吉视传媒手机营业厅技术方案

目录

[1. 需求分析 4](#_Toc416305574)

[1.1. 注册登录 5](#_Toc416305575)

[1.2. 会员中心 5](#_Toc416305576)

[1.3. 在线缴费 6](#_Toc416305577)

[1.4. 产品订购 7](#_Toc416305578)

[1.5. 业务办理 8](#_Toc416305579)

[1.6. 特惠特价 8](#_Toc416305580)

[1.7. 积分商城 9](#_Toc416305581)

[1.8. 线下活动 9](#_Toc416305582)

[1.9. 我的订单 10](#_Toc416305583)

[1.10. 故障报修 10](#_Toc416305584)

[2. 系统设计 11](#_Toc416305585)

[2.1. 系统整体设计 11](#_Toc416305586)

[2.1.1. 数据库层 11](#_Toc416305587)

[2.1.2. 数据库适配层 12](#_Toc416305588)

[2.1.3. 业务应用层 12](#_Toc416305589)

[2.1.4. 代理层 12](#_Toc416305590)

[2.1.5. 展示层 12](#_Toc416305591)

[2.2. 业务模块设计 13](#_Toc416305592)

[2.2.1. 用户中心 13](#_Toc416305593)

[2.2.2. 运营中心 13](#_Toc416305594)

[2.2.3. 支付中心 13](#_Toc416305595)

[2.2.4. VAS系统 14](#_Toc416305596)

[2.2.5. CMS系统 14](#_Toc416305597)

[2.2.6. 统计系统 15](#_Toc416305598)

[2.3. 硬件架构设计 15](#_Toc416305599)

[2.3.1. 各系统模块硬件架构设计 15](#_Toc416305600)

[2.3.2. 硬件清单汇总 21](#_Toc416305601)

[3. 特殊方案设计 22](#_Toc416305602)

[3.1. 数据安全 22](#_Toc416305603)

[3.1.1. 数字签名加密 22](#_Toc416305604)

[3.1.2. 接口协议加密 22](#_Toc416305605)

[3.1.3. 会话数控制 22](#_Toc416305606)

[3.1.4. 流量控制 23](#_Toc416305607)

[3.1.5. 业务数据安全 23](#_Toc416305608)

[3.2. 高并发解决方案 23](#_Toc416305609)

[3.2.1. 只读高并发 24](#_Toc416305610)

[3.2.2. 读写高并发 24](#_Toc416305611)

[3.2.3. 业务优化 25](#_Toc416305612)

[3.3. 离线数据分析 26](#_Toc416305613)

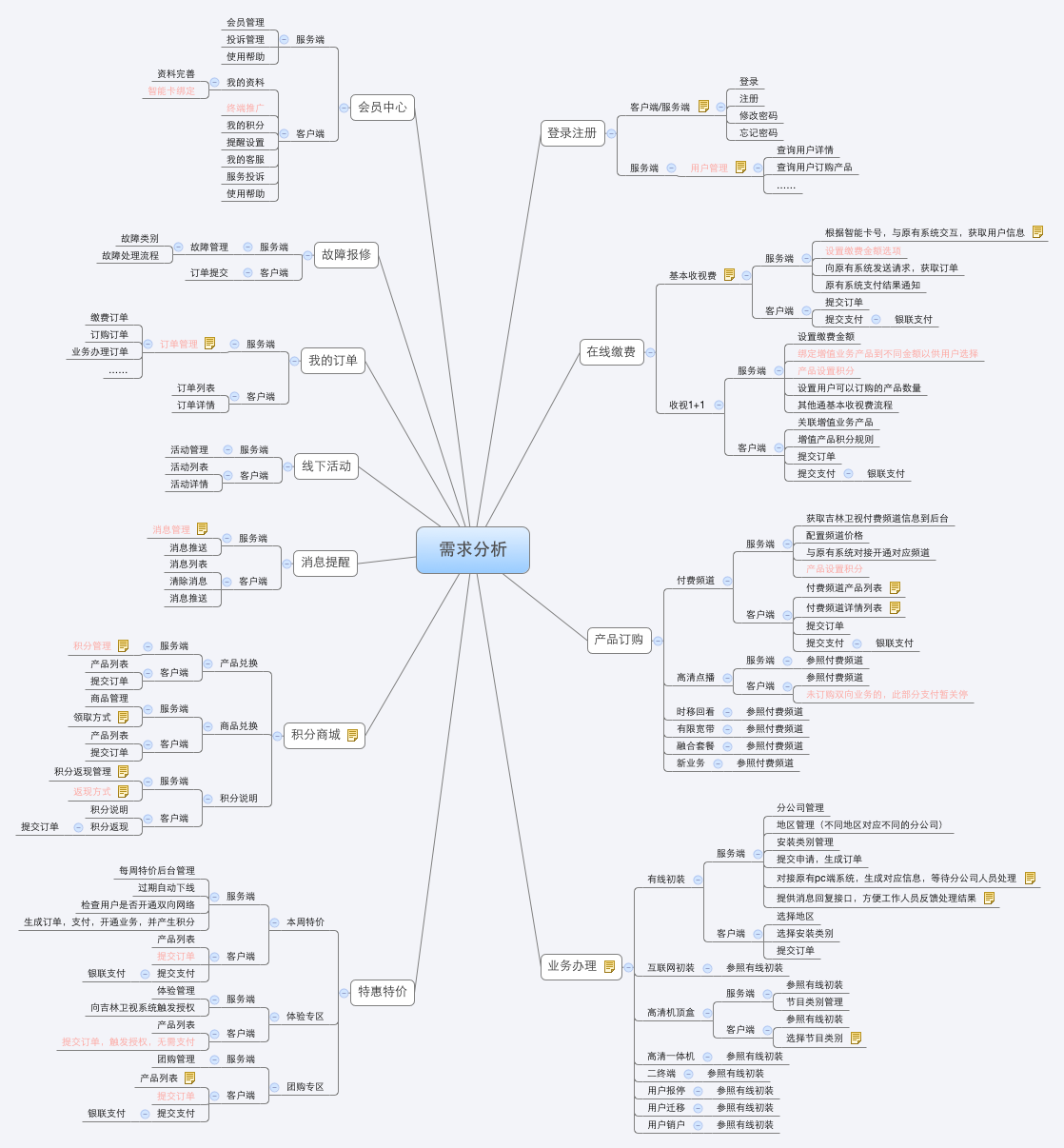
[3.3.1. 原有业务统计 26](#_Toc416305614)

[3.3.2. 自身业务统计 26](#_Toc416305615)

[4. 附件 26](#_Toc416305616)

