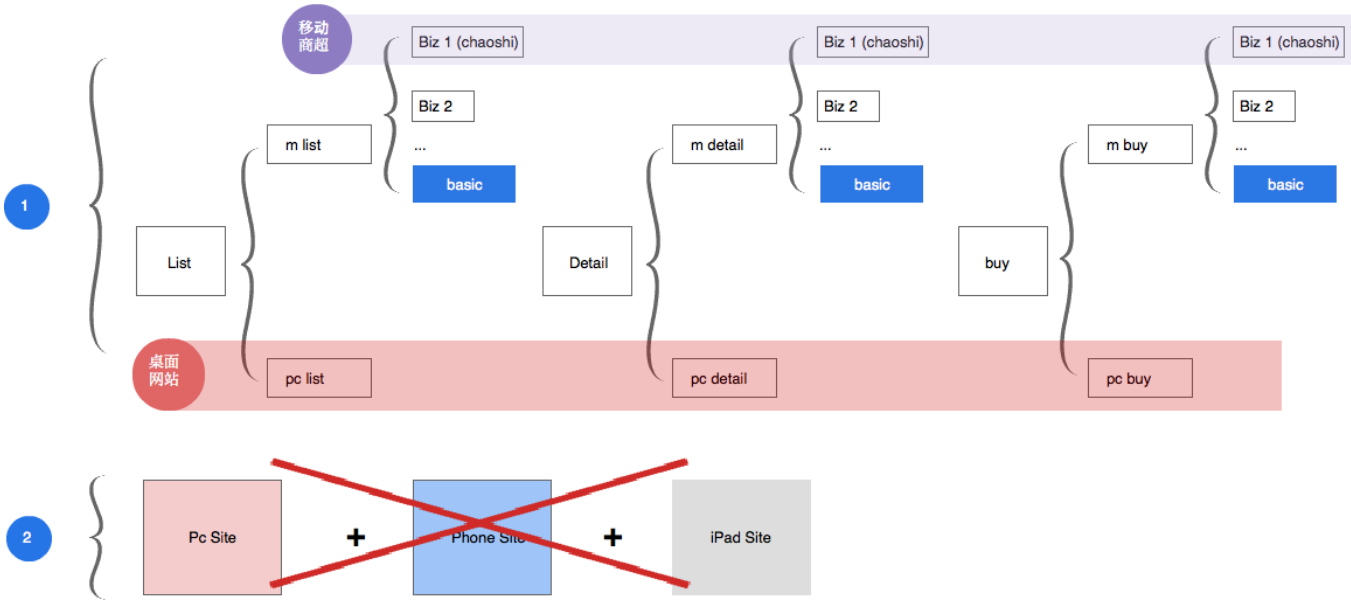
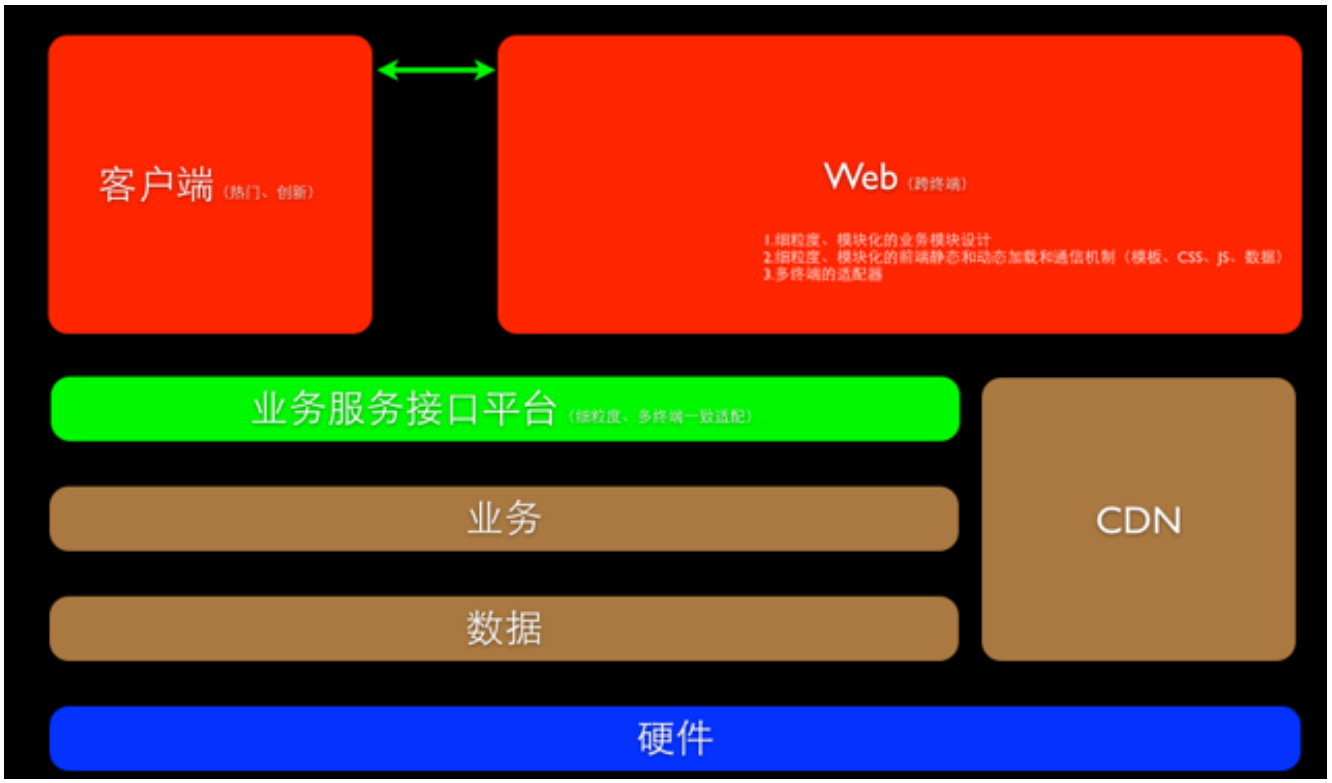


