

**本科毕业论文(设计)**

**题目：**气象大数据分析与预测平台设计与实践

**学 院** 管理科学与工程学院

**专 业** 信息管理与信息系统

**班 级** 2014级2班

**学 号** 20140614107

20140614234

20140614204

**姓 名** 傅腾达 王丽 程前

**指导教师** 张贵杰

山东财经大学教务处制

二Ｏ 一八 年 四 月

**山东财经大学学士学位论文原创性声明**

本人郑重声明：所呈交的学位论文，是本人在导师的指导下进行研究工作所取得的成果。除文中已经注明引用的内容外，本论文不含任何其他个人或集体已经发表或撰写过的研究成果。对本文的研究做出重要贡献的个人和集体，均已在论文中作了明确的说明并表示了谢意。本声明的法律结果由本人承担。

学位论文作者签名：

年 月 日

**山东财经大学关于论文使用授权的说明**

本人完全了解山东财经大学有关保留、使用学士学位论文的规定，即：学校有权保留、送交论文的复印件，允许论文被查阅，学校可以公布论文的全部或部分内容，可以采用影印或其他复制手段保存论文。

指导教师签名： 　 论文作者签名：

　　 年 月 日　　　 　　 年 月 日

实时气象大数据分析与预测

摘 要

随着信息化时代的进步，气象分析与预测越来越与人们息息相关。从上世纪二、三十年代开始，传统的无线电技术兴起，可以将高空中观测到的气象数据传递给地面，由此人类可以获取到很多观测资料，通过对这些数据的简单处理，就可以进行气象预测。对这些数据进一步统计处理分析，并根据流体学、动力学等物理学的原理，从而构建起一套基本的气象学科框架，并随着时代的进步，逐步对其进行完善。因此，气象学科是建立在观测资料的基础上，并由气象学家通过各种先进的统计方法进行深入的科学分析研究，发现其中的内在机理，进而形成最终的一套完善的气象学科理论体系。随着资料的不断丰富完善，数据越来越大，人工智能的兴起，单台计算机已经不足以处理如此众多的气象数据，分布式存储、分布式计算逐步的发展，逐步也引入到了气象学科。

关键词：实时、气象、大数据、分析、预测。

Financial Wallet Platform

ABSTRACT

With the progress of the information age, meteorological analysis and prediction are more and more closely related to people.From 20 to 30 s of the last century, the traditional radio technology, the observed meteorological data can be high up in the air is passed to the ground, the man can get a lot of observation data, by means of the simple processing of the data, you can proceed to the weather forecast.Further statistical processing of the data analysis, and according to the fluid study, dynamics, such as the principle of physics, so as to build up a basic framework of meteorological science, and make progress with The Times, to improve step by step.Meteorological science, therefore, is established on the basis of observation data, and by meteorologists through a variety of advanced statistical methods in-depth scientific analysis and research, found that the internal mechanism, and finally

a set of perfect weather discipline theory system.Constantly enrich and perfect as the data is more and more big, the rise of artificial intelligence, a single computer has not enough to deal with so many meteorological data, distributed storage, the development of distributed computing, gradually, gradually also introduced to the weather.

**Keywords**：runtime、weather、bigdata、analyse

**目 录**

1. 引言 i
2. 市场分析 i
3. 设计目的 i
4. 设计目标 i
5. 项目技术应用 i
6. 系统分析 iii
7. 系统工程时间预算 iv
8. 需求分析 iv
9. 系统生命周期图 vi

本系统的启动需要集群的支持，所以先要启动HDFS、Spark、ZooKeeper和Kafka，之后我们会分别运行数据生产者任务和数据消费者任务，到此数据端启动完毕。 vi

数据端启动完毕后，便可以启动Node服务器了，同时启动Express WebAPI服务，到此后端启动完毕。 vi

最后启动前端服务，浏览器便可以访问了。 vi

1. 系统用例 vii
2. 系统设计 vii
3. 系统功能详细分析 viii
4. 系统组件图 viii
5. 系统流程图 viii
6. 系统实施 xv
7. 系统测试 xv
8. 系统效果图 xvi
9. 系统工程决算 xvi
10. 结束语 xvi
11. 致谢 xvii

参考文献 xviii

山东财经大学本科毕业论文（设计）中期检查表 xix

山东财经大学本科毕业论文（设计）开题报告 xx

山东财经大学本科毕业论文（设计）评阅审核表 xxii

1. 引言
2. 市场分析

目前我国互联网平台处于高速发展期，互联网+、大数据这些关键字称谓互联网界的热点。而我国每年新增的气象数据已经达到了PB的级别，相较于上个世纪90年代已然增长了数千倍了。同时在深改和经济转型的经济大环境下，互联网+天气的发展势头迅猛，国家发展气象信息的整体已经非常明确。同时，信息产业低资产运营，投资风险相对较小，气象创业公司已经成为各种资本市场的共同选择。可以预见，像提供更为精准的天气服务，精确到分钟，给物流、农业甚至金融都会带来极大的影响，因此未来气象大数据的市场必定会空间巨大。

