

学 号： 2013220801009

姓 名： 胡晗

专业方向： 软件工程（数字动漫）

企业名称： 成都莲合创想科技股份有限公司

实习岗位名称： JAVA研发工程师

企业指导教师： 楚金

院内代管教师： 肖堃

**企业实习总结报告**

**信息与软件工程学院**

摘 要

随着社会经济的发展，参与医疗保险的人数越来越多，“骗保”现象也越来越多。而传统的基于人工的检查方法费时费力，所以如何更有效的识别出“骗保”现象，成为医疗领域一个热门研究方向。此次实习参与的“智能医保审核系统”项目，使用规则引擎技术去识别“骗保”现象。此系统经过部署测试发现，能够更加有效、高效的识别出“骗保”现象。

**关键词：医疗保险** **骗保** **智能** **审核**

**Abstract**

With the development of the society and economy, more and more people begin to buy medical insurance in China, while ‘medical fraud’ appears more frequently, too. Traditional method to recognize ‘medical fraud’ is laborious, so it has been a key issue to recognize ‘medical insurance’ more efficient in the field of health care. The ‘Intelligent medical insurance audit system’ utilize Internet technology to solve the problem. I code the module ‘System manager’ and ‘Data transfer’. The experience result shows the system can recognize ‘medical fraud’ more efficient.

**keywords:** ‘medical insurance’ ‘medical fraud’ intelligence audit

目 录

[第1章 企业实习概要](#_Toc6910)

[1.1. 实习单位情况](#_Toc30796)

[1.2. 项目背景与岗位职责](#_Toc32499)

[1.2.1 项目需求分析](#_Toc26359)

[1.2.2 岗位职责](#_Toc29680)

[1.3. 实习目标、任务与复杂工程问题](#_Toc4906)

[1.4.术语与定义](#_Toc25468)

[第2章 项目综述](#_Toc10037)

[2.1. 项目可行性分析](#_Toc16875)

[2.1.1规则定义引擎](#_Toc29787)

[2.1.1.1技术说明](#_Toc30916)

[2.1.1.2解决方案](#_Toc28173)

[2.1.2医保数据分析](#_Toc17815)

[2.1.2.1技术说明](#_Toc7156)

[2.1.2.2解决方案](#_Toc14428)

[2.1.2.2.1医保费用预测方法](#_Toc8534)

[2.1.2.2.2参保人及单位类别确定](#_Toc5572)

[2.1.2.2.3异常参保人员发现](#_Toc11418)

[2.1.3时间和资源可行性分析](#_Toc17705)

[2.2. 项目引用的标准和规范](#_Toc30711)

[2.3.项目拟制定的标准和规范](#_Toc28425)

[2.4. 项目总体架构](#_Toc3220)

[2.4.1.总体设计原则](#_Toc22561)

[2.4.2. 总体技术路线](#_Toc17311)

[2.4.3. 架构概览](#_Toc9898)

[2.4.4. 架构遵从](#_Toc7460)

[2.4.4.1 应用架构](#_Toc27215)

[2.4.4.2. 数据架构](#_Toc694)

[2.4.4.3 技术架构](#_Toc17599)

[2.5. 接口](#_Toc31768)

[2.5.1. 接口设计准则](#_Toc16413)

[2.5.2. 外部接口](#_Toc20115)

[2.6. 系统数据视图](#_Toc2451)

[2.6.1. 数据模型](#_Toc924)

[2.6.1.1. 规则管理数据视图](#_Toc14802)

[2.6.1.2. 系统管理数据视图](#_Toc31215)

[2.6.1.3. 系统表关系数据视图](#_Toc25163)

[2.6.2. 非结构化数](#_Toc31826)

[2.6.3. 数据存储和分布](#_Toc3418)

[2.7. 系统物理部署视图](#_Toc29593)

[2.7.1. 容量规划](#_Toc4296)

[2.7.2. 硬件环境设计](#_Toc13733)

[2.7.3. 软件环境设计](#_Toc19782)

[2.8. 项目收获](#_Toc16748)

[2.8.1 知识技能的学习](#_Toc2766)

[2.8.2 职业素养的学习与对软件工程的认识](#_Toc30810)

[2.8.3 部分核心代码](#_Toc24851)

[第3章 结束语](#_Toc10111)

[参考文献](#_Toc21605)

[致谢](#_Toc1895)

## 第1章 企业实习概要

### 1.1. 实习单位情况

我于2016年1月至2016年9月在成都莲合创想科技股份有限公司实习，担任JAVA研发实习生职务。

成都莲合创想科技股份有限公司起步于1995年，2015年完成股份制改革在新三板上市。公司是一家致力于成为西部乃至全国首屈一指的IT服务提供商；是国家认定的双软 企业和高新技术企业。主要从事应用软件的设计、开发、实施集成及IT服务和培训，为用户提供完整的解决方案；业务涉及社会保险、HIS、卫生信息平台、新型农村合作医疗等诸多领域,是全国最早进入社会保险、农村新型合作医疗、医疗卫生领域的专业计算机公司之一。目前公司包含智能医保审核、数字生活平台、银保通等多个项目开发组，我所在的项目组为智能医保审核组。在企业指导老师的指导下，同项目组开发《智能审核分析配置管理系统》项目。

### 1.2. 项目背景与岗位职责

#### 1.2.1 项目需求分析

医疗保险是我国社会保险的重要组成部分。它涉及面广、业务量大、政策性强，关系到城镇职工的切身利益。随着社会保险覆盖面的扩大、保障水平的提高、服务便利性的提升，在利益驱动下，一些地方开始出现不法分子违规操作套取社保基金。然而由于参保人员就医总量巨大、医疗服务机构分布广泛、医保结算实时性强、临床诊疗专业性强、医疗违规行为相对隐蔽，各地医保监管及反欺诈工作面临稽核人员不足、技术手段落后、审核效率低下、惩戒威慑力度偏弱等现实困难。

本项目是医保审核系统，主要包括医保数据的接收、存储、审核、展示模块。解决现在医保数据量大，审核效率低等问题。软件接收从网络发送来的医保数据，调用系统中现有规则库对数据进行实时分析审核，并将这些数据存储到数据库中。审核后的处理信息，如预警等级、描述等，支持通过web远程查看，帮助医保审核人员快速定位不合格单据，实现对大量医保数据的实时审核和处理。