1. 需求分析

本节将根据客户提供的《产品功能文档》，分析系统业务功能需求，并简要设计对应的系统功能模块，为系统设计提供设计依据。



上图是基于《产品功能文档》所做的需求分析，内部包括承载了各个功能模块的业务流程，以及用以支撑业务流程的后台业务模块。下面将对各个业务功能模块，逐一进行分析介绍：

* 1. 注册登录

该功能为系统基本功能，以下是详细业务流程分析：

|  |  |  |  |
| --- | --- | --- | --- |
|  | 服务端 | 客户端 | 备注 |
| 登录 | 根据客户端上传的username和password，进行查库校验，校验成功则生成加密后的access token（令牌，用于标记用户的登陆状态，用户每次授权请求都需要该令牌），通过消息返回给客户端，如果失败，则返回错误消息 | 调用API接口，系统验证成功后，返回access token，后续的操作都需要带上该令牌（置于请求头） |  |
| 注册 | 1. 校验用户信息； 2. 注册用户信息至数据库，并调用母平台注册接口； |  |  |
| 修改密码 | 1. 校验原密码； 2. 校验失败，则直接返回错误消息；校验成功，则修改数据库，更新用户密码，同时调用母平台修改密码接口 |  |  |
| 忘记密码 | 1. 验证消息接口，根据用户名查询出其手机号 / 邮箱，生成密码重置参数链接，调用短信接口 / 邮箱服务器，发送重置链接给用户； 2. 重置密码接口，用户通过验证链接请求重置页面，系统校验参数合法性，如果参数合法，则返回重置页面，并提供密码重置功能；否则返回错误消息页面。 | 1. 选择找回密码方式（手机 / 邮箱）； 2. 输入用户名，并发送重置密码请求； 3. 获取密码重置链接（手机 / 邮箱），点击链接进入密码重置页面，输入新密码，完成密码重置操作。 |  |
|  |  |  |  |

* 1. 会员中心

该模块以用户为中心，提供与用户设置相关的所有服务，如个人信息、消息管理、积分情况以及系统使用帮助等。

|  |  |  |  |
| --- | --- | --- | --- |
|  | 服务端 | 客户端 | 备注 |
| 个人信息 | 1. 后台系统模块：用户信息管理； 2. 前台服务接口：个人信息更新接口； | 1. 个人资料完善； 2. 智能卡绑定； |  |
| 积分管理 | 1. 后台系统模块：积分规则管理； 2. 前台服务接口：积分查询接口 | 可以查询本人的积分情况 |  |
| 投诉反馈 | 1. 后台系统模块：投诉信息管理； 2. 前台服务接口：投诉反馈接口、投诉进度查询接口； | 可以投诉服务，并查询投诉处理进度 |  |
| 推送消息 | 1. 后台系统模块：推送消息模块、已发布消息管理； 2. 前台服务接口：收集客户端数据接口，消息推送接口； | 客户端提供推送消息设置功能 | 1. 为降低研发和维护成本，使用第三方推送平台（APNS / 个推）推送消息; 2. 业务端推送支持按位置、标签和用户等级等策略进行推送； |
|  |  |  |  |

* 1. 在线缴费

在线缴费暂时分为“基本收视费”和“增值业务”两种类型，为挖掘用户消费需求，拓展增值业务，建议设立类似于余额宝的用户个人账户管理中心，不用每次支付都需要通过银联支付平台，简化用户的缴费流程，提高用户的消费体验，同时也为后续的积分转换、增值业务的开展奠定基础。由于“在线缴费”属于自动化流程，整个业务只需要客户与系统交互，不需要后台管理员配合，同时由于本系统与母平台直接的对接关系不太明确，所以本处只设计了简单的业务流程。

|  |  |  |  |
| --- | --- | --- | --- |
|  | 服务端 | 客户端 | 备注 |
| 基本收视费 | 1. 接收客户端提交的订单信息，根据智能卡号，与原有系统交互，获取用户信息； 2. 保存订单信息至数据库； 3. 请求母平台支付接口，提交支付订单； 4. 用户转跳银联支付系统，完成支付操作； 5. 银联平台完成支付业务处理后，后台通知母平台支付回执接口，母平台更新订单状态，完成支付业务闭环； 6. 客户端请求订单状态信息，系统根据订单号查询母平台接口，获取订单状态，整合数据后，返回消息给客户端； | 1. 填写订单（智能卡号、缴费事项、缴费金额等）； 2. 提交订单； 3. 查看订单信息； |  |
| 收视1+1 | 1. 提供增值业务和积分规则信息查询接口； 2. 接收到客户端提交的订单后，根据增值业务和积分规则，更新响应的业务数据； 3. 重复以上“基本收视费”的业务流程； | 1. 查看增值业务产品介绍、积分规则等信息； 2. 选择购买，填写相关订单信息； 3. 提交订单； 4. 查看订单信息； | 该项业务的套餐规则、积分规则待定 |
|  |  |  |  |

* 1. 产品订购

该模块为增值业务的核心模块，为“在线缴费”的下游服务模块，为“在线缴费”提供实质性的商品服务。

|  |  |  |  |
| --- | --- | --- | --- |
|  | 服务端 | 客户端 | 备注 |
| 付费频道 | 1. 后台业务：a、定时同步吉林卫视付费频道信息至系统商品库；b、系统与母平台付费频道开通接口联调，保证用户完成支付操作后，能及时开通付费频道； 2. 后台系统模块：付费频道的价格管理，套餐管理，积分规则管理； 3. 前台服务接口：付费频道列表接口、频道详情信息接口、“产品订购”模块和“在线缴费”模块的对接接口（引导用户完成支付操作）； | 1. 获取付费频道产品列表（产品汇总数量、浏览次数、订购用户数等信息）； 2. 查询某个付费频道的产品简介、价格信息、缴费方式等； 3. 选择产品，并提交订单； 4. 进入“在线缴费”模块，根据系统提示，完成支付操作； |  |
| 高清点播 | 参考“付费频道” | 参考“付费频道” | 未订购双向业务的，此部分支付暂时关闭 |
| 时移回看 | 参考“付费频道” | 参考“付费频道” |  |
| 有限带宽 | 参考“付费频道” | 参考“付费频道” |  |
| 融合套餐 | 参考“付费频道” | 参考“付费频道” |  |
| 新业务 | 参考“付费频道” | 参考“付费频道” |  |
|  |  |  |  |

* 1. 业务办理

该模块暂不执行线上支付，属于服务指南，本系统只提供简单的业务

|  |  |  |  |
| --- | --- | --- | --- |
|  | 服务端 | 客户端 | 备注 |
| 有线初装 | 1. 后台系统模块：分公司管理、地区管理（不同地区对应不同的分公司）、安装类别管理和有线初装申请管理； 2. 前台服务接口：有线初装申请接口、申请缴费对接“在线缴费”接口、初装处理结果反馈接口； | 1. 选择地区； 2. 选择安装类别； 3. 提交初申请； |  |
| 互联网初装 | 参考“有线初装” | 参考“有线初装” |  |
| 高清机顶盒 | 1. 后台系统模块：节目类别管理； 2. 前台服务接口：参考“有线初装”； | 参考“有线初装” |  |
| 高清一体机 | 参考“有线初装” | 参考“有线初装” |  |
| 二终端 | 参考“有线初装” | 参考“有线初装” |  |
| 用户报停 | 参考“有线初装” | 参考“有线初装” |  |
| 用户迁移 | 参考“有线初装” | 参考“有线初装” |  |
| 用户销毁 | 参考“有线初装” | 参考“有线初装” |  |
|  |  |  |  |

