# 基于Html5以及MongoDB的视频、照片墙研究

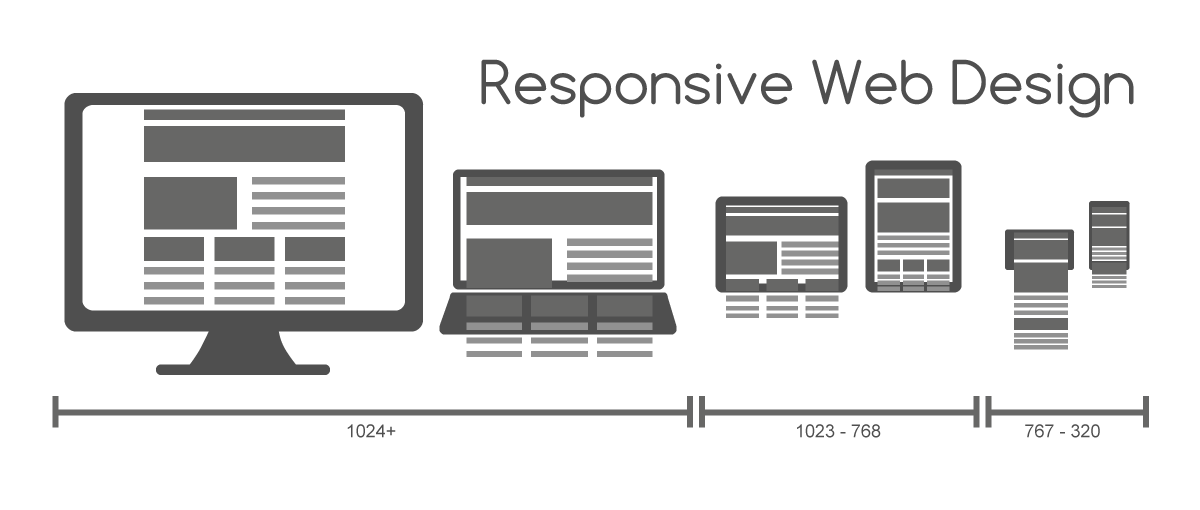
## 系统目标

随着互联网技术的快速发展，大众对网络的依赖性也越来越强，接入互联网的设备也快速增长。各种基于互联网的手机 App 如雨后春笋般纷纷拔地而起。同时，巨大的网络带宽压力，高延迟，各种虚拟化技术的选择，这些问题接踵而至，这也对较晚步入互联网时代的我国来说无疑也是一个巨大的挑战。

在这个信息爆炸的时代中，很多人都沉迷于自己的世界之中，如何保持人与人之间的沟通，记录并分享每个人遇到的新奇的，好玩的事物变得尤为重要。因此，Fudan MSE Pins Board 从加强人与人之间的分享与沟通为切入点，做到了拍摄，存储，分享，管理等一体化集成，打造一个别具一格的照片视频墙。

## 系统架构

### 前端

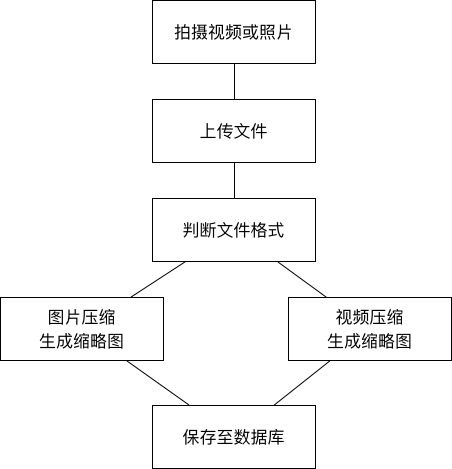
**响应式布局**  
据中国互联网中心报告，国内网民达6.49亿，其中86%为移动端用户。因此，移动端的用户显得比以往更为重要。使用响应式布局，只需要开发一份版本即能适应桌面版本、移动端版本。  
  
HTML5 Video播放器  
传统网站使用flash开发视频播放器，客户端必须安装flash播放器，而且iOS平台上不支持播放flash。使用HTML5 Video播放器可以跨平台支持各种设备，无需安装任何插件。  
  
**AngularJS**  
Demo版本中使用了jQuery 获取服务端数据并展示，实际开发中前端架构更为复杂，单纯使用jQuery无法满足复杂的前端架构，使用AngularJS的MVVM的分层模式以及双向绑定的特性能更规范、更快速的开发复杂的前端页面。

Gulp  
使用Gulp前端构建工具，可满足前端自动化测试，CSS、JS文件校验、压缩、打包的各种自动化构建操作。大大增加前端构建的效率。

### 后端 Node.JS 随着近年来JavaScript人气高涨，Node.js逐渐成为了主流的服务端开发语言。 它统一了编程语言和数据格式（JSON），令前后端开发协作更为便捷、快速。 并且，Node.JS的高并发处理，高扩展性非常适合Web开发。因此，我们在Demo项目中选择了使用Node.js作为服务端开发语言。