图1-3：跨终端的技术架构



天猫的核心交易链路list、detail、buy都有Mobile版本，而每个业务的Mobile开发都是由对应Desktop业务开发团队完成的，即detail的前端团队不仅仅完成desktop detail的业务，还负责mobile的开发维护，list、buy同样如此。这样做的原因是：从业务角度讲，desktop和mobile其实都是同一产品，要实现同样的核心功能，不同终端只是因为终端展现的人机交互不同而已，同样的产品没有理由每个终端由单独的团队来维护。举个例子，我们不可能因为页面需要兼容IE和Firefox，就把前端

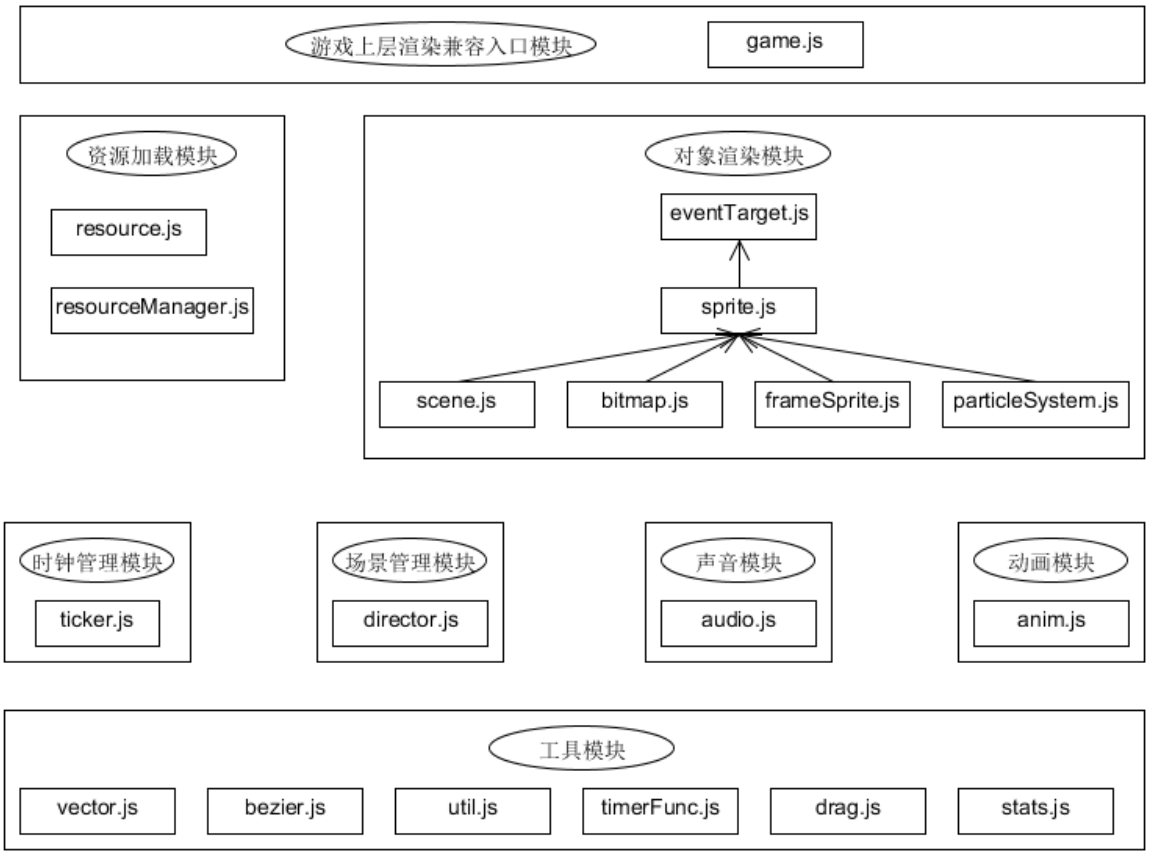
团队分为一部分人做IE，一部分做Firefox，对于desktop和mobile也是如此；另一方面，一个团队维护可以节约大量的成本和对业务更深度的认识，只有一个团队的维护开发，才更有可能从整体上去看待desktop、mobile开发中的问题，考虑如何优化，提升效率，也由于本身业务的一致性，开发desktop的经验对于开发mobile也省去大量对于业务理解的成本。所以基于以上原因，我们对于跨终端的实现从组织架构角度是通过一个团队来完成，而双11也印证了这种做法的成功。

今年双十一预热阶段有一个大型游戏“狂欢城”，往年对于这种大型游戏都是外包给外部广告公司通过Flash完成。基于今年跨终端的考虑，使用Flash技术无法支持Mobile，所以我们今年采用Web技术的方式实现，具体是:

- 占比 66% 的高级浏览器使用原生canvas
- IE6/7/8 浏览器方案，使用flashcanvas兼容

对于低级浏览器兼容方面对比过几款常见canvas兼容方案，相对flashcanvas是比较完善的方式。这对于天猫是首次尝试，业界类似的案例也非常少见。为了降低风险我们很早就开始准备技术方案，包括通过Web技术把去年的Flash版的狂欢城完全实现了一遍。实际开发中也遇到很多挑战：flashcanvas的一些bug，如何解决游戏性能效果等。从最终结果看，无论从游戏的跳失率、互动率还是转化率等业务指标都较去年有了大幅提升，同时也为公司节省了一笔不小的外包费用，最重要的是这为以后大型互动游戏的跨终端做好铺垫。最终我们沉淀了一套游戏框架，见下图:

图1-4：游戏框架

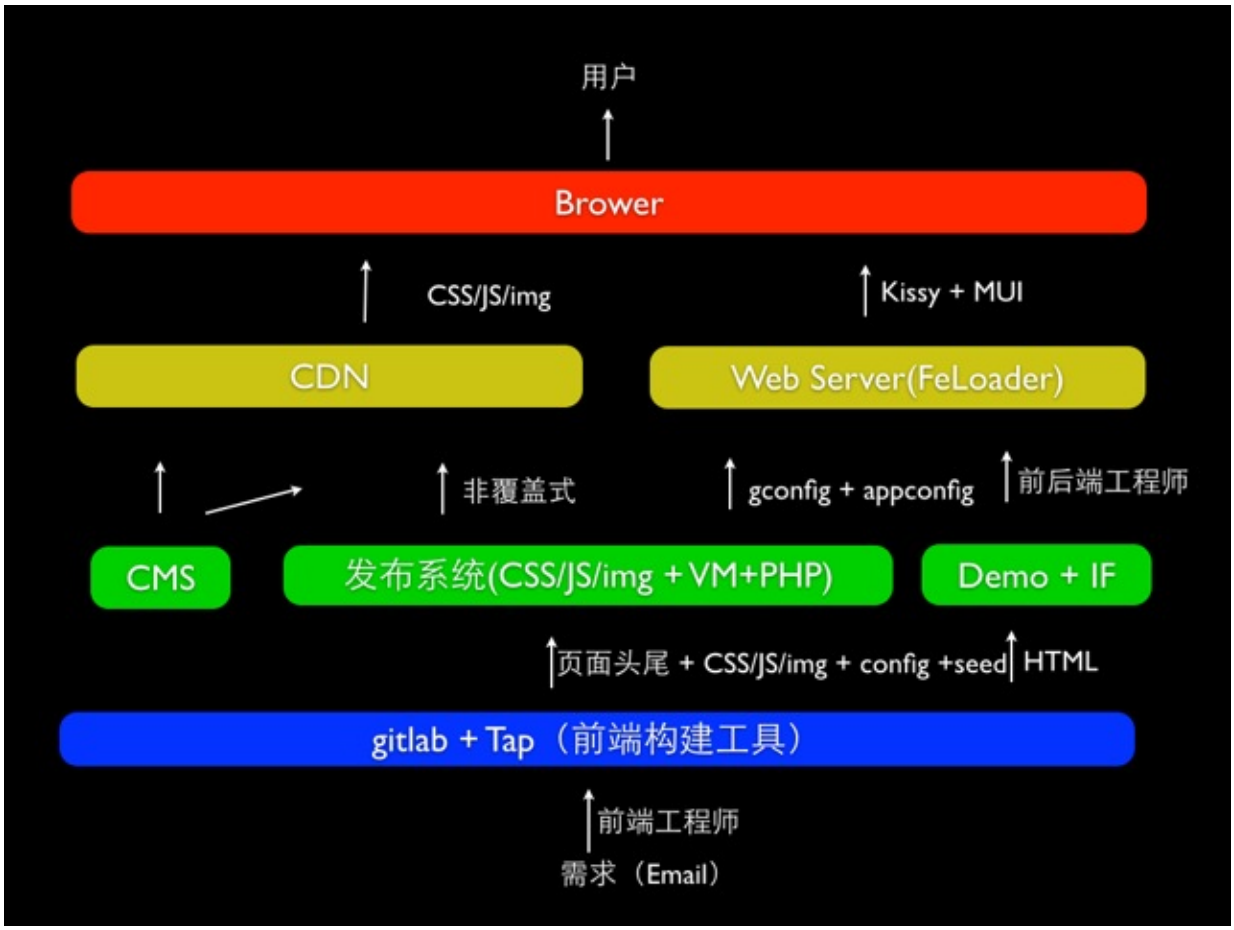


双十一当天，在天猫首页还有捉猫猫的游戏，实际上这个游戏前端只用了1天时间就完成了Desktop、Pad、Phone的完全兼容，这得益于初期对于游戏的准备，也再次证明了可以通过一致的开发方案，低成本地解决跨终端问题。

二、一致高效灵活新鲜的前端架构和发布机制MAP

跨终端的前端解决方案的前提是相同业务在不同终端上的业务本质和人机交互本质是一致的，为了保证能够把Web快速地推进到各个终端，我们需要建立统一的前端架构和发布机制（我们称之为MAP（Tmall Front-end Architecture & Published Mechanism的简写））。MAP目标不只是为双11服务，但每年的双11绝对是验证这一年前端架构和发布机制改进成果的最佳时机，满足了双11的苛刻环境就意味着为下一年常态化运作打下了坚实的基础。今年MAP实现了从1.0到2.0的巨大升级，目标就是一致高效灵活新鲜的前端开发环境。基础设施的一致性是高效自动化的基础，也是前端代码能够自由流动的基础，而一致性对本质和未来的探讨是达到灵活的前提，同时保持一致性实现方案的新鲜度和先进性，不断地自我革新，是前端架构和发布机制能够长期存活的关键因素。

