

开源项目创新赛——自由组

项目技术文档

|  |  |
| --- | --- |
| 项目名称 | 基于python的数据获取和  相关性性分析系统 |
| 队伍名称 | 勇敢汪汪队 |
| 队员姓名 | 刘航 李超杰 梁颖秋  陈湘嵋 朱倩倩 |

2021年11月

目 录

一、项目简介 ..3

二、研究背景 3

三、技术路线 4

3.1 爬虫 4

3.2 django 5

3.3 统计分析工具 7

四、测试 9

4.1 测试目的 9

4.2 测试需求分析 9

4.3 单元测试 10

4.4 功能测试 10

4.5 系统测试 10

五、数据备份方案 12

5.1 备份 12

5.2 恢复 12

六、项目管理 13

6.1 项目组织 13

6.2 需求管理 14

6.3 范围管理 16

6.4 时间管理 16

6.5 成本管理 16

6.6 风险管理 18

6.7 质量管理 22

6.8 人力资源管理 25

6.9 配置管理 26

七、成本分析 27

7.1 成本构成分析 27

7.2 软件开发与维护成本 29

7.3 固定成本及实施费 30

7.4 总成本 32

**一.项目简介**

当前环境下，数据分析的基本流程是，首先在数据网站上下载相关数据，然后在本地进行数据的分析，如果下载的数据不符合最终的实验要求，还需要重新返回网站下载其他的相关数据，因此整体的数据分析效率低下。

针对这个问题，本项目开发了一个在线数据分析的平台。首先针对的是国家统计局的所有数据，在该平台中可以下载其中任意数据，并且可以直接选择数据进行在线的统计分析。平台中提供了一些基础的统计分析工具，比如线性回归主成分分析，QQ图检验的。用户可以首先根据这些统计工具进行数据的检验，进而选取合适的数据进行下载。可以大大提高数据分析效率。

**二.研究背景**

数据是经济分析的基础。严格的经济理论和预测判断均需要以详实合理的数据做支撑。认识理解经济数据是进行经济研究所需的基本素质。宏观经济涉及多层面庞杂的数据，系统全面的掌握宏观数据体系，以及阅读、处理、分析、判断经济数据对宏观研究至关重要，卓越的经济理论以及准确的经济预测均建立在对数据的准确把控上。特别对于转型时期的中国，宏观数据体系基本上依据市场经济需求建立，但尚未完全摆脱计划经济影响，理解中国经济数据更具复杂性。

宏观经济分析可以把握证券市场的总体变化趋势，尤其要关注货币政策和财政政策的变化。把握证券市场的投资价值。掌握宏观经济政策对市场的影响力度与方向。宏观经济对股市的影响不同于成熟市场经济，中国股市的特殊性。对影响宏观经济运行总量指标的因素及其变动规律进行分析，如对国民生产总值、消费额、投资额、银行贷款总额及物价水平等的变动规律的分析，进而说明整个经济的状态和全貌。

宏观经济走势的基本指标：

1.国民经济总体指标

国内生产总值GDP

工业增加值

失业率

通货膨胀——居民消费价格指数CPI、[生产者价格指数PP](https://www.zhihu.com/search?q=%E7%94%9F%E4%BA%A7%E8%80%85%E4%BB%B7%E6%A0%BC%E6%8C%87%E6%95%B0PP&search_source=Entity&hybrid_search_source=Entity&hybrid_search_extra=%7B%22sourceType%22%3A%22article%22%2C%22sourceId%22%3A34745661%7D" \t "_blank)I

宏观经济运行经期指标——采购经理指数PMI

[国际收支](https://www.zhihu.com/search?q=%E5%9B%BD%E9%99%85%E6%94%B6%E6%94%AF&search_source=Entity&hybrid_search_source=Entity&hybrid_search_extra=%7B%22sourceType%22%3A%22article%22%2C%22sourceId%22%3A34745661%7D" \t "_blank)——经常项目、资本项目

2.投资指标

固定资产投资

投资主体——政府投资、企业投资、外商投资

3.消费指标

社会消费品零售总额

城乡居民储蓄存款余额

居民可支配收入

这些指标都可以在国家统计局的网站上获得。在进行宏观数据分析时，往往会从国家统计局数据网站上扒取这些数据，然后对这些数据使用相关软件进行分析，比如。

国际上使用最广泛的web分析工具，免费版就已有足够多的功能供大家深挖使用。但其服务器又在国外。在国内访问速度慢，且不稳定。国内最大的中文网站分析平台。如果你使用百度的竞价排名，那用百度统计做分析是不错的选择，天然首要位置突出分析自家的竞价排名流量。

Hadoop是一个能够对大量数据进行[分布式](https://www.zhihu.com/search?q=%E5%88%86%E5%B8%83%E5%BC%8F&search_source=Entity&hybrid_search_source=Entity&hybrid_search_extra=%7B%22sourceType%22%3A%22article%22%2C%22sourceId%22%3A375781196%7D" \t "_blank)处理的软件框架。但是Hadoop是以一种可靠、高效、可伸缩的方式进行处理的。Hadoop是可靠的，因为它假设计算元素和存储会失败，因此它维护多个工作数据副本，确保能够针对失败的节点重新分布处理。Hadoop是高效的，因为它以并行的方式工作，通过[并行处理](https://www.zhihu.com/search?q=%E5%B9%B6%E8%A1%8C%E5%A4%84%E7%90%86&search_source=Entity&hybrid_search_source=Entity&hybrid_search_extra=%7B%22sourceType%22%3A%22article%22%2C%22sourceId%22%3A375781196%7D" \t "_blank)加快处理速度。Hadoop还是可伸缩的，能够处理PB级数据。此外，Hadoop依赖于社区服务器，因此它的成本比较低，任何人都可以使用。

[思迈特软件Smartbi](https://link.zhihu.com/?target=https%3A//www.smartbi.com.cn/" \t "_blank)：融合传统BI、自助BI、智能BI，满足BI定义所有阶段的需求；提供数据连接、数据准备、数据分析、数据应用等全流程功能；提供复杂报表、数据可视化、自助探索分析、[机器学习](https://link.zhihu.com/?target=https%3A//www.smartbi.com.cn/gn/wjjm" \t "_blank)建模、预测分析、[自然语言分析](https://www.zhihu.com/search?q=%E8%87%AA%E7%84%B6%E8%AF%AD%E8%A8%80%E5%88%86%E6%9E%90&search_source=Entity&hybrid_search_source=Entity&hybrid_search_extra=%7B%22sourceType%22%3A%22article%22%2C%22sourceId%22%3A375781196%7D" \t "_blank)等全场景需求；满足数据角色、分析角色、管理角色等所有用户的需求。

Bokeh：这套可视化框架的主要目标在于提供精致且简洁的图形处理结果，用以强化大规模数据流的交互能力。其专门供[Python](https://link.zhihu.com/?target=https%3A//www.smartbi.com.cn/wiki/3242" \t "_blank)语言使用。

Storm：Storm是自由的开源软件，一个分布式的、容错的实时计算系统。Storm可以非常可靠的处理庞大的数据流，用于处理Hadoop的[批量数据](https://www.zhihu.com/search?q=%E6%89%B9%E9%87%8F%E6%95%B0%E6%8D%AE&search_source=Entity&hybrid_search_source=Entity&hybrid_search_extra=%7B%22sourceType%22%3A%22article%22%2C%22sourceId%22%3A375781196%7D" \t "_blank)。Storm很简单，支持许多种编程语言，使用起来非常有趣。Storm由Twitter开源而来，其它知名的应用企业包括Groupon、淘宝、支付宝、[阿里巴巴](https://www.zhihu.com/search?q=%E9%98%BF%E9%87%8C%E5%B7%B4%E5%B7%B4&search_source=Entity&hybrid_search_source=Entity&hybrid_search_extra=%7B%22sourceType%22%3A%22article%22%2C%22sourceId%22%3A375781196%7D" \t "_blank)、乐元素、Admaster等等。

通过调研发现，这些工具都无法提供宏观经济数据的在线分析。

**三.技术路线**

**3.1爬虫**

3.1.1selenium+chromdriver简介

当今Web技术的大环境下，为了网站的性能、可扩展性、用户友好等方向，几乎所有的Web页面都在使用浏览器脚本动态生成内容，所以如果没有脚本执行环境就没有数据。

无头浏览器类似于在流行的Web浏览器的环境中提供对网页的自动控制，但通过命令行接口或使用网络通信来执行。它们特别适用于测试网页，因为它们能够以浏览器相同的方式呈现和解释HTML，包括页面布局、颜色、字体选择和JavaScript和Ajax的执行等样式元素，当使用其它测试方法时，这些元素通常是不可用的。  
简单来说就是用浏览器来对目标URL进行HTML解析，CSS渲染，Js执行，借由API甚至可以模拟用户行为（鼠标点击，键盘输入），但不提供UI渲染。

本项目使用selenium+chromdriver爬取国家统计局数据。Selenium 是 ThoughtWorks 提供的一个强大的基于浏览器的开源自动化测试工具。  
Selenium 是一个用于 Web 应用程序测试的工具，测试直接自动运行在浏览器中，就像真正的用户在手工操作一样。支持的浏览器包括 IE、Chrome 和 Firefox 等。这个工具的主要功能包括：测试与浏览器的兼容性 - 测试您的应用程序看是否能够很好地工作在不同浏览器和操作系统之上；测试系统功能 - 创建回归测试检验软件功能和用户需求；支持自动录制动作，和自动生成 .NET、Perl、Python、Ruby 和 Java 等不同语言的测试脚本。

Selenium 2，又名 WebDriver，它的主要新功能是集成了 Selenium 1.0 以及 WebDriver， 是两个项目的合并，既兼容 Selenium API 也支持 WebDriver API。

WebDriver 曾经是 Selenium 的竞争对手（最开始是google的一个人弄的，主要用于避免在JavaScript的沙箱环境里存在的各种限制），他主要是通过利用浏览器原生API的方式来操控浏览器执行各种动作（还包括系统级别的调用来模拟用户输入）。

Selenium WebDriver 就是对浏览器提供的原生API进行封装，使其成为一套更加面向对象的Selenium WebDriver API。  
使用这套API可以操控浏览器的开启、关闭，打开网页，操作界面元素，控制Cookie，还可以操作浏览器截屏、安装插件、设置代理、配置证书等。由于使用的原生API，其速度与稳定性都会好很多。但浏览器厂商各不相同，提供的驱动各异（ChromeDriver、FirefoxDriver(xpi插件)、InternetExplorerDriver(exe)等），API也会有差异（好像都走JSON Wire Protocol，并且向W3C标准靠拢）。Selenium 对不同厂商的各个驱动进行了封装，如：selenium-chrome-driver、selenium-edge-driver、selenium-firefox-driver等。  
还包括了对移动应用进行测试的AndroidDriver和iOS WebDriver，以及一个基于HtmlUnit的无界面实现HtmlUnitDriver。