**FFmpeg**视频转换使用Linux下开源的视频处理、编码工具: FFmpeg，将客户端上传的不同的视频格式统一转换成H264编码的mp4文件，可用于直接在支持html5的客户端播放。同时使用FFmpeg截取视频截图作为视频照片墙的缩略图。  
由于FFmpeg编码过程中，每个cpu核心只能同时处理一个视频，真实项目中并发量大的场景下，如果在硬件条件允许的情况下可使用分布式进行视频压缩，如果条件有限可使用队列消息的方式对视频进行依次编码。

**ImageMagick**  
图片转换使用Linux下的开源工具： ImageMagick，将客户端上传的图片进行压缩，并裁剪生成缩略图用于视频照片墙的显示。

以下是用户上传视频或照片的后端业务处理逻辑：  
****

#### 数据库

MongoDB  
MongoDB 是一个基于分布式文件存储的数据库，为 WEB 应用提供可扩展的高性能数据存储解决方案。使用JSON格式插入及查询数据，与node.js无缝交互，很大程度地提高了开发效率。

### 硬件

应用云主机

数据库云主机

### 开发环境

#### 操作系统

OS X EI Capitan（版本： 10.11.4）

#### 开发工具

Sublime Text 3 Build 3103

Vim version 7.4.944

#### 浏览器

Chrome

Edge

Firefox

Safari

IE

#### 测试工具

easyABC

polygraph

#### 代码托管工具

git

## 网络和硬件基础架构

### 虚拟机

VPS使用的虚拟技术种类有很多，如OpenVZ、Xen、KVM、Xen和HVM与PV。在XEN中pv是半虚拟化，hvm是全虚拟化，pv只能用于Linux内核的系统，效率更高，hvm可以虚拟所有常见操作系统(可以使用Windows)，理论效率比pv略低，另外，hvm需要cpu虚拟化指令支持，pv无此要求。KVM是新兴的虚拟化项目，出售KVM的VPS商家不多，但由于KVM虚拟技术是比较强悍的。而OpenVZ是一个类似于Linux-VServer的操作系级全虚拟化解决方案，目前基于Xen和OpenVZ的VPS服务商比较多。所以 Fudan MSE Pins Board 在选择虚拟机的时候做了以下比较。

#### OpenVZ

OpenVZ是操作系统级别的虚拟化技术，是底层操作系统上的一层应用，这意味着易于理解和低权重开销，一般来说也意味着更优的性能。但是OpenVZ配置起来比较灵活，给服务商改低限制的机会。

优势：这种类型的虚拟机，许可的内存都非常大，CPU也是十分强劲，而且卖家很多，可比性也很强。性价比超高。

劣势：超卖，没有不超卖的，导致各种石头盘，钻石盘。连带的cpu也被过度分割导致性能升值不如其标明的1/10。再有就是内存，基本上OpenVZ技术没有独占的，都是共享，别人多了你就少了，而且这种技术最大的劣势就是内存下来后直接就是当机。还有开设vpn各种不方便。

#### Xen

Xen是半虚拟化技术，它并不是一个真正的虚拟机，而是相当于自己运行了一个内核的实例，可以自由的加载内核模块，虚拟的内存和IO，稳定而且可预测。分Xen+pv+和Xen+hvm，区别就是pv只支持Linux，而hvm支持win系统。

优势：内存独占，虽然小但是保证分配的到。部分虚拟技术决定了就算是超卖也不会超太离谱。所以一般的母鸡就算超了，也很少导致石头盘。当然小鸡有pt就除外了。另外就是即使内存再低也不会当掉，只是会无响应一段时间。

劣势：内存小，硬盘小，带宽小（以上3点均和同价位的OpenVZ对比）。因为没法超卖了，为了赚钱只能把这3个基本配置降下来了。还有就是供应商参差不齐，而且大部分在说Xen的时候都木有明说是pv还是hvm，其实pv的性能是优于hvm的。

#### KVM

优势：和Xen类似，比Xen更好的一点是，KVM是完全虚拟的，所以不分pv和hvm的区别，所有的KVM类型的虚拟技术都可以装各种Linux的发行版和各种win的发行版，不管供应商在主页有没有写明是否支持win，只要你配置够win运行，那就肯定可以装的上去，只是方法的问题而已。

劣势：恰恰因为KVM可以装任意类型的操作系统，导致了折腾帝甚至在128m的机器上装了win2003（上去后直接所有的cpu和内存都用于硬盘IO了，不明白这些人想干神马）。其结果就是所有的KVM邻居都得看你周围邻居的脸色。这么说把，一个node下只要有5、6台这种折腾帝，对不起，你的硬盘基本就是石头盘了。