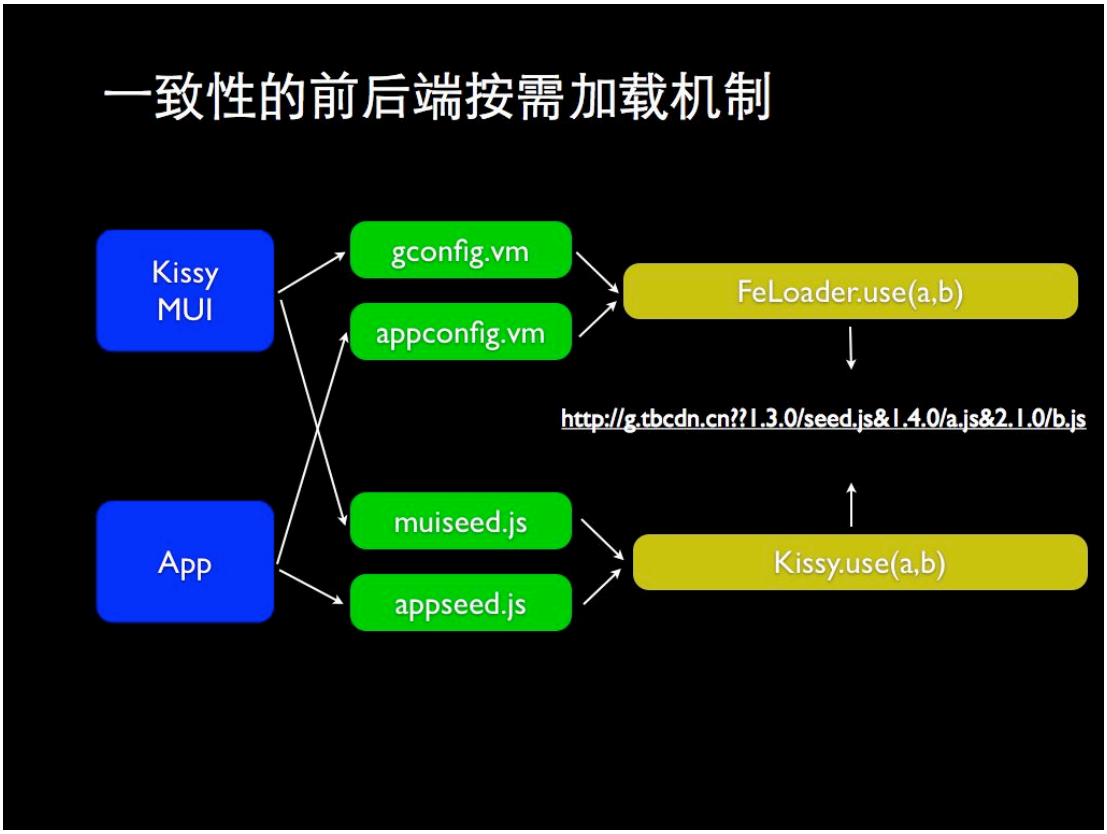
图2-1：MAP2.0的架构



MAP2.0主要体现在以下4个方面：

1. CDN静态文件全部是非覆盖式发布，且采用语义化路径。使用CDN存储静态文件是Web性能优化最关键的步骤，也导致了静态文件和动态文件发布的差异性，通过静态文件的非覆盖式发布，就能实现静态文件和动态文件发布的分离，静态文件开发完毕可提前随时发布，从而避开前后端发布无法同步而导致短时间内错乱问题，同时在出现问题时可以轻易地实现前端和后端各自独立回滚。我们把语义化版本号引入到文件路径作为非覆盖式发布的标识，比如你查看天猫首页原文件会发现类似文件路径 http://g.tbcdn.cn/tm/fp/1.7.1/core.js，不需任何解释，前端也能很快知晓这个路径表达的含义。通过语义化版本路径能够让前端和合作伙伴不借助任何说明就能知晓代码的架构和含义，这种规划不仅仅全面的提升了认知和理解效率，也为自动化发布工具打下了坚实的基础及同多版本迭代开发的项目管理有机地结合起来。
2. 双层扁平化前端代码组织结构和前后端一致的静态文件引用方式。整个天猫前端代码的组织基础是Kissy（基础前端框架）和MUI（天猫前端业务组件），通过保证Kissy + MUI在整个天猫前端的一致性实现，业务组件能够一致快速地部署到全站，比如双11页面

左侧的天猫工具栏和全站顶通。Kissy和MUI都有大量的组件，我们通过全站统一的配置文件实现这些组件引用的部署，这样每个页面只需要按需加载组件即可。由于性能优化的需要，首屏首次访问需在服务端直接渲染完后推送到客户端，其他内容可以是JavaScript动态渲染，所以天猫的前后端都部署了Loader机制，实现了前后端一致且灵活的加载机制。比如后端引用前端文件代码是FeLoader.use(a,b)，在前端则是Kissy.use(a,b)，其执行之后输出的代码是一致的，都类似 http://g.tbcdn.cn??1.3.0/seed.js&1.4.0/a.js&2.1.0/b.js，从这个URL看出，通过Loader机制能够同时在前后端自动实现一致的依赖关系处理和合并加载。图2-2：一致性的前后端按需加载机制