1. 设计目的

本次项目的设计充分体现了各个子项目解耦的特性，更符合当前云服务、微服务化的发展趋势。

目前大数据处理技术已经日趋成熟，大数据所带来的经济效益已经在各大互联网公司得到印证，但如何让大数据服务社会，实现其社会价值，是我们需要思考的，这也是本次项目的最终目的。

1. 设计目标

实现一个基于大数据的气象大数据平台，提供实时气象状况查询、历史气象查询、气象分析以及气象预测等功能。本项目将分四部分组成——数据端、后端、前端、测试。

数据端：主要负责数据的获取、处理、转换、导出，为前端提供各种个性化指标以及气象预测信息。

后端：主要负责提供高性能的Web API接口，为前端提供时间、地区等维度的查询能力，是数据端与前端沟通的桥梁。

前端：主要负责数据的可视化展示，让使用者不仅能在其中获取数据，更能探索数据，让用户能够看到数据中蕴含的秘密。

测试：主要负责测试代码的稳定性、可用性，让接口在高密度IO的条件小，依然能够稳定、快速的工作。

1. 项目技术应用

数据架构：hadoop+spark

数据组件：hdfs+spark+spark-streaming+kafka

数据技术栈：

1：离线大数据储存——Hadoop

Hadoop是一个可扩展的、可靠的分布式的大数据储存与计算框架，这使得它成为当下云平台架构下的首选。Hadoop主要由两部分组成——HDFS和MapReduce，在本篇文章中，只用到了HDFS。HDFS的高容错、高可靠性允许用户将Hadoop部署在低廉的硬件上，从而搭建起分布式系统，轻松利用集群的性能完成海量数据的处理。

在本项目中，Hadoop的作用主要体现在分布式储存上，即储存我们的原始气象数据集。

2：快速大数据分析——Spark

Hadoop和Sparkl联合构成了当今的大数据世界，Hadoop主要负责数据储存和资源管理，Spark负责一体化、多元化的不同规模的数据计算。Spark是一个通用的大规模数据快速处理引擎，相比于传统的以Hadoop为基石的第一代大数据技术生态而言，Spark无论是性能还是方案的统一性都具有极为显著的优势。

本项目中，Spark的主要作用体现在各种气象指标的分析。

3：高性能消息中间件——Kafka

Kafka是一个分布式的发布-订阅消息系统，具有高吞吐、低延迟、可扩展性、持久性、可靠性、容错性、高并发性等特点，主要由话题Topic、生产者Producer、消费者Consumer、服务代理Broker组成。生产者将数据发送到Broker代理，Broker代理有多个话题topic，消费者从Broker获取数据。

本项目中，Kafka的作用主要是模拟真实情况下，实时数据的生产与消费。

4：实时数据分析——Spark-Streaming

Spark-Streaming是Spark系统中用于处理流数据的分布式流式处理框架，能够以最低500ms的时间间隔对流数据进行处理，处理的延迟大概在1s，是一个准实时处理系统。并且，Spark-Streaming是一个非常出色的容错系统，具有极为高效的错误处理和恢复能力。

本项目中，Spark-Streaming的作用主要是实时消费Kafka中的气象数据。

后端架构：nodejs+redis+mongodb

后端组件：express

后端技术栈:

1：高性能内存数据库——Redis

Redis是一个Key-Value内存数据库，支持string、list、set、zset等多种value类型，这些数据类型都支持push、pop、add、remove等操作，且这些操作都是原子性的。Redis还支持从简单到设置的主从异步复制，具有非常快的非阻塞的第一次同步，在网络分割上自动重新连接和部分重新同步。

本项目中，Redis的作用主要体现在储存实时产生的气象数据

2：高性能海量数据存取数据库——MongoDB

Mongodb是一个适用于敏捷开发的数据库，可以用于各种规模的企业、个人行业以及各类应用程序的开源数据库，他为开发者提供类传统数据库的二级索引功能、完整的查询系统以及严格一直性等。它是可拓展的、高性能、高可用性的NOSQL数据库，他可以从单服务器部署逐渐扩充到集群部署的多数据中心架构，同时利用其内存优势，MongoDB可以提供高性能的数据库读写功能。

在本项目中，MongoDB的作用主要体现在储存所有产生的气象数据，以用于气象分析使用。

3：事件驱动非阻塞服务器——Node.js

Nodejs是一个基于Chrome V8 JavaScript引擎的JavaScript运行时。nodejs使用一种事件 驱动的、非阻塞的输入/输出模型，使其轻量级和高效。nodejs的包管理npm，是世界 上最大的开源库生态系统。