#### 1.2.2 岗位职责

在此项目中，我的岗位职责如下：

(1)负责基于WebService技术的项目对外接口程序编写

(2)负责基于Java Web技术的项目后台程序编写

(3)负责“项目管理”模块代码编写

(4)负责两个软件著作权自己研发部分书写

(5)负责相关模块使用说明书编写

(6)服务器Hadoop分布式架构学习、部署、维护

### 1.3. 实习目标、任务与复杂工程问题

实习的目标是我能够和正式员工一起参与整个项目的研发，并且在整个过程中能够担当项目核心研发之一。系统的总体目标，是实现一套医保智能审核系统，能够以医学角度出发，根据医学要求、药品、检查项目的使用限制（包括禁忌、适用症、除外内容等），对两定机构上传的特病门诊（含门诊视同住院）、住院的所有费用明细进行审核，将医保审核人员头脑中的“经验”抽取出来，完成自动审核，辅助进行医保基金管理，以减轻医保审核的负担，加快医保审核速度，从而加大医保审核范围并提升准确度。

同时系统使用业务规则，能够将业务逻辑从具体程序代码中抽取出来，支持业务人员在业务系统之外，对规则进行修改、增加、删除等基本维护操作，并且能够根据定义好的规则，实现医保数据的审核，同时，借助大数据分析，完成规则的自学习，对规则知识库进行补充，实现规则知识库的自丰富。

项目的任务主要有下面几大点：

一、是可以实现将监控逻辑从无组织的思想“固化”为规则条件，为医保机构构造费用监控知识库；

二、是用规则来表示监控逻辑可以更好的适应系统的扩展性，这样修改监控逻辑一般就不需要修改代码；

三、是具有一定的知识推理能力，能依据监控逻辑对己知的费用信息进行推理，实现对现有数据的判断和分析；

四、是具有规则自学习功能，能够根据历史大数据通过一定的数据挖掘算法产生新规则，从而实现规则的自丰富。

### 1.4.术语与定义

表1-1 术语与定义

|  |  |  |
| --- | --- | --- |
| **序号** | **名词** | **相关解释** |
| 1 | 规则引擎 | 规则引擎是一种根据规则中包含的指定约束机制，判断其能否匹配运行时刻的实时条件，来执行规则中所规定的动作的引擎。 |
| 2 | 规则 | 描述和约束业务的语句，用来刻画业务的结构或控制和影响业务的行为。 |
| 3 | 元规则 | 元规则是规则的最小单元，一条元规则可以单独进行审核，也可以组成复合规则进行审核。 |
| 4 | 指标 | 对规则、原始数据、分析资料数据库中的字段、生命周期等进行描述。 |

## 第2章 项目综述

### 2.1. 项目可行性分析

在第1章的项目背景部分已经进行了项目的需求分析，下面将从“规则引擎”、“医保数据”、“时间和资源的可行性”三个方面对项目进行全面分析。

#### 2.1.1规则定义引擎

##### 2.1.1.1技术说明

使用传统的应用系统开发和实施方法，业务规则相对固定，不易改动、医保业务规则动态变更的特点使传统的解决方案越来越难以满足实际需求。引入规则引擎之后把业务规则从具体的程序代码中剥离出来。业务规则存储在规则库中，完全独立于程序、业务人员通过规则语言和规则编辑器可以像管理数据一样对业务规则进行管理。

规则引擎是推理引擎的一种，它是基于规则的专家系统的一部分，也可以说它起源于基于规则的专家系统。专家系统使用试探性的方法进行推理，从而模仿人类的推理方式。通过使用易于人类理解的证明方式和术语证明它的推理结论。基于规则、神经网络和基于案例推理是三个最重要的专家系统。

##### 2.1.1.2解决方案

前市场上使用最多、最成熟的是Rete算法，它是一个演绎法推理算法，许多规则引擎都是基于Rete算法来实现它们的推理机，例如著名的开源规则引擎Drools。

目前主流规则引擎有JBoss旗下的Drools开源规则引擎，IBM旗下的著名商业规则引擎ILog，小巧灵活的Java规则引擎Jess和基于反向推理(归纳法)的纯Java规则引擎Mandarax 。他们的比较如表2-1所示。

表2-1 主流规则引擎比较



目前最好的商用规则管理系统BRMS是ILOG JRules，最普遍使用的开源规则引擎是Drools，Java规则引擎的规范是JSR94。 本项目采用Drools规则引擎。

#### 2.1.2医保数据分析

##### 2.1.2.1技术说明

数据挖掘在国内外医疗保险行业的应用案例十分丰富。IBM研究中心的Marisa等人基于澳大利亚医疗保险机构采集的数据，将关联规则和神经分割技术应用于医疗保险信息系统中，从GB级的数据中获取未知模式。MohitKumar等使用数据挖掘和机器学习技术预测和预防保险公司在处理医疗保险申诉过程中的支付错误，用以降低日益增长的医疗保险开销。

国内有上海市医疗保险信息中心秦德霖基于 SOA 和动态数据仓库技术，利用数据挖掘和分析技术，针对基金运行管理的主要环节和基金风险的主要因素，建立对医保基金风险防控基础技术平台。为控制医保基金的风险、保障基本医疗、促进医疗保险可持续发展提供强有力的支撑。国防科学技术大学朱攀利用人工神经网络模型对医保定点医疗机构的信用等级进行学习，并且根据学习过程中出现的问题，对人工神经网络做了改进，克服了医保定点医疗机构信用等级评价网络原有的不足。并以医保信息系统形成的大量数据为基础，利用LOF算法对大量数据进行挖掘，找出了医保定点医疗机构的违规行为。翁滔华等通过利用数据挖掘软件SPSS11.0对病毒性肝炎的住院费用情况进行分析，并分别给出了病毒性肝炎费用控制的上下限，发现能起到控制医院的住院费用的作用。黄晶晶等利用数据挖掘技术制定医保定额指标并进行相关数据的分析，结果发现数据挖掘技术能够制定动态的定额指标，加快分析反馈的速度，并做出及时的分析返溯。

由于数据规则多种多样，人工往往不能全面的分析和定义，通过利用数据挖掘和分析手段，生成新的不易人工发现或人工定义的新规则，完成规则的自学习和自动生成，作为对人工规则定义的补充，另外，在发现规则的同时，还能够发现医保数据中隐含的有价值信息。