WebDriver API可以通过Python、Ruby、Java和C#访问

WebDriver是一个用来进行复杂重复的web自动化测试的工具。意在提供一种比Selenium1.0更简单易学，有利于维护的API。它没有和任何测试框架进行绑定，所以他可以很好的在单元测试和main方法中调用。一旦创建好一个Selenium工程，你马上会发现WebDriver和其他类库一样：它是完全独立的，你可以直接使用而不需要考虑其他配置，这个Selenium RC是截然相反的。

3.1.2关键代码

关键代码如下所示;

from selenium import webdriver

driver = webdriver.Chrome()

driver.get('https://www.baidu.com')

input = driver.find\_element\_by\_id('kw') #获取输入框

searchButton = driver.find\_element\_by\_id('su') #获取搜索按钮

常用查找元素方法：

find\_element\_by\_id # ID

find\_elements\_by\_class\_name # class

find\_elements\_by\_tag\_name # 标签名

find\_elements\_by\_name # name

find\_elements\_by\_link\_text # a标签中的text查找（精确匹配）

find\_elements\_by\_partial\_link\_text #a标签中的text查找（部分匹配即可）

find\_elements\_by\_css\_selector # css选择器查找

find\_elements\_by\_xpath # find\_elements\_by\_xpath("//input")，请翻阅文档

输入并点击

inputElement.send\_keys("Python") #输入框输入"Python"

searchButton.click() #搜索

3.2django

3.2.1django介绍

Django 最初是被开发用来管理劳伦斯集团旗下一些以新闻内容为主的网站。2003 年，Lawerence Journal-World 报社的 Web 开发者艾德里安和威利森开始用 Python 语言去开发应用。新闻界的快节奏使他们必须提高产品的开发速度，于是他们两个人想办法构建出了节省开发时间的框架，这个框架将能复用的组件或者模块放在一起，使开发者不必重复的工作，这是 Django 诞生的根本原因。后来他们所在的开发小组决定将这个框架发布成一个开源软件。经过发展和时间的积累，Django 有了数以万计用户和贡献者，现在 Django 成为了 Web 开发者中最流行的框架.总之，Python 的诞生是为了减少重复性工作，节省开发者时间，可以让开发者把有限的精力投入到关键的开发环节中。

通过上面的介绍，可以给出 Web 框架的概念：它是一套组件，提供通用的设计模式，能够最大程度地降低开发 Web 站点开发的难度，Django 的设计目标就是使复杂的工作变得简单。

相对于 Python 的其他 Web 框架，Django 的功能是最完整的，Django 定义了服务发布、路由映射、模板编程、数据处理的一整套功能。这也意味着 Django 模块之间紧密耦合。

3.2.2django特点

Django 的主要特点如下：

完善的文档：经过 10 余年的发展和完善，Django 官方提供了完善的在线文档，为开发者解决问题提供支持。

集成 ORM 组件：Django 的 Model 层自带数据库 ORM 组件，为操作不同类型的数据库提供了统一的方式。

URL 映射技术：Django 使用正则表达式管理URL映射，因此给开发者带来了极高的灵活性。

后台管理系统：开发者只需通过简单的几行配置和代码就可以实现完整的后台数据管理Web控制台。

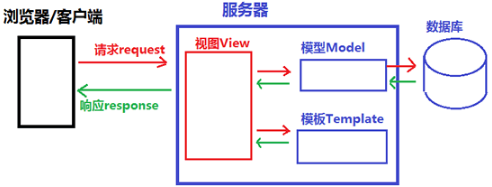
错误信息提示：在开发调试过程中如果出现运行异常，Django 可以提供非常完整的错误信息帮助开发者定位问题。

Django 借鉴了经典的 MVC 模式，它也将交互的过程分为了 3 个层次，也就是 MTV 设计模式；

Model：数据存储层，处理所有数据相关的业务，和数据库进行交互，并提供数据的增删改查；

Template：模板层（也叫表现层）具体来处理页面的显示；

View：业务逻辑层，处理具体的业务逻辑，它的作用是连通Model 层和 Template 。

  
图3-1：Django MTV 设计模式示意图

我们按照 MVC 的设计模式对 MTV 进行分析，MTV 设计模式中，用 View 层取代了 Controller 层的位置，用 Template 层取代了原来 View 层的位置。  
初次接触 Django 的设计模式的人，可能会对 Template 层产生疑问，其实 Template 英文的含义就是“模板”的意思，你可以简单理解成，它是一个 HTML 页面 ，HTML 页面的渲染在视图层完成。  
同样我们也对 MTV 设计模式的请求与响应过程进行描述：

用户通过浏览器对服务器发起 request 请求，服务器接收请求后，通过 View 的业务逻辑层进行分析，同时向 Model 层和 Template 层发送指令；

Mole 层与数据库进行交互，将数据返回给 View 层；

Template 层接收到指令后，调用相应的模板，并返回给 View 层；

View 层接收到模板与数据后，首先对模板进行渲染（即将相应的数据赋值给模板），然后组织成响应格式返回给浏览器，浏览器进行解析后并最终呈现给用户。

通过以上两种设计模式的比较， 我们可以得出 MTV 是 MVC 的一种细化，将原来 MVC 中的 V 层拿出来进行分离，视图的显示与如何显示交给 Template 层，而 View 层更专注于实现业务逻辑。其实在 Django 是有 Controller 层的，只不过它由框架本身来实现，所以我们不用关心它。Django 更关注于M、T 和 V。

在 Django 中，视图函数是一个 Python 函数或者类，开发者主要通过编写视图函数来实现业务逻辑。视图函数首先接受来自浏览器或者客户端的请求，并最终返回响应，视图函数返回的响应可以是 HTML 文件，也可以是 HTTP 协议中的 303 重定向。接下来编写一个简单的视图函数：

from django.http import HttpResponse

def Hello\_my\_django(request):

return HttpResponse('<html><body>Hello my Django</body></html>')

下面针对以上 3 行代码做解析：

1）HttpResponse视图响应类型

从 django.http 模块中导入 HttpResponse，从它简单的名字我们可以得知，它是一种视图的响应类型。

2）视图函数参数request

我们定义了一个名为“Hello\_my\_django”的函数，Django 规定了，视图函数至少有一个参数，第一个参数必须是 request，request 是 HttpRequest 请求类型的对象，它携带了浏览器的请求信息，所以视图函数的第一个参数必须为 request。

3）return视图响应

视图函数要返回响应内容，这里的响应内容是我们用 HTML 标签编写的，把它作为 HttpResponse 的对象返回给浏览器。

3.2.3Django的工作机制：

1.用manage .py runserver 启动Django服务器时就载入了在同一目录下的settings .py。该文件包含了项目中的配置信息，如前面讲的URLConf等，其中最重要的配置就是ROOT\_URLCONF，它告诉Django哪个Python模块应该用作本站的URLConf，默认的是urls .py

2.当访问url的时候，Django会根据ROOT\_URLCONF的设置来装载URLConf。

3.然后按顺序逐个匹配URLConf里的URLpatterns。如果找到则会调用相关联的视图函数，并把Http[Request对象](https://baike.baidu.com/item/Request%E5%AF%B9%E8%B1%A1/5357566" \t "_blank)作为第一个参数(通常是request)

4.最后该view函数负责返回一个Http[Response对象](https://baike.baidu.com/item/Response%E5%AF%B9%E8%B1%A1/7465134" \t "_blank)。

3.3统计分析工具

3.3.1基本统计量

统计量是统计理论中用来对数据进行分析、检验的变量。[宏观量](https://baike.baidu.com/item/%E5%AE%8F%E8%A7%82%E9%87%8F/2112968" \t "_blank)是大量[微观量](https://baike.baidu.com/item/%E5%BE%AE%E8%A7%82%E9%87%8F/2113002" \t "_blank)的统计[平均值](https://baike.baidu.com/item/%E5%B9%B3%E5%9D%87%E5%80%BC/8353298" \t "_blank)，具有统计平均的意义，对于单个微观粒子，宏观量是没有意义的．相对于微观量的统计平均性质的宏观量也叫统计量。需要指出的是，描写宏观世界的[物理量](https://baike.baidu.com/item/%E7%89%A9%E7%90%86%E9%87%8F/9984692" \t "_blank)例如速度、[动能](https://baike.baidu.com/item/%E5%8A%A8%E8%83%BD/1678605" \t "_blank)等实际上也可以说是宏观量，但宏观量并不都具有统计平均的性质，因而宏观量并不都是统计量。

数据经典统计学分析包括计算数据的极值、平均值、中位数、标准差、方差、变异系数、偏度与峰度等常用统计学指标。

3.3.1线性回归

线性回归是利用数理统计中回归分析，来确定两种或两种以上变量间相互依赖的定量关系的一种统计分析方法，运用十分广泛。其表达形式为y = w'x+e，e为误差服从均值为0的正态分布。

[回归分析](https://baike.baidu.com/item/%E5%9B%9E%E5%BD%92%E5%88%86%E6%9E%90/2625498" \t "_blank)中，只包括一个自变量和一个因变量，且二者的关系可用一条直线近似表示，这种回归分析称为[一元线性回归](https://baike.baidu.com/item/%E4%B8%80%E5%85%83%E7%BA%BF%E6%80%A7%E5%9B%9E%E5%BD%92" \t "_blank)分析。如果回归分析中包括两个或两个以上的自变量，且因变量和自变量之间是线性关系，则称为[多元线性回归](https://baike.baidu.com/item/%E5%A4%9A%E5%85%83%E7%BA%BF%E6%80%A7%E5%9B%9E%E5%BD%92/10702248" \t "_blank)分析。

3.3.2皮尔逊系数

皮尔逊相关也称为积差相关（或积矩相关）是英国统计学家皮尔逊于20世纪提出的一种计算直线相关的方法。

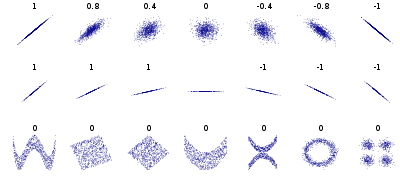
[](https://baike.baidu.com/pic/ç®å°éç¸å³ç³»æ°/12712835/0/6d81800a19d8bc3ec5831d39898ba61ea9d345bd?fr=lemma%26ct=single)

