

学号： 2013220402008

姓名： 尹俊民

专业方向： 大型主机

企业名称： 作业帮教育科技（北京）有限公司

毕设课题名称： 作业帮高并发场景

下的稳定服务端功能实现

企业指导教师： 陆海霞

院内代管教师： 刘玓

**信息与软件工程学院**

**毕业设计（顶岗实习）**

**中期报告**

目 录

[1. 毕业设计的进展情况 1](#_Toc448332878)

[1.1. 课题工作完成情况 1](#_Toc448332879)

[1.2. 知识技能学习情况 1](#_Toc448332880)

[1.3. 职业素养学习培养 1](#_Toc448332881)

[2. 存在问题与解决方案 2](#_Toc448332882)

[2.1. 存在的主要问题 2](#_Toc448332883)

[2.2. 解决方案与可行性研究 2](#_Toc448332884)

[3. 前期任务完成度与后续实施计划 3](#_Toc448332885)

[参考文献 4](#_Toc448332886)

说明:

1.报告要求6000字以上。

2.本模板仅为基本参考，请各位同学根据个人情况进行目录结构扩展。

3.封面和目录单面打印，从正文第1页开始双面打印。

## 1. 毕业设计的进展情况

### 1.1. 课题工作完成情况

近期的数个项目开发中，都已顺利的进行开发并已提测，在黑板报feed的改版过程中，实现了近期的新旧的数据迁移，将原有的腾讯新闻接口请求的内容转移到原有的作业帮自己生成的内容表里，将两份数据合一，优化存储。并将原有的简单读操作从读库到读缓存，优化缓存刷新机制，降低请求时长的抖动。让用户在请求黑板报feed流的过程中更加流畅。内容也较之前更加丰富。同时还增加了弹幕与点赞功能。由于弹幕和点赞功能属于高并发但数据准确性要求不高的接口，所以这些数据不会实时的入库，而是先进入缓存在缓存到达一个值时以批次将数据入库。从而保证了两个接口的效率与响应时长。

关于第三方登入，成功将现有的上亿用户的session、password迁移到新表中以实现新功能的需求。成功实现了客户端与服务端联调，实现了通过oauth2.0协议获取用户信息，从第三方得到用户的open id并以其作为唯一主键作为索引第三方用户的登录注册功能。

具体实现步骤（A）用户访问客户端，后者将前者导向认证服务器。 （B）用户选择是否给予客户端授权。（C）假设用户给予授权，认证服务器将用户导向客户端事先指定的”重定向URI”（redirection URI），同时附上一个授权码。 （D）客户端收到授权码，附上早先的”重定向URI”，向认证服务器申请令牌。这一步是在客户端的后台的服务器上完成的，对用户不可见。 （E）认证服务器核对了授权码和重定向URI，确认无误后，向客户端发送访问令牌（access token）和更新令牌（refresh token）（F）服务端存储用户的openid、头像等用户信息 （G）生成session存储用户登陆信息并将session的key返回客户端进行存储。

在实现第三方登陆时最主要的问题是，之前必须兼容依靠手机号进行用户表分表的数据。因此再次之上新增了与其平级的依靠openid来分表的用户表以支撑第三方登陆功能。并将session、password等用户信息从旧有的依靠手机号进行分表的用户表里迁移到一个新增表。为此我们提前一天去将必要的上亿行必要字段导出，并为了保持一个干净的数据在用户访问量低的凌晨实行了系统维护并进行增量数据的导出。在导出后将数据灌入新表才实现了重要数据的迁移。

拓展作业帮现有的广告平台来源并在与广告平台对接中新增一层审核接口，已保存投放广告的优质性。原有的广告请求逻辑是我们的客户端直接请求第三方的广告，此类请求难免会有一些广告对用户产生不良影响，所以我们的广告请求转而打到我们自己的服务端，让服务端转而请求第三方广告平台的数据。待接口返回数据，又去请求我们自己的广告判优劣接口，优质通过审核的广告啧直接展现，不是则请求下一条广告。关于广告审核的接口则是当广告请求过来是根据广告素材与广告主等信息从缓存中读取是否有相关审核信息，如果有则直接返回结构，没有则把广告入口并添加至缓存然后返回未审核状态。同时我们也新增一个广告后台将广告放入后台并让公司内部人员进行审核。审核成功则同步修改缓存和db的数据的状态。同时添加白名单机制，在服务端存储一些优质广告主信息，若判断第三方请求为此广告主的请求则直接返回审核通过。