##### 2.1.2.2解决方案

###### 2.1.2.2.1医保费用预测方法

建立医保结算费用预测模型，实现对医保结算总费用的预测，并能详细到各区县、各级别医院医保结算费用的预测，为医保基金监管提供方向。

对医保费用的预测，可采用时间序列预测方法完成。

**一、时间序列预测方法**

 时间序列预测是一种最基本、最常用的预测方法。它是运用事物的历史数据资料,按时间顺序排成动态数列进行分析,预测事物未来的发展变化。时间序列预测是根据事物的发展变化具有一定延续性这一原则进行的,就是说近期内事物的发展趋势是其过去历史的延伸。只要掌握事物一定的数据资料,就能对事物近期内的发展变化作出预测。时间序列预测的特点是着重研究事物发展变化的内因,很少考虑或不考虑外界环境和条件变化对事物的影响,因此这种方法只适用於短期预测。

###### 2.1.2.2.2参保人及单位类别确定

对参保人及单位的类别确定，可采用聚类与分类方法结合来完成。

**二、聚类分析方法**

聚类分析将数据划分成若干个聚类，使得在同一聚类中的对象比较相似，而不同聚类中的对象差别很大。聚类中要划分的类别是未知的，它是一种不依赖于预先定义的类和带类标号的训练数据集的非监督学习(unsupervised learning)，无需背景知识，其中类的数量由系统按照某种性能指标自动确定。

聚类算法的选择应由数据类型、聚类目的和应用决定。主要的聚类方法有层次聚类、划分聚类、基于密度的聚类、基于网格的聚类和基于模型的聚类。

1. **层次聚类**

层次方法是对给定的数据对象集合进行层次分解，层次方法可以分为凝聚和分裂。该方法在合并、分裂的时候要检测大量的记录和簇，因而伸缩性比较差。比较常见的方法有四种：BIRCH ,CURE,ROCK和Chameleon。

1. **划分聚类**

划分聚类事先需要制定数据分为几类，若将其划分为K个组，这K个组需要满足以下两个条件：a.每个组至少包含一个对象；b.每个对象必须且只属于一个组。但近年来的一些算法中，第二个条件可以被适当放松。K-Means, K-Medoids是划分聚类常见的方法。

1. **基于密度的聚类**

基于密度的方法与其他方法的一个本质区别是：它不是基于距离作为相似性度量的，而是基于密度的。这样就能克服基于距离的算法只能发现类球状聚类的缺点。其主要思想是只要临近区域密度超过某个阈值，则继续聚类。代表算法有DBSCAN、DENCLUE和Ankerst等人提出的OPTIC。

1. **基于网格的聚类**

基于网格的聚类是首先把对象空间向量化为有限个单元，从而形成一个网格结构，代表算法有STING。

1. **基于模型的聚类**

基于模型的方法首先假设数据集是按一定概率分布的，给定一个模型，然后去寻找能够满足这个模型的数据集。代表方法有COBWEB、SOMS。

采用聚类方法，对参保人的经济情况、健康状况和信用情况等聚类，将医保参保人聚合成多个簇，其中每个簇中参保人具有较大的相似特征，而分属不同簇的参保人则存在明显的特征差异，为参保人的分类提供依据。

###### 2.1.2.2.3异常参保人员发现

异常参保人员发现，主要是对具有就医聚集行为的人群一方面能够对特殊疾病人群提供针对性的管理和服务，另一方面能有效提高对违规人群的监督力度。

将就医聚集行为的表现形式简化为某种一致性: 在就医时多张医保卡在同一时间段内( 细化为一天) 在同一家医院就医。这些医保卡若一致性消费过于频繁则可以认为是一种异常现象，将被列为重点监管对象。

对异常就医聚集行为的挖掘，常用方法是采用关联规则算法（频繁模式挖掘）。关联规则是寻找在同一个事件中出现的不同项的相关性，即找出事件中频繁发生的项或属性的所有子集及它们之间的关联性。关联规则的挖掘过程主要有两个步骤：第一阶段是频繁项集的建立；第二阶段是关联规则的建立。

**（1）Apriori算法**

针对第一阶段的频繁集的建立，研究者提出了不少的算法，其中Apriori算法是较为成功的一种。Apriori算法的基本思想是：统计多种商品在一次购买中共同出现的频数，然后将出现频数多的搭配转换为关联规则。Apriori算法的核心是：用前一次扫描数据库的结果产生本次扫描的候选项目集，从而提高搜索的效率。可能产生大量候选集和可能需要重复扫描数据库，是Apriori算法的两大缺点。

**（2）FP-Grwoth算法**

针对Apriori算法的固有缺陷，J. Han等提出了小产生候选挖掘频繁项集的方法：FP-Growth频集算法。它采用了一种简洁的数据结构，叫做frequent-pattern tree。FP-Growth频集算法采用分而治之的策略，第一遍扫描的过程中把数据库中的频集压缩进一棵频繁模式树(FP-tree)，同时依然保留其中的关联信息，随后再将FP-tree分化成一些条件库，每个库和一个长度为1的频集相关，然后再对这些条件库分别进行挖掘，当原始数据量很大的时候，可以结合划分的方法，使得一个FP-tree可以放入主存中。与Apriori算法相比，FP-Growth算法采用前缀树的形式来表征数据，减少了扫描事务数据库的次数，通过递归地生成条件FP-tree来挖掘频繁项。

采用频繁模式挖掘算法，可以发现关联度较大的医保卡，对这些对象进行重点关注与预警。

此外，通过各种数据挖掘算法，还可进行用药模式等方面的挖掘，以辅助和改善因人工的方式定义规则而带来的不全面性。进一步对规则库进行补充，完善规则引擎。

**（3）LOF算法**

LOF离群点检测是一种基于局部密度的离群点检测算法。该算法不再把离群看成是一个二元属性，而是通过计算对象局部异常因子（LOF）来表征异常的程度。局部异常因子越大表示该值为离群点的可能性越大，反之则越小。通常异常的用户所产生的医保数据往往也不同于其他的用户，通过这种方式可以检测异常数据，从而分析异常用户。

#### 2.1.3时间和资源可行性分析

预计开发人员：医保项目组（包括正式员工4人、电子科大实习生4人）

预计开发时间：约一年

资源可行性：系统开发只需要一般PC即可完成，规则引擎可采用目前使用广泛的开源Droool；而后台数据分析部分，可利用开源大数据分析平台hadoop实施，集群可由廉价PC构建，因此在经济方面完全可行。

