Hybrid APP离线缓存方案整理

一、项目背景

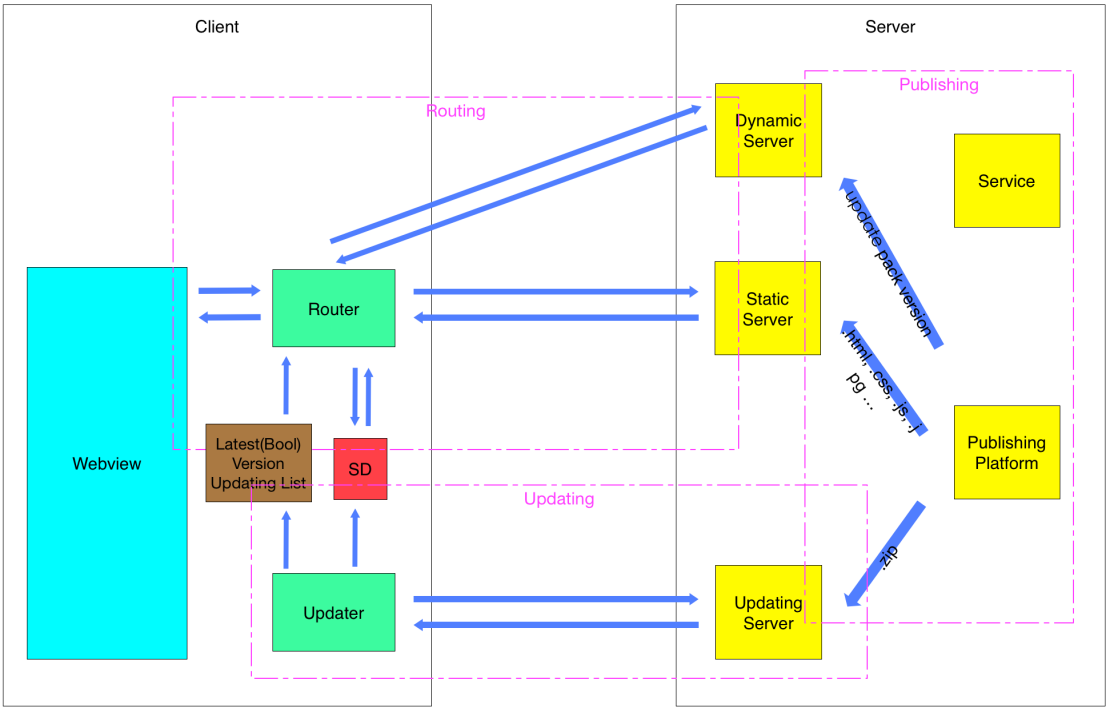
为了解决Hybrid APP页面资源加载缓慢的问题，我们利用APP本地离线缓存等技术，提供了一套性能好、安全高、请求少、更新及时、容错性高，并提供一套支持用户灰度、平滑发布、无网访问、紧急停服等功能的完整解决方案。