图3-2.皮尔逊相关系数举例

皮尔逊相关系数是比欧几里德距离更加复杂的可以判断人们兴趣的相似度的一种方法。该相关系数是判断两组数据与某一直线拟合程序的一种试题。它在数据不是很规范的时候，会倾向于给出更好的结果。

3.3.3qq图

分位数图示法（Quantile Quantile Plot，简称 Q-Q 图）

统计学里Q-Q图（Q代表分位数）是一个概率图，用图形的方式比较两个概率分布，把他们的两个分位数放在一起比较。首先选好分位数间隔。图上的点（x,y）反映出其中一个第二个分布（y坐标）的分位数和与之对应的第一分布（x坐标）的相同分位数。因此，这条线是一条以分位数间隔为参数的曲线。如果两个分布相似，则该Q-Q图趋近于落在y=x线上。如果两分布线性相关，则点在Q-Q图上趋近于落在一条直线上，但不一定在y=x线上。Q-Q图可以用来可在分布的位置-尺度范畴上可视化的评估参数。

从定义中可以看出Q-Q图主要用于检验数据分布的相似性，如果要利用Q-Q图来对数据进行正态分布的检验，则可以令x轴为正态分布的分位数，y轴为样本分位数，如果这两者构成的点分布在一条直线上，就证明样本数据与正态分布存在线性相关性，即服从正态分布。

3.3.4正态比例数据

标准正态分布又称为u分布，是以0为[均数](https://baike.baidu.com/item/%E5%9D%87%E6%95%B0" \t "_blank)、以1为[标准差](https://baike.baidu.com/item/%E6%A0%87%E5%87%86%E5%B7%AE" \t "_blank)的正态分布，记为N（0，1）。

标准正态分布曲线下面积分布规律是：在-1.96～+1.96范围内曲线下的面积等于0.9500，在-2.58～+2.58范围内曲线下面积为0.9900。统计学家还制定了一张统计用表（自由度为∞时），借助该表就可以估计出某些特殊u1和u2值范围内的曲线下面积。

3.3.5主成分分析

主成分分析（Principal Component Analysis，PCA）， 是一种[统计方法](https://baike.baidu.com/item/%E7%BB%9F%E8%AE%A1%E6%96%B9%E6%B3%95/10694433" \t "_blank)。通过正交变换将一组可能存在相关性的变量转换为一组线性不相关的变量，转换后的这组变量叫主成分。

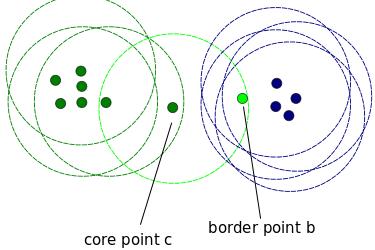
在实际课题中，为了全面分析问题，往往提出很多与此有关的变量（或因素），因为每个变量都在不同程度上反映这个课题的某些信息。

主成分分析首先是由K.皮尔森（Karl Pearson）对非[随机变量](https://baike.baidu.com/item/%E9%9A%8F%E6%9C%BA%E5%8F%98%E9%87%8F" \t "_blank)引入的，尔后H.霍特林将此方法推广到随机向量的情形。信息的大小通常用[离差平方和](https://baike.baidu.com/item/%E7%A6%BB%E5%B7%AE%E5%B9%B3%E6%96%B9%E5%92%8C/918673" \t "_blank)或[方差](https://baike.baidu.com/item/%E6%96%B9%E5%B7%AE/3108412" \t "_blank)来衡量。

3.3.6寻找异常点

DBScan是一种将数据分组的聚类算法。它也可以作为一种基于密度的异常检测方法，无论是单维数据还是多维数据。其他的聚类算法，如k-means 和hierarchal聚类也可以用来检测异常值。在本例中，我将向您展示一个使用DBScan的示例，但是在开始之前，让我们先介绍一些重要的概念。DBScan有三个重要概念：

核心点：为了理解核心点的概念，我们需要关注一些用于定义DBScan作业的超参数。第一个超参数（HP）是min\_samples。这只是组成集群所需的最小核心点数量。第二重要的超参数HP是eps。eps是两个样本被视为在同一个聚类之间的最大距离。边界点与核心点在同一个集群中，但距离集群中心远得多。



其他的一切都被称为噪声点，那些是不属于任何簇的数据点。它们可以是异常的或非异常的，需要进一步的研究。

3.3.7k-means聚类

算法流程：

1、首先确定一个k值，即我们希望将数据集经过聚类得到k个集合。

2、从数据集中随机选择k个数据点作为质心。

3、对数据集中每一个点，计算其与每一个质心的距离（如欧式距离），离哪个质心近，就划分到那个质心所属的集合。

4、把所有数据归好集合后，一共有k个集合。然后重新计算每个集合的质心。

5、如果新计算出来的质心和原来的质心之间的距离小于某一个设置的阈值（表示重新计算的质心的位置变化不大，趋于稳定，或者说收敛），我们可以认为聚类已经达到期望的结果，算法终止。

6、如果新质心和原质心距离变化很大，需要迭代3~5步骤。

四.测试

4.1测试目的

主要目的在于明确测试目标及针对于项目中的核心功能模块的测试思路，对系统进行架构分析并设计测试过程及实现方案，保证项目测试进度。测试包含单元测试，系统测试，性能测试和压力测试，分别检查功能的容错性，正确性，检查功能的性能情况以及并发下的处理能力，从而从多方面保证产品的平稳正常运行。

4.2测试需求分析

4.2.1测试策略与目标

测试重点：仓库模型、风险预测、定价系统以及电商平台基本功能。

测试难点：用户请求多种多样，合理划分等价类。

测试分层：将程序分为请求处理层，业务层，数据层等，分层进行测试，每层针对自己独有的情况设计等价类进行测试。

目标：覆盖95%以上的情况，90%以上的代码分支。

4.2.2测试范围

测试广度：覆盖全部核心功能模块。

测试深度：采用黑盒测试和白盒测试的方法，使用分支分析和等价类划分两种方案进行代码函数级覆盖。

4.2.3测试环境

硬件环境:

Intel® Core™ i5-6300U CPU @ 2.40GHz（开发环境）

软件环境:

Windows 10 专业版

Ubuntu 14.04.5 LTS , Python 3.5.5 (Travis CI)

测试工具：

LoadRunner12、Selenium、Junit4

4.3单元测试

举例

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 用例编号 | 001 | | | |
| 测试单元描述 | 下载 | | | |
| 用例目的 | 检查用户能否下载 | | | |
| 前提条件 | 已经登录主界面 | | | |
| 特殊的规程说明 | 无 | | | |
| 用例间的依赖关系 | 无 | | | |
| 步骤 | 输入 | 期望输出 | 实际输出 | 备注 |
| 1 | 下载内容 | 下载成功 | 下载成功 | 已事先添加到数据库 |

4.4功能测试

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 用例编号 | 058 | | | |
| 测试单元描述 | 回归分析 | | | |
| 用例目的 | 完成基本的回归分析 | | | |
| 前提条件 | 用户选择数据 | | | |
| 特殊的规程说明 | 无 | | | |
| 用例间的依赖关系 | 无 | | | |
| 步骤 | 动作 | 期望输出 | 实际输出 | 备注 |
| 1 | 分析 | 分析结果 | 分析结果 |  |

4.5系统测试

4.5.1模型性能测试

举例：

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 用例编号 | 063 | | | |
| 性能描述 | 主成分分析性能 | | | |
| 用例目的 | 判断主成分分析性能 | | | |
| 前提条件 | 无 | | | |
| 特殊的规程说明 | 无 | | | |
| 用例间的依赖关系 | 无 | | | |
| 步骤 | 输入/动作 | 期望的性能  (平均值) | 实际性能  (平均值) | 备注 |
| 1 | 主成分分析 | 1s | 1.272s | 测试1000次 |

结果：

模型的性能测试结果符合项目的最低要求，所以我们认为该模型可用。

但是模型的不稳定性依旧存在，需要继续完善模型，提高准确率与响应速度。

4.5.2请求性能测试

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 用例编号 | 067 | | | |
| 性能描述 | 请求性能测试 | | | |
| 用例目的 | 测试请求响应速度 | | | |
| 前提条件 | 无 | | | |
| 特殊的规程说明 | 关闭调试模式 | | | |
| 用例间的依赖关系 | 无 | | | |
| 步骤 | 输入/动作 | 期望的性能(平均值) | 实际性能  (平均值) | 备注 |
| 1 | 无载荷请求 | 0.005s | 0.003s | 无并发, 测试1000次 |
| 2 | 500用户载荷请求 | 1.500s | 1.946s | 1000用户并发，测试1000次 |

测试结果综合分析及建议

在简单内容请求下，平台性能符合预期，对项目整体提供了好的基础条件。由于项目初期采用阿里云服务器，载荷能力较弱，所以项目响应速度下降，在今后可以通过硬件设施的增强从而提高响应速度。

4.5.3压力测试

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 功能 | 分析 | | | | |
| 目的 | 检测分析模块压力 | | | | |
| 方法 | 每一项测试1000次，取3轮测试平均值 | | | | |
| 并发用户数与事务执行情况 | | | | | |
| 并发用户数 | 事务平均响应时间 | 事务最大响应时间 | 平均每秒处理事务数 | 事务成功率 | 平均流量(字节/秒) |
| 1 | 0.003s | 0.009s | 331.17 | 100% | 53.69K |
| 25 | 0.078s | 0.096s | 317.83 | 100% | 51.52K |
| 100 | 0.315s | 0.355s | 316.66 | 100% | 51.33K |
| 500 | 1.946s | 2.178s | 256.90 | 100% | 41.65K |

分析结果：性能符合低并发量下的需求，在测试环境的硬件条件下，超过500并发时会出现明显的性能下降。

测试结果综合分析及建议

并发性能和压力测试是系统的关键测试部分，在本次性能测试中，系统没有出现崩溃的情况，说明系统在低并发下的可靠性符合需求，但性能会出现下降。未来应考虑使用更高性能的测试平台重复压力测试，以检验分布式的使用必要。

4.5.4测试总结

经过单元测试、功能测试以及系统测试，我们可以初步断定，项目的基本功能与非功能性要求大部分已实现，模型的准确率与响应速度也满足项目的需求。但项目依旧存在这可以改进的地方，例如模型准确率的提高，系统并发措施的完善，个性化服务的完善等环节。在今后的项目开发和维护中，这些方面应该被重点考虑。

五.数据备份方案

5.1备份

平台正式上线后主要包括四个部分：硬件设备，软件环境，应用系统和数据。下面从这四个方面介绍下系统的数据备份方案。

（1）硬件设备

对于硬件设备故障，并非简单的加强管理就可以避免的，往往必须增加投资，进行硬件设备的冗余，以增强系统的高可用性。columba使用多机集群，存储设备采用 RAID。