* 1. 特惠特价

|  |  |  |  |
| --- | --- | --- | --- |
|  | 服务端 | 客户端 | 备注 |
| 本周特价 | 1. 后台业务：定时任务，定期检查过期特价产品，过期则自动下架，并更相关数据； 2. 后台系统模块：每周特价管理 3. 前台服务接口：产品列表接口、特价产品购买接口（对接“在线缴费”模块）； | 1. 获取产品列表； 2. 选择产品，提交订单，进入“在线缴费”流程； |  |
| 体验专区 | 1. 后台系统模块：体验服务管理、服务授权管理（体验服务的开通需要向吉林卫视系统触发授权）； 2. 前台服务器接口：体验服务列表接口、体验开通接口（只需要触发授权，无需支付）； | 1. 获取体验服务列表； 2. 选择需要体验的服务，像开通接口提交订单； | 体验服务开通模式需要具体讨论才能确定 |
| 团购专区 | 1. 后台系统模块：团购管理； 2. 前台服务接口：团购列表接口、团购购买接口（对接“在线缴费”模块）； | 1. 获取团购列表； 2. 选择团购服务，向“团购购买接口”提交订单； |  |
|  |  |  |  |

* 1. 积分商城

积分系统在电子商务中一直是一个比较复杂的系统，因为其业务触角延伸得非常广泛，其数据采集点可能会贯穿于整个电子商务系统中，由于子系统业务具有不可控性，所以对于积分系统的设计基本上是采用，数据采集、消费基于接口调用，转换规则统一管理的策略。

|  |  |  |  |
| --- | --- | --- | --- |
|  | 服务端 | 客户端 | 备注 |
| 产品兑换 | 1. 后台系统模块：积分产品兑换规则管理（定义积分与产品的兑换关系）； 2. 前台服务接口：可兑换产品列表接口、产品兑换接口； | 1. 获取可兑换产品列表； 2. 选择可兑换的产品，向兑换接口提交订单； | 此处的产品可为吉林卫视的付费频道、吉视传媒的特色服务等 |
| 商品兑换 | 1. 后台系统模块：积分商品兑换规则管理； 2. 前台服务接口：可兑换商品列表接口、产品兑换接口； | 1. 获取可兑换商品列表； 2. 选择商品，向商品兑换接口提交订单； | 此处的商品为实物商品 |
| 积分说明 | 1. 后台系统模块：CMS系统，内容参考产品兑换和商品兑换； 2. 前台服务页面：积分规则页面、积分兑换、返现页面； | 客户端使用webview显示服务端返回的页面数据 |  |
|  |  |  |  |

* 1. 线下活动

线下活动属于不经常更新的业务，主要还是使用CMS（Content Manage System）系统，生成页面，客户端直接使用webview加载显示即可，好处是使用B / S架构，更改页面时，不需要升级客户端版本。

|  |  |  |  |
| --- | --- | --- | --- |
|  | 服务端 | 客户端 | 备注 |
| 发布线下活动 | 1. 后台系统模块：CMS系统，可以编辑、发布线下活动内容； 2. 前台服务页面：线下活动页面 | 客户端使用webview显示服务端返回的页面数据 |  |
|  |  |  |  |

* 1. 我的订单

该模块是站在用户角度，对前面的各个与缴费相关的模块的数据汇总，通过该模块，用户可以轻松地管理前面各个业务的订购业务。客户端可以对所有的订单做分类显示和管理，以下是对各个。

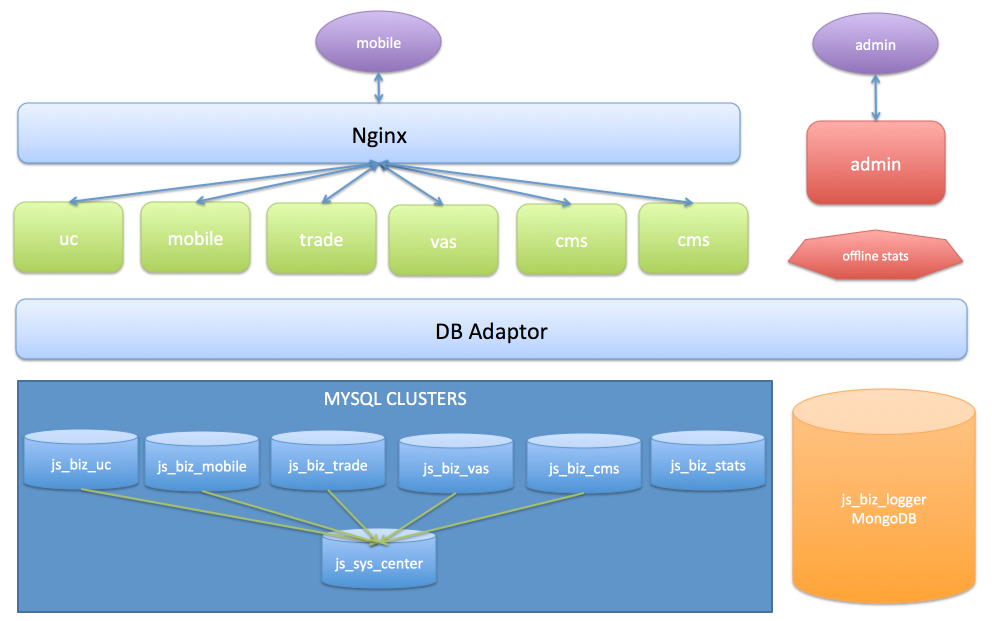
|  |  |  |  |
| --- | --- | --- | --- |
|  | 服务端 | 客户端 | 备注 |
| 缴费订单 |  |  |  |
| 订购订单 |  |  |  |
| 业务办理订单 |  |  |  |
|  |  |  |  |
|  |  |  |  |

* 1. 故障报修

|  |  |  |  |
| --- | --- | --- | --- |
|  | 服务端 | 客户端 | 备注 |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

1. 系统设计
   1. 系统整体设计

下图为应用系统的架构设计图：



由上图所示，从垂直方向，系统一共分为以下4层：

* + 1. 数据库层

出于不同的系统设计目的，我们需要根据具体的业务情况使用不同类型的数据库，对于业务安全性要求高，数据之间的业务逻辑复杂，需要做很多连表查询的业务，我们可以考虑使用关系型数据库；而对于业务逻辑不是很复杂，但要求高性能，高吞吐量的业务，我们则应该考虑采用非关系型数据库，如MongoDB、HBase等。结合本系统的项目需求，我们将数据库层的架构设计如下：