二、初步方案

1. 方案概述

1. 系统架构

1.1基本结构图



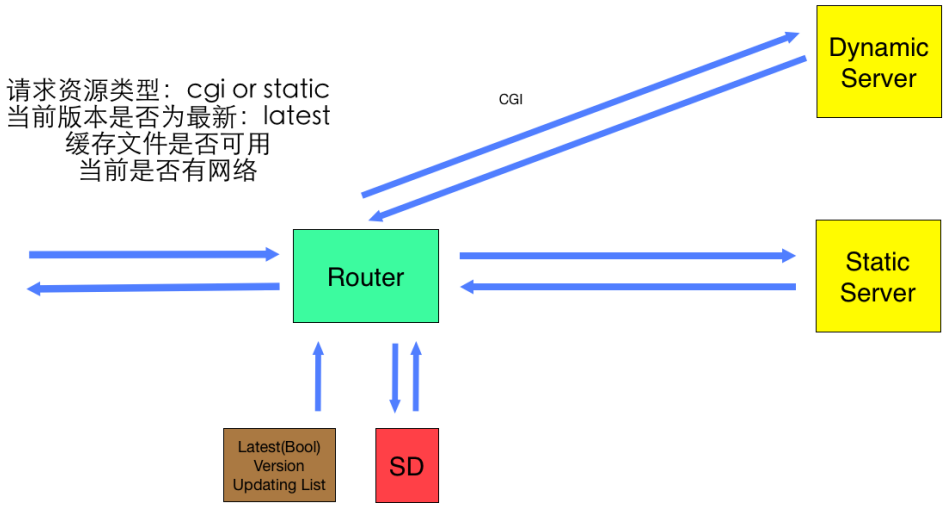
1.2 web 前端

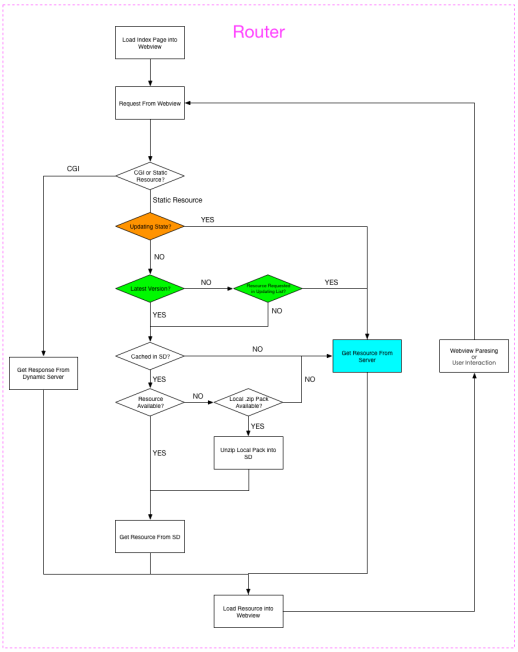
web 前端负责请求静态资源和动态CGI数据，并在webview中渲染页面。

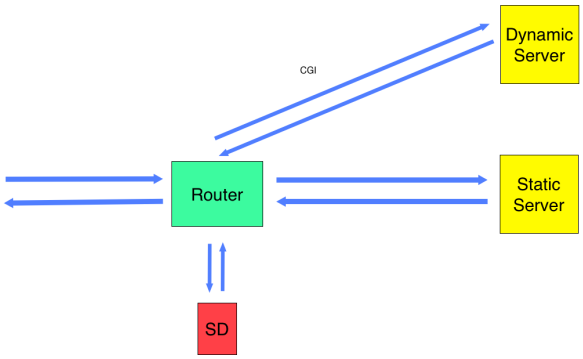
1.3 客户端

1.3.1 Router

Router截获前端(Webview)发送的请求，根据当前网络状况（有网无网）、前端请求的数据参数（强制使用线上）、缓存包的状态和可用性、服务器指令（dynamic server & updating server）（路由策略），来决定返回的资源是使用客户端缓存包资源、请求服务器资源、还是直接使用前端localstorage的存储资源。