（2）软件环境

软件的故障一般是由于人为操作因素引起的，影响应用系统的正常运行，对于该类故障，有两种解决方式：

启动冗余备份系统，即将当前生产机离线，将备份机启动，再逐步恢复原系统机器的软件环境。

从软件备份中恢复系统，在这种情况下，要求对原系统的操作系统、数据库系统等已经定时做好完整的备份这样可以是恢复成为简单机械、按部就班的工作。

（3）应用系统

应用系统的故障大多也是由于人为操作引起的。应用系统的备份和恢复和软件环境的备份恢复类似。

（4）数据

数据是本系统的关键，为了系统的完整性考虑，在条件允许的情况下，将所有系统数据全部备份。同时进行分级备份，视可重复性、重要性、数据量、处理的时间和难度据顶备份的级别。数据备份定期进行，所有数据每周备份一次，重要数据每天备份一次。

5.2恢复

备份的目的是在出现故障的情况下能尽快的恢复系统，为保证能快速、准确地正常恢复，我们应该明确地定义恢复的策略。根据不同的故障，我们有以下的恢复策略。

5.2.1 全崩溃恢复机制

因为本平台设置了远程灾难备份中心，也有一定的冗余备份，出现此类崩溃需要启动冗余机器，然后迅速修复原有机器，就可以将系统恢复到灾难发生前的状态。恢复步骤如下：

（1）启用备份机器和系统；

（2）修复服务器硬件设备；

（3）修复网络硬件设备；

（4）从系统备份磁盘中恢复服务器软件环境；

（5）从应用系统备份磁盘中恢复应用系统；

（6）从冗余机器中恢复数据；

（7）按照需要恢复应用系统的日志。

5.2.2 服务器崩溃恢复机制

服务器出现崩溃，此时因为数据存储在磁盘阵列中，并没有受到影响，只要快速恢复服务器上的操作系统、数据库、应用系统即可。

（1）修复服务器硬件设备；

（2）从系统备份磁盘中恢复服务器操作系统、数据库系统；

（3）从应用系统备份磁盘中恢复应用系统；

（4）按照需要恢复应用系统的日志。

5.2.3 磁盘阵列崩溃恢复机制

磁盘阵列崩溃，数据将全部丢失，恢复的对象主要是数据。恢复步骤如下：

（1）修复磁盘阵列；

（2）从最近的全备份磁盘恢复到全备份时数据状态；

（3）按照全备份后进行的增量备份的次序逐个恢复。

5.2.4 系统软件全崩溃恢复机制

软件系统全崩溃的情况下，由于数据存储在磁盘阵列中，可能受到损失，也可能为受到损失。恢复步骤如下：

（1）从系统备份磁盘中恢复服务器操作系统、数据库系统；

（2）从应用系统备份磁盘中恢复应用系统；

（3）进行数据库检查，如果数据未损坏，则恢复已完成。否则执行下述步骤：

（4）从最近的全备份磁盘恢复到全备份时数据状态；

（5）按照全备份后进行的增量备份的次序逐个恢复增量备份；

（6）按照需要恢复应用系统的日志。

5.2.5 操作系统崩溃恢复机制

操作系统崩溃的情况下，由于数据存储在磁盘阵列中，可能受到损失，也可能为受到损失。恢复步骤如下：

（1）从系统备份磁盘中恢复服务器操作系统；

（2）从应用系统备份磁盘中恢复应用系统；

（3）进行数据库检查，如果数据未损坏，此时恢复已完成，否则执行以下步骤：

（4）最近的全备份磁盘灰度到全备份时数据状态；

（5）按照全备份后进行的增量备份的次序逐个恢复增量备份；

（6）按照需要恢复应用系统的日志。

5.2.6 数据库系统崩溃恢复机制

数据库系统崩溃的情况下，由于数据存储在磁盘阵列中，可能受到损失，也可能未受到损失。恢复步骤如下：

（1）从系统备份磁盘中恢复数据库系统；

（2）进行数据库检查，如果数据未损坏，此时恢复已完成，否则执行以下步骤；

（3）从最近的全备份磁盘灰度到全备份时数据状态；

（4）按照全备份后进行的增量备份的次序逐个恢复增量备份；

（5）按照需要恢复应用系统的日志。

六.项目管理

6.1项目组织

6.1.1团队结构

每一个成员可以根据自己能力的突出点，去承担一部分工作。在每一个责任块中，模糊上下级的概念。每个人都有责任义务去完成这项工作。在整个工作开展过程中，坚持短期，中期，长期的组内交流会。确保贯穿项目始终都能有新思想涌入。团队的结构分布如下图 5-1 所示。

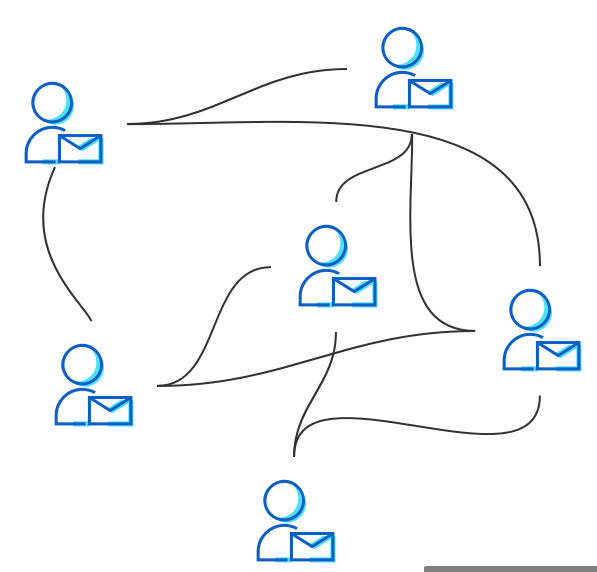


图6-1 团队结构

6.1.2成员分工

团队的成员责任范围（如表6-1）。

表6-1 成员分工

|  |  |  |
| --- | --- | --- |
| **责任范围** | **成员名称** | **主要职责** |
| **项目整体** | 成员1 | 负责项目的管理、进度跟踪和人力资源调度 |
| 组织项目组成员对项目规模、工作量、工期进行估计，制定相应的执行计划，定义项目目标 |
| 对组织资源进行动态的管理，按照项目计划组织项目验收 |
| 建立各种专业管理体系,并组织实施 |
| 组织小组讨论会议、评审会议等 |
| **技术方面** | 成员2 | 负责项目的整体架构和技术路线的制定 |
| 负责团队成员技术培训 |
| 组织团队成员进行编码 |
| 分析涉及技术的可行性 |
| **产品方面** | 成员3 | 负责项目的可行性分析以及用户体验的设计 |
| 系统的界面设计 |
| 负责与客户进行沟通 |
| **业务方面** | 成员4 | 负责项目的业务需求分析 |
| 确定领域边界和领域内容 |
| **测试方面** | 成员5 | 制定系统的测试计划 |
| 编写测试报告 |

6.2 需求管理

6.2.1 需求追踪

在需求的跟踪过程中，建立与维护需求跟踪矩阵，需求跟踪矩阵（如表6-2）。

表6-2 项目需求追踪矩阵（片段）

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **序号** | **需求规格说明书**  **（版本，日期）** | **设计文档**  **（版本，日期）** | **代码**  **（版本，日期）** | **测试用例**  **（版本，日期）** |
| **1** | （V0.1,2021-01-15） | （V0.1,2021-01-15） | （V0.1,2021-01-15） | （V0.1,2021-01-15） |
| **2** | （V0.2,2021-01-31） | （V0.2,2021-01-31） | （V0.2,2021-01-31） | （V0.2,2021-01-31） |
| **3** | （V0.3,2021-02-12） | （V0.3,2021-02-12） | （V0.3,2021-02-12） | （V0.3,2021-02-12） |
| **4** | （V0.4,2021-02-23） | （V0.4,2021-02-23） | （V0.4,2021-02-23） | （V0.4,2021-02-23） |
| **5** | （V0.5,2021-03-10） | （V0.5,2021-03-10） | （V0.5,2021-03-10） | （V0.5,2021-03-10） |
| **6** | （V0.6,2021-03-18） | （V0.6,2021-03-18） | （V0.6,2021-03-18） | （V0.6,2021-03-18） |
| **7** | （V0.7,2021-03-22） | （V0.7,2021-03-22） | （V0.7,2021-03-22） | （V0.7,2021-03-22） |
| **8** | （V0.8,2021-03-31） | （V0.8,2021-03-31） | （V0.8,2021-03-31） | （V0.8,2021-03-31） |

6.2.2 需求变更

团队在综合考虑之后决定制定相应的需求变更的防范管理策略，在开发过程当中，建立了需求变更追踪矩阵，需求变更追踪矩阵如表6-3所示。

表 6-3 项目需求变更记录文档（实例）

文档信息

|  |  |  |  |
| --- | --- | --- | --- |
| 编写者 | 成员2 | 编写日期 | 2021-01-18 |
| 审核者 | 成员3 | 审核日期 | 2021-02-14 |
| 批准者 | 成员1 | 批准日期 | 2021-02-16 |

变更记录

|  |  |  |  |
| --- | --- | --- | --- |
| 日期 | 作者 | 版本 | 变更说明 |
| 2021-01-18 | 成员2 | V0.2 | 商品展示模块增加智能推荐系统 |
| 2021-01-20 | 成员3 | V0.3 | 优化合同交易模块 |
| 2021-01-22 | 成员4 | V0.4 | 设置多维度的风险管理体系 |
| 2021-01-25 | 成员5 | V0.5 | 增加智能决策系统 |

变更审阅

|  |  |  |
| --- | --- | --- |
| 日期 | 审阅者 | 意见 |
| 2021-01-18 | 成员1 | 同意变更 |
| 2021-01-20 | 成员1 | 同意变更 |
| 2021-01-22 | 成员1 | 同意变更 |
| 2021-01-25 | 成员1 | 同意变更 |