3. 扁平化细粒度的代码仓库和开发构建工具。整个天猫的前端代码都是通过git管理起来，每个仓库都是最小的独立业务发布单元，代码结构扁平和细粒度，保证了前端发布独立且去中心化，再集合基于Node.js的前端构建工具TAP，能容易地把代码快速地发布到仓库、Demo服务器、测试服务器和线上环境，同时能在各个发布环节进行合法性验证，保证发布质量。这一切就是因为基础机制的一致性才能够轻易地实现开发和发布的自动化，保证双11能够享受到快速且高品质的发布。
4. 端到端的通用接口规范。随着前端开发从兼容多浏览器到跨终端，使得开发从一套数据一个View层变成了一套数据多个View层，前

后端分离变成必然，这其中最关键点就是前后端的数据接口，所以我们建立名为IF的规范，它提供一套数据接口规范（Web前端与后端服务的接口、Web服务端模板（如VM）同后端服务的接口、Web前端与Native客户端的接口、Native客户端与后端服务的接口和后端服务之前的接口），提供了一个接口工具集支持整个开发生命周期中的接口需求。通过IF，保证了基本数据接口的一致性和合法性，从而大大的避免了接口带来的风险，提升了整个前端产品的稳定性。

天猫工具栏部署在天猫全站（页面左侧），这是一个采用“框架+插件”机制开发且业务复杂的组件，它既是 MAP 的试金石，同时也可能是未来全站 Web App 的雏形。：1. 由于 MAP 保证了全站一致的基础库，工具栏才免于对基础库做兼容处理，节约超过10%的时间，保证了项目的高质量上线。2. 工具栏本身采用的“框架+插件”的架构思想和 MAP “扁平化细粒度”的设计理念保持一致。所有插件被设计为可由独立的代码仓库管理、并可单独发布，而工具栏框架可以实现插件“热拔插”；这样就保证了部署于全站的框架的稳定性，同时又适应了工具栏业务频繁变更的需求（每个插件承载一项业务）。3. 工具栏中存在的20多个异步接口直接使用 IF维护接口的定义、文档、mock 数据及校验，确保了数十人协同开发、接口频繁改动、存在接口降级的情况下在双十一没有产生一起由接口直接导致的线上故障。

三、前端降级预案和限流机制

在双十一大并发、高流量的业务场景下，前端还需要考虑如何保证用户的优雅体验及系统稳定的平衡，如后端系统当天有可能出错或响应缓慢。如何在这种情况下给到用户可用的体验，是前端面临的挑战。我们技术方案上主要从降级和限流两个维度：

在业务层面，我们会降级掉一些不太重要当天对用户影响小的业务，如：list的minisite展示、list关键词的关联推荐、首页的个性化推荐等，这些在双十一当天都被降级不显示，这样可以减少额外的系统压力，让我们的后端机器优先保障核心交易链路的稳定。对于前端在技术层面在开发各个功能模块时就需要考虑：模块不展示的可能，需要实现模块细粒度的“热拔插”。对于双十一，需要梳理全站前端可能降级的情况，制定前端的降级的模块列表，针对这些模块进行降级和取消降级的测试。对于降级备案的执行分为两种情况：双十一之前执行和双十一当天判断。如前面提到的list关键词关联推荐，就属于11月初即降级；而比如list的小图展现降级方案，是在双十一当天根据CDN流量情况选择执行（最终没有执行，CDN容量较充足）。总之，前端为双十一备战的一项重要的工作是梳理全站（首页、list、detail、登录、交易等）降级预案，这需要充分依赖于前端架构的细粒度和灵活性。

除了降级预案，还要需要考虑另一种有意思的场景，就是后端接口限流。为了保证后端服务的稳定性，在后端无法支撑高流量或并发的情况下，后端会启用一种保护机制，对部分请求直接返回静态的内容，避开对后端应用造成压力，另一方面，前后端的通讯接口是通过异步请求(ajax,jsonp等)的，如果不做任何处理，限流时接口会和普通页面一样返回一个静态HTML内容，这样就会引起脚本解析报错，甚至流程无法走通。所以必须对异步请求返回特定的数据，标识这种情况，以便前端拿到标识做相应处理。对于异步请求，首先需要解决的问题是如何识别，涉及两种：一种通过XMLHttpRequest发出，这种后端可以通过 HTTP header中 X-Requested-With=XMLHttpRequest 识别；另一种jsonp请求，我们的做法是：对于URL中存在callback=jsonpxxx的情况即认为jsonp请求，后端识别后即可针对异步请求返回特定json数据，整体方案如下图：

图3-1：限流的技术方案

举个例子：天猫首页焦点图，有一张是依赖后端接口返回数据展示，在接口被限流的情况下，前端根据标识使用备份的打底图片展示，而不影响用户体验；而去年当用户被限流时只能js报错，无法处理。