* MYSQL集群：考虑到像“用户中心”、“运营中心”以及“支付中心”这些模块需要承载大量的业务流程，业务逻辑较为复杂，同时对于吞吐量的要求并不是很高，所以，使用关系型数据库较为合理，当业务量上涨，数据库压力变大时，可以考虑将同一个库的表拆分到不用的数据库中，通过横向扩展数据库集群的方式提升系统的容量。同时，由图可以看出，关系型数据库分为业务数据库和系统数据库，业务数据库（如js\_biz\_uc, js\_biz\_mobile,js\_biz\_trade等）负责处理专业性比较强的模块业务，比如存储用户信息，存储订单数据，存储增值服务数据等，而系统数据库js\_sys\_center则负责存储一些公共的元数据，比如字典表、地区数据、后台管理员等，各个业务数据库都依赖于系统数据库。
* MongoDB：对于系统日志分析，运营数据统计报表，用户行为数据分析等需要大规模的存储，和大量的计算的业务，可以考虑使用非关系型数据库，同时结合分布式计算框架，通过离线数据分析技术，定时将数据从数据库中取出，然后进行数据分析，并将分析结果存入关系型数据库（如js\_biz\_stats）中。当管理员需要查看统计数据和运营报表时，可以在统计好的数据库的基础上，做简单的展示，或者进行二次即时统计，极大地加快了系统数据的分析、处理能力。
  + 1. 数据库适配层

为了更好地管理代码，提高系统的复用性、扩展性，以及可维护性，我们在“业务应用层”和“数据库层”之间增加了一个“数据库适配层”。通过该层的适配，业务应用层可以很轻松地实现跨库多数据源查询操作，避免了各个业务模块之间接口调用的麻烦，提高了产品的研发效率。

* + 1. 业务应用层

业务应用层是系统的核心，承载了几乎所有的系统业务逻辑，是系统设计和系统研发中耗时最多，维护最困难的一层。

为了保证系统的可维护性、可拓展性，我们在设计该层系统时，必须遵循高内聚，低耦合的系统设计理念，将系统按照业务功能切分为一个个相互独立的小模块，在研发时采用测试驱动的方式进行开发，力求测试样例代码能够覆盖小模块的各个代码边界，保证系统的严谨性，和稳定性。

对于业务应用层的详细结构设计，请参考章节：[业务模块设计](#业务模块设计)。

* + 1. 代理层

为了提高前端服务（业务应用层）的横向拓展能力，一般在生产环境中，都会在业务应用层的前面增加一个代理层，用于负载均衡，将请求分发给后面的业务服务器集群进行处理。同时，对于拥有大量页面资源的站点，可以考虑在代理层使用页面静态化技术，将大量页面静态化在代理服务器上，以提高系统的响应速度。

* + 1. 展示层

结合我们的项目需求，该层实际上为手机客户端软件层，通过使用android / ios客户端，优化数据传输结构，降低数据传输量，同时优化的系统的使用体验，是整个系统中不可或缺的一环。

* 1. 业务模块设计

综合前面的需求分析，可以将系统按照业务功能切分为以下几个业务模块。

* + 1. 用户中心

用户中心（UC）在整个系统中处于基础核心的地位，其他的业务流程，例如在线缴费、产品订购、业务办理等，都离不开用户中心，都需要与用户中心的数据进行交互，

根《产品功能文档》分析，用户中心功能较为简单，其前台服务压力也不会太大，用户中心主要提供注册、登陆、账户绑定、以及权限验证等，如果后续需要进行系统整合，则用户中心应提供SSO通行证功能。

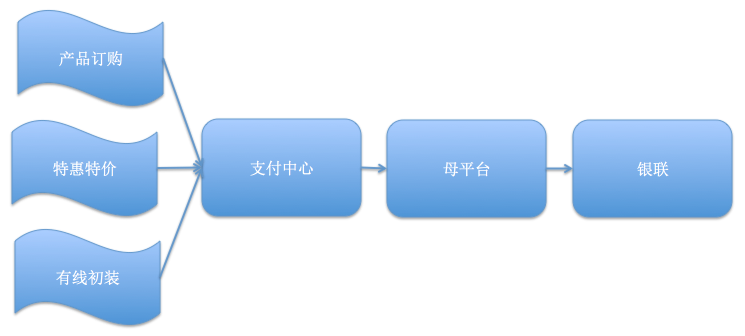
* + 1. 运营中心

该模块（mobile）主要为手机客户端提供接口API支持，内容涵盖了软件下载、版本升级、业务办理、投诉建议、故障报修等基础业务，该模块前段服务的客户端访问量比较大，不过其业务数据大多更新操作较少，在生成环境中，可以考虑在数据库和业务层之间增加一个数据缓存层，以减轻数据库的压力，提高API的访问速度。

* + 1. 支付中心

支付中心（trade）也是本系统的一个核心模块，业务涉及在线缴费、产品订购、特惠特价、积分商城，以及我的订单等业务流程。由于对吉视传媒现有系统的架构不熟悉，我们根据以往的设计经验，设计了以下2种业务流程：

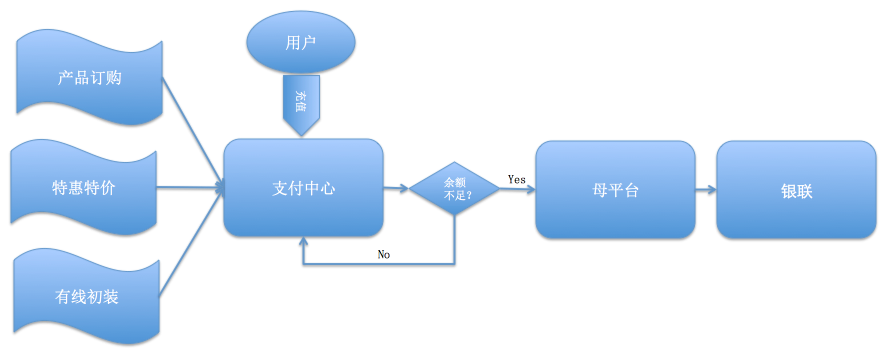
* 即时支付模式：



如上图所示，在该模式下，每一个支付流程，如产品订购，都需要调用支付中心的支付接口，获取数据验证后，由母平台将用户转接至银联支付平台，当用户在银联平台完成支付操作后，银联通过后台异步消息通知母平台更新订单业务状态，母平台再以同样的方式通知支付中心，更新支付中心的订单状态，然后通知对应的业务模块（如VAS模块）更新业务状态（如开通某个付费频道），完成整个支付流程。

这种模式的优点是支付中心的结构设计简单，与母系统的对接也比较容易，缺点是用户每次支付行为都需要走：支付中心 🡪 母平台 🡪 银联，这整个支付链条，支付接口访问频率高，操作繁琐，用户体验非常不好。

* 账户模式：



账户模式是对即时支付模式的优化，即在原支付模式的基础上，增加了一个账户的概念，即用户可以通过特定的充值入口向支付中心充钱，当用户的账户里有足够的金币时，用户购买任何一项服务和商品，都不再需要转接母平台和银联平台。

* + 1. VAS系统

VAS（value add service）是增值服务模块，《产品功能文档》里的“产品订购”、“特惠特价”，以及“积分商城”等业务都在该系统的业务范畴之内，该系统内的业务属于系统的外围业务，虽然地位不高，但是业务逻辑确实最复杂的，同时也可能会高并发的可能性，如当搞“特惠特价”促销活动时，由于可能会吸引大量的用户抢购，所以极有可能会引发高并发访问，要解决该问题，可以参考章节：[高并发解决方案专题](#高并发解决方案)。

* + 1. CMS系统

CMS（content manage system）是内容管理系统，该系统设计的目的是应对系统信息发布、网站内容建设，该系统涵盖了特惠特价、线下活动等业务流程，该系统建设得好坏将直接影响VAS业务的开展。