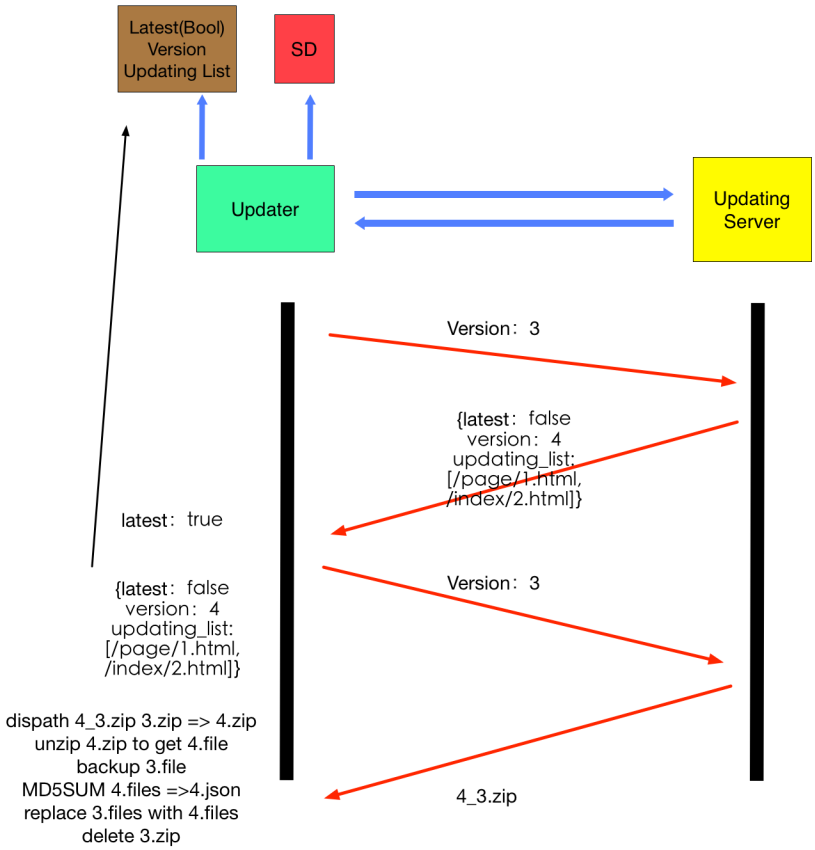


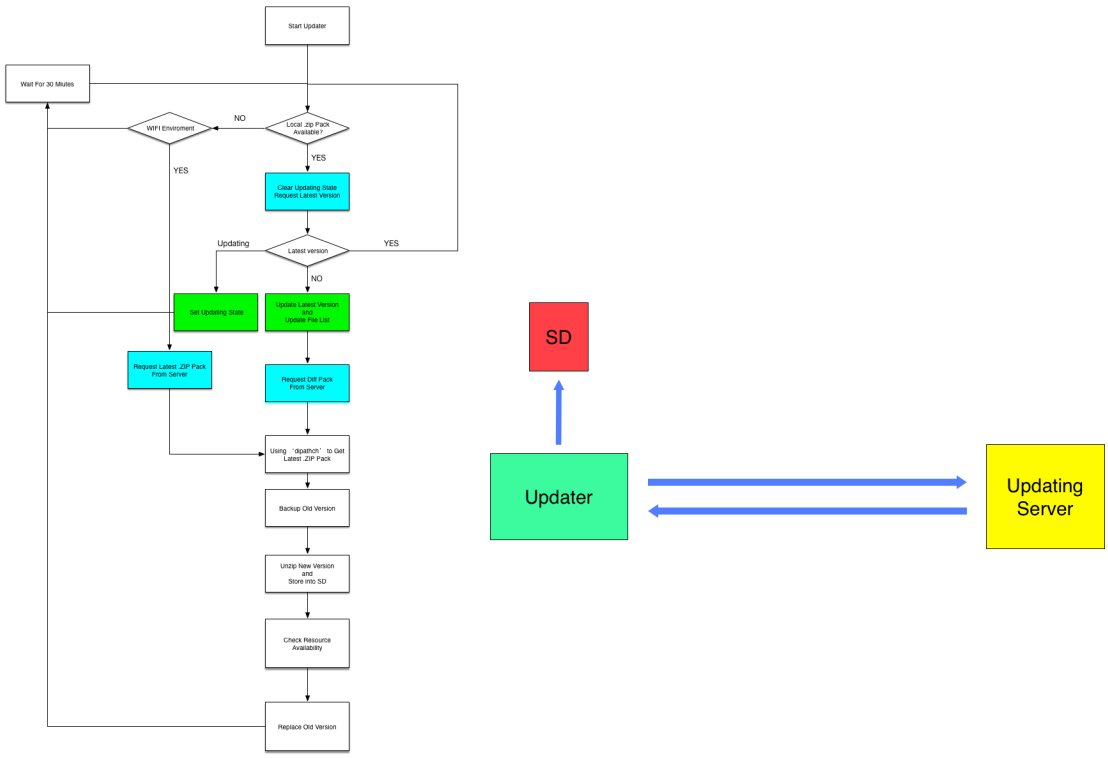




1.3.2 Updater

Updater根据客户端更新策略进行启动更新、定时更新，向updating server请求获取缓存包的更新状态、版本号和更新文件列表。并在需要更新时，发送更新请求。





1.4 Server端

1.4.1 Dynamic Server

> 获取客户端的请求，先判断是否处于停服状态、是否为灰度用户、接口使用的版本是否过期，之后根据相应业务返回相应的请求接口。

1.4.2 Static Server

> 查询是否为灰度用户，提供相应的`html`、`js`、`css`、图片和字体等静态资源服务。

1.4.3 Updating Server

> 查询是否为灰度用户，并根据用户提交的版本号，返回获取缓存包的更新状态、版本号和更新文件列表；并根据用户更新包请求，返回相应的增量包或全量包。

1.4.4 Pack Service Platform

> 发布新版本的静态资源、资源包、增量包以及相应的包信息，并同步更新到`Static Server`、`Updating Server`、 `Dynamic Server`。