### 2.2. 项目引用的标准和规范

表格2-2 引用的标准和规范

| **序号** | **引用位置** | **标准名称** |
| --- | --- | --- |
| 1 | 系统架构设计 | 1、MVC标准  2、SSH/SSH2架构  3、JavaEE架构  4、Web Service  5、ZERO MQ（消息传送机制） |
| 2 | 系统设计与代码实现部分 | 1、SVN代码管理  2、莲合创想软件编码规范  3、UML标准建模语言 |
| 3 | 数据库 | 字典定义的标准：  药品目录、诊疗项目目录、服务设施目录（持续补充） |
| 4 | 数据预处理 | Python框架 |
| 5 | 外部接口 | SOAP协议；  Json标准 & XML标准；  CXF框架。 |

### 2.3.项目拟制定的标准和规范

表2-3 拟制定的标准和规范

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 序号 | 标准名称 | 标准摘要 | 拟替换的标准 | 责任单位 |
| 1 | 规则的定义规范 | 包括元规则定义规范、组合规则的定义规范 |  |  |
| 2 | 针对医保数据的数据清洗规范 | 一套较好的针对医保数据清洗的流程和优化方法 |  |  |

### 2.4. 项目总体架构

#### 2.4.1.总体设计原则

**(1) 可扩展性原则**

系统的各功能模块按照松耦合设计，各功能之间保持相对独立，各个子系统符合相关技术条件和数据传输规约的要求，同时系统具备良好的扩展性，满足软件功能扩充等要求，系统各功能模块可分期、分层建设，并可根据发展应用的需要进行平稳完善和升级。

**(2) 可移植性原则**

基于可移植性的要求，系统采用J2EE企业架构标准进行建设，使用B/S架构模式，通过标准化接口，基于标准化传输方式、采用标准化协议进行调用，支持与其它系统的数据交换和共享遵循TCP/IP协议，这些技术使得集成设计人员无需了解应用的内部工作原理。同时在底层技术标准上实现开放，采用松耦合的模块化建设模式，使用通用软件开发平台，具备良好的开放性和可移植性，适应于未来计算机技术和网络技术发展的需求。

**(3) 可靠性原则**

软硬件资源均能保障系统7×24小时不间断、可靠运行，同时配备完善的可靠性措施设计，保证系统高度可靠的运行。系统中涉及到的数据库、主机、应用部署、网络等关键环节配备多种高可用性方案。

**(4) 准确性及完整性原则**

在系统实际运行中，对数据的分析与处理均完整准确，涉及到的数据准确性、数据完整性均满足相关技术标准。

**(5) 标准化**

在设计中，需要遵循一定规范，保证系统的每个功能之间、子模块之间的统一化、标准化，保证系统后续功能开发，以便根据发展应用的需要，进行平稳完善和升级。

#### 2.4.2. 总体技术路线

系统总体设计路线将沿袭面向对象的软件开发，采用JavaEE技术架构，提供多层开发模式，将系统的行为或功能以层为组织单位来进行划分，不同层具有不同的抽象级别和特性，各层内的元素仅依赖当前层和上下相邻层中的元素，封装变化、功能清晰，实现系统松散耦合和高内聚的设计要求，同时增加了系统的灵活性和重用性。系统逻辑划分为：展现层、交互层、业务处理层和数据持久层，每层提供采用相应的技术标准为系统提供服务。

表2-4 总体技术路线

|  |  |
| --- | --- |
| **技术** | **要求** |
| 技术平台 | JavaEE, JDK1.7 |
| 技术标准 | 展现层：JavaScript、Ajax  交互层：Struts2  业务处理层：Spring  数据持久层：Hibernate |
| 开发工具 | MyEclipse8.5以上 |
| 数据库 | Oracle 11g |
| 中间件 | Tomcat/ Web Logic |
| 客户端 | 支持HTML5的浏览器 |

#### 2.4.3. 架构概览



图 2-1 架构概览图

#### 2.4.4. 架构遵从

**2.4.4.1 应用架构**

表2-5 应用架构信息

|  |  |  |
| --- | --- | --- |
| 应用架构：应用域 | 全民医保信息管理平台 | |
| 应用架构：应用 | 医保智能审核系统 | |
| 系统架构：一级功能 | 应用架构：一级应用功能 | 遵从说明 |
| 系统管理 | 日志管理 | 细化 |
|  | 权限管理 | 细化 |
|  | 数据库管理 | 细化 |
|  | 用户信息管理 | 细化 |
| 规则管理 | 元规则管理 | 细化 |
|  | 指标规则管理 | 细化 |
|  | 组合规则管理 | 细化 |
| 数据审核 | 审核 | 细化 |
|  | 记录审核结果 | 细化 |
| 数据清洗 | 约束检查 | 细化 |
|  | 数据错误检查 | 细化 |
|  | 冲突检测 | 细化 |
|  | 一致性检测 | 细化 |
|  | 分析资料数据库管理 | 细化 |

**2.4.4.2. 数据架构**

系统数据主要由系统各个模块间的数据组成，分别是：系统管理模块数据、规则管理模块数据（规则数据）、原始数据、结果数据、清洗后的数据。

表2-6 系统管理数据架构信息

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 数据域 | SYSTEM | | | |
| 数据主题 | 系统管理 | | | |
| 系统架构：一级功能 | 应用架构：一级应用功能 | 系统架构：数据实体 | 数据架构：数据实体 | 遵从说明 |
| 系统管理 | 日志管理 | 系统日志实体 | Log | 细化 |
|  | 权限管理 | 权限实体 | TRight | 细化 |
|  | 数据库管理 | 系统内所有实体 |  | 细化 |
|  | 用户信息管理 | 用户实体  用户权限实体 | UserInfo  UserRight | 细化 |

表格2-7 数据库的字典数据架构信息

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 数据域 | SYSTEM | | | |
| 数据主题 | 系统管理 | | | |
| 系统架构：一级功能 | 应用架构：一级应用功能 | 系统架构：数据实体 | 数据架构：数据实体 | 遵从说明 |
| 系统管理 | 应用项目 | 应用实体 | SysAppTable | 细化 |
|  | 表描述 | 表实体 | TableDescrible |  |
|  | 列描述 | 列实体 | ColumnDescribe |  |
|  | E-R关系描述 | 系统内所有列关系实体 | ColumnRelation |  |