本项目中，Node.js的作用主要体现在书写后端代码

4：web 开发框架——Express

Express是一个基于Node.js的web开发框架，它灵活、极简，可以帮助我们快速创建各

种APP和移动设备应用。同时它具有丰富的HTTP快捷方法和任意排列组合的Connect

中间件，让我们可以快速创建健壮、友好的API服务。 Express不对Node.js已有的特性

进行二次抽象，我们只是在它上面扩展了web应用所需要的基本功能

本项目中，Express的作用主要体现在提供web api接口

前端架构：nodejs+Vue

前端组件：ElementUI+Vuex+VueRouter+Axios

前端技术栈：

1：渐进式MVVM框架——Vue

Vue是一个渐进式的MVVM框架，简单却不失优雅，小巧而不乏大匠，有着极高的运行时性能，它具有VirtualDom、Reactive、Composable等特性，我们可以将注意力集中保持在核心库上，而将其他功能如路由和全局状态管理交给相关的库。

2：基于Vue2.0的桌面端组件布局框架——ElementUI

Element是一套为开发者、设计师和产品经理准备的前端组件布局框架，基于Vue2.0，可以和Vue完美配合使用。它的设计符合一致性、反馈性、效率性、可控性的原则，可以帮助用户快速构建起一个后台管理系统。

3：全局状态管理框架——Vuex

Vuex是一个转为Vue应用程序开发的全局状态管理模式。它可以集中的管理所有应用内的所有状态，并以相应的规则保证状态以一种可预测的方式发生变化。通过Vuex我们可以方便的管理应用的所有全局状态，让每个组件都能方便的获取、更改全局状态。

4：单页面应用路由框架——VueRouter

VueRouter是一个单页面应用的路由管理框架，使用Vue+VueRouter我们可以方便的创建单页面应用，我们需要做的就是将Vue组件映射到Router路由上，然后告诉VueRouter我们在哪里渲染他们。

1. 系统分析

系统设计是对整个系统的结构的规划以及宏观控制的关键环节，主要是根据系统需求，系统的功能，用例等，分析应该设计什么模块，大体规划使用什么框架去实现，是进行系统评估的重要环节。前期的分析和设计非常重要，性能和优雅是个需要权衡的东西，前期的分析与设计就要设计好，优雅的设计可以减少后期很多工作，使得系统更容易实现低耦合高内聚的设计目标，同时要确保了代码对内部修改的关闭，和对扩展的开放，符合OO的设计思想，模块化管理。

1. 系统工程时间预算

系统的开发工作总体分做四个阶段：设计阶段，开发阶段，集成阶段和测试阶段。这四个阶段也可能交叉进行，比如开发阶段也会进行简单的单元测试，一个模块开发好了也会进行集成测试，中间开发也可能因为前期设计的不合理，而要进行重新设计。

在完成系统层级的开发后，鉴于测试工作量庞大而难以开展，因此测试方法采用小样本白盒测试和大样本黑盒测试的方式，并对测试结果进行相应的记录。

整体工程进度评估如下：设计阶段10日，开发阶段7日，集成阶段2日，测试阶段7日，工程评估26日。

1. 需求分析

本项目由四部分组成：数据端、后端、前端、测试，其中各自的需求分析如下：

数据端：获取原始气象数据，导入Kafka消息队列，准备消费。再使用Spark消费Kafka数据, 进行ETL转化, 转化成一份符合标准规范的日志, 并持久化进Redis数据库，最后使用Spark、SparkStreaming分析日志，计算各种指标

|  |  |  |
| --- | --- | --- |
| 部分 | 需求名称 | 进度 |
| 数据端 | 获取原始数据集 | 完成 |
| 搭建大数据分析服务器 | 完成 |
| 原始数据使用Shell脚本导入HDFS | 完成 |
| 使用SparkStreaming模拟实时数据的产生并导入Kafka | 完成 |
| 使用Spark消费Kafka数据，进行ETL转化，并导入Redis | 完成 |
| 完成实时数据的ETL |  |
| 完成预测数据的ETL |  |
|  |  |

后端：使用Node+Express构建高性能WebAPI服务，借助Node的事件驱动、非阻塞IO的特性，为前端提供稳定的支持高并发IO的接口服务。使用数据端提供的数据，建立起数据端与前端沟通的桥梁。此外，实时数据和历史数据是分别储存在Redis和MongoDB里的，这是为了充分利用这两个数据库的特性——实时查询和海量数据查询。

|  |  |  |
| --- | --- | --- |
| 部分 | 需求名称 | 进度 |
| 后端 | 搭建Node+Express服务 | 完成 |
| Redis、MongoDB数据库设计 | 完成 |
| 完成Dao层代码设计 | 完成 |
| 完成Route层代码设计 | 完成 |
| 完成Controller层代码设计 | 完成 |
| 完成城市列表接口 | 完成 |
| 完成实时数据接口 |  |
| 完成不同时间筛选粒度的数据接口 |  |
| 完成预测数据接口 |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |

前端：使用Node+Vue构建一个基于MVVM设计模式的气象大数据可视化平台。使用Vuex管理全局应用状态，是的每个组件都能方便的获取、更改全局状态。使用VueRouter管理路由与组件的映射关系。

|  |  |  |
| --- | --- | --- |
| 部分 | 需求名称 | 进度 |
| 前端 | 搭建Node+Vue服务 | 完成 |
| 完成Navbar组件 | 完成 |
| 完成Sidebar组件 | 完成 |
| 完成Filter组件 | 完成 |
| 完成今日天气页面 | 完成 |
| 完成气象温度页面 |  |
| 完成气象风速页面 |  |
| 完成平均降水页面 |  |
| 完成气压页面 |  |
| 完成空气质量页面 |  |
| 完成历史数据页面 |  |
|  |  |
|  |  |