个人比较倾向于Xen，因为它是对供应商和和客户来说都是可以接受的折衷方案。OpenVZ超卖太厉害，KVM等看RP，这2种都不适合那些不想折腾只想做站的站长。

### 负载均衡

#### Nginx



随着互联网信息的爆炸性增长，负载均衡（load balance）已经不再是一个很陌生的话题，顾名思义，负载均衡即是将负载分摊到不同的服务单元，既保证服务的可用性，又保证响应足够快，给用户很好的体验。快速增长的访问量和数据流量催生了各式各样的负载均衡产品，很多专业的负载均衡硬件提供了很好的功能，但却价格不菲，这使得负载均衡软件大受欢迎，Nginx就是其中的一个。

Nginx第一个公开版本发布于2004年，2011年发布了1.0版本。它的特点是稳定性高、功能强大、资源消耗低，从其目前的市场占有而言，Nginx大有与apache抢市场的势头。其中不得不提到的一个特性就是其负载均衡功能，这也成了很多公司选择它的主要原因。

Nginx的负载均衡策略可以划分为两大类：内置策略和扩展策略。内置策略包含加权轮询和IP Hash，在默认情况下这两种策略会编译进Nginx内核，只需在Nginx配置中指明参数即可。扩展策略有很多，如Fair、通用Hash、consistent Hash等，默认不编译进Nginx内核。下面来分别分析一下各种策略的特点。

##### 加权轮询

轮询的原理很简单，首先我们介绍一下轮询的基本流程。如下是处理一次请求的流程图：



图中有两点需要注意，第一，如果可以把加权轮询算法分为先深搜索和先广搜索，那么Nginx采用的是先深搜索算法，即将首先将请求都分给高权重的机器，直到该机器的权值降到了比其他机器低，才开始将请求分给下一个高权重的机器；第二，当所有后端机器都down掉时，Nginx会立即将所有机器的标志位清成初始状态，以避免造成所有的机器都处在timeout的状态，从而导致整个前端被夯住。

##### IP Hash

IP Hash是Nginx内置的另一个负载均衡的策略，流程和轮询很类似，只是其中的算法和具体的策略有些变化，如下图所示：



Hash值既与IP有关又与后端机器的数量有关。经过测试，上述算法可以连续产生1045个互异的value，这是该算法的硬限制。对此Nginx使用了保护机制，当经过20次Hash仍然找不到可用的机器时，算法退化成轮询。因此，从本质上说，IP Hash算法是一种变相的轮询算法，如果两个IP的初始Hash值恰好相同，那么来自这两个IP的请求将永远落在同一台服务器上，这为均衡性埋下了很深的隐患。

##### Fair

Fair策略是扩展策略，默认不被编译进Nginx内核。其原理是根据后端服务器的响应时间判断负载情况，从中选出负载最轻的机器进行分流。这种策略具有很强的自适应性，但是实际的网络环境往往不是那么简单，因此要慎用。

##### 通用Hash、一致性Hash

这两种也是扩展策略，在具体的实现上有些差别，通用Hash比较简单，可以以Nginx内置的变量为Key进行Hash，一致性Hash采用了Nginx内置的一致性Hash环，可以支持Memcache。

##### Nginx负载均衡小结

下图为IP Hash策略，IP Hash是Nginx内置策略，可以看做是前两种策略的特例：以来源IP为Key。由于测试工具不便于模拟海量IP下的请求，因此这里截取线上实际的情况加以分析，如下图所示：



图中前1/3使用轮询策略，中间段使用IP Hash策略，后1/3仍然是轮询策略。可以明显的看出，IP Hash的均衡性存在着很大的问题。原因并不难分析，在实际的网络环境中，有大量的高校出口路由器IP、企业出口路由器IP等网络节点，这些节点带来的流量往往是普通用户的成百上千倍，而IP Hash策略恰恰是按照IP来划分流量，因此造成上述后果也就自然而然了。



不难发现，无论哪种策略都不是万金油，在具体的场景下应该选择哪种策略一定程度上依赖于使用者对这些策略的熟悉程度。

#### F F5 BIG-IP

F5负载均衡器是应用交付网络的全球领导者F5 Networks公司提供的一个负载均衡器专用设备，F5 BIG-IP LTM 的官方名称叫做本地流量管理器，可以做4-7层负载均衡，具有负载均衡、应用交换、会话交换、状态监控、智能网络地址转换、通用持续性、响应错误处理、IPv6网关、高级路由、智能端口镜像、SSL加速、智能HTTP压缩、TCP优化、第7层速率整形、内容缓冲、内容转换、连接加速、高速缓存、Cookie加密、选择性内容加密、应用攻击过滤、拒绝服务(DoS)攻击和SYN Flood保护、防火墙—包过滤、包消毒等功能。