1.4.5 Gray(?) Query Server

> 提供灰度用户查询服务

2 通信接口

2.2 Web前端与Router

2.3 Router与Updater

2.4 Router 与Dynamic Server

2.5 Router 与Static Server

2.6 Updater与Updating Server

2.7 Dynamic Server 与 Pack Service Platform

2.8 Static Server与 Pack Service Platform

2.9 Updating Server 与 Pack Service Platform

三、路由和更新策略

1. 路由策略

1.1 检查网络环境：

> 无网环境使用本地缓存资源和localstorage数据；

1.2 检查前端请求参数：

> 强制请求线上资源，则直接请求线上Static Server资源并返回；

1.3 检查前端请求类型：

> cgi：请求dynamic Server；
> 静态资源：请求本地或Static Server

1.4 检查客户端缓存是否可用:

> 如果可用则使用客户端本地缓存；如果不可用则请求static server资源。

2. 客户端更新策略

客户端更新策略指的是 客户端的updater更新请求时，如何返回相应的更新包的策略。一共有启动更新、定时更新、下推更新三种基本方式。由于下推更新会对服务器造成较大的压力，故我们采用了自动更新和定时更新2种方案。

启动更新：客户端再启动时则向updating server发送更新请求；
定时更新：客户端在启动更新后，每隔30分钟则向updating server发送更新请求。

3. 服务器更新策略

服务器更新策略指的是updating server再收到客户端的updater更新请求时，如何返回相应的更新包的策略。作为基础策略有静态更新策略和动态更新策略2种，由于其缺点我们选择了将2种策略结合的方式进行版本发布与更新。

静态更新

每一次版本发布，都需要将当前版本和之前的所有版本（使用二进制比对工具bsdiff）进行比对，并更新版本增量。如v0.3.zip版本发布时，更新增量v0.3_0.1.zip, v0.3_0.2.zip。
客户端updater每次带着版本号（如：v0.1时），向后端请求版本更新时；后端返回给v0.3_0.1.zip增量文件进行更新。客户端收到v0.3_0.1.zip时，将v0.3_0.1.zip文件和v0.1.zip文件合并成新的，v0.3.zip文件并缓存到本地。