6.3 范围管理

6.3.1 工作分解结构

根据业务需求，按产品范围制作下表，可以确定本系统的产品范围。围绕项目阶段设计，确定项目范围（如图6-3）。



图6-3 WBS工作分解结构图

6.4 时间管理

本系统的详细进度安排（如表6-4）。

表6-4 项目进度安排

|  |  |  |
| --- | --- | --- |
| 阶段 | 起止时间 | 任务安排 |
| 项目启动阶段 | 2021/1/15-2021/1/22 | 提出设想，编写《项目计划书》 |
| 需求分析阶段 | 2021/1/22-2021/1/25 | 编写《系统需求分析说明书》和《系统概要设计说明书》， |
| 设计阶段 | 2021/1/25-2021/2/14 | 编写《系统详细设计说明书》，《项目详细实施计划》 |
| 系统集成 | 2021/2/14-2021/3/21 | 进行系统集成，同时整理出第一版的资料 |
| 第一期验收 | 2021/3/21-2021/3/27 | 上交初期成品 |
| 试运行阶段 | 2021/3/27-2021/4/4 | 系统试运行并进行相应调整 |
| 第二期验收 | 2021/4/4-2021/4/10 | 完成第二期验收 |
| 第三期设计实现 | 2021/4/10-2021/4/20 | 完成第三期验收 |
| 试运行阶段 | 2021/4/2-2021/4/28 | 系统试运行并进行相应调整 |
|  | 2021/4/28-2021/5/2 | 完成最终验收 |

6.5 成本管理

软件项目成本管理的内容包括成本估算、成本预算、成本控制。

现实中，软件项目经常成本超支，这是因为：项目需求含糊，经常会由于客户不断变化的实际要求而变更计划；项目成本结构复杂，成本核算方法和实施难度大；成本的估算不合理，行业标准不明确，尤其是间接成本的估算没有标准成熟的方法和科学依据；项目涉及新的技术或商业过程，有很大的内在风险。

结合IT项目的成本特点，应用恰当的项目成本管理技术和方法可以很有效地改变成本超支状况。

6.5.1成本构成

|  |  |
| --- | --- |
| 成本名称 | 具体内容 |
| 软硬件购置成本 | 这部分费用虽然可以作为固定资产，但因技术折旧太快，需要在项目开发中分摊一部分费用。 |
| 人工成本（软件开发、系统集成费用） | 主要是指开发人员、操作人员、管理人员的工资福利费等。在软件项目中人工费用总是占有相当大的份额，有的可以占到项目总成本的80％以上。 |
| 维护成本 | 维护成本是在项目交付使用之后，承诺给客户的后续服务所必需的开支。可以说，软件业属于服务行业，其项目的后期服务是项目必不可少的重要实施内容。所以，维护成本在项目生命周期成本中，占有相当大的比例。 |
| 培训费 | 培训费是项目完毕后对使用方进行具体操作的培训所花的费用。 |
| 管理及服务费 | 这部分费用是指项目应分摊管理层、财务等人员的费用。 |
| 其他费用 | 包括：基本建设费用，如新建、扩建机房、购置计算机机台、机柜等的费用；材料费，如打印纸、磁盘等购置费；水、电、气费；资料、固定资产折旧费及咨询费等等。 |
| 直接成本 | 与具体项目的开发直接相关的成本。如人员的工资、外包外购成本等。又可细分为开发成本、管理成本、质量成本等。 |
| 间接成本 | 不归属于一个具体的项目，是运营成本，分摊到各个项目中。如房租、水电、保安、税收、福利，等等。 |

6.5.2项目成本跟踪管理

在软件开发项目实施过程中，人力不足现象是经常发生的问题，确切的说是满足项目所需技能的人力不足现象。以人力不足为风险来管控，深层含义是项目成本管理。

项目成本跟踪表如下显示：

表6-5 项目成本跟踪表

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 更新日期 | 项目阶段 | 目前累计（人时） | | | 本周统计值 | | | |
| AC | PV | EV | SV% | CV | SPI | CPI |
| 第1周 | 项目策划 |  |  |  |  |  |  |  |
| 第2周 | 项目策划 |  |  |  |  |  |  |  |
| 第3周 | 需求开发 |  |  |  |  |  |  |  |
| 第4周 | 需求开发 |  |  |  |  |  |  |  |
| 第5周 | 需求开发 |  |  |  |  |  |  |  |
| 第6周 | 软件设计 |  |  |  |  |  |  |  |
| 第7周 | 软件设计 |  |  |  |  |  |  |  |
| 第8周 | 编码测试 |  |  |  |  |  |  |  |
| 第9周 | 编码测试 |  |  |  |  |  |  |  |
| 第10周 | 编码测试 |  |  |  |  |  |  |  |
| 第11周 | 编码测试 |  |  |  |  |  |  |  |
| 第12周 | 编码测试 |  |  |  |  |  |  |  |

PV Planned Value：计划值，应该完成多少工作, （截止目前应该花费的预算）

AC Actual Cost：实际成本，完成工作的实际成本是多少 （截止目前实际的花费）

EV Earned Value：挣值，完成了多少预算的工作 （实际完成的工作，按照预算标准应该有的花费）以上三个指标是成本管理最基本的三个概念；

CV% = CV/EV = （EV-AC）/EV = 1- 1/CPI，CV 成本差异，CV = EV–AC CV>0代表成本节约；

SV% = SV/PV = （EV-PV）/PV= SPI-1，SV 进度差异 ，SV = EV–PV SV>0代表进度提前；

CPI 成本绩效指数 ，CPI = EV/AC CPI>1代表工作价值高；

SPI 进度绩效指数 ，SPI = EV/PV SPI>1代表实际进度快。

6.5.3项目规模度量

软件的规模是影响软件项目成本和工作量的主要因素。准备采用功能点（Function Point，FP）来根据功能点数来衡量软件系统的规模。根据技术复杂度因子（权）对功能点进行调整，产生产品规模的度量结果。

计算公式：FP =UFC\*TCF

UFC（Unadjusted Function Point Count）未调整功能点计数

TCF（Technical Complexity Factor）技术复杂度因子

表6-6计数项的复杂度权重

|  |  |  |  |
| --- | --- | --- | --- |
| 功能计数项 | 复杂度权重 | | |
| 简单 | 中等 | 复杂 |
| 外部输入 | 3 | 4 | 6 |
| 外部输出 | 4 | 5 | 7 |
| 外部查询 | 3 | 4 | 6 |
| 外部接口文件 | 5 | 7 | 10 |
| 内部文件 | 7 | 10 | 15 |

6.6 风险管理

项目风险是指在软件开发过程中遇到的预算和进度等方面的问题以及这些问题对软件项目的影响。软件项目风险会影响项目计划的实现，如果项目风险变成现实，就有可能影响项目的进度，增加项目的成本，甚至使软件项目不能实现。

风险管理涉及的主要过程包括：风险识别，风险量化，风险应对计划制定和风险监控。风险识别在项目的开始时就要进行，并在项目执行中不断进行。就是说，在项目的整个生命周期内，风险识别是一个连续的过程。

PMI项目风险管理过程如图所示：

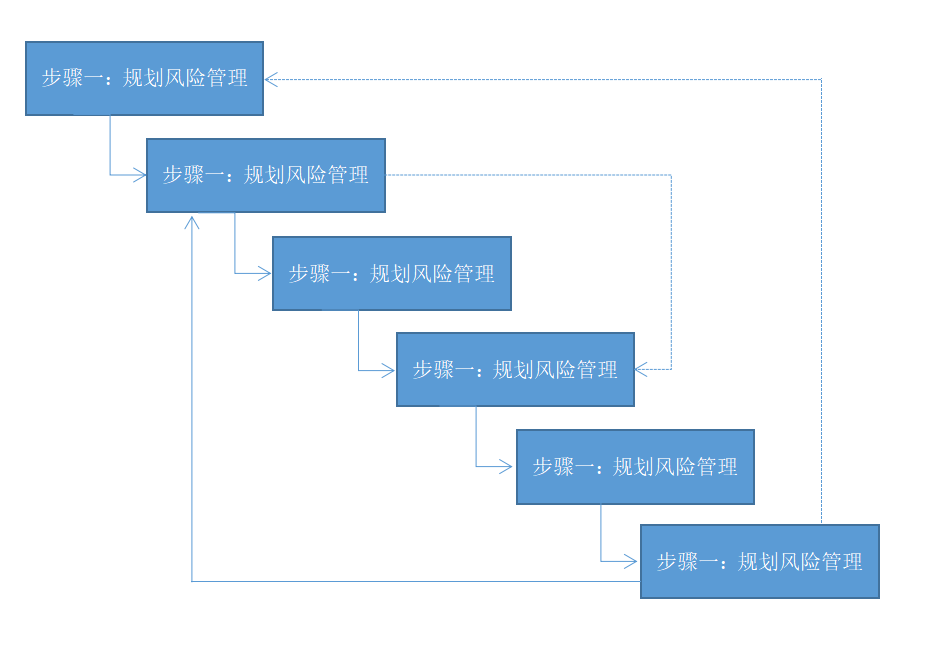


图6-4 项目风险管理过程

6.6.1风险识别

软件项目的风险无非体现在以下四个方面：需求、技术、成本和进度。所以对这四方面进行控制，通过头脑风暴，Delphi法进行风险识别。

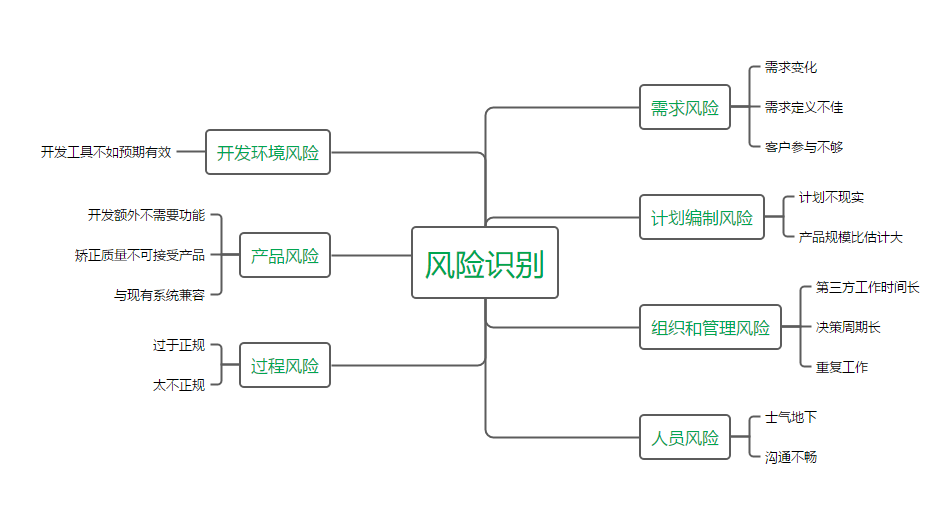


图6-5 各知识领域相关的可能风险

6.6.2风险量化

针对本系统潜在的商业风险、管理风险和技术风险团队估计各风险发生的概率、估计它们对项目的影响以及发现降低各风险可采取的策略。其中风险量化可以采用风险因子计算，PERT估计等方法进行量化。