图3-2：不做处理时首页焦点图情况

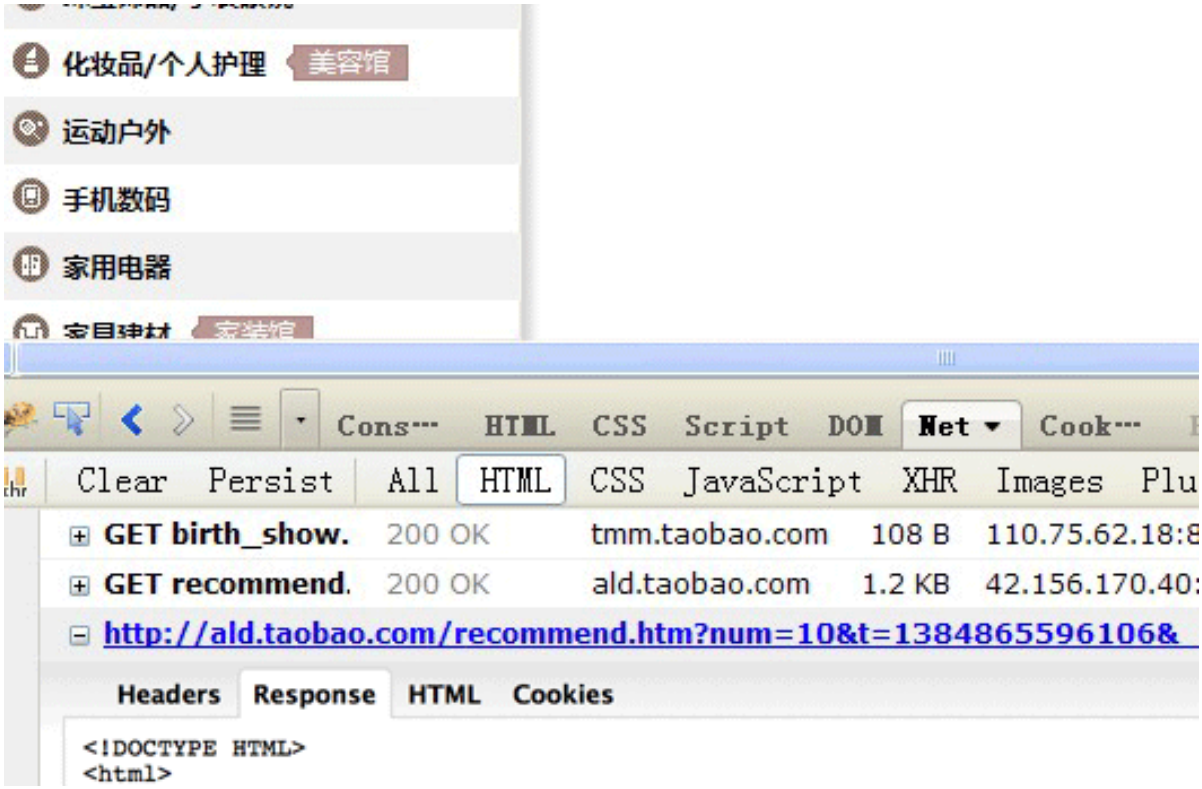


图3-3：做了处理时天猫首页焦点图接口被限流情况

四、前端质量检查和监控平台猫须

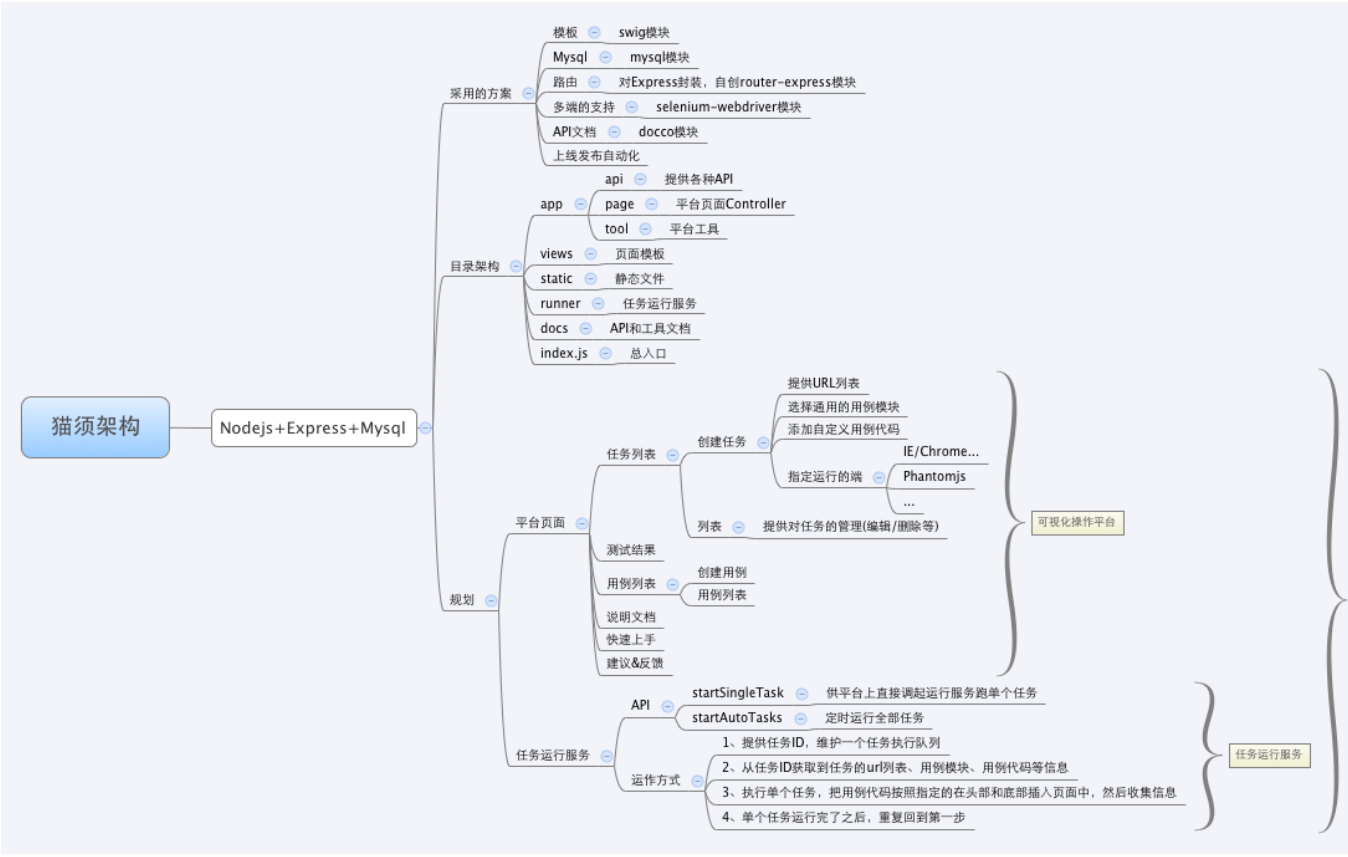
双十一活动中前端短时间需要产出大量的页面，这其中涉及到多个角色的共同协作，面对消费者极高的要求，如何更好地保证页面质量是一个挑战。我们从几个维度来迎接这个挑战: 内容填写时、发布前和发布后。对于活动页面，大量内容是运营同学在后端填写配置的，其中很难避免