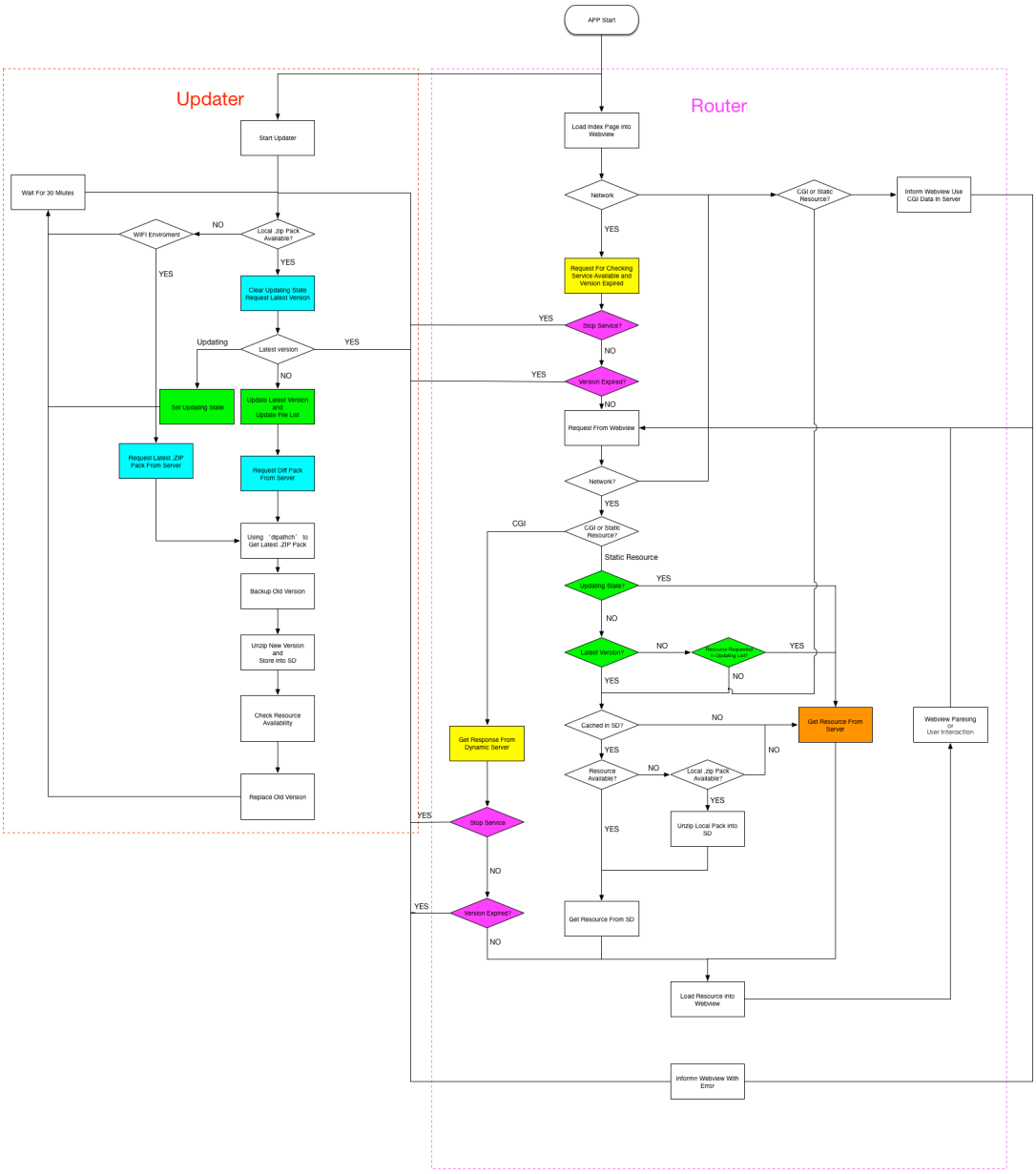
动态更新

客户端updater每次带着版本号（如：v0.1时），向后端请求版本更新时；服务器将最新版本与之比对，动态计算获取差量，进行增量版本更新。并将生成的差量版本缓存。

静态＋动态更新（推荐方案）

将最新版本，最近的所有小版本进行比对（20个），生成所有小版本与最新文件的增量文件。客户端请求时，如果客户端请求的版本属于所有小版本，则采用静态更新策略；如果不在所有小版本文件中，则使用动态更新。

四、系统流程



五、应用场景

1. 首次安装

首先 Start Updater 检查需要全量更新；
全量下载完毕后（体验优化，检测是否WIFI环境，非WIFI环境强提醒）Start Router

2. 启动时无更新

Updater检查无更新；
Router检查缓存可用，返回webview缓存内容。

启动时无更新，但是本地资源损坏（包括zip包）
Updater在WIFI环境下下载全量包，非WIFI环境下不请求。
Router请求线上资源；

3. 启动时有更新

Updater检查有更新，请求更新资源包，下载增量包；
Router启动时检查版本过期，直接访问线上资源，更新完成后使用本地缓存。

4. 使用中有更新（正式上代码前30分钟开始）（包括频繁更新）

Updater收到正在更新中的回包，设定更新中状态；
Router检查到更新中的状态，直接请求线上资源。

更新完毕后
Updater检查发现版本过期，下载增量包；
Router，直接访问线上资源，更新完成后使用本地缓存。

5. 紧急上线

Updater设置更新中状态；
Router直接访问线上资源。
Dynamic Server判断当前版本是否为最新版本。

6. 启动遇到停服

Updater设置更新中状态；
Router直接访问线上资源或展示停服页面。

7. 使用中遇到停服

Updater设置更新中状态；
Router在未调用AJAX接口时，使用本地缓存；调用AJAX后，收到后端返回的停服指令，展示停服页面。

8. 无网环境

```
Updater无返回；
Router  对于静态资源直接使用本地缓存，若本地缓存不可用，通知前端展示错误；对于CGI数据，直接通知前端使用localstorage，若没有存储，展示相应错误。
```

五、其他

缓存内容

```
大部分的静态资源，包括但不限于js、css、img、font
大部分html（活动页面不建议缓存）
部分cgi数据（分级缓存）
```

版本差量对比算法

```
bsdiff & bspatch
```

六、遗留问题

紧急上线的时效性

dynamic server 与 updating server的同步性

灰度服务查询

参考文献

[1]. 15年双11手淘前端技术巡演 - H5性能最佳实践

[2]. 手机 QQ Hybrid App 优化新思路