风险发生概率分析如表6-7所示：

表6-7 风险发生概率分析表

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 风险 | | 概率 | 容忍度 | 可预测性 |
| 需求风险 | 需求已经成为项目基准，但需求还在继续变化 | 1% | 中 | 4 |
| 需求定义欠佳，而进一步的定义会扩展项目范畴 | 4% | 高 | 2 |
| 产品定义含混的部分比预期需要更多的时间 | <1% | 底 | 7 |
| 在做需求中客户参与不够 | 2% | 中 | 8 |
| 缺少有效的需求变化管理过程 | <1% | 高 | 6 |
| 计划编制风险 | 计划、资源和产品定义全凭客户或上层领导口头指令，并且不完全一致 | 8% | 底 | 4 |
| 计划是优化的，是"最佳状态"，但计划不现实，只能算是"期望状态" | 8% | 中 | 5 |
| 计划基于使用特定的小组成员，而那个特定的小组成员其实指望不上 | 4% | 高 | 7 |
| 产品规模（代码行数、功能点、与前一产品规模的百分比）比估计的要大 | 10% | 中 | 3 |
| 完成目标日期提前，但没有相应地调整产品范围或可用资源 | 2% | 高 | 2 |
| 涉足不熟悉的产品领域，花费在设计和实现上的时间比预期的要多 | 1% | 高 |  |
| 组织和管理风险 | 仅由管理层或市场人员进行技术决策，导致计划进度缓慢，计划时间延长 | 15% | 中 | 5 |
| 低效的项目组结构降低生产率 | 2% | 高 | 4 |
| 管理层审查 决策的周期比预期的时间长 | 15% | 中 | 3 |
| 预算削减，打乱项目计划 | 26% | 高 | 5 |
| 管理层作出了打击项目组织积极性的决定 | 5% | 底 | 6 |
| 缺乏必要的规范，导致工作失误与重复工作 | 5% | 中 | 7 |
| 非技术的第三方的工作（预算批准、设备采购批准、法律方面的审查、安全保证等）时间比预期的延长。 | 7% | 高 | 2 |
| 人员风险 | 作为先决条件的任务（如培训及其他项目）不能按时完成 | 10% | 底 | 3 |
| 开发人员和管理层之间关系不佳，导致决策缓慢，影响全局 | 15% | 中 | 4 |
| 缺乏激励措施，士气低下，降低了生产能力 | 4% | 高 | 5 |
| 某些人员需要更多的时间适应还不熟悉的软件工具和环境 | 3% | 中 | 7 |
| 项目后期加入新的开发人员，需进行培训并逐渐与现有成员沟通，从而使现有成员的工作效率降低 | 4% | 高 | 8 |
| 由于项目组成员之间发生冲突，导致沟通不畅、设计欠佳、接口出现错误和额外的重复工作 | 12% | 底 | 6 |
| 没有找到项目急需的具有特定技能的人 | 5% | 中 | 4 |
| 开发环境风险 | 开发工具不如期望的那样有效，开发人员需要时间创建工作环境或者切换新的工具 | 8% | 高 | 5 |
| 新的开发工具的学习期比预期的长，内容繁多 | 6% | 底 | 7 |
| 客户风险 | 客户对规划、原型和规格的审核 决策周期比预期的要长 | 5% | 中 | 3 |
| 客户没有或不能参与规划、原型和规格阶段的审核，导致需求不稳定和产品生产周期的变更 | 7% | 高 | 2 |
| 过程风险 | 前期的质量保证行为不真实，导致后期的重复工作 | 15% | 中 | 6 |
| 太不正规（缺乏对软件开发策略和标准的遵循），导致沟通不足，质量欠佳，甚至需重新开发 | 2% | 高 | 5 |
| 过于正规（教条地坚持软件开发策略和标准），导致过多耗时于无用的工作 | 15% | 底 | 4 |
| 撰写进程报告占用开发人员的时间比预期的多 | 26% | 中 | 3 |
| 风险管理粗心，导致未能发现重大的项目风险 | 5% | 高 | 5 |
| 设计和实现风险 | 计质量低下，导致重复设计 | 5% | 中 | 6 |
| 代码和库质量低下，导致需要进行额外的测试，修正错误，或重新制作 | 8% | 高 | 7 |
| 过高估计了增强型工具对计划进度的节省量 | 4% | 中 | 2 |
| 代码和库质量低下，导致需要进行额外的测试，修正错误，或重新制作 | 10% | 高 | 3 |
| 产品风险 | 矫正质量低下的不可接受的产品，需要比预期更多的测试、设计和实现工作 | 2% | 底 | 4 |
| 开发一种全新的模块将比预期花费更长的时间 | 5% | 中 | 5 |

6.6.3风险应对计划

对不同的风险采用不同的应对方法，确保项目的可靠进行，并且保证处理风险带来的收益小于为应对这些风险带来的成本。

表6-8风险参照应对表

|  |  |  |
| --- | --- | --- |
| **重要性**  **可能性** | 影响较小 | 影响较大 |
| 可能性较小 | 对这类风险可以忽视，并选择接受 | 对这类风险，尽量做到转移与规避 |
| 可能性较大 | 对这类风险尽量减小 | 对这类风险要额外关注 |

6.6.4风险监控

涉及整个项目管理过程中的风险进行应对。该过程的输出包括应对风险的纠正措施以及风险管理计划的更新。这其中包括核对表，定期项目评估，挣值分析。采用如表6-9所示：

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 序号 | 风险事件 | 成因 | 初始风险 | | | 风险处理措施 | 残留风险 | | | 残留风险处理措施 | 风险处理负责人 | 填写日期 | 备注 |
| 概率 | 后果 | 等级 | 概率 | 后果 | 等级 |
| 1 | 需求变动 | 客户提出新的需求 | 5 | 4 | 极高 | 更改计划，重新进行系统设计 | 2 | 2 | 中度 | 缩短其余任务的时间，增加人力资源。 | 成员1 | 2021.01.24 | 无 |
|  |  |
|  |  |

表6-9 风险核对表

6.7质量管理

软件质量就是软件与用户需求相一致的程度。具体地说，软件质量是软件符合明确叙述的功能和性能需求、以及所有专业开发的软件都应具有的隐含特征的程度。

软件的质量形成于软件的整个开发过程中，所以在产品的生产过程中，设置稳定过程的上下控制界，防止质量问题的出现。

在这个过程中确保质量成本，其中包括预防成本，评价成本，失效成本。低于质量管理带来的收益。

对软件质量方面着重关注的方面采用 McCall软件质量模型，如下图所示：

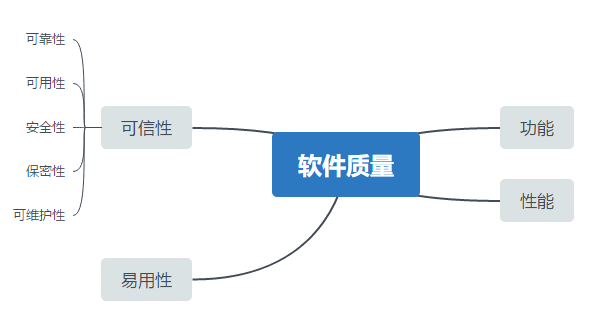


图6-6 McCall模型

6.7.1PAF质量成本模型

表6-10 PAF质量成本模型

|  |  |  |
| --- | --- | --- |
| 预防 | 评价 | 失效 |
| 培训 | 审查 | Regression Test |
| 计划 | 测试 | 服务 |
| 过程研究和改进 | 监控 | 修复 |
| 咨询 | 度量 | 缺陷分析 |
| 获得资格 | 验证 | 投诉处理解决 |
|  | 分析 |  |
|  | 确认 |  |

6.7.2技术评审管理

就是对工作成果进行审查和分析，发现其中的缺陷，并帮助开发人员及时消除缺陷。

技术评审的主要对象：需求和设计规格说明、代码、测试计划、用户手册等。

技术评审分为正式技术评审和非正式技术评审两种基本类型，前者比较严格，需要举行评审会议，参加人员比较多，后者的形式比较灵活，通常在同伴之间开展，不必举行评审会议，参与人员相对较少。

一般来说，对重要性和复杂性较高的工作成果，应进行正式技术评审，对重要性和复杂性相对较低的工作成果，可进行非正式技术评审。

技术评审管理流程如下表所示

表6-11 技术评审管理流程

|  |  |
| --- | --- |
| 关键人员 | 主要内容 |
| 评审组长 | 评审组长把待评审的材料分发给每个评审者，评审者（包括评审组长）审查材料，记下相关的要点，为评审会议做准备。 |
| 评审组长 | 开评审会议。评审会议由评审组长、评审者、评审对象的开发者参加。其中的一个评审者充当记录员，负责记录会议中发现的所有问题。 |
| 开发小组 | 由开发小组对提交的评审对象进行讲解。同时评审者可对开发者提问，提出建议和要求，展开讨论。  在讨论中如果发现了问题和错误，由记录员记录下来。 |
| 参会人员 | 决定作出后，所有参加会议的人员签字，确认会议结果。 |
| 评审组长 | 技术评审会议后，要完成一个“评审总结报告”，其内容包括：评审对象是什么？谁参加了评审？评审的结论是什么？有哪些重要发现？ |
| 开发者 | 跟踪与审核。开发者修改工作成果，消除已发现的缺陷。由指定的审查人员跟踪每个缺陷的状态，直到工作成果合格为止。 |

技术评审注意事项

评审产品，而不是评审人。评审会议的气氛要轻松和愉快，注意提出问题时的方式和态度，不要让产品开发者产生被审问的感受。

制订评审会议的议程并遵守进度。不要让会议过分拖延。问题的具体解决方案可以在会后讨论。

使用检查清单。为不同的软件产品（需求、设计、代码等）开发检查清单，在检查清单中列出所有重要的、常见的问题，这样可以使评审会议聚焦于一些重要问题。

6.7.3软件测试管理

软件测试是通过执行软件来发现缺陷，它是控制软件质量的重要手段和关键活动。

测试进度安排，如表6-12所示。

表6-12 测试进度安排

|  |  |  |
| --- | --- | --- |
| 测试活动 | 计划开始日期 | 结束日期 |
| 制定测试计划 | 2021-02-01 | 2021-02-12 |
| 设计用例 | 2021-02-12 | 2021-03-11 |
| 实施测试 | 2021-03-11 | 2021-03-23 |
| 对测试进行评估 | 2021-03-23 | 2021-03-24 |
| 完成测试文档 | 2021-03-24 | 2021-03-26 |