表2-8 规则管理数据架构信息

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 数据域 | RULE | | | |
| 数据主题 | 规则管理 | | | |
| 系统架构：一级功能 | 应用架构：一级应用功能 | 系统架构：数据实体 | 数据架构：数据实体 | 遵从说明 |
| 规则管理 | 元规则管理 | 元规则实体 | RuleTable | 细化 |
|  | 指标规则管理 | 指标规则实体  字典实体 | IndexTable  DictionaryTable | 细化 |
|  | 组合规则管理 | 组合规则实体 | ComplexRuleTable | 细化 |

表2-9 原始数据架构信息

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 数据域 | RAW DATA | | | |
| 数据主题 | 原始数据 | | | |
| 系统架构：一级功能 | 应用架构：一级应用功能 | 系统架构：数据实体 | 数据架构：数据实体 | 遵从说明 |
| 原始数据 |  |  |  |  |
|  |  |  |  |
|  |  |  |  |

表2-10 结果数据架构信息

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 数据域 | RESULT DATA | | | |
| 数据主题 | 结果数据 | | | |
| 系统架构：一级功能 | 应用架构：一级应用功能 | 系统架构：数据实体 | 数据架构：数据实体 | 遵从说明 |
| 结果数据 |  |  |  |  |
|  |  |  |  |
|  |  |  |  |

表2-11 清洗后数据架构信息

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 数据域 | CLEARN DATA | | | |
| 数据主题 | 清洗后数据管理 | | | |
| 系统架构：一级功能 | 应用架构：一级应用功能 | 系统架构：数据实体 | 数据架构：数据实体 | 遵从说明 |
| 清洗后数据管理 |  |  |  |  |
|  |  |  |  |
|  |  |  |  |

**2.4.4.3 技术架构**



图 2-2 技术架构总览图

表2-12 技术架构遵从对照表

|  |  |  |
| --- | --- | --- |
| 产品标准 |  | |
| 系统架构：软件产品 | 技术架构：软件产品 | 遵从说明 |
| 应用系统操作系统 | Windows Server 2008 | 遵从 |
| 应用系统数据库 | Oracle 11g | 遵从 |
| 应用系统中间件 | Weblogic、Tomcat6.0 | 遵从 |
| 应用系统JAVA虚拟机 | JRE1.7 | 遵从 |
| 页面标签组件 | JSTL | 遵从 |
| 页面脚本组件 | JavaScript | 遵从 |
| 页面样式组件 | CSS | 遵从 |
| 异步通信组件 | Ajax | 遵从 |
| 简单对象组件 | JSON | 遵从 |
| 交互控制组件 | Struts | 遵从 |
| 对象管理组件 | Spring | 遵从 |
| 数据访问组件 | Hibernate | 遵从 |

### 2.5. 接口

#### 2.5.1. 接口设计准则

需要避免的接口设计特性：

僵化性：很难对系统进行改动，因为每个改动都会迫使许多对系统其他部分的其它改动。

脆弱性：对系统的改动会导致系统中和改动的地方在概念上无关的许多地方出现问题。

牢固性：很难解开系统的纠结，使之成为一些可在其他系统中重用的组件。

粘滞性：做正确的事情比做错误的事情要困难。

复杂性(不必要的)：设计中包含有不具任何直接好处的基础结构。

重复性(不必要的)：设计中包含有重复的结构，而该重复的结构本可以使用单一的抽象进行统一。

晦涩性：很难阅读、理解。没有很好地表现出意图。

需要遵循的接口设计原则 ：

1. **单一职责原则-Single Responsibility Principle(SRP)**

就一个类而言，应该仅有一个引起它变化的原因。 (职责即为“变化的原因”。)

1. **开放-封闭原则 - Open Close Principle(OCP)**

软件实体（类、模块、函数等）应该是可以扩展的，但是不可修改。(对于扩展是开放的，对于更改是封闭的。关键是抽象。将一个功能的通用部分和实现细节部分清晰的分离开来。 开发人员应该仅仅对程序中呈现出频繁变化的那些部分作出抽象。拒绝不成熟的抽象和抽象本身一样重要。)

1. **里氏替换原则 - Liskov Substitution Principle(LSP)**

子类型(subclass)必须能够替换掉它们的基类型(superclass)。

1. **依赖倒置原则(IoCP) 或 依赖注入原则 - Dependence Inversion Principle(DIP)**

抽象不应该依赖于细节。细节应该依赖于抽象。 (Hollywood原则: "Don't call us, we'll call you"。程序中所有的依赖关系都应该终止于抽象类和接口。针对接口而非实现编程。任何变量都不应该持有一个指向具体类的指针或引用。任何类都不应该从具体类派生。任何方法都不应该覆写他的任何基类中的已经实现了的方法。)

1. **接口隔离原则(ISP)**

不应该强迫客户依赖于它们不用的方法。接口属于客户，不属于它所在的类层次结构。 (多个面向特定用户的接口胜于一个通用接口。) ----包内聚原则

1. **重用发布等价原则(REP)**

重用的粒度就是发布的粒度。

1. **共同封闭原则(CCP)**

包中的所有类对于同一类性质的变化应该是共同封闭的。 一个变化若对一个包产生影响， 则将对该包中的所有类产生影响，而对于其他的包不造成任何影响。

1. **共同重用原则(CRP)**

一个包中的所有类应该是共同重用的。 如果重用了包中的一个类，那么就要重用包中的所有类。(相互之间没有紧密联系的类不应该在同一个包中。) ----包耦合原则

1. **无环依赖原则(ADP)**

在包的依赖关系图中不允许存在环。

**(10)稳定依赖原则(SDP)**

朝着稳定的方向进行依赖。应该把封装系统高层设计的软件（比如抽象类）放进稳定的包中， 不稳定的包中应该只包含那些很可能会改变的软件（比如具体类）。

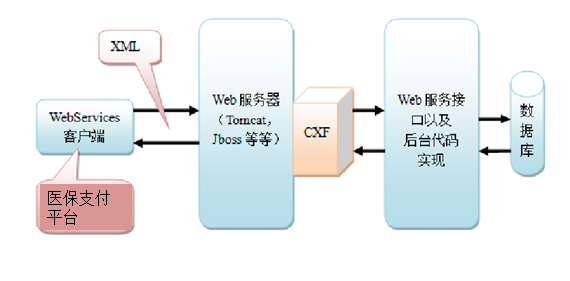
**(11) 稳定抽象原则(SAP)**

包的抽象程度应该和其稳定程度一致。 (一个稳定的包应该也是抽象的，一个不稳定的包应该是抽象的。)