以下是F5 BIG-IP用作HTTP负载均衡器的主要功能：

F5 BIG-IP提供12种灵活的算法将所有流量均衡的分配到各个服务器，而面对用户，只是一台虚拟服务器。

F5 BIG-IP可以确认应用程序能否对请求返回对应的数据。假如F5 BIG-IP后面的某一台服务器发生服务停止、死机等故障，F5会检查出来并将该服务器标识为宕机，从而不将用户的访问请求传送到该台发生故障的服务器上。这样，只要其它的服务器正常，用户的访问就不会受到影响。宕机一旦修复，F5 BIG-IP就会自动查证应用已能对客户请求作出正确响应并恢复向该服务器传送。

F5 BIG-IP具有动态Session的会话保持功能。

F5 BIG-IP的iRules功能可以做HTTP内容过滤，根据不同的域名、URL，将访问请求传送到不同的服务器。

##### 多链路的负载均衡和冗余

与互联网络相关的关键业务都需要安排和配置多条ISP接入链路以保证网络服务的质量,消除单点故障,减少停机时间｡多条ISP接入的方案并不是简单的多条不同的广域网络的路由问题,因为不同的ISP有不同自治域,所以必须考虑到两种情况下如何实现多条链路的负载均衡:内部的应用系统和网络工作站在访问互联网络的服务和网站时如何能够在多条不同的链路中动态分配和负载均衡,这也被称为OUTBOUND流量的负载均衡｡互联网络的外部用户如何在外部访问内部的网站和应用系统时也能够动态的在多条链路上平衡分配,并在一条链路中断的时候能够智能地自动切换到另外一条链路到达服务器和应用系统,这也被称作为INBOUND流量的负载均衡｡

F5 的BIG-IP LC可以智能的解决以上两个问题:对于OUTBOUND流量,BIG-IP LC接收到流量以后,可以智能的将OUTBOUND流量分配到不同的INTERNET接口,并做源地址的NAT,可以指定某一合法IP地址进行源地址的 NAT,也可以用BIG-IP LC的接口地址自动映射,保证数据包返回时能够正确接收｡对于INBOUND流量,BIG-IP LC分别绑定两个ISP 服务商的公网地址,解析来自两个ISP服务商的DNS解析请求｡BIG-IP LC不仅可以根据服务器的健康状况和响应速度回应LDNS相应的IP地址,还可以通过两条链路分别与LDNS建立连接,根据RTT时间判断链路的好坏,并且综合以上两个参数回应LDNS相应的IP地址｡

##### 防火墙负载均衡

考虑到绝大多数的防火墙只能达到线速的30%吞吐能力,故要使系统达到设计要求的线速处理能力,必须添加多台防火墙,以满足系统要求｡然而,防火墙必须要求数据同进同出,否则连接将被拒绝｡如何解决防火墙的负载均衡问题,是关系到整个系统的稳定性的关键问题｡F5的防火墙负载均衡方案,能够为用户提供异构防火墙的负载均衡与故障自动排除能力｡典型的提高防火墙处理能力的方法是采用“防火墙三明治"的方法,以实现透明设备的持续性｡这可满足某些要求客户为成功安全完成交易必须通过同一防火墙的应用程序的要求,也能够维护原来的网络安全隔离的要求｡

##### 服务器负载均衡

对于所有的对外提供服务的服务器,均可以在BIG-IP上配置Virtual Server实现负载均衡,同时BIG-IP可持续检查服务器的健康状态,一旦发现故障服务器,则将其从负载均衡组中摘除｡BIG-IP利用虚拟IP地址(VIP由IP地址和TCP/UDP应用的端口组成,它是一个地址)来为用户的一个或多个目标服务器(称为节点:目标服务器的IP地址和TCP/UDP应用的端口组成,它可以是internet的私网地址)提供服务｡因此,它能够为大量的基于TCP/IP的网络应用提供服务器负载均衡服务｡根据服务类型不同分别定义服务器群组,可以根据不同服务端口将流量导向到相应的服务器｡BIG-IP连续地对目标服务器进行L4到L7合理性检查,当用户通过VIP请求目标服务器服务时,BIG-IP根椐目标服务器之间性能和网络健康情况,选择性能最佳的服务器响应用户的请求｡如果能够充分利用所有的服务器资源,将所有流量均衡的分配到各个服务器,我们就可以有效地避免“不平衡"现象的发生｡利用UIE+iRules可以将TCP/UDP数据包打开,并搜索其中的特征数据,之后根据搜索到的特征数据作相应的规则处理｡因此可以根据用户访问内容的不同将流量导向到相应的服务器,例如:根据用户访问请求的URL将流量导向到相应的服务器｡