表2-1 系统需求表

1. 系统生命周期图

本系统的启动需要集群的支持，所以先要启动HDFS、Spark、ZooKeeper和Kafka，之后我们会分别运行数据生产者任务和数据消费者任务，到此数据端启动完毕。

数据端启动完毕后，便可以启动Node服务器了，同时启动Express WebAPI服务，到此后端启动完毕。

最后启动前端服务，浏览器便可以访问了。

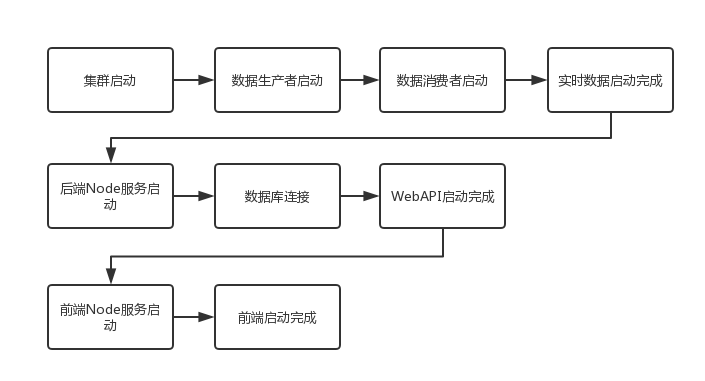


图2-1 系统生命周期图

1. 系统用例

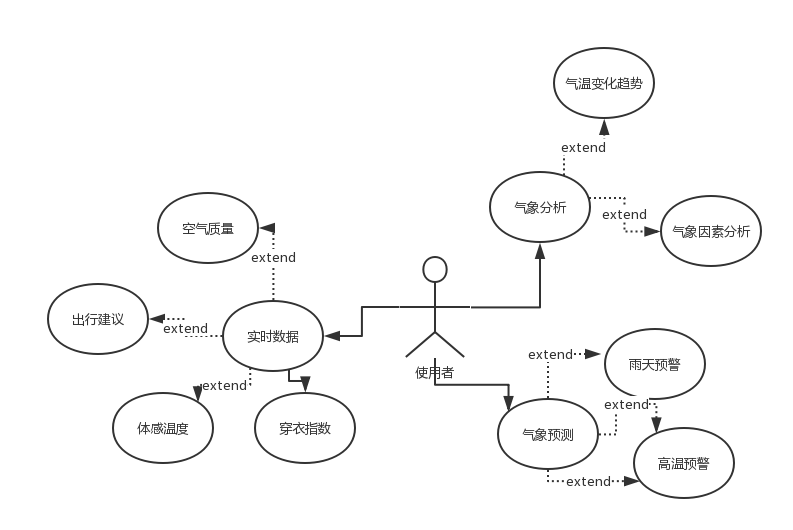


图2-2 系统用例图

|  |  |
| --- | --- |
| 系统用例说明表 | |
| 用例名称 | 系统整体用例图 |
| 用例正流程 | 1.用户注册登录  2.后端校验信息  3.进入气象可视化平台  4.进行Ajax请求  5.后端接收请求，连接数据库，获取数据  6.根据数据渲染页面 |

1. 系统设计

功能流程是程序设计的重要工作环节，对于业务流程详细，合理的设计是确保系统稳定性，健壮性和友好性的重要保证。功能流程以用例规约表和活动图为主要设计手段，必要时以流程图为辅助。

流程设计的设计应力求简明扼要，清晰准确的描述系统功能的流程结构。以下为本系统的功能流程设计图表。

1. 系统功能详细分析
2. 系统组件图

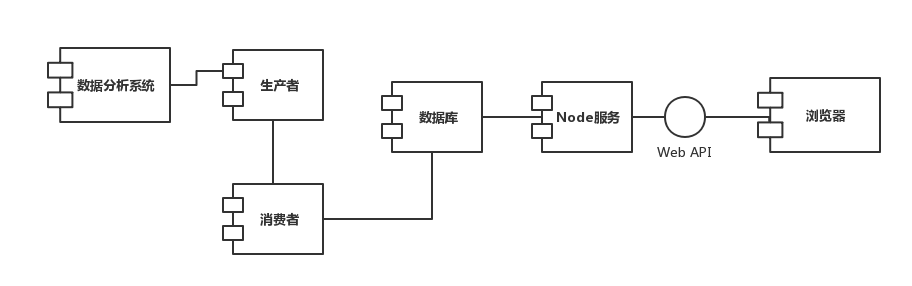


图3-1 系统组件图

1. 系统流程图

整体：从整体上来看，我们的后端作为一个桥梁沟通着数据端和前端。数据端会产生各种指标数据与预测数据存入数据库，同时后端会搭建WebAPI接口服务，响应前端页面的Ajax请求，对数据库进行增删查改的操作。