（2）测试环境，如下表所示。

表6-13 测试环境

人力资源

|  |  |  |
| --- | --- | --- |
| 角色 | 姓名 | 具体职责或注释 |
| 测试项目  负责人 | 成员2 | 管理监督测试项目，确保与相关开发人员的交互，为测试提供一个稳定的基础环境。 |
| 测试  分析员 | 成员3，  成员4 | 确定测试计划、测试内容、测试方法、测试数据生成方法、测试（软、硬件）环境、测试工具，评估测试工作的有效性 |
| 测试  设计员 | 成员1 | 负责网络方面的测试，需要维护云，本地和云的混合网络。 |
| 测试员 | 成员5 | 执行测试、记录测试结果 |

（3）编写并完成功能测试用例，如下表所示。

表6-13功能测试用例

|  |  |  |  |
| --- | --- | --- | --- |
| 项目名称 |  | 程序版本 | V0.1 |
| 测试目的 |  | | |
| 功能特性 |  | | |
| 预置条件 |  | | |
| 测试环境 | 软件环境：PC系统 | | |
| 参考信息 |  | 特殊规程 |  |
| 设计人 | 成员2 | 设计时间 | 1天 |
| 测试记录人 | 成员1 | 测试时间 | 1天 |

6.7.4缺陷跟踪

缺陷跟踪是指从缺陷被发现开始到被改正为止的整个跟踪流程。

准备使用Bugzilla软件辅助进行缺陷跟踪。

以下是缺陷跟踪的流程图。

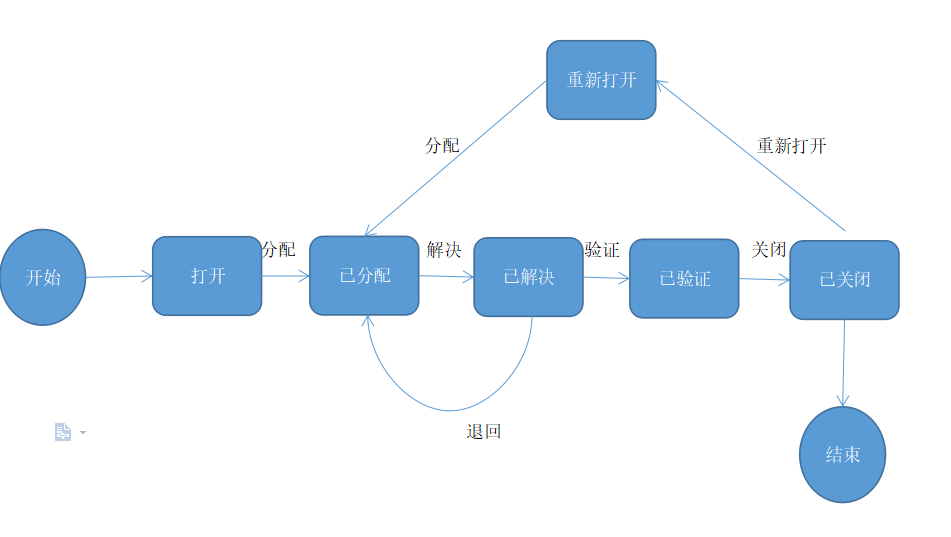


图6-7 缺陷跟踪流程图

6.8人力资源管理

项目的实施需要多种资源，包括人力、自然、资本和信息资源，其中人力资源是最基本、最重要的资源。人力资源管理是指根据[企业发展战略](https://baike.baidu.com/item/%E4%BC%81%E4%B8%9A%E5%8F%91%E5%B1%95%E6%88%98%E7%95%A5" \t "https://baike.baidu.com/item/%E4%BA%BA%E5%8A%9B%E8%B5%84%E6%BA%90%E7%AE%A1%E7%90%86/_blank)的要求，有计划地对人力资源进行合理[配置](https://baike.baidu.com/item/%E9%85%8D%E7%BD%AE" \t "https://baike.baidu.com/item/%E4%BA%BA%E5%8A%9B%E8%B5%84%E6%BA%90%E7%AE%A1%E7%90%86/_blank)，通过对项目中成员的[培训](https://baike.baidu.com/item/%E5%9F%B9%E8%AE%AD" \t "https://baike.baidu.com/item/%E4%BA%BA%E5%8A%9B%E8%B5%84%E6%BA%90%E7%AE%A1%E7%90%86/_blank)、使用、[激励](https://baike.baidu.com/item/%E6%BF%80%E5%8A%B1" \t "https://baike.baidu.com/item/%E4%BA%BA%E5%8A%9B%E8%B5%84%E6%BA%90%E7%AE%A1%E7%90%86/_blank)、调整等一系列过程，调动成员的积极性，发挥成员的[潜能](https://baike.baidu.com/item/%E6%BD%9C%E8%83%BD" \t "https://baike.baidu.com/item/%E4%BA%BA%E5%8A%9B%E8%B5%84%E6%BA%90%E7%AE%A1%E7%90%86/_blank)。

6.8.1责任分配矩阵

责任分配矩阵如下表所示：





表6-14 责任分配矩阵

6.8.2各阶段人员配置计划

各阶段人员配置计划如图6-8所示：

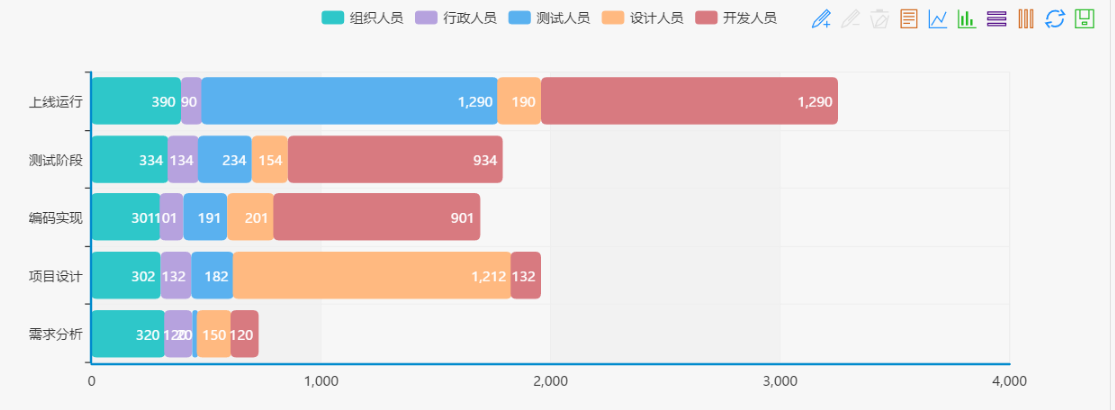


图 6-8 各阶段人员配置计划图

6.9配置管理

软件配置管理是一种标识、组织和控制修改的技术。软件配置管理应用于整个软件工程过程。我们知道，在软件建立时变更是不可避免的，而变更加剧了项目中软件开发者之间的混乱。软件配置管理活动的目标就是为了标识变更、控制变更、确保变更正确实现并向其他有关人员报告变更。从某种角度讲，软件配置管理的目的是使错误降为最小并最有效地提高生产效率。

6.9.1 环境和工具

在该项目中将采用GIT托管服务进行软件配置管理，不需要自己搭建服务器，方便项目的启动；不需要自己管理和维护服务器，有利于项目的顺利开发。

6.9.2 配置的检查和评审

表6-16 审核内容

|  |  |  |
| --- | --- | --- |
| 审核分类 | 审核内容 | 检查情况 |
| 发布审核 | 发布文档是否清楚地定义发布的范围，包括应被纳入的更改请求？ |  |
| 所有已知缺陷/毛病（bug）是否已文档化？ |  |
| 是否有适当的文档，它标识重建该发布所需的环境（编译器版本、OS版本、compilation flags，等等）？ |  |
| 是否有适当的文档，它说明构成该发布的成分及成分的版本？ |  |
| 发布的所有项是否彼此同步（在时间上一致）？ |  |
| 是否采用正确存储库中的正确成分的正确版本生成发布？ |  |
| 存储库/配置项审核 | 存储库是否按SCM计划定义？ |  |
| 是否已经进入正确的库？ |  |
| 是否按SCM计划中规定的命名约定项命名？ |  |
| 是否按照SCM计划，规定项的版本号？ |  |
| 是否按照SCM计划中规定的时间已经将所有项入库？例如：测试完成、客户的审评意见已采纳？ |  |
| 项是否有所要求的文档以识别项、版本和更改历史？ |  |
| 更改实施审核 | 是否全部所要求的更改请求均已结束？ |  |
| 是否更改请求标识出全部拟更改的项？ |  |
| 更改请求中所标识的全部要更改的项均已更改，被QC和在所要求的QC后入库？ |  |
| 是否可能在项目的任何两个版本中间区分更改？ |  |
| 项目的文档是否足够，能向后跟踪更改到相应的更改请求？ |  |
| 是否有恰当方法能回到以前的版本？ |  |
| 审核的其他方面 | 是否对数据库作了恰当的备份？ |  |
| 是否已测试过从备份中恢复？ |  |
| 在群组成员的工作的目录中是都有任何未经许可的成分？ |  |
| 是否有恰当的保密/批准手续以保证只有经授权的群组成员才能进行入库/出库？ |  |
|  |

在审核过程中，提供所需要的配置管理计划及相关资料，在项目开发结束后，需要提交所有关于项目的软件配置库。

七.成本分析

7.1成本构成分析

软件的成本构成如下图7-1所示。

图7-1 软件成本构成图

软件开发成本D估算模型

软件开发成本 = 开发工作量 \* 开发费用 ∕ 人·月

开发工作量 = 项目软件功能点 \* 开发成本系数 ∕ WD ∕ WM

开发费用∕人·月=（工资+奖金+福利+办公成本+资源储备+基础建设+税收利润）\*（1 + 管理费用百分比）= 综合系数 \* B（其中，B为软件人员平均工资）

（其中，开发成本系数单位为 人·小时∕ FP（功能点）、WD（每个工作日工作时长）、WM（每个月有效工作日数））

开发成本系数的大小主要考虑项目的非技术难度（开发周期、协调难度、业务的复杂程度、需求的不确定性）根据对实际数据的测算，开发成本系数取值范围一般为：

表7-1 开发成本系数

|  |  |
| --- | --- |
| 功能点数（FP） | 开发成本系数（人工时/FP） |
| <3000 | 3.5～4.0 |
| 3000<FP<=8000 | 4.0～4.5 |
| >8000 | 4.5～5.0 |

对于功能点的估算，大致有以下工作流程：

①确定计算范围：确定功能点的计算规范、划定应用程序的边界。

②功能点分析：识别和估算与软件数据和事务功能有关的各种要素及其数量。要确定功能点的数目，需要对软件的用户输入数（EI）、用户输出数（EO）、用户查询表（EQ）、内部逻辑文件数（ILF）、外部逻辑文件数（EIF）的数量进行评估。

③功能点计算（初步）：预估出五个要素的数量后，根据复杂度加权因子，计算出初步的功能点数UFC，复杂度加权因子表如表7-1。

④确定技术复杂度因子：根据项目具体情况，对14个技术复杂度参数进行调整。得出技术复杂度调整参数TCF，技术复杂度因子表如表7-2.