##### 系统高可用性

系统高可用性主要可以从以下几个方面考虑:

4.1.设备自身的高可用性:F5 BIG-IP专门优化的体系结构和卓越的处理能力保证99.999%的正常运行时间,在双机冗余模式下工作时可以实现毫秒级切换,保证系统稳定运行,另外还有冗余电源模块可选｡在采用双机备份方式时,备机切换时间最快会在200ms之内进行切换｡BIG-IP 产品是业界唯一的可以达到毫秒级切换的产品, 而且设计极为合理,所有会话通过Active 的BIG-IP 的同时,会把会话信息通过同步数据线同步到Backup的BIG-IP,保证在Backup BIG-IP内也有所有的用户访问会话信息;另外每台设备中的watchdog芯片通过心跳线监控对方设备的电频,当Active BIG-IP故障时,watchdog会首先发现,并通知Backup BIG-IP接管Shared IP,VIP等,完成切换过程,因为Backup BIG-IP中有事先同步好的会话信息,所以可以保证访问的畅通无阻｡

4.2.链路冗余:BIG-IP可以检测每条链路的运行状态和可用性,做到链路和ISP故障的实时检测｡一旦出现故障,流量将被透明动态的引导至其它可用链路｡通过监控和管理出入数据中心的双向流量,内部和外部用户均可保持网络的全时连接｡

4.3.服务器冗余,多台服务器同时提供服务,当某一台服务器故障不能提供服务时,用户的访问不会中断｡BIG-IP可以在OSI七层模型中的不同层面上对服务器进行健康检查,实时监测服务器健康状况,如果某台服务器出现故障,BIG-IP确定它无法提供服务后,就会将其在服务队列中清除,保证用户正常的访问应用,确保回应内容的正确性｡

##### 高度的安全性

BIG-IP采用防火墙的设计原理,是缺省拒绝设备,它可以为任何站点增加额外的安全保护,防御普通网络攻击｡可以通过支持命令行的SSH或支持浏览器管理的SSL方便､安全的进行远程管理,提高设备自身的安全性;能够拆除空闲连接防止拒绝服务攻击;能够执行源路由跟踪防止IP欺骗;拒绝没有ACK缓冲确认的SYN防止SYN攻击;拒绝teartop和land攻击;保护自己和服务器免受ICMP攻击;不运行SMTP､FTP､TELNET或其它易受攻击的后台程序｡

BIG-IP的Dynamic Reaping特性可以高效删除各类网络DoS攻击中的空闲连接,这可以保护BIG-IP不会因流量过多而瘫痪｡BIG-IP可以随着攻击量的增加而加快连接切断速率,从而提供一种具有极强适应能力､能够防御最大攻击量的解决方案｡BIG-IP的Delay Binding技术可以为部署在BIG-IP后面的服务器提供全面地SYN Flood保护｡此时,BIG-IP设备作为安全代理来有效保护整个网络｡BIG-IP可以和其它安全设备配合,构建动态安全防御体系｡BIG-IP可以根据用户单位时间内的连接数生成控制访问列表,将该列表加载到其它安全设备上,有效控制攻击流量｡F5负载均衡功能6.SSL加速,在每台BIG-IP上,都具有SSL硬件加速芯片,并且自带100个TPS的License,用户可以不通过单独付费,就可以拥有100个TPS的SSL 加速功能,节约了用户的投资｡在将来系统扩展时,可以简单的通过License升级的方式,获得更高的SSL加速性能｡

##### 系统管理

BIG-IP提供HTTPS､SSH､Telnet､SNMP等多种管理方式,用户客户端只需操作系统自带的浏览器软件即可,不需安装其它软件｡可以通过支持命令行的SSH或支持浏览器管理的SSL方便､安全的进行远程管理｡直观易用的Web图形用户界面大服务降低了多归属基础设施的实施成本和日常维护费用｡BIG-IP包含详尽的实时报告和历史纪录报告,可供评测站点流量､相关ISP性能和预计带宽计费周期｡管理员可以通过全面地报告功能充分掌握带宽资源的利用状况｡另外,通过F5 的i-Control 开发包,目前国内已有基于i-Control开发的网管软件x-control, 可以定制针对系统服务特点的监控系统,比如服务的流量情况､各种服务连接数､访问情况､节点的健康状况等等,进行可视化显示｡告警方式可以提供syslog､snmp trap､mail等方式｡