* + 1. 统计系统

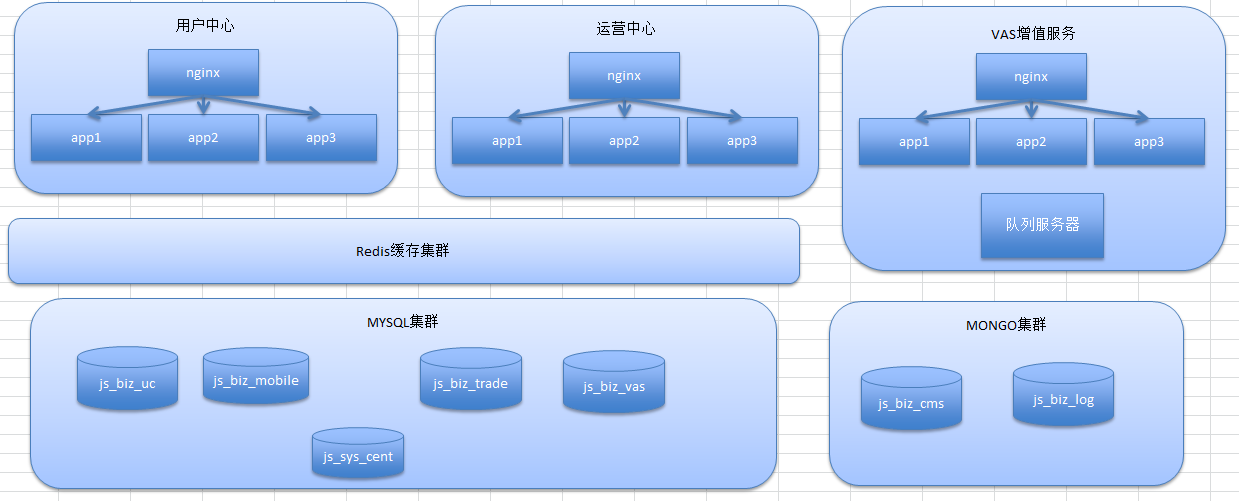
对于统计模块，由于很多统计结果对实时性要求不高，所以为了提高系统的运行效率，可以将数据采集点采集到的数据存储到非关系型数据库（如js\_biz\_logger）中，定时分析数据库里的数据，并将分析结果存储到关系型数据库中，以节省大量无谓的即时统计操作，具体请参考章节：[离线数据分析](#离线数据分析)。

* 1. 硬件架构设计

手机营业厅系统，在硬件平台建设要考虑高效性、安全性等关键因素；同时利用虚拟化等技术建设成拥有横向扩展能力的平台，本章节主要从硬件部署的角度分析应用系统的架构设计。

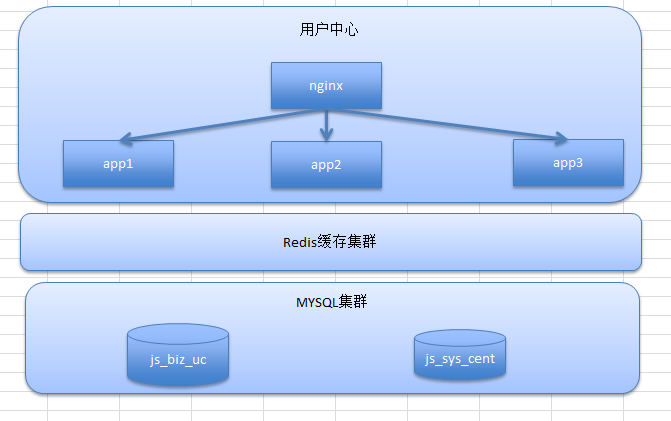
* + 1. 各系统模块硬件架构设计

下图为系统硬件架构总拓扑图



* + - 1. 用户中心

硬件拓扑图

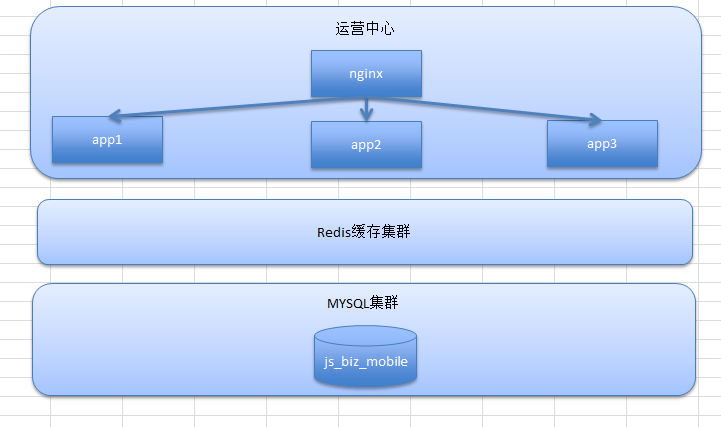


服务器清单

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  | 应用说明 | 硬件类型 | Core数 | 内存（GB） | 数量 | 部署方式 |
| 代理服务器 | Nginx，负责代理、分发请求 | 小型机 | 16 | 32 | 2 | 集群 |
| 应用服务器 | 负责处理业务流程，及其动态数据 | pc机 | 4 | 8 | 2 | 集群 |
| 缓存服务器 | 用于缓存更新频率较少的数据，以降低数据库的访问压力 | pc机 | 4 | 8 | 2 | 与应用服务器使用同一主机 |
| 数据库服务器 | 负责存储业务数据 | 小型机 | 8 | 16 | 2 | Master-Slave |
|  |  |  |  |  |  |  |

* + - 1. 运营中心

硬件拓扑图

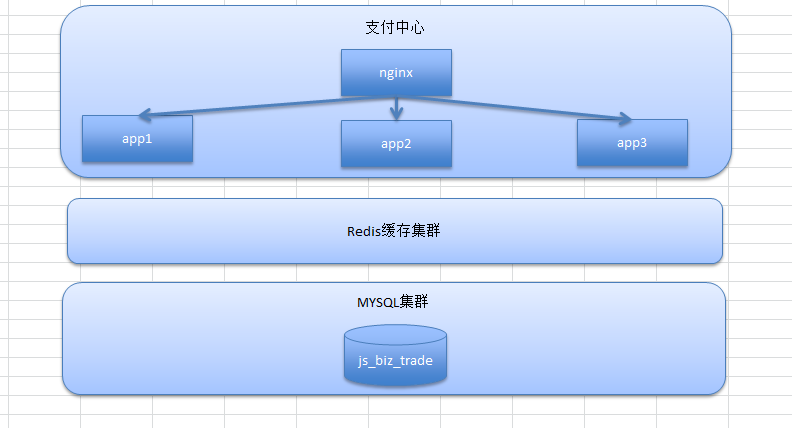


服务器清单

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  | 应用说明 | 硬件类型 | Core数 | 内存（GB） | 数量 | 部署方式 |
| 代理服务器 | Nginx，负责代理、分发请求 | 小型机 | 16 | 32 | 2 | 集群 |
| 应用服务器 | 负责处理业务流程，及其动态数据 | pc机 | 4 | 8 | 2 | 集群 |
| 缓存服务器 | 用于缓存更新频率较少的数据，以降低数据库的访问压力 | pc机 | 4 | 8 | 2 | 与应用服务器使用同一主机 |
| 数据库服务器 | 负责存储业务数据 | 小型机 | 8 | 16 | 2 | Master-Slave |
|  |  |  |  |  |  |  |