同时还进行了一些作业帮运营活动的server端开发，后端同步渲染活动页并在用户触发页面时让前段请求相应的接口调用相关数据接口实现相关功能。如背包减负活动则是将用户点击相关卡片时请求后端接口，后端判断相关用户的状态，在确认状态后通过rpc接口将相关活动奖励（其他业务线的优惠券等）发放给用户。

### 1.2. 知识技能学习情况

整体上是对现有开发环境使用的工具进行熟悉，在服务器层面上，主要使用的是百度公有云提供的集群，服务器的操作系统主要为CentOS，web服务器使用的是主流的nginx，数据库使用的是mysql稳定版本5.6，使用dbproxy进行数据的读写分离，代理连接。后端框架使用的是odp，其集成了saf等拓展框架，

版本控制工具则是git、svn都有使用，缓存系统啧是经典redis+memcached。

Odp：是百度内部开发的web框架。集成nginx，hhvm,内部的php框架，日志系统，fis控件等等技术。了解ODP的使用是进入公司最初的任务。（手册：<http://man.baidu.com/inf/odp/tutorial/>）。了解百度odp，帮助了之前习惯使用轻量级框架开发后端了解巨型框架开发的一些知识。了解了完整的一套集成开发环境非常有助于对于web整个流程的开发有一个更深的理解。Odp对于内部的PHP框架也是通过c来重写的。第一次接触这种类型的框架有利于了解对于PHP语言本身的理解。另外不时的分析日志，查询nginx信息，也帮助对于服务器相关的知识的掌握。而前后端交互虽然并没有什么特别的技术。不过基于fis控件。更清晰的了解了相关数据的展现。

Uda:百度知道内部的数据分析系统，用于大规模的数据分析的图形化数据平台。基于hive实现，简化了复杂的sql查询。

Fis：fis是百度前端技术部开发的，运用于百度各个项目的前端构建工具。用于解决前端开发中自动化工具、性能优化、模块化框架、开发规范、代码部署、开发流程等问题。FIS3采取了类似CSS语法一样的配置风格，易于理解与上手。编译配置不再枯燥而令人生畏，构建项目变得如此简单。提供本地调试服务器，方便快速预览、本地数据模拟等功能。支持文件监听、自动刷新技术。方便的资源定位机制、css语法般的目录和发布配置。等等技术都有助于对于前端技术的了解。为了配置fis，还需配置node的相关环境。还让我顺便了解node相关的知识。Fis的资源内嵌，文件校验、压缩、合并内容嵌入可以为工程师提供诸如图片base64嵌入到css、js里，前端模板编译到js文件中，将js、css、html拆分成几个文件最后合并到一起的能力。这些功能让我更好的了解了前端文件的调度，压缩。Fis的命令也非常简单，对于文件的名打MD5戳的功能也帮助对于浏览器缓存的理解。静态资源管理应当能适配php、java等各种后端方案，通过与后端的结合也让我更加理解了实现网站资源的管理。按需加载、延迟加载、预加载等多种方式灵活控制，从性能优化角度也给我了不少提示。

Dbproxy: 目前业界正在面临最大的难题，面对的数据量在爆炸性的增长，用户对互联网服务提出了更多的要求，因此在背后的架构上都是很大的挑战。随着数据库流量和服务器数量增长，数据库集群面临很多的问题，如何实现应用程序和集群的解耦，降低运维成本?如何实现数据库服务的高可用?如何将并发控制前移，保护数据库系统?如何降低数据库开发的成本，用中间曾实现基本的技术，降低了开发成本。Dbproxy是百度内部开发的数据库中间件技术可以轻松做到数据库热备份，读写分离。虽然DBproxy为dba开发的工具。但作为后端开发者，了解DBproxy有帮助于了解MysQL集群读写分离，连接池，负载均衡，访问控制，可集群化的部署。还能明白运维关键要求：支持高并发和低延迟。了解了DBproxy四个方面1配置信息热加载。2支持连接多个数据库集群。3、将自身压力信息准确的输出。4流量控制，不至于压垮数据库。

Redis：如今除了非常简单的架构，几乎所有的web系统都有缓存系统，可以说缓存是提升服务器的一个保障。而主流的缓存系统主要为memcached和redis.而开源的redis如今已在各个方面超越了memcached。学习redis并用于构建高性能，是开发可扩展的Web应用程序的基础。