##### 其它

内存扩充能力:F5 BIG-IP 1000以上设备单机最大可扩充到2G内存,此时可支持400万并发回话｡升级能力:F5 所有设备均可通过软件方式升级,在服务有效期内,升级软件包由F5公司提供｡F5 NETWORKS已经发布其系统的最新版本BIG-IP V9.0,主要有以下特性:虚拟 IPV4 / IPV6 应用､加速Web应用高达3倍､减少66%甚至更多的基础架构成本､确保高优先级应用的性能､确保更高级别的可用性､大幅提高网络和应用安全性､强大的性能,简单的管理方式､无以匹敌的自适应能力和延展能力和突破的性能表现力｡其强大的HTTP压缩功能可以将用户下载时间缩短50%,节省80%的带宽｡IP地址过滤和带宽控制:BIG-IP可以根据访问控制列表对数据包进行过滤,并且针对某一关键应用进行带宽控制,确保关键应用的稳定运行｡配置管理及系统报告:F5 BIG-IP提供WEB 界面配置方式和命令行方式进行配置管理,并在其中提供了丰富的系统报告,更可通过i-Control自行开发复杂的配置及报告生成｡

#### Citrix NetScaler

#### 负载均衡小结

方案优缺点对比

基于硬件的方式(F5)

优点：能够直接通过智能交换机实现,处理能力更强，而且与系统无关，负载性能强更适用于一大堆设备、大访问量、简单应用

缺点：成本高，除设备价格高昂，而且配置冗余．很难想象后面服务器做一个集群，但最关键的负载均衡设备却是单点配置；无法有效掌握服务器及应用状态.

硬件负载均衡，一般都不管实际系统与应用的状态，而只是从网络层来判断，所以有时候系统处理能力已经不行了，但网络可能还来 得及反应（这种情况非常典型，比如应用服务器后面内存已经占用很多，但还没有彻底不行，如果网络传输量不大就未必在网络层能反映出来）

基于软件的方式(Nginx)

优点：基于系统与应用的负载均衡，能够更好地根据系统与应用的状况来分配负载。这对于复杂应用是很重要的，性价比高，实际上如果几台服务器，用F5之类的硬件产品显得有些浪费，而用软件就要合算得多，因为服务器同时还可以跑应用做集群等。

缺点：负载能力受服务器本身性能的影响，性能越好，负载能力越大。

综述：对我们管理系统应用环境来说，由于负载均衡器本身不需要对数据进行处理，性能瓶颈更多的是在于后台服务器，通常采用软负载均衡器已非常够用且其商业友好的软件源码授权使得我们可以非常灵活的设计，无逢的和我们管理系统平台相结合。

### CDN

随着 CDN 技术的发展，网宿科技、上海帝联、七牛云存储等各大 CDN 提供商的节点部署，我们已不在需要在其原理上投入过多精力，只需简单的提供需要缓存的数据以及修改DNS即可。

## 软件功能模块架构

## 系统技术路线和实现方案

## 系统设计

## 接口设计

## 业务流程

## 系统实现方案

## 大数据分析

### 目的

随着云时代的到来，大数据的威力逐渐为人们所了解，以及多媒体内容识别技术的日渐成熟，对于即将面临海量多媒体数据信息的 Fudan MSE Pins Board 来说，这样的机会同样不能错过。

由于原始照片中的EXIF包含着大量的可分析数据，例如：光圈大小、快门速度、ISO感光度、白平衡、曝光补偿、镜头焦距、闪光GN值、前后帘同步、闪光补偿、测光方式、相机型号、镜头型号、闪光灯型号、快门次数、拍摄时间、GPS等。

通过这些信息，我们可以分析出用户拍摄的设备的品牌分布、拍摄地点分布、拍摄风格、设备分辨率、设备大约寿命，人脸识别、物体识别等，非常多的信息。从而与设备厂商、旅游景点合作，分析种种平时难以准确获取的数据信息。

### 选择

Apache 基金会下的 Spark 再次引爆了大数据的话题。带着比 Hadoop MapReduce 速度要快 100 倍的承诺以及更加灵活方便的 API，一些人认为这或许预示着 Hadoop MapReduce 的终结。

作为一个开源的数据处理框架，Spark 是如何做到如此迅速地处理数据的呢？秘密就在于它是运行在集群的内存上的，而且不受限于 MapReduce 的二阶段范式。这大大加快了重复访问同一数据的速度。

Spark 既可以单独运行，也可以运行在 Hadoop YARN 上（注：Hadoop第二代框架中的改进框架，用于将资源管理和处理组件分开，基于YARN的结构不受 MapReduce 约束），此时 Spark 可以直接从 HDFS （Hadoop Distributed File System 分布式文件系统）中读取数据。 诸如 Yahoo（雅虎）、Intel（因特尔）、Baidu（百度）、Trend Micro（趋势科技）和 Groupon（高朋）等公司已经在使用 Spark 了。