功能点调节：计算出经调节后的功能点数：FP=UFC × TCF

软件维护成本W估算模型

维护内容包括：运行管理、系统平台维护、应用软件维护、数据维护等

运行维护工作量（人月） = （开发工作量+实施工作量）\*维护系数w

= （开发工作量+开发工作量\*实施系数s）\*维护系数w

= D\*（1+s）\*w

维护系数w取值范围为0.15-0.20，具体取值依项目维护难度而定。

针对个别项目，如果对维护有特殊要求（这些特殊要求是一般项目中从未出现过的），则经专业机构或者专家评估，维护成本系数可以不受此限制。

固定成本F估算模型

固定成本相对于其他成本模型来说，成本总额在一定时期，一定的业务量范围内，不受业务量增减变动影响而保持不变。在本项目开发过程中，对整个项目规模进行分析，大致将硬件及系统软件成本等费用包括在内。

实施费S估算模型

实施阶段的工作量可依据开发阶段工作量、实施系数来计算。

实施工作量（人月）= 开发工作量D \* 实施系数s

根据项目是集中式实施还是分布式实施，实施系数s的取值有所不同。

集中式实施的项目：实施系数s与“用户数”相关。设n为用户数，一般情况下：当 0<n≤100时，s=0.2；否则，s=0.2 +（（n-100）/100）\*q（四舍五入取两位小数）；q是调节因子，取值范围为：0.03≤q≤0.05，具体取值依项目实施难度而定。

分布式实施的项目：实施系数s与“实施单位（点）数”相关。设n为需要实施的单位（点）数，一般情况下：s =0.2 +（n - 1）\*qq是调节因子,一般取值范围0.08≤q≤0.15，具体取值依项目实施难度而定。

个别项目，如果对实施有特殊要求（这些特殊要求是一般项目中从未出现过的或有本地化开发工作的），或者实施环境、条件、难度等方面因素的影响，则经专业机构或者专家评估，实施系数可以超出此范围上限的限制。

如果软件项目是系统集成项目中的一部分，实施时需要整体考虑，则可将实施费抽出另算。一种是将软件实施费并入到整个集成项目的实施费用中，另一种就是在软件实施费中加入项目集成的实施费用。

对于本项目而言，可以认为是集中式实施的项目。

7.2软件开发与维护成本

功能性度量方法是一种独立于编程语言的软件规模度量方式，使用这种方法可在早期根据明确功能需求来对最终产品的规模进行估算。在对软件开发环境校准以后，功能性度量的结果可以为评估开发工作量和软件产品的成本提供很好的指标。

7.2.1计算系统功能点

功能点规模公式为FP = UFC × VAF

其中UFC表示未调整功能点数，具体取值如表7-2所示。

表7-2 未调整功能点数计算表

|  |  |  |  |
| --- | --- | --- | --- |
| 测量参数 | 数量 | 权重因素 | 总计 |
| 简单 平均 复杂 |
| 用户输入数 | 105 | 3 4 6 | 420 |
| 用户输出数 | 143 | 4 5 7 | 715 |
| 用户查询表数 | 9 | 3 4 6 | 36 |
| 内部逻辑文件数 | 3 | 7 10 15 | 30 |
| 外部接口文件数 | 4 | 5 7 10 | 28 |
| 全局统计（总UFC数） | | | 1229 |

而其中的VAF表示值调整因子，计算公式为VAF = 0.65 + 0.01∑Fi，取值如表7-2所示。

表7-3 值调整因子计算表

|  |  |  |
| --- | --- | --- |
| i | Description | values |
| 01 | 剂量系统需要可靠的备份和恢复 | 5 |
| 02 | 是否需要数据通信 | 4 |
| 03 | 是分布式处理功能 | 3 |
| 04 | 性能至关重要 | 5 |
| 05 | 系统是否会在现有的，利用率很高的操作系统中运行 | 2 |
| 06 | 剂量系统需要在线输入 | 5 |
| 07 | 在线数据输入需要在多个屏幕或操作上构建输入事务 | 3 |
| 08 | 主文件是否在线更新 | 2 |
| 09 | 输入，输出，文件或查询是否复杂 | 3 |
| 10 | 内部处理是否复杂 | 3 |
| 11 | 代码是否设计为可重用的 | 4 |
| 12 | 转换和安装是否包含在设计中 | 3 |
| 13 | 该系统是否针对不同组织中的多个安装而设计 | 3 |
| 14 | 该应用程序是否旨在促进用户的更改和轻松 | 3 |
| total values: | | 48 |

上表中，values的取值范围在0-5之间，其中“0”表示此因素没有影响，“5”表示此因素有很大的影响。

最终得到

VAF = 1.12

根据以上计算结果可以计算出系统的功能点规模为FP = UFC × VAF = 1376.48。

7.2.2项目软件成本

（1）计算参数取值表

表7-4 软件成本计算参数取值表

|  |  |  |
| --- | --- | --- |
| 系数 | 描述 | 取值 |
| 开发成本系数  （人·小时∕FP） | 3000个功能点以下3.5人·小时∕FP~4.0人·小时∕FP  3000 ~ 8000个功能点数4.0人·小时∕FP ~ 4.5人·小时∕FP | 4.0 |
| WD | 每个工作日有效工作时长 | 7.5 |
| WM | 每个月有效工作日数 | 22 |
| 综合系数  （开发费用∕人·月） | 综合国家标准，一个开发工作人员在软件开发期间的费用与其所在地区软件人员平均工资的倍数关系 | 3.23 |
| B | 软件工作人员平均工资（xx地区） | 5000 |

（2）软件开发估算成本D =开发工作量（人月） \* 开发费用∕人·月

（1376.48 \* 4.0/7.5/ 22） \* 3.23 \* 5000= 538912.8

开发成本约为53.89 （万元）。

7.2.3项目软件维护成本

每一年的软件维护成本约为开发成本的10%，则软件维护成本W = 53.89 \* 10% =5.389 万元∕年

7.3固定成本及实施费

7.3.1硬件及系统软件成本

项目根据实际情况进行分期部署，在项目投入初期考虑用户规模可能偏少，所以采用阿里巴巴提供的服务器；随着项目的投入使用和用户规模的稳步增加，如果阿里巴巴提供服务器不能再维持软件正常运行，则在项目的第二期会购买服务器，建立自身独立机房，这样便于长远发展同时维护起来相对来说方便可控；在随后的发展中，每当并发量稳步增加至一定数量时，都会新增对应数量的服务器进行应对，对于节假日等用户量激增的情况，会采用云服务器租赁的方式来满足用户需求。以上是不同时期的硬件成本采购情况。

在2000以下用户并发量的情况下（第一期），暂时使用阿里云公司原有服务器。当用户并发量大于2000时，会采用下面的推荐解决方案。

表7-5 2000-4000用户并发量推荐配置（第二期）

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 名称 | 型号 | 描述 | 数量 | 备注 |
| 硬件配置推荐 | | | |  |
| 服务器1 | IBM System X3850 M4 | CPU：16核1.9GHz  内存：32GB | 1 | 应用服务器 |
| 服务器2 | IBM System X3850 M4 | CPU：16核1.9GHz  内存：32GB | 2 | 数据库服务器 |
| 软件配置推荐 | | | |  |
| 服务器操作系统 | Linux | 免费 | - |  |
| 应用服务器 | Tomcat | 免费 | - |  |
| 数据库软件 | MySQL | 免费 | - |  |
| 概算：10万 | | | |  |

表7-6 每增加2000用户并发量推荐增加配置（第三期）

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 名称 | 型号 | 描述 | 数量 | 备注 |
| 硬件配置推荐 | | | |  |
| 服务器1 | IBM System X3850 M4 | CPU：16核1.9GHz  内存：32GB | 1 | 数据库服务器 |
| 服务器2 | IBM System X3850 M4 | CPU：16核1.9GHz  内存：64GB | 1 | 应用服务器 |
| 软件配置推荐 | | | |  |
| 服务器操作系统 | Linux | 免费 | - |  |
| 应用服务器 | Tomcat | 免费 | - |  |
| 数据库软件 | MySQL | 免费 | - |  |
| 概算：8万 | | | |  |

注：由于硬件更新较快，上述硬件作为参考，贵公司可选购同品牌性能相当的机器。

7.3.2 项目实施费成本

由于该软件项目第一期采用东软睿道公司已有服务器，所以地点比较唯一，而服务器在第一期中预期仅有一台，考虑到实施难度偏小，因此我们设定比例因子K为0.08，n取值为1；在项目的第二期中，我们采用购买服务器的方式，所以需要架设自身机房，而在第二期的系统实施部署中，系统升级，难度偏大，因此我们设定比例因子K为0.10，n取值为2（服务器数），另外，机房所需的建设费用由东软睿道公司支付，具体费用不在此处说明；在项目以后的硬件升级中，设定比例因子不变为K = 0.09，n取值与增加的服务器数一致。

因此由公式：实施费S =（项目功能点 \* 实施成本系数 ∕WD ∕WM ）\* 开发费用∕人·月，可以计算得出项目实施成本费用，具体见表7-7。

表7-7 项目实施成本费用

|  |  |
| --- | --- |
|  | **实施成本费用** |
| **第一期** | 4.5万 |
| **第二期** | 27.36万 |
| **第三期（每次）** | 18.24万 |

7.4 总成本

在成本预算中，我们考虑将硬件配置分阶段增加，以避免不必要的损失。我们首先假设在第一年的后半年会达到2000用户的并发量，并且以后每年以100%的增加量来进行假设。根据以上假设，对应所需成本如下。

在初步考虑情况下，我们预期第一期满足项目正式运营前三个月的需求，第二期满足第一年的客户需要，而以后每年都增加一套第三期的配置以此来维持系统的运行。

根据公式

项目总估算成本C = 软件开发估算成本D + 软件维护估算成本W + 固定估算成本F（硬件成本 + 系统软件成本） + 实施费S +宣传费用P

我们得出以下总成本表7-7。

表7-8 项目总成本费用（五年为一周期）

|  |  |
| --- | --- |
|  | **成本费用** |
| **第一年** | 585.52万 |
| **第二年** | 720.89万 |
| **第三年** | 1028.22万 |
| **第四年** | 1124.7万 |
| **第五年** | 1200.2万 |