我们讨论的场景依然是对可用性有着极高要求的web大型站点，这种类型的web站点通常前后端的代码是分离的(前端指JS、CSS代码，后简称样式代码; 后端指模板、Java等后端语言) — 各自有独立的代码分支，且部署在不同的服务器上。这种的架构能带给我们更好的代码可维护性，以及对静态和动态资源更好的服务能力。但也不免会带来一些问题：

　　例如需求同时涉及到前后端代码的改动，且与线上的当前版本不兼容，这种情下况我们是希望前后端代码能够“同时上线“生效的，但由于两者部署在不同的服务器，发布流程通常也是分开进行，要做到“同时上线”并不是一件容易的事情，就会造成一定时间内页面发生异常的情况(大型web站点的后端服务器通常是一个集群，完成一次代码的集群部署有时需要20–30分钟)。

**一： 常见的解决方案**

　　在网站开发的过程中，上述的场景是非常常见的，页面总是会改版，一个区块被反复多次修改那不是什么新鲜事~ 我们的开发同学在遇到这样的情况时，常见的解决方案有这几种：

　　1) 对出现不兼容的JS或是CSS文件重新命名，代码分支中多提交一个物理文件，同时更新后端模板上的引用地址。

　　2) 不重新命名JS或是CSS文件，直接把新需求所需的新的样式或是JS追加在原有文件中，本次发布完成后再清理上个版本的历史代码。

　　3) 直接忽略前后端代码发布过程中的20-30分钟的时间差，用户不投诉就好：)

　　以上的几种处理方式要么降低了代码可维护性，要么牺牲了用户体验甚至可用性，都不是很好的解决方案，我们理想的解决方案需要做到：

　　●前端代码SVN分支中不出现为兼容线上版本而保留的历史代码或是文件。

　　●完全消除前后端发布过程中因代码不同时上线而出现的可用性问题，做到平滑过渡。

　　●保持前后端发布流程的独立性，不带来新问题(比如多次发布的问题)。

　　●平滑发布方案做到自动化，对开发者透明。

**二： 总体解决思路**

　　其实解决这个问题的总体思路比较简单，如果发布系统能做到对前端代码的增量发布(线上同时存在多个版本)，则可以在发布时先发布前端的CSS和JS文件，然后再控制后端的代码生效，那么无论后端的服务器集群有多少台，也能够做到对不兼容需求发布的平滑切换。

　　思路简单，但其中有几个个关键问题要解决：

　　1) 前端代码增量发布的方案?

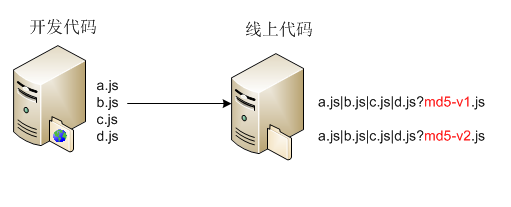
　　2) 后端服务器何时更新对前端资源文件(CSS、JS)的引用?

　　3)上述的两点如何由发布系统自动完成?且对开发者透明?

**三： 关键问题解决**

　　●**前端代码的增量方案**前端代码发布增量方案有多种方式，经过我们多次讨论比较后，我们选择了对合并后的JS、CSS文件增量发布的方案。

　　为便于理解，大家可以先打开阿里巴巴搜索list页面 查看其源代码，你会发现这个页面的CSS文件及JS文件引用路径很长，每个长路径在静态资源服务器上会对应着一个物理文件，它在开发状态的SVN分支中是不存在的，而是在页面加载时通过动态合并服务 把页面上的各个JS或CSS小文件合并而成 。合并的原理这里不细说，但最终一定是在静态服务器上生成了一个物理文件。我们可以对这个合并后的文件做增量机制，比如当动态合并服务检测到有JS、CSS小文件发生变化时(前端代码发布成功后)，就生成一份新的合并代码的物理文件，而同时保留之前的那份老的合并文件，那么在一段时间内就会有两份前端代码同时提供服务，达到了增量的目的，示意图：



　　这种增量方案相比其它增量方案的优势在于：

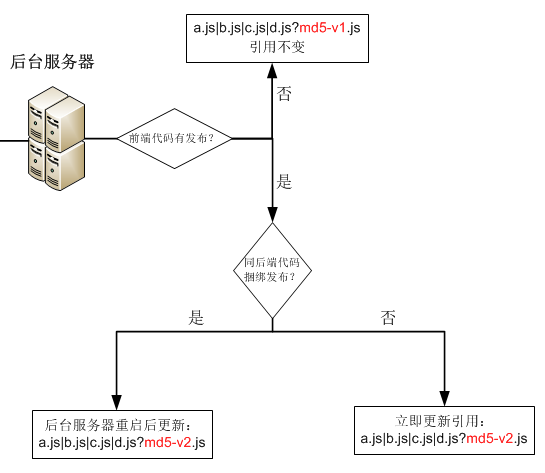
　　1)发布系统自动完成

　　2) 避免了对单个文件增量所需要的文件映射关系数据库的维护。

　　3) 对静态资源服务器上文件数量的影响到最小，在完成平滑发布后，老版本的合并文件可以由发布系统自动删除。

●**后端服务器模板自动更新JS、CSS引用路径机制**

　　有了多份同时在线的前端文件，就具备了前后端应用平滑发布的基础，接着要解决的就是何时让后端服务器切换引用路径的问题了。对此，我们的总体更新策略如下图所示：



　　其中有两个判断节点：**前端代码是否有发布** & **同后端代码捆绑发布**

　　前端代码是否有发布的判断我们可以通过动态合并服务定时轮询静态资源服务器上的style版本索引文件，版本索引文件的生成原理很简单，是通过扫描style文件内容生成对应的md5码，内容变了，生成的md5码也就变了，动态合并服务就能够知道是否有前端代码发布。

　　是否与后端代码捆绑发布的判断，是通过在前后端项目发布的初期，对需要同时发布的前后端应用设定了一个**锁定标识** 来实现 (发布系统自动完成)。若需要同时发布，则对应后台应用标识为lock=true，否则lock=false，动态合并服务通过后台应用名查询到该标识状态即可。

　　当后台应用处于**非锁定状态** 时，当检测到有JS、CSS文件发布时立即更新模板上的样式文件引用地址。

　　若后台应用处于**锁定状态** ，则延迟对样式文件引用地址的更新，直至应用重启。后台服务的发布完成通常是伴随着一次服务重启的，重启后的应用初始化就是更新样式文件引用地址的时机。

　　至此，平滑发布的几个关键问题已经解决，一套自动化、对开发者友好的平滑发布方案设计完成。

**四： 总体流程图**