听上去好像 Spark 已经注定要取代 Hadoop MapReduce 了。但真的是这样吗？

### 性能

Spark 在内存中处理数据，而 Hadoop MapReduce 是通过 map 和 reduce 操作在磁盘中处理数据。因此从这个角度上讲 Spark 的性能应该是超过 Hadoop MapReduce 的。

然而，既然在内存中处理，Spark 就需要很大的内存容量。就像一个标准的数据库系统操作一样， Spark 每次将处理过程加载到内存之中，然后该操作作为缓存一直保持在内存中直到下一步操作。如果 Spark 与其它资源需求型服务一同运行在 Hadoop YARN 上，又或者数据块太大以至于不能完全读入内存，此时 Spark 的性能就会有很大的降低。

与此相反， MapReduce 会在一个工作完成的时候立即结束该进程，因此它可以很容易的和其它服务共同运行而不会产生明显的性能降低。

当涉及需要重复读取同样的数据进行迭代式计算的时候，Spark 有着自身优势。 但是当涉及单次读取、类似 ETL （抽取、转换、加载）操作的任务，比如数据转化、数据整合等时，MapReduce 绝对是不二之选，因为它就是为此而生的。

小结：当数据大小适于读入内存，尤其是在专用集群上时，Spark 表现更好；Hadoop MapReduce 适用于那些数据不能全部读入内存的情况，同时它还可以与其它服务同时运行。

### 使用难度

Spark 有着灵活方便的Java，Scala和 Python 的API，同时对已经熟悉 SQL 的技术员工来说， Spark 还适用 Spark SQL（也就是之前被人熟知的 Shark）。多亏了 Spark 提供的简单易用的构造模块，我们可以很容易的编写自定义函数。它甚至还囊括了可以即时反馈的交互式命令模式。

Hadoop MapReduce 是用 Java 编写的，但由于其难于编程而备受诟病。尽管需要一定时间去学习语法，Pig 还是在一定程度上简化了这个过程， Hive也为平台提供了 SQL 的兼容。一些 Hadoop 工具也可以无需编程直接运行 MapReduce 任务。Xplenty 就是一个基于 Hadoop 的数据整合服务，而且也不需要进行任何编程和部署。

尽管 Hive 提供了命令行接口，但 MapReduce 并没有交互式模式。诸如 Impala，Presto 和 Tez 等项目都在尝试希望为 Hadoop 提供全交互式查询模式。

安装与维护方面， Spark 并不绑定在 Hadoop 上，虽然 在 Hortonworks（HDP 2.2 版） 和 Cloudera（CDH 5 版） 的产品中 Spark 和 Hadoop MapReduce 都包含在其分布式系统中。（注： Cloudera， Hortonworks 及 MapR 是 Hadoop 领域三大知名的初创公司，致力于打造更好的 Hadoop 企业版应用）。

小结：Spark 更易于编程，同时也包含交互式模式；Hadoop MapReduce 不易编程但是现有的很多工具使其更易于使用。

### 成本

Spark 和 Hadoop MapReduce 都是开源的，但是机器和人工的花费仍是不可避免的。

Spark 集群的内存至少要和需要处理的数据块一样大，因为只有数据块和内存大小合适才能发挥出其最优的性能。所以如果真的需要处理非常大的数据，Hadoop 绝对是合适之选，毕竟硬盘的费用要远远低于内存的费用。

考虑到 Spark 的性能标准，在执行相同的任务的时候，需要的硬件更少而运行速度却更快，因此应该是更合算的，尤其是在云端的时候，此时只需要即用即付。

在技术人员方面，即使 Hadoop 从 2005 年就开始普及，但是 MapReduce 方面的专家仍然存在着短缺。而对于从 2010 年才开始普及的 Spark ，这又意味着什么呢？ 或许投身 Spark 学习的人正在快速增加，但是相比于 Hadoop MapReduce 仍然存在着更大的技术人才的缺口。

进一步讲，现存了大量的 Hadoop 即服务的资料和基于 Hadoop 的服务（比如我们 Xplenty 的数据整合服务），这些都降低对技术人员能力和底层硬件知识的要求。相比之下，几乎没有现有可选的 Spark 服务，仅有的那些也是新产品。

小结：根据基准要求， Spark 更加合算， 尽管人工成本会很高。依靠着更多熟练的技术人员和 Hadoop 即服务的供给， Hadoop MapReduce 可能更便宜。

### 兼容性

Spark 既可以单独运行，也可以在 Hadoop YARN 上，或者在预置 Mesos 上以及云端。它支持实现 Hadoop 输入范式的数据源，所以可以整合所有 Hadoop 支持的数据源和文件格式。 根据 Spark 官方教程， 它还可以通过 JDBC 和 ODBC 同 BI（商业智能） 工具一起运行。 Hive 和 Pig 也在逐步实现这样的功能。