* + - 1. 支付中心

硬件拓扑图



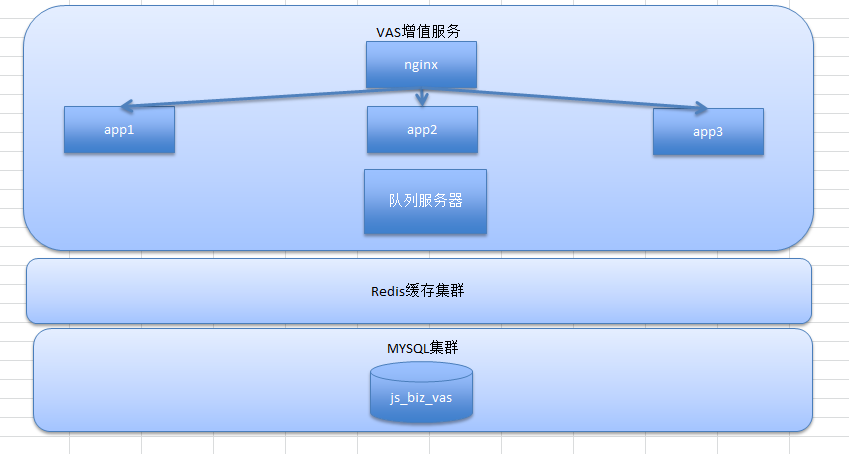
服务器清单

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  | 应用说明 | 硬件类型 | Core数 | 内存（GB） | 数量 | 部署方式 |
| 代理服务器 | Nginx，负责代理、分发请求 | 小型机 | 16 | 32 | 2 | 集群 |
| 应用服务器 | 负责处理业务流程，及其动态数据 | pc机 | 4 | 8 | 2 | 集群 |
| 缓存服务器 | 用于缓存更新频率较少的数据，以降低数据库的访问压力 | pc机 | 4 | 8 | 2 | 与应用服务器使用同一主机 |
| 数据库服务器 | 负责存储业务数据 | 小型机 | 8 | 16 | 2 | Master-Slave |
|  |  |  |  |  |  |  |

支付系统对于安全性要求比较高，所以在数据库层级，需要对数据进行容灾处理，保证数据的安全。在数据库部署时，业内一般采用主备方案（Master 🡪 Slaver），当主服务器宕机时，备用服务器自动顶替主服务器，提供服务，防止服务中断，避免数据丢失，同时，系统会自动发送系统失效邮件，通知运维人员进行系统修复。

* + - 1. VAS系统（特惠特价）

该业务属于VAS（增值服务）类业务，该业务有可能出现高并发的情况，如特价抢购活动，对于该类业务，可以参考中国铁道部的官方售票网站12306的高并发解决方案，采用请求分流、请求队列处理等策略，从空间、时间上对高并发进行压力疏散，故在进行硬件架构设计时，我们可以在原来典型的集群拓扑架构的基础上，增加一个队列服务器集群，专门处理应用服务器的异步请求，解决高并发压力。



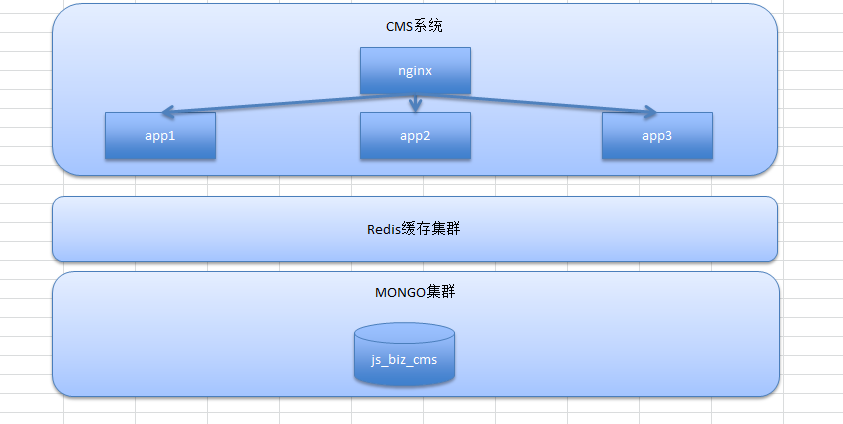
服务器清单

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  | 应用说明 | 硬件类型 | Core数 | 内存（GB） | 数量 | 部署方式 |
| 代理服务器 | Nginx，负责代理、分发请求 | 小型机 | 16 | 32 | 10(100W)  30(300W)  50(500W)  100(1000W) | 集群 |
| 应用服务器 | 负责处理业务流程，及其动态数据 | pc机 | 4 | 8 | 1000(100W)  3000(300W)  5000(500W)  10000(1000W) | 集群 |
| 缓存服务器 | 用于缓存更新频率较少的数据，以降低数据库的访问压力 | pc机 | 4 | 8 |  | 与应用服务器使用同一主机 |
| 数据库服务器 | 负责存储业务数据 | 小型机 | 8 | 16 | 2 | 集群 |
| 队列服务器 | 负责处理读写并发的业务，如抢票、抢单等。 | 小型机 | 16核 | 32g | 2(100W)  4(300W)  6(500W)  10 (1000W) | 集群 |
|  |  |  |  |  |  |  |

考虑到异步服务的性能，队列服务器也设计成分布式，采用一致哈希算法对同步数据进行横向切分，将需要同步的数据分散到队列服务器集群中，解决高并发的数据一致性问题。

* + - 1. CMS系统

硬件拓扑图

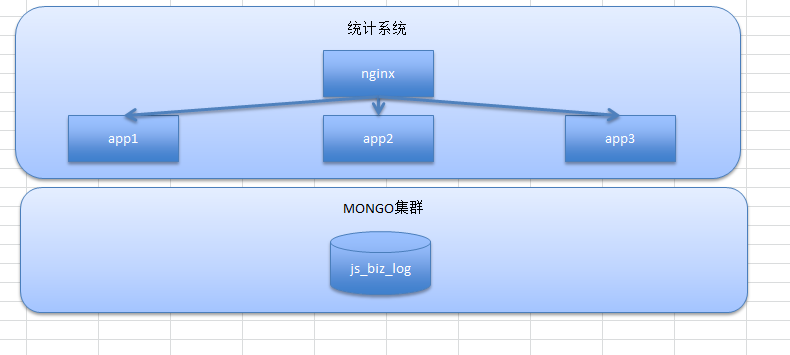


服务器清单

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  | 应用说明 | 硬件类型 | Core数 | 内存（GB） | 数量 | 部署方式 |
| 代理服务器 | Nginx，负责代理、分发请求 | 小型机 | 16 | 32 | 2 | 集群 |
| 应用服务器 | 负责处理业务流程，及其动态数据 | pc机 | 4 | 8 | 2 | 集群 |
| 缓存服务器 | 用于缓存更新频率较少的数据，以降低数据库的访问压力 | pc机 | 4 | 8 | 2 | 与应用服务器使用同一主机 |
| MONGO服务器 | 负责存储发布内容，这些非事务性数据 | 小型机 | 8 | 16 | 2 | 主从 |
|  |  |  |  |  |  |  |