Mysql：作为后端开发最重要的技术之一，在学习接触新知识时，进一步加深对mysql的理解应用是非常重要的。如何能发挥mysql的高效，建立最优化的索引执行高效的查询都是非常重要的。为更加理解mysql，我对于mysql其的数据结构，数据类型，索引优化等都进行了更加深的学习。

PHP:PHP作为一门非常容易上手的语言编写起来其实是非常简单的。但作为web服务的基础，我们所追求的是最优、是高效，深入的了解PHP特性。其中对于代码分析然后进行优化是一个非常重要的过程。在这个过程中我了解了许多代码分析工具： xdebug, xhprof C/C++: gprof, Google Performance Tools Others: strace, pstack。

前端基础的(html+css+js):作为一名web开发工程师必须得掌握HTML、CSS和JavaScript。只懂其中一个或两个还不行，必须对这三门语言都很熟悉。也不是说必须对这三门语言都非常精通，但至少要能够运用它们完成大多数任务，而无需地频繁地寻求别人的帮助，因此整理了下想要学习的知识路线

服务器基础运维知识：虽然内部有着百度的noah等系统支持服务器的良好运行，但了解基础的服务器运维知识也是必要的。熟悉各式shell命令，了解如何优化服务器负载缓存都是在职期间学习到的东西。

Awk、sed等linux下的基础文本分析工具:这些工具简单轻便且强大，在写脚本进行文本分析时，这些都是非常可靠的工具

版本控制工具--git、svn:了解其基础命令，进一步了解其工作机制对于团队开发都是十分必要的。Subversion一个通用系统，可以管理任何类型的文件集。对于经常对软件改来改去的程序员来说，它早就是重要的工具。

### 1.3. 职业素养学习培养

在辛苦的开发过程中，一点特别铭记在心：尊重他人的智慧。其他如：树立正确的技能观。决不能利用自己的技能去从事危害公众利益的活动，包括构造虚假信息和不良内容、制造电脑病毒、参与盗版活动、黑客活动等。具有良好的工作责任性，不能以追求个人利益为目的，不随意向他人泄露工作和客户机密。

如何做到简单可依赖，我认为是工程师最重要的一环。

web开发技术一更新命的特别快，新技术、新技巧在不断地被发明出来。我们讲作为web工程师的成长，我认为，主要在两个方面，一部分是“能力”，一部分是“知识”。 我个人的观点，能力占百分之八十，知识占百分之二十。

我认为占重点是能力，是非常稳定的，我认为能力是三大块：编程能力、架构能力、工程能力、编程能力，就是用代码解决问题的能力，你编程能力越强，就能解决越复杂的问题，细分又有调试、OS 原理、算法、数据结构、这些的支撑，你才能解决各种麻烦的问题、架构能力，则是解决代码规模的问题，当一个系统足够复杂，你会写每一块，能解决每一个问题，不、接口隔离，也包含认识业务建立抽象模型，也有一些常见的模式，比如经典的 MVC，还有设计层面，面向对象、设计模式、于你能搞定整个系统，这就需要架构能力，架构能力包含了一些意识，比如解耦。最后工程能力，则是解决协作的问题，当系统规模更大，光靠一个人，是没办法完成的，如何保证几个高手互相能够配合好？如何保证项目里面水平最差的人不拖后腿？这个工程化建设，往往会跨越多个业务，以汇报关系上的团队为单位来做。 包括前后端解耦，模块化，质量保证，代码风格。

其实不难看出来，这三项，其实是有顺序的，低、级、小团队，编程能力一项就能应付，越资深的前端，越大的公司和团队，越是需要后面的技能，但是这里其实资深前端，大团队，对能力的需求，是非常看重的，不是说资深的前端，编程能力就可以变差。社区总会拥有一些声音，对工程能力，对架构能力持有一种抵触的态度，觉得比较虚，觉得不需要。实际上以某些人所在的岗位来说，也没错，毕竟公司、团队的状态确实可能用不到，但是以个人成长的角度来看，就是大错特错。