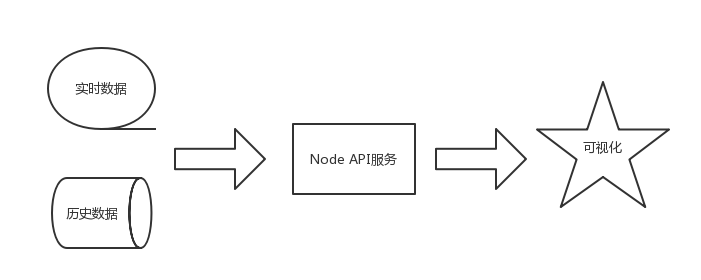
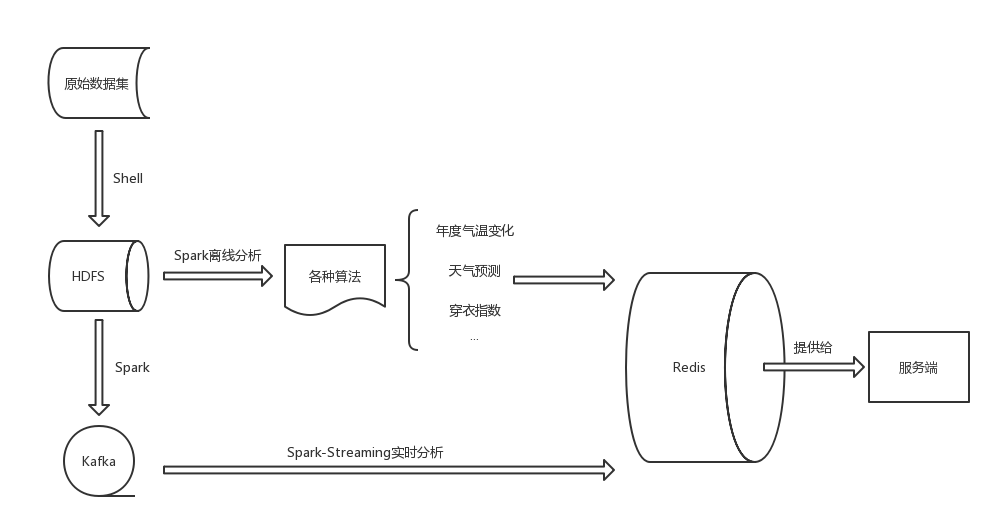
数据端：在数据端内部，先把原始气象数据通过shell脚本存入HDFS中备用。之后，为了模拟实时数据的产生，我们会启动一个Spark任务去读取HDFS上的所有气象数据，并每隔3s导入到Kafka消息队列中。同时，我们会另起一个SparkStreaming任务去实时分析Kafka的weather topic中的数据，并实时导入Redis中，以此来模拟实时数据的产生。另一方面，为了满足各种个性化指标的展现要求，我们会使用Spark对HDFS的气象数据进行离线分析，通过特点算法得出各种个性化数据，这些数据也会被导入数据库中，供后端使用。

图3-3 系统数据端流程图

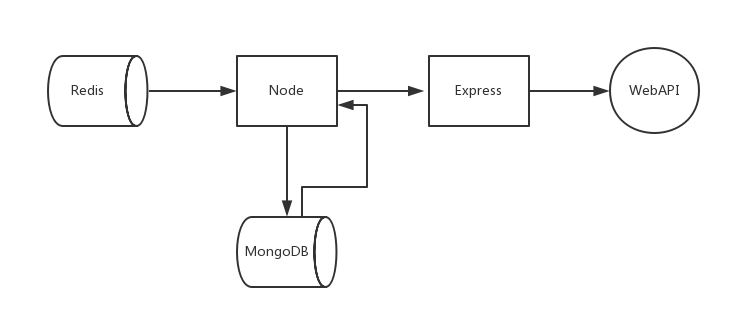
后端：在后端内部，我们启动了一个Node服务器，使用Express做了一个WebAPI服务。它会实时接收前端的请求，并不断向Redis中读取实时数据，与此同时，读取的实时数据也会往MongoDB中导入一份，这是为了给气象分析使用。