小结： Spark 和 Hadoop MapReduce 具有相同的数据类型和数据源的兼容性。

### 数据处理

除了平常的数据处理，Spark 可以做的远不止这点：它还可以处理图和利用现有的机器学习库。高性能也使得 Spark 在实时处理上的表现和批处理上的表现一样好。这也催生了一个更好的机遇，那就是用一个平台解决所有问题而不是只能根据任务选取不同的平台，毕竟所有的平台都需要学习和维护。

Hadoop MapReduce 在批处理上表现卓越。如果需要进行实时处理，可以利用另外的平台比如 Storm 或者 Impala，而图处理则可以用 Giraph。MapReduce 过去是用 Mahout 做机器学习的，但其负责人已经将其抛弃转而支持 Spark 和 h2o（机器学习引擎）。

小结：Spark 是数据处理的瑞士军刀；Hadoop MapReduce 是批处理的突击刀。

### 容错

和 MapReduce 一样， Spark 会重试每个任务并进行预测执行。然而，MapReduce 是依赖于硬盘驱动器的，所以如果一项处理中途失败，它可以从失败处继续执行，而 Spark 则必须从头开始执行，所以 MapReduce 这样节省了时间。

小结：Spark 和 Hadoop MapReduce 都有着较好的容错能力，但是 Hadoop MapReduce 要稍微更好一点。

### 安全性

在安全性上， 此时的 Spark 还略显不足。 授权验证由共享秘钥机制支持，网络用户接口则通过 servlet 过滤器和事件日志保护。Spark 可以运行在 YARN 上并配合使用 HDFS， 这也就意味着它同时还拥有 Kerberos 认证授权验证，HDFS 文件许可机制和节点间的加密机制。

Hadoop MapReduce 拥有所有 Hadoop 支持的安全机制，同时也整合了其它基于 Hadoop 的安全项目， 比如 Knox 网关和 Sentry。志在解决 Hadoop 安全的 Rhino 项目也只是在添加 Sentry 支持时添加了 Spark 支持。否则 Spark 开发者们只能自己去提升其安全性了。

小结： Spark 的安全机制仍处在发展期。 Hadoop MapReduce 拥有更多安全控制机制和项目。

### 总结

Spark 是大数据领域冉冉升起的新星，但是 Hadoop MapReduce 仍有着较广的应用领域。

在内存中进行数据处理使得 Spark 具有较好的性能表现，也比较高效合算。它兼容所有 Hadoop 的数据源和文件格式， 支持多种语言的简单易用的 API 也使人们更快速的可以上手。 Spark 甚至实现了图处理和机器学习工具。

Hadoop MapReduce 是一个更加成熟的平台，为进行批处理而生。当遇到确实非常大的数据以至于无法完全读入内存，又或是依靠着大量对该平台有经验的技术人员，它可能会比 Spark 更加合算。 而且围绕 Hadoop MapReduce 的衍生系统正在依靠着更多的支撑项目、工具和云服务而更加壮大。

但是，即使看上去 Spark 像是最终的赢家，问题在于我们永远不会单独使用它—我们需要 HDFS 存储数据，或许还会需要用到 HBase，Hive，Pig，Impala 或其他 Hadoop 项目。这意味着在处理非常大的数据的时候，Spark 仍然需要同 Hadoop 和 MapReduce 共同运行。

而且，对于 Fudan MSE Pins Board 项目来说，在实际生产环境中将面临大量多媒体数据（照片，视频）的处理，若使用 Spark 将不得不面临海量的内存购买，成本将十分可观，因此，在这样的情况下，Hadoop 的 MapReduce 将是更为合理的选择。

## 总结与展望

本次团队课程设计的课题是《基于Html5以及MongoDB的视频、照片墙研究》，完成了系统的需求分析、设计、编码实现，从功能和性能两方面对系统进行测试。

本系统着重研究了无需 App 情况下利用纯HTML5 以及 MongoDB 跨平台对多媒体数据的处理，具有一定的实用性和通用性。

由于软件开发是所有本团队成员的本职工作，因此本课题的完成紧密结合实际，经过研究尝试以及生产环境的检验，证明这样的视频、照片墙是能够适应复杂的环境的。

但同时本系统还有诸多地方有待完善，其界面布局的和理性、程序的稳定性和执行效率也有一定幅度的可提升空间。

作为三名专门从事软件开发行业工作的学生，通过本次团队课程设计，完成了对无App移动开发方向的一次深入探索。不仅仅是对我们自己在首个学期，阶段学习的成果的肯定，也锻炼了我们自己的自学和研究能力，同时还开阔了眼界，了解到自身尚存在诸多不足之处，需要在以后的工作和学习生活中进一步改进和提高。

## 参考文献

* Google 开发者社区
* 开源天空
* TopGeek 社区
* Python 开发者
* 百度文库
* 51CTO