作为一个程序员、一名工程师，“建立自己的知识体系”是非常重要的。

第一步，寻找线索。你要了解一个知识，可以先找一本书，看看别人都写了什么，了解这个知识学习路径、拓展性。找一些比较准确的，你可以确定它真的

足够全面的资料当作线索。如用wsc的标准、作为线索，我就很有信心同样可能比较适合做的资料，还有一些标准文档的附录，和源代码里的结构定义。

第二步，是建立联系。操作同一组数据，正是面向对象的核心概念，建立对应关系，依据高效原则来面对新知识

第三步，是分类。建立联系以后，我们依据知识之间的联系，进行分类，就可以得到一张图谱，在这个图里面，你就可以非常清楚地知道，哪些知识，是非常重要的，哪些，其实是可以互相替代的，而一旦有你之前没见过的东西，你又能通过把它放到图谱里，来快速理解它，或者找出一些很好的替代方案。

第四步，是追本溯源。当我对一个知识体系的全貌有了概念以后，占了全面两个字，接下来需要确认它的准确性。 很多知识，在社区，会拥有很多的争议，该相信谁呢，这是个问题。 而我的答案，就是追本溯源，去找它最初的讨论和定义。

做到这些之后，想必内心将是有一番更为深刻的体验，找准自己的定位，然后付诸行动，将知识化为实践。

## 2. 存在问题与解决方案

### 2.1. 存在的主要问题

随着作业帮现有的业务线不断增多，用户量的不断增加，如今作业帮的集群压力正不断加强。接口的总请求量已经从最初的数千qps到数万qps。单检索请求就已经到数千qps。接口也从最初的几十个，到单一业务线就有上千接口。新废接口混合，业务线的不断扩大对于开发带来了许多问题。单点峰值流量，在并发系统中，除了请求整体的并发量高，还常见单一热点资源的并发请求量很高。对于查询请求量大的时候，大家知道一定有缓存，可以复用可重用的计算结果。但当单一接口读写量太大时，redis或memcached集群也可能会有雪崩的情况。在各种挑战情况下如何保证服务的稳定便是最大的问题

### 2.2. 解决方案与可行性研究

在高并发场景下最重要的便是稳定的架构设计，在服务架构层面有三个重要的工具。一是 RPC 服务组件二是中间件中间件.消息起的作用：可以把两个模块之间的交互异步化，其次可以把不均匀请求流量输出为匀速的输出流量，所以说消息中间件 异步化 解耦 和流量削峰的利器。三是配置管理，它是 代码级灰度发布以及 保障系统降级的利器。

在接口层最重要的就是无状态。我们在电商网站购物，在这个过程中很多情况下是有状态的，比如我浏览了哪些商品，为什么大家又常说接口层是无状态的，其实我们把状态从接口层剥离到了数据层。像用户在电商网站购物，选了几件商品，到了哪一步，接口无状态后，状态要么放在缓存中，要么放在数据库中，其实它并不是没有状态 ，只是在这个过程中我们要把一些有状 态 的 东 西抽离出来到了数据层。

另外要设计好良好的数据结构，数据层比服务层更需要设计，如果你的数据结构开始设计不合理，将来数据结构的改变会花费你数倍的代价，老的数据格式向新的数据格式迁移会让你痛不欲生，既有工作量上的，又有数据迁移跨越的时间周期。要满足用户量快速增长的，同时线上业务又不能停

在分布式系统我们会做MySQL分库分表，我们要从不同的库和表中读取数据，使用db-proxy等组件。二是使用加速网络的CDN，它是做速度方面的性能提升，刚才我们也提到从CPU、内存、IO、网络四个方面来考虑，CDN本质上一个是做网络智能调度优化，另一个是多级缓存优化。还要做到服务化，各个大网站转型过程中一定会做服务化，其实它就是做抽象和做服务的拆分。第四个看一下消息队列，本质上还是做分类，只不过不是两个边际清晰的类，而是把两个边际不清晰的子系统通过队列解构并且异步化。

## 3. 前期任务完成度与后续实施计划

负责平台活动、第三方登陆、黑板报持续改版等项目开发已经完成并已通过qa的测试和线上的回归。各个项目都在排期内顺利完成虽然会在测试阶段暴露一些问题可随着一段时间的熟悉，虽然仍有不足但作为团队的一员已发挥了相应的价值。后续也会对产品方提出的各式需求一一进行分析、开发。并不断优化代码架构，拓展代码的健壮性，优化数据存储。更多的理解服务器端的各种知识，对于系统缓存设计，redis等进一步的学习与应用。进一步的优化架构设计，实现各个需求的分析、设计和开发。

## 参考文献

1. 尹博学.百度数据库中间层详解[R]北京：北京国际会议中心，2012-04-13