#### 2.5.2. 外部接口

本系统采用WebService的方式对外提供接口，拟采用CXF框架，使用该框架主要是基于以下几点考虑：

1. CXF支持多种WebService标准，实现了JAX-WS API；
2. CXF支持轻量级服务器；
3. CXF可将WebService与Spring集成；
4. 高性能、可扩展性良好、开发简单、部署方便。



**Json**

图 2-3 用户请求WebService流程

本系统主要是与现有的医保支付平台对接，采用共享数据模型定义了本系统与医保支付平台进行交互时，需要传递的数据实体及相互关系。

医保支付平台向本系统发送的数据主要包含以下几类：

1. 人员档案
2. 医疗机构
3. 就诊主记录
4. 病案
5. 处方

……

**接口功能需求分析:**

1. 数据审核请求

该接口由审核系统提供，医保支付平台利用该接口向审核系统发送待审核数据。

1. 返回审核结果

该接口由审核系统提供，用于向医保支付平台返回审核结果。

**接口定义:**

接口名称：数据审核

接口提供者：医保智能审核系统

接口使用者：医保支付平台

接口使用方式：采用WebService方式提供接口，采用打包的方式发送多条待审核数据。

**接口需要的信息：**

|  |  |
| --- | --- |
| **数据项** | **说明** |
| 共享数据模型描述的待审核数据 | 含有待审核数据的各个字段 |

**接口返回信息：**

|  |  |
| --- | --- |
| **数据项** | **说明** |
| 数据审核结果 | 含有审核数据的编号、结果、说明等 |

### 2.6. 系统数据视图

#### 2.6.1. 数据模型

描述本系统数据实体逻辑模型，包括数据实体间的逻辑关系，数据实体关键数据属性，数据实体键。该数据模型比较直观的模拟了智能审核系统实体，容易为人理解，便于计算机实现，对于每一逻辑数据实体，通过下述表格描述实体详细信息。

##### 2.6.1.1. 规则管理数据视图



图2-4 规则管理数据模型

表13 DictionaryTable定义

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 编号-名称 | 001-DictionaryTable | | | | |
| 所属数据域 | Rule | | | | |
| 所属数据主题 | 规则管理信息 | | | | |
| 是否主数据 | 是 | | | | |
| 代码 | DictionaryTable | | | | |
| 与其他实体关系 |  | | | | |
| 备注 |  | | | | |
| 字段名称 | 字段代码 | 数据类型 | 数据长度 | 数据精度 | 主键/外键 |
| DictionaryID | DictionaryID | Number |  |  | pk |
| IndexDicID | IndexDicID | Number |  |  | Fk1 |
| TypeName | TypeName | Varchar | 1024 |  |  |
| TypeDescibe | TypeDescibe | Text |  |  |  |

表2-14 RuleTable定义

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 编号-名称 | 002- RuleTable | | | | |
| 所属数据域 | Rule | | | | |
| 所属数据主题 | 规则管理信息 | | | | |
| 是否主数据 | 是 | | | | |
| 代码 |  | | | | |
| 与其他实体关系 |  | | | | |
| 备注 |  | | | | |
| 字段名称 | 字段代码 | 数据类型 | 数据长度 | 数据精度 | 主键/外键 |
| RuleID | RuleID | number | 100 |  | pk |
| IndexOneID | IndexOneID | number |  |  | Fk1 |
| IndexTwoID | IndexTwoID | number |  |  |  |
| RuleOperatorsID | RuleOperatorsID | number | 3 |  | Fk2 |
| Result | Result | Varchar | 10 |  |  |
| RuleDescribe | RuleDescribe | Text |  |  |  |
| CreateTime | CreateTime | Date |  |  |  |
| Creator | Creator | varchar | 1024 |  |  |
| CreatorID | CreatorID | Number |  |  |  |
| ChangeFlag | ChangeFlag | Varchar |  |  |  |
| Deadline | Deadline | Date |  |  |  |

表2-15 OperatorsTable定义

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 编号-名称 | 003- OperatorsTable | | | | |
| 所属数据域 | Rule | | | | |
| 所属数据主题 | 规则管理信息 | | | | |
| 是否主数据 | 是 | | | | |
| 代码 |  | | | | |
| 与其他实体关系 |  | | | | |
| 备注 |  | | | | |
| 字段名称 | 字段代码 | 数据类型 | 数据长度 | 数据精度 | 主键/外键 |
| OperatorsID | OperatorsID | number | 3 |  | pk |
| OperatorsName | OperatorsName | Varchar | 100 |  |  |
| RalationSign | RalationSign | Varchar | 4 |  |  |

表2-16 IndexTable定义

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 编号-名称 | 004- IndexTable | | | | |
| 所属数据域 | Rule | | | | |
| 所属数据主题 | 规则管理信息 | | | | |
| 是否主数据 | 是 | | | | |
| 代码 |  | | | | |
| 与其他实体关系 |  | | | | |
| 备注 |  | | | | |
| 字段名称 | 字段代码 | 数据类型 | 数据长度 | 数据精度 | 主键/外键 |
| IndexID | IndexID | number | 38 |  | pk |
| IndexName | IndexName | Varchar | 1024 |  |  |
| IndexDescribe | IndexDescribe | Text |  |  |  |
| IndexCreatorID | IndexCreatorID | number |  |  |  |
| IndexCreatorTime | IndexCreatorTime | Date |  |  |  |
| IndexDeadline | IndexDeadline | Date |  |  |  |

表2-17 OperationRuleTable定义

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 编号-名称 | 002- OperationRuleTable | | | | |
| 所属数据域 | Rule | | | | |
| 所属数据主题 | 规则管理信息 | | | | |
| 是否主数据 | 是 | | | | |
| 代码 |  | | | | |
| 与其他实体关系 |  | | | | |
| 备注 |  | | | | |
| 字段名称 | 字段代码 | 数据类型 | 数据长度 | 数据精度 | 主键/外键 |
| RuleID | RuleID | number | 100 |  | pk |
| IndexOneID | IndexOneID | number |  |  | Fk1 |
| IndexTwoID | IndexTwoID | number |  |  |  |
| RuleOperatorsID | RuleOperatorsID | number | 3 |  | Fk2 |
| Result | Result | Varchar | 10 |  |  |
| RuleDescribe | RuleDescribe | Text |  |  |  |
| CreateTime | CreateTime | Date |  |  |  |
| Creator | Creator | varchar | 1024 |  |  |
| CreatorID | CreatorID | Number |  |  |  |
| Deadline | Deadline | Date |  |  |  |
| DictionaryRuleID | DictionaryRuleID | Number |  |  |  |
| Value | Value |  |  |  |  |
| CompareType | CompareType |  |  |  |  |