人为的疏忽错误，比如链接填写错误、图片大小不符合规范、标签嵌套错误等。我们总结了保证内容质量的一些规则，在CMS系统中填写完数据后，依照这些规则，对内容进行检验，来避免数据层面的一些常见错误。还有一些代码层面的问题，比如：代码中是否存在安全漏洞(XSS，UTF-7漏洞，JSONP、Ajax等异步请求的漏洞)，是否存在一些内部的静态资源引用等，所以在发布前还有道检查。在发布后，需要检查线上是否正常，线上系统是否有冲突。我们积累了大量的这方面的规则和经验，这次把这些规则和经验自动化起来，通过猫须实现发布前检测和发布后监控，通过把对应规则写成测试用例，部署到在线系统，然后定时自动检查，产出功能及性能测试报告，并跟内部统一的BUG跟踪流程平台打通，提给相应同学进行跟踪改进。实时且高效的上线前检查和上线后监控非常有效，尤其是在双11这种场景下能够快速地保证大量运营页面的品质和性能。

猫须分为两部分：操作平台和运行服务。提供操作平台的关键目的是提供各种丰富的工具去自动化生成用例代码，这也是自动化思想的一个体现，比如录制交互行为、录制样式、自动创建XSS URL等等，这些都可以非常方便的自动化生成用例，对于这方面的监控需求，可以做到无需手写用例代码，这也很大的减小用例代码的维护成本，大不了重新录制一遍而已。在平台上，还有一个非常重要的思想是：汇聚优质的通用用例模块，增加用例代码的重复利用度，统一维护，为创建任务服务。运行服务是运行任务的服务，跟平台可以结合也可以单独部署运行，方便后续调度任务的需要。在运行的时候，任务运行完了之后，如果该任务下有错误信息，会及时通过邮件等方式通知到任务的创建者，推动快速解决问题。

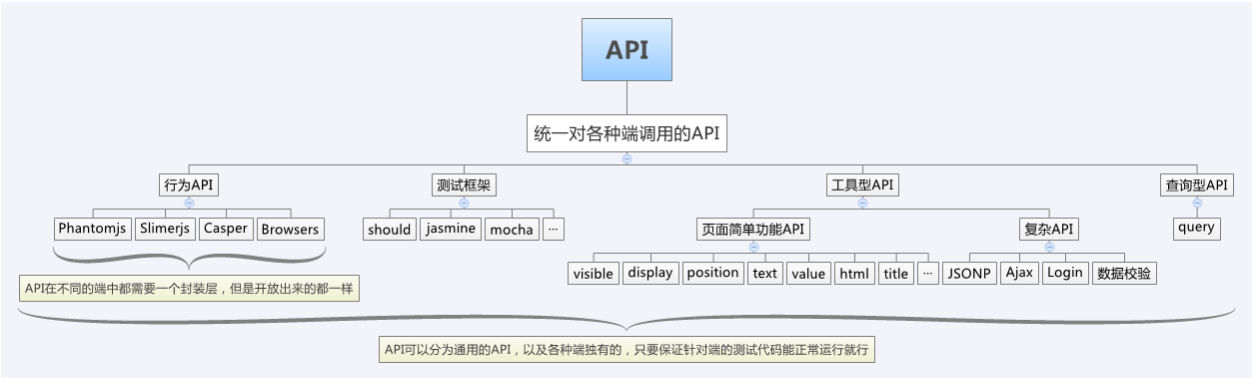
猫须从九月初开始建设，需要在十月初投入使用，需要在一个月左右的时间内，完成对双十一众多会场页面、重要业务页面的监控任务。之所以能够这么快的实现，是因为猫须是构建在大量业界开源项目之上，我们依据需要进行了深度开发，其架构如下。

图4-1：猫须架构图



猫须采用Nodejs+Express架构开发，采用Nodejs的原因是可以快速地架构和开发，非常多成熟的服务端模块可以投入使用，在数据库上采用了Mysql，Nodejs里面有mysql模块可以做很好的支持。猫须支持多种浏览器，当然也包括了Phantomjs，这个得益于selenium-webdriver，但只利用它来做启动浏览器和关闭浏览器，并不依赖它做提供的操作浏览器和页面的API，猫须有自己非常丰富的API。下面是猫须中提供的API:

图4-2：猫须的API



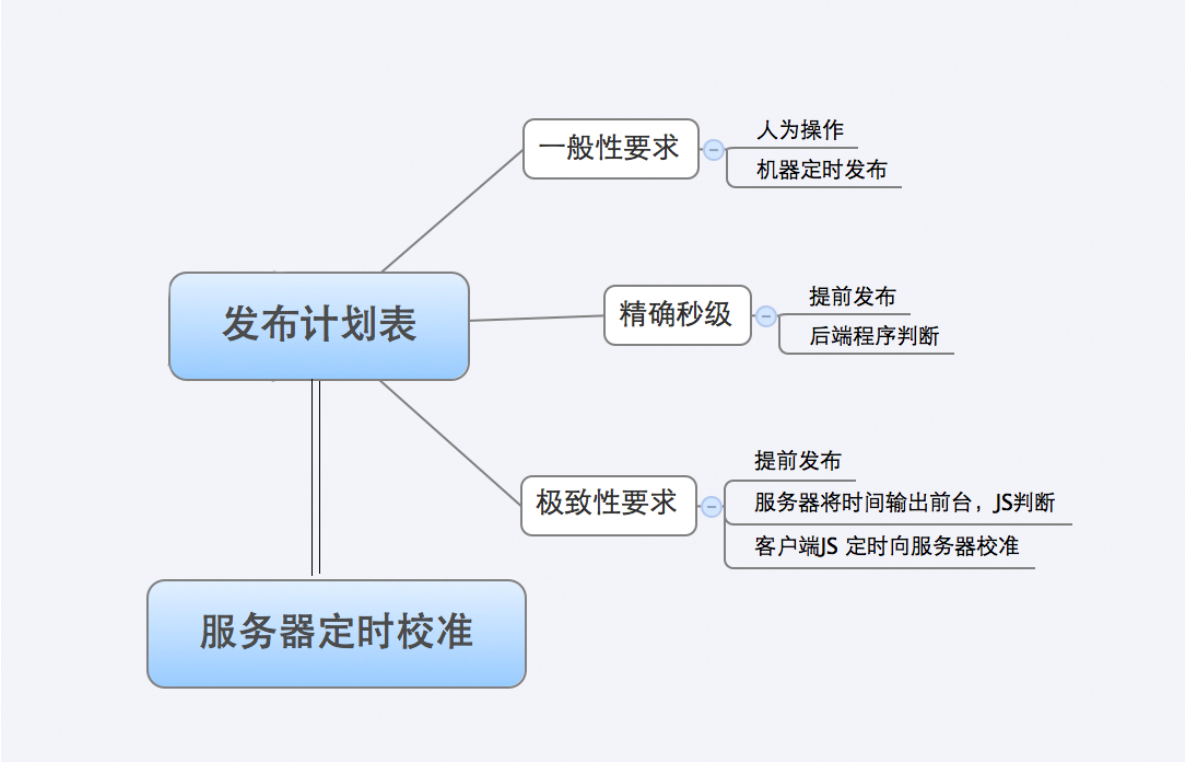
总之、猫须的核心目标是上线前检测，上线后对线上页面的质量做到实时监控，及时发现问题，并及时的解决问题；同时也要最大的降低用例代码的维护成本；创建一套适合于天猫的前端标准规则体系。

五、精确到秒的发布计划和实时发布的能力

双11就是一场战争，所以在当天的安排上，所有的准备都是为这一天我们能够活着度过。当天的发布是有非常苛刻的时间要求，我们制定了精确到秒的发布计划，包括：几分几秒需要发布哪些页面，由谁具体操作，double check人员是谁，影响范围等。双十一当天所有前端严格按照时间发布表执行，并更新每次发布状态，做到信息同步。除了组织安排上的保证，由于发布到CDN或服务器都存在一定延时，所以技术上还需要保证时间的准确性，整体上把其分成如下三种情况：

1. 对于时间要求不高的发布，如：会场页面预热到正式的切换，通过人为、内部cms系统定时发布完成
2. 对于精确的秒级的业务展示，如当天00:00:00秒需要出现的顶通，我们是提前发布，通过后端代码判断服务器时间展示，对于服务器每分钟都会自动校准时间，保证服务器时间稳定性。
3. 对于用户时实性要求极高，如：00:00:00秒天猫首页出现的开幕式，单单靠第二种处理还不够，因为用户有可能之前已经打开首页，如果不刷新将看不到开幕式，而开幕式又是持续极短，我们希望尽量所有用户看到，对于这种情况，就需要前端Javascript代码轮询判断时间来给到用户展现，但Javascript本身执行时间有不准确问题，为了保证准确性，JS再不断地去服务器校准时间，以保证用户端可以准时看到效果。

图5-1：精确到秒的时间方案



战争的魅力之一就是意外，所以除了精确到秒的前端发布外，还有大量的依据当天状态的临时冲锋，在保证大流量高并发下用户继续完成购物需求外，还需要快速地进行特别内容的开发和随时整站上线，这个过程虽然和平时几乎一样，但当时能够快速稳定完成就是依赖于平时所做的一致性的前端架构和发布机制MAP，充分显示日常开发的能力。

总之，双十一有类似前端降级和限流这些独特的挑战，也有类似跨终端解决方案这样的前端创新。在这个充满挑战和压力的环境中，无论是个人还是团队都得到了极大的锻炼和成长，累的要死但依旧激情满怀，这是其非常迷人的地方。