* + - 1. 统计系统

硬件拓扑图



服务器清单

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  | 应用说明 | 硬件类型 | Core数 | 内存（GB） | 数量 | 部署方式 |
| 代理服务器 | Nginx，负责代理、分发请求 | 小型机 | 16 | 32 | 2 | 集群 |
| 应用服务器 | 负责处理业务流程，及其动态数据 | pc机 | 4 | 8 | 2 | 集群 |
| 缓存服务器 | 用于缓存更新频率较少的数据，以降低数据库的访问压力 | pc机 | 4 | 8 | 2 | 与应用服务器使用同一主机 |
| MONGO服务器 | 负责存储发布内容，这些非事务性数据 | 小型机 | 8 | 16 | 2 | 主从 |
|  |  |  |  |  |  |  |

* + 1. 硬件清单汇总

由上面对各个业务的分析，得知系统的主要瓶颈在VAS这一块，而其他的业务系统（用户中心，运营中心，支付中心，CMS系统，统计系统）可以部分共享同一服务器。

得出以下业务清单

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 并发量 | 代理服务器  （16核32G/小型机） | 应用服务器/缓存服务器  （4核8G/pc） | 队列服务器  （16核32G/小型机） | MONGO服务器  （8核16G/小型机） | 数据库服务器  （8核16G/小型机） |
| 100W | 12 | 1008 | 2 | 4 | 4 |
| 300w | 32 | 3008 | 4 | 4 | 4 |
| 500w | 52 | 5008 | 6 | 4 | 4 |
| 1000w | 102 | 10008 | 10 | 4 | 4 |

以上清单为服务器配置清单，具体服务器的采购型号，请自行定义。

1. 特殊方案设计
   1. 数据安全

随着互联网的不断发展，网络安全问题越来越突出，如何防范服务被攻击，防止数据被盗用，有效地保证数据安全，已经成为每个企业需要特别关注的头等大事。本节描述了企业在信息化过程中可能遇到的数据安全问题，以及相应的对策。

* + 1. 数字签名加密

客户端软件与后台之间采用基于TCP/IP的HTTP协议。为保证通讯安全性，传输层可以采用HTTPS通讯，避免路由层级数据泄露

* + 1. 接口协议加密

系统会给客户端软分配一个appkey和appsecret，appkey用于唯一识别手机客户端软，appsecret是针对这个appkey的一个安全码，用于参数签名，不能暴露在用户可接触到的地方，否则会带来安全问题。

* + 1. 会话数控制

session是每个客户端请求必须的参数，是用来标识一个客户端会话的。服务端需要维护一个sessionKey列表，用来保存所有有效的会话。当每次客户端要求建立会话的时候，根据会话列表就可以很容易的控制会话数，当超过会话数的时候就返回登录失败。

* + 1. 流量控制

流量控制主要是控制单个会话在单位时间内的请求频率。例如在一分钟内请求次数不能超过40次。在服务端为每个会话维护一个请求时间队列，当有一个请求到来，首先要检查该队列，删除超过某个时间段的时间例如删除超过1分钟的时间，然后判断队列的大小是否超过某个值例如是否超过40，如果超过就返回服务器忙错误，如果没超过将当前服务器时间放入该队列。请求时间队列如下图所示：