图3-4 系统服务端流程图

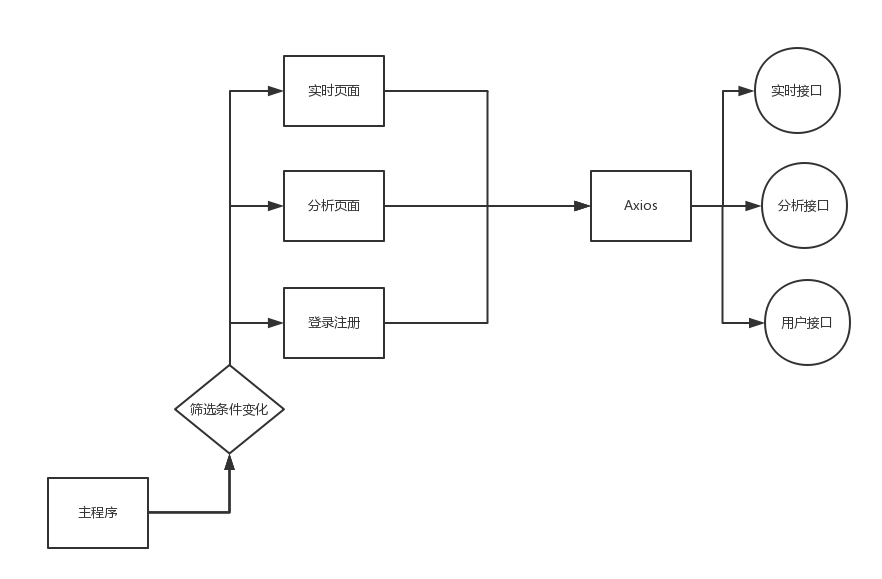
前端：在前端内部，我们启动了一个Node热加载服务器和Vue单页面应用程序。整个前端页面采用了MVVM的设计模式，数据的改变会自动触发新的网络请求，从而更新当前页面的数据，数据会像水一样在程序内部流动，十分优雅。

图3-5 系统前端流程图

3. 系统设计实现关键代码

数据端：

|  |  |
| --- | --- |
| 关键代码：分析数据导入Kafka | |
| 说明 | 我们会启动多个线程，分别取模拟多个城市的气象站同时去获取实时气象数据。 |
| 片段1 | //提交5个线程  cities.foreach { city =>  threadPool.execute(new ThreadTask(city))  } |
| 片段2 | // 获取HDFS数据并导入Kafka  val reader = Try(HdfsUtils.readGZFile(fileSystem, s"/root/clouddata/$year/${city.\_1}-$year.op.gz"))  reader match {  case Success(r) =>  // 一行是一天  var lines = Stream.continually(r.readLine()).takeWhile(\_ != null)  lines.toArray.slice(1, lines.length).foreach { line =>  logger.info(s"### sending kafka msg $city ###")  val newline = line.replace(city.\_1.replace("-", " "), city.\_2)  producer.send(new ProducerRecord(topic, newline))  logger.info(s"### sent kafka msg: $newline ###")  Thread.sleep(3000)  }  case Failure(f) =>  logger.warn(s"### year: $year-$city 获取BufferdReader失败###");  f.printStackTrace()  } |

|  |  |
| --- | --- |
| 关键代码：实时消费Kafka数据 | |
| 说明 | 我们会启动一个SparkStreaming任务去实时消费Kafka数据 |
| 片段1 | // 连接Kafka Streaming  val kafkaStream = KafkaUtils.createDirectStream[String, String, StringDecoder, StringDecoder](ssc, kafkaParams, topics) |
| 片段2 | // 遍历streamData这个RDD集合并将数据导入Redis  streamData.foreachRDD { rdd =>  rdd.foreach { data =>  val jedis = pool.getResource  jedis.select(1)  logger.info(s"### writing into redis with data: ${new Gson().toJson(data.get)}")  jedis.rpush(data.get.city, new Gson().toJson(data.get))  jedis.close()  }  } |

后端：

|  |  |
| --- | --- |
| 关键代码：区间数据查询 | |
| 说明 | 进行区间数据的查询 |
| 片段1 | // 连接Redis  client = redis.createClient(redisPort, redisIp); |
| 片段2 | // 根据日期为index的方式进行区间查询  var startIndex = await insertValueSearch(startTime, false, true, lindexAsync, keyName, count);  if (startIndex == -1)  throw "no find start in the range";  var endIndex = await insertValueSearch(endTime, true, false, lindexAsync, keyName, count);  if (endIndex == -1)  throw "no find end in the range";  ret = await lrangeAsync(keyName, startIndex, endIndex); |

|  |  |
| --- | --- |
| 关键代码：api路由注册 | |
| 说明 | 注册api路由关键代码 |
| 片段1 | // 注册/query路由  router.get('/query', async (request, response, next) => {  let city = request.query.city;  if (!city) {  city = "济南";  }  let startTime = parseInt(request.query.startTime);  let endTime = parseInt(request.query.endTime);  if (isNaN(startTime) || startTime < 0)  startTime = 0;  if (isNaN(endTime))  endTime = 20190101;  var dataArray = [];  let queryRes = await queryDataByRange(startTime, endTime, city)  for (var i in queryRes) {  var dataString = queryRes[i];  dataArray.push(JSON.parse(dataString));  }  sendJson(response, 0, "invoke ok!", dataArray);  } |