##### 2.6.1.2. 系统管理数据视图



图2-4 系统管理数据模型

表2-18 UserInfo定义

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 编号-名称 | 005-UserInfo | | | | |
| 所属数据域 | SYSTEM | | | | |
| 所属数据主题 | 系统管理 | | | | |
| 是否主数据 | 是 | | | | |
| 代码 |  | | | | |
| 与其他实体关系 |  | | | | |
| 备注 |  | | | | |
| 字段名称 | 字段代码 | 数据类型 | 数据长度 | 数据精度 | 主键/外键 |
| 用户编号 | UserID | Int |  |  | pk |
| 用户名 | UserName | Varchar | 30 |  |  |
| 真实姓名 | RealName | Varchar | 50 |  |  |
| 密码 | Password | Varchar | 50 |  |  |
| 所属区域 | RegionID | Int |  |  | Fk |
| 注册时间 | RegiDate | Date |  |  |  |
| 有效标志 | Flag | Number | 1 |  |  |
| 上次登录时间 | LastLoginTime | Date |  |  |  |

表2-19 Region定义

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 编号-名称 | 006-Region | | | | |
| 所属数据域 | SYSTEM | | | | |
| 所属数据主题 | 系统管理 | | | | |
| 是否主数据 | 是 | | | | |
| 代码 |  | | | | |
| 与其他实体关系 |  | | | | |
| 备注 |  | | | | |
| 字段名称 | 字段代码 | 数据类型 | 数据长度 | 数据精度 | 主键/外键 |
| 区域编号 | RegionID | varchar | 6 |  | pk |
| 区域名 | RegionName | Varchar | 50 |  |  |
| 上级区域 | ParentRegion | Varchar | 6 |  | fk |

表2-30 Log定义

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 编号-名称 | 007-Log | | | | |
| 所属数据域 | SYSTEM | | | | |
| 所属数据主题 | 系统管理 | | | | |
| 是否主数据 | 是 | | | | |
| 代码 |  | | | | |
| 与其他实体关系 |  | | | | |
| 备注 |  | | | | |
| 字段名称 | 字段代码 | 数据类型 | 数据长度 | 数据精度 | 主键/外键 |
| 日志编号 | LogID | Varchar | 20 |  | pk |
| 操作类型 | OperationType | Varchar | 20 |  |  |
| 操作时间 | Time | Date |  |  |  |
| 用户ID | UserID | Int |  |  | Fk |
| 操作描述 | OperateInfo | Varchar | 500 |  |  |
| 表名 | TabName | Varchar | 100 |  |  |

表2-31 Role定义

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 编号-名称 | 008-Role | | | | |
| 所属数据域 | SYSTEM | | | | |
| 所属数据主题 | 系统管理 | | | | |
| 是否主数据 | 是 | | | | |
| 代码 |  | | | | |
| 与其他实体关系 |  | | | | |
| 备注 |  | | | | |
| 字段名称 | 字段代码 | 数据类型 | 数据长度 | 数据精度 | 主键/外键 |
| 角色ID | RoleID | Int |  |  | pk |
| 角色名 | RoleName | Varchar | 30 |  |  |
| 创建者ID | CreaterID | Int |  |  | fk |
| 创建者名 | CreaterName | Varchar | 30 |  |  |
| 创建时间 | CreateTime | Date |  |  |  |
| 备注 | Remark | varchar | 100 |  |  |
| 生效标志 | Flag | Number | 1 |  |  |

表2-32 UserRole定义

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 编号-名称 | 009-UserRole | | | | |
| 所属数据域 | SYSTEM | | | | |
| 所属数据主题 | 系统管理 | | | | |
| 是否主数据 | 是 | | | | |
| 代码 |  | | | | |
| 与其他实体关系 |  | | | | |
| 备注 |  | | | | |
| 字段名称 | 字段代码 | 数据类型 | 数据长度 | 数据精度 | 主键/外键 |
| 编号 | UserRoleID | Int |  |  | pk |
| 用户ID | UserID | Int |  |  | Fk |
| 角色ID | RoleID | Int |  |  | Fk |
| 有效标志 | Flag | Number | 1 |  |  |

表2-33 Right定义

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 编号-名称 | 010-Right | | | | |
| 所属数据域 | SYSTEM | | | | |
| 所属数据主题 | 系统管理 | | | | |
| 是否主数据 | 是 | | | | |
| 代码 |  | | | | |
| 与其他实体关系 |  | | | | |
| 备注 |  | | | | |
| 字段名称 | 字段代码 | 数据类型 | 数据长度 | 数据精度 | 主键/外键 |
| 编号 | RightID | Int |  |  | pk |
| 权限名 | RightName | Varchar | 60 |  |  |
| 备注 | Remark | Varchar | 100 |  |  |

表2-34 RoleRight定义

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 编号-名称 | 010-RoleRight | | | | |
| 所属数据域 | SYSTEM | | | | |
| 所属数据主题 | 系统管理 | | | | |
| 是否主数据 | 是 | | | | |
| 代码 |  | | | | |
| 与其他实体关系 |  | | | | |
| 备注 |  | | | | |
| 字段名称 | 字段代码 | 数据类型 | 数据长度 | 数据精度 | 主键/外键 |
| 角色权限编号 | RoleRightID | Int |  |  | pk |
| 权限编号 | RightID | int |  |  | fk |
| 角色编号 | RoleID | int |  |  | fk |

##### 2.6.1.3. 系统表关系数据视图



图2-5 规则管理数据模型

表2-35 TableDesTable定义

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 编号-名称 | 001-TableDesTable | | | | |
| 所属数据域 | SYSTEM | | | | |
| 所属数据主题 | 系统表关系信息 | | | | |
| 是否主数据 | 是 | | | | |
| 代码 | TableDesTable | | | | |
| 与其他实体关系 |  | | | | |
| 备注 |  | | | | |
| 字段名称 | 字段代码 | 数据类型 | 数据长度 | 数据精度 | 主键/外键 |
| TableID | TableID | Number |  |  | pk |
| TableName | TableName | Varchar |  |  |  |
| TableDescribe | TableDescribe | Text |  |  |  |