统计一分钟以内的请求数量

删除一分钟以前的数据

请求 时间戳

1428503600744

……

请求时间戳

142850720744

注意：上图中的时间戳是服务端时间，不是客户端时间。

* + 1. 业务数据安全

在一般性的数据处理过程中，简单的应用同步——锁机制，即可处理读写的非原子性操作导致的数据错误。在并发量大的情况下，需要使用特殊的处理方式。详情见高并发解决方案。

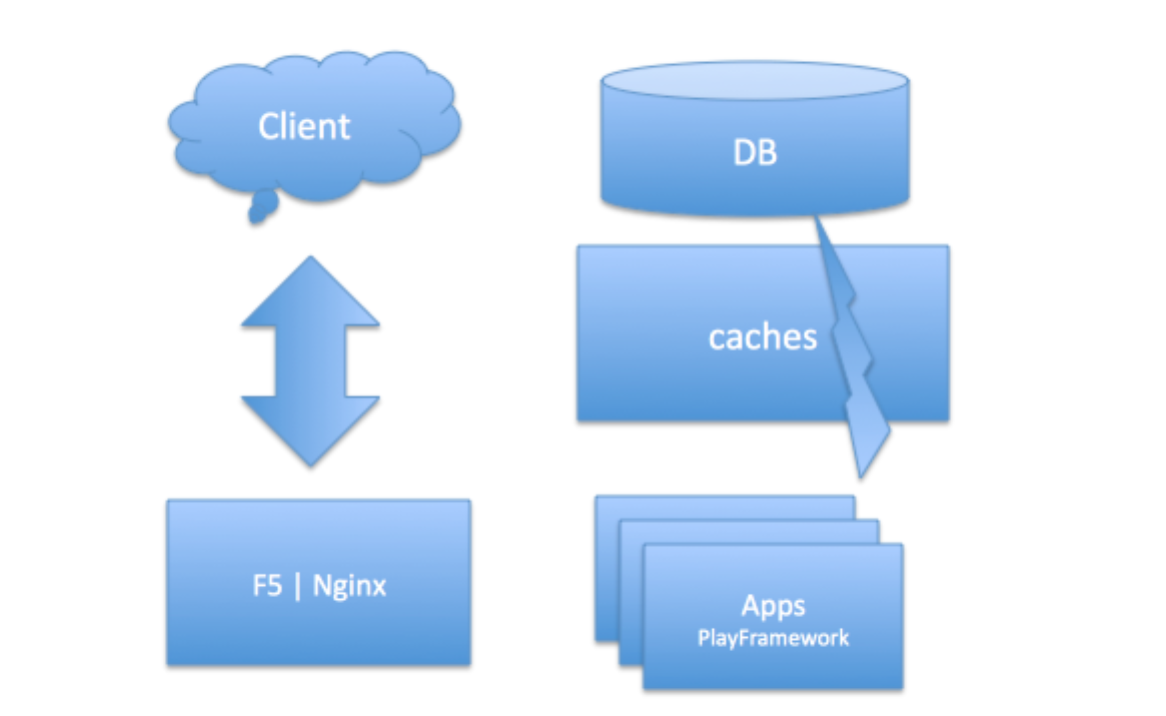
* 1. 高并发解决方案

在实际的业务运营中，可能会遇到部分业务在某些特殊时段会遭遇大规模并发请求的情况，本节将重点阐述针对该种类型的业务，应该如何进行系统设计。

要解决服务器领域的高并发问题，首先要分析涉及高并发的业务是属于只读高并发，还是属于读写高并发，对于不同类型的高并发，可以根据实际情况采用不同的解决方案。

* + 1. 只读高并发

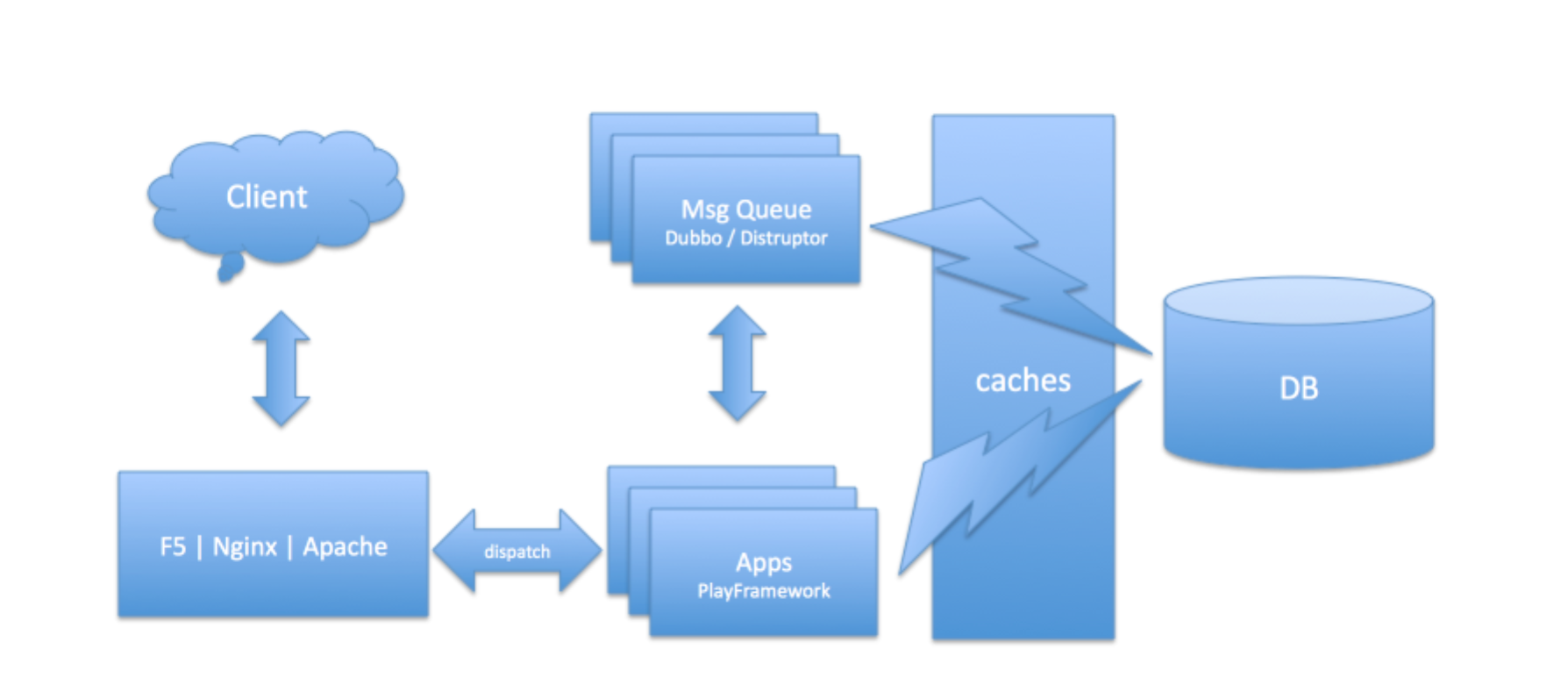
对于只读高并发，由于不涉及数据同步问题，所以可以简单地通过横向拓展应用集群的方式解决服务的高并发问题，具体的架构图如下：



如上图所示，对于只读高并发，我们只需要使用负载均衡器（如果企业资金雄厚可以考虑性能优异的硬件级代理服务器F5，如果资金不足可以使用软代理Nginx）搭建应用服务器集群（Apps），这种架构原则上讲可以通过简单堆叠服务器的方式实现系统性能的横向拓展。同时，对于数据更新少的API接口，可以考虑在接口和DB之间增加缓存Caches层，以降低业务服务器集群对数据库的压力。

* + 1. 读写高并发

由于读写高并发涉及到数据同步（数据一致性）的问题，所以，需要在“只读高并发”架构的基础上增加一个请求消息队列服务器，使用队列堵塞的方式，从应用级别解决读写高并发问题。



如上图所示，增加了请求消息队列服务器集群（Msg Queue）后，对于要求数据一致性（多个请求同时对同一份资源数据进行读写操作）的业务请求，可以将请求通过异步消息的方式，发送给消息服务器，并阻塞等候。消息集群服务器接收到消息后，放入消息队列中，同时不断从消息队列中批量读取数据，调用系统接口处理相应的业务。这样的做法可以在保证数据一致性的前提下，将高并发的请求以队列的形式在时间维度上分摊了系统压力，避免瞬时的高并发造成系统崩溃。

* + 1. 业务优化

前面只是对一般性业务做的高并发方案设计，在实际开发中，可以根据具体的业务做特定的优化处理，优化主要从以下几方面入手：

* 缓存和静态化：对于没有频繁写操作的业务数据，可以考虑在业务层和和DB层之间增加一个数据缓存层，用以提高业务层读取数据的速度。对于B /S 架构的系统，由于业务服务器生成动态页面需要消耗大量的计算性能，所以，可以考虑将频繁访问的动态页面持久化到服务器本地硬盘，再配合使用高性能的静态页面服务器，可以让系统的访问速度提高一个数量级。
* 业务切分：当使用系统的承载量达到每秒数千万级的并发量，使用以上的架构系统已经无法满足需求时，可以针对具体的业务做业务切分，将用户的访问请求分流给多个不同的应用入口，以横向拓展系统的吞吐量。例如，对于特价商品的抢购，可以根据不同的商品设置不同的访问url入口，避免在抢购时刻，出现大量的用户请求同时请求同一个系统入口，造成系统超载崩溃。
* 异步处理：对于某些不需要即时响应的用户请求，可以考虑在用户请求系统时，将用户的请求异步提交给系统任务池，由系统根据资源空闲情况进行业务处理，然后立即返回消息给用户，结束用户访问请求，降低系统的访问压力。
  1. 离线数据分析

系统存在自身的统计和原有系统的统计离线报表的要求。

* + 1. 原有业务统计

定时从原系统抽取业务数据，按照报表的业务，统计对应的结果，直接保存，生成对应的数据，提高访问速度

例如：原来的按日，按月，按年统计，可以添加新的业务表，通过程序，计算指定时间点之前的统计结果，保存到对应表中，这样在查询数据的时候，就不用实时计算。

* + 1. 自身业务统计

统计系统采用异步，批量的方式采集用户数据，

1.客户端记录用户操作，批量发送操作日志到服务器。

2.服务器异步处理客户端操作日志

3.把日志存储到nosql类型的数据库中

4.由统计程序，定时处理数据。

1. 附件