数据库：

|  |  |
| --- | --- |
| 关键代码：数据库索引优化 | |
| 说明 | 对数据库的时间列进行索引优化，加快时间区间筛选的速度 |
| Redis差值查找算法 | // redis list中数据是按时间排序的，二分查找仍为log2n复杂度，目前使用了比二分还快的查找算法，即插值查找，按照数据值根据比例划分查找，达到秒查的效率。  //let  let partition = parseInt((value - dataStartTime) \* (endIndex - startIndex + 1) / (dataEndTime - dataStartTime));  if (partition < startIndex)  partition = startIndex;  if (partition > endIndex)  partition = endIndex;  let partitionJson = JSON.parse(await lindexAsync(redisKeyName, partition));  let partitionValue = parseInt(partitionJson.date);  if (value < partitionValue) {  endIndex = partition - 1;  } else if (value > partitionValue) {  startIndex = partition + 1;  }  else {  findIndex = partition;  break;  } |
| MongoDB |  |

前端：

|  |  |
| --- | --- |
| 关键代码：基于MVVM的设计模式 | |
| 说明 | 基于MVVM，筛选值变化会自动发起Ajax请求，重新渲染页面 |
| 片段1 |  |
| 片段2 |  |

|  |  |
| --- | --- |
| 关键代码：使用Vuex进行全局状态管理 | |
| 说明 | Vuex来管理全局的筛选条件，是的所有组件都能够方便使用 |
| 片段1 |  |
| 片段2 |  |

|  |  |
| --- | --- |
| 关键代码：使用VueRouter管理路由 | |
| 说明 | VueRouter进行页面组件和路由的映射 |
| 片段1 |  |
| 片段2 |  |

1. 系统实施
2. 系统测试

系统测试是保证系统业务流程符合预期，系统健壮性符合设计要求的工作环节，也是系统开发流程中必须做的工作。本系统的系统测试工作分为黑盒测试和白盒测试两阶段，测试工作以黑盒测试为主，白盒测试为辅，主要通过RESTful风格的单元测试进行测试。因篇幅原因，此处不再对测试用例和测试案例进行展示，仅提供单元测试结果表和相应的单元测试结果数据。

测试工作预算：测试工作时间预算为测试用例评估1日，单元测试4日，黑盒测试3日，出具测试报告1日。

单元测试数据：系统单元测试共测试40个功能用例，编写测试用例119个，一次通过率87%，修复通过率11%，用例废弃率2%。

黑盒测试数据：黑盒测试共测试40个功能用例，一次通过率70%，修复通过率28%，用例废弃率2%。

白盒测试数据：白盒测试共测试40个功能用例，一次通过率90%，修复通过率8%，用例废弃率2%。

测试工作决算：测试工作时间决算为测试用例评估3日，单元测试6日，黑盒测试2日，出具测试报告1日。

本系统测试工作以确保系统可使用，保证系统健壮性和稳定性为目标。测试工作以单元测试为主要途径，黑盒测试和白盒测试相结合的方式进行测试。

整体代码质量参照以下代码质量缺陷分析报告和代码标准分析质量报告：

图4-1 代码质量缺陷分析报告

图4-2 代码标准分析报告

1. 系统效果图
2. 系统工程决算

工程预算和决算时间表如下：

设计阶段用例设计2日，流程设计5日，系统组件构架1日，需求复核评审1日，预算9日。实际用时用例设计5日，流程设计5日，系统组件架构2日，需求符合评审1日，实际12日，超标33%。

开发阶段系统架构搭建2日，技术评审1日，代码开发工作17日，预算20日。系统架构搭建2日，技术评审2日，代码开发工作26日，超标33%。

集成阶段前后端对接4日，系统覆盖率测试1日，出具集成报告1日，预算6日。实际用时前后端对接8日，系统覆盖率测试2日，出具集成报告1日，超标98%。

测试阶段测试案例设计1日，单元测试与白盒测试8日，出具测试报告1日，预算10日。实际用时案例设计3日，单元测试与白盒测试8日，出具测试报告1日，超标12日，超标20%。

总开发预算工时45日，实际66日，超标46%。

表4-29 系统工程决算表

|  |  |  |  |
| --- | --- | --- | --- |
| **系统工程预决算表** | | | |
| 工程阶段 | 预算数 | 决算数 | 超标值 |
| 用例设计 | 2 | 5 | 150% |
| 系统组件架构 | 1 | 2 | 100% |
| 需求符合评审 | 1 | 1 | — |
| 系统架构搭建 | 2 | 2 | — |
| 技术评审 | 1 | 2 | 100% |
| 代码开发工作 | 17 | 26 | 53% |
| 前后端对接 | 4 | 8 | 100% |
| 系统覆盖率测试 | 1 | 2 | 100% |
| 出具集成报告 | 1 | 1 | — |
| 测试案例设计 | 1 | 3 | 200% |
| 单元测试与白盒测试 | 8 | 8 | — |
| 出具测试报告 | 1 | 1 | — |

（单位：日）

1. 结束语

本系统以气象大数据平台为设计目标，截止至2018年4月25日工程结束本系统已经初步具备数据分析系统平台架构以及包含基本模块。

通过张贵杰老师的指导以及吸纳学院各位老师中肯的意见和建议，通过本次毕业设计，我们不仅在需求分析，系统设计，系统编码以及系统测试等一系列系统开发的全生命周期流程都有更深刻的学习和认识，并且实践工作能力也得到了提升，对于书面知识有着更深刻的理解。同时，我们也对数据分析的需求和业务流程有一定的了解和认识。

通过本次毕业设计，我们三人对自己学习的领域有了更加深刻的理解。我们尝试了很多在工作中根本不会用到的特性，没有了工作上的压力，让我们有了更大的勇气去尝试、去探索，这也使我们的知识面得到了很大的扩充，这是我们工作中不可能得到的。

本次项目不免也有许多遗憾，由于资金有限，我们无法租到多台高性能主机，只能用一个2G内存的小服务器，这使得我们数据分析能力大打折扣。由于时间有限，有许多细节我们也无法顾及，但是我们还是完成了本次项目的核心代码。

总之，感谢本次毕设的经历！

1. 致谢

本次毕业设计的成功离不开学院以及指导老师的教诲与帮助，尤其张贵杰老师对我们的工作进行了深入而有见地的指导。同时本次毕业设计期间，我们也得到了学院许多其他老师的指导和帮助。本次毕业设计之所以取得我们预想的成果，与学院的各位老师的大力帮助密不可分，在此首先向给予本次毕业设计指导帮助的各位老师，特别是我们的指导老师——张贵杰老师表达诚挚的感谢！

四年来，在校园中我们感受到了学院领导以及各位导师的衷心投入与悉心栽培，四年的大学学习即将落幕，曾经遇到的困难和受到的指导鼓励仍然历历在目，因此我们大学四年得以全面成长，在理论、实践以及社会见闻上等方面获得全面的发展。相信我们走上工作岗位以后，我们也将继续承载着这份栽培与教诲，带着曾经在校园中收获的知识与感悟不忘初心，继续前行。

最后，再次向张贵杰老师以及为此次毕业设计提供各种支持帮助的学院老师们致以诚挚的感谢！

感谢各位老师的指导！

参考文献

[1] 朴灵.深入浅出Node.js [M]. 人民邮电出版社，2013（1）：10-188.

## 山东财经大学本科毕业论文（设计）中期检查表

学院：管理科学与工程学院 填表时间： 年 月 日

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 姓 名 | 付腾达  王丽  程前 | 学 号 | 20140614107  20140614234  20140614204 | 专 业 | 信息管理与信息系统 |
| 班 级 | 1402班 | 联系方式 |  | E-mail |  |
| 论文（设计）题目 | 气象大数据分析与预测平台设计与实践 | | | | |
| 已完成的工作： | | | | | |
| 未完成的工作： | | | | | |
| 计划完成时间和拟采取措施： | | | | | |
| 指导老师意见：  签名：  年 月 日 | | | | | |

## 山东财经大学本科毕业论文（设计）开题报告

学院：管理科学与工程学院 填表时间： 年 月 日

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 姓 名 | 付腾达  王丽  程前 | 专 业 | 信息管理与信息系统 | | 学 号 | | 20140614107  20140614234  20140614204 |
| 班 级 | 1402班 | 联系方式 |  | | E-mail | |  |
| 指导教师姓名 | 宋健 | 联系方式 |  | | E-mail | |  |
| 论文(设计)题目 | 气象大数据分析与预测平台设计与实践 | | | | | | |
| 计划完成时间 | 2018-04-27 | | | 计划撰写字数 | | 10k字 | |
| 本选题研究的目的、意义和实用价值： | | | | | | | |
| 本选题国内外研究现状： | | | | | | | |
|  | | | | | | | |

|  |
| --- |
| 围绕本选题已做哪些准备工作，计划再做的工作： |
| 拟解决的关键问题： |
| 主要参考文献： |
| 指导教师意见：  指导教师签名：  年 月 日 |

## 山东财经大学本科毕业论文（设计）评阅审核表

学院： 管理科学与工程学院 填表时间： 年 月 日

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| 姓 名 | | 傅腾达  王丽  程前 | 学 号 | 20140614107  20140614234  20140614204 | 专 业 | 信息管理与信息系统 |
| 班 级 | | 1402班 | 联系方式 |  | E-mail |  |
| 指导教师姓名 | | 张贵杰 | 联系方式 |  | E-mail |  |
| 论文（设计）  题目 | | 气象大数据分析与预测平台设计与实践 | | | 字数 |  |
| 指  导  老  师  意  见 | 评阅要点(主要包括：1.选题的理论和现实意义；2.作者运用所学知识分析问题和解决问题的能力；3.论文写作的规范性及文字表达能力；4.存在的问题；5.建议成绩)  指导教师： （签名）  年 月 日 | | | | | |

|  |  |
| --- | --- |
| 评 阅 老 师 意 见 | 评阅要点(主要包括：1.论文选题的实际价值与理论意义；2.论文的主要工作及成果；3.论文的文字表述、写作规范情况；4.论文的主要不足及修改意见；5.建议成绩)  评阅人： （签名）  年 月 日 |
| 答辩委员会（小组）意 见 | **该论文成绩综合评定为：**  答辩委员会（小组）负责人： （签名）  年 月 日 |
| 学 院 审 核 意 见 | 学院负责人： （签名）  （公章）  年 月 日 |