表2-36 ColumnDesTable定义

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 编号-名称 | 002- ColumnDesTable | | | | |
| 所属数据域 | System | | | | |
| 所属数据主题 | 系统数据表信息 | | | | |
| 是否主数据 | 是 | | | | |
| 代码 |  | | | | |
| 与其他实体关系 |  | | | | |
| 备注 |  | | | | |
| 字段名称 | 字段代码 | 数据类型 | 数据长度 | 数据精度 | 主键/外键 |
| ColumnID | ColumnID | number | 100 |  | pk |
| LocTableID | LocTableID | number |  |  | Fk1 |
| IsNull | IsNull | Boolean |  |  |  |
| IsPK | IsPK | Boolean |  |  | Fk2 |
| ColumnDescribe | ColumnDescribe | text |  |  |  |
| ColName | ColName | Varchar | 1024 |  |  |
| Table\_ID | Table\_ID | Number |  |  |  |
| Table\_Name | Table\_Name | varchar | 1024 |  |  |
| ColType | ColType | Varchar |  |  |  |
| ColNullable | ColNullable | Char | 1 |  |  |
| ColPkey1 | ColPkey1 | Char | 1 |  |  |
| ColPkey2 | ColPkey2 | Char | 1 |  |  |
| ColIdex1 | ColIdex1 | Char | 1 |  |  |
| ColIdex2 | ColIdex2 | Char | 1 |  |  |

表2-37 ColumnRelTable定义

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 编号-名称 | 003- ColumnRelTable | | | | |
| 所属数据域 | System | | | | |
| 所属数据主题 | 系统数据关系信息 | | | | |
| 是否主数据 | 是 | | | | |
| 代码 |  | | | | |
| 与其他实体关系 |  | | | | |
| 备注 |  | | | | |
| 字段名称 | 字段代码 | 数据类型 | 数据长度 | 数据精度 | 主键/外键 |
| ColumnRelID | ColumnRelID | number |  |  | pk |
| Table\_ID | Table\_ID | number |  |  |  |
| Rel\_Tab\_Left | Rel\_Tab\_Left | Varchar | 1024 |  |  |
| Rel\_Left\_Lambda | Rel\_Left\_Lambda | Varchar | 1024 |  |  |
| Rel\_Tab\_Right | Rel\_Tab\_Right | Varchar | 1024 |  |  |
| Rel\_Right\_Lambda | Rel\_Right\_Lambda | Varchar | 1024 |  |  |
| RelType | RelType | Number |  |  |  |
| RelDescrible | RelDescrible | Text |  |  |  |

#### 2.6.2. 非结构化数

无

#### 2.6.3. 数据存储和分布

表2-38 数据存储清单

|  |  |
| --- | --- |
| 数据实体 | 存储系统名称 |
| Rule | 规则管理 |
| System | 系统管理 |

表2-39数据分布清单

|  |  |  |
| --- | --- | --- |
| 数据实体 | SG-MDM | 中文解释 |
| Rule | DictionaryTable | 字典信息表 |
| Rule | RuleTable | 元规则表 |
| Rule | RelationTable | 关系符表 |
| Rule | TargetTable | 指标信息表 |
| System | UserInfo | 用户信息表 |
| System | Role | 角色表 |
| System | Log | 系统日志 |
| System | UserRole | 用户角色表 |
| System | Right | 权限表 |
| System | Region | 区域/机构表 |
| System | RoleRight | 角色权限表 |

### 2.7. 系统物理部署视图

#### 2.7.1. 容量规划

表2-40容量规划清单

|  |  |
| --- | --- |
| **指标分类** | **容量** |
| 表空间 | 4G |
| Web服务器内存 | 16G |
| 数据库服务器内存 | 16G |

#### 2.7.2. 硬件环境设计

表2-41硬件环境

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 物理部署节点编号-名称 | 硬件 | 配置 | 位置 | 备注 |
| XH01-数据库服务器部署节点 | 数据库服务器 | CPU主频：不低于2.4GHz 内存容量：不低于6GB 硬盘容量：不低于1TB，RAID5 操作系统：Windows Server 2008 |  |  |
| XH02-应用服务器部署节点 | 应用服务器 | CPU主频：不低于2.4GHz 内存容量：不低于4GB 硬盘容量：不低于300GB，RAID0/1 操作系统：Windows Server 2008 |  |  |

#### 2.7.3. 软件环境设计

表2-42 物理部署视图软件清单

|  |  |  |  |
| --- | --- | --- | --- |
| 物理部署节点编号-名称 | 软件名称 | 版本 | 备注 |
| XH01-数据库服务器部署节点 | 操作系统 | Windows Server 2008 |  |
| 数据库 | Oracle 11g |  |
| XH02-应用服务器部署节点 | 操作系统 | Windows Server 2008 |  |
| 应用服务器 | Tomcat 8.0.24 |  |

### 2.8. 项目收获

#### 2.8.1 知识技能的学习

经过这一段时间的实习，自己职业技能和素养毫无疑问得到了大大的加强。系统的技术架构图如表2所示，由于自己在学校有Java基础，所以学习的主要目标是JavaEE、Spring、Struts2、Hibernate等基于Java的开发框架。此外，由于自己并不负责展现层和数据库，所以与此相关的技术，自己只做了解。

表2-43 项目总体技术架构

|  |  |
| --- | --- |
| 技术 | 要求 |
| 技术平台 | JavaEE, JDK1.7 |
| 技术标准 | 展现层：JavaScript、Ajax  交互层：Struts2  业务处理层：Spring  数据持久层：Hibernate |
| 开发工具 | MyEclipse8.5以上 |
| 数据库 | Oracle 11g |
| 中间件 | Tomcat/ Web Logic |
| 客户端 | 支持HTML5的浏览器 |

#### 2.8.2 职业素养的学习与对软件工程的认识

自己的学习过程采用“需求推动”、“螺旋上升”的方法，在项目开始阶段，我对SSH2（Spring、Struts2、Hibernate）三大框架的基础知识进行学习，让自己对三大框架有一个初步的认识。然后便进入项目研发阶段，项目需要什么，就马上去学习什么，运用研发间歇期进行回顾总结，咀嚼消化。学习文献自己主要参考网上的技术博客与官方的开发文档，比如CSDN，博客园等。由于互联网技术变化很快，所以书籍学习方式使用次数并不多。

经过这一段时间的学习以及和公司同事交流，自己愈发发现程序员是一个终生学习的职业，学习可能会伴随自己的整个职业生涯，所以学习能力也许是自己以后职业生涯能不能走入一个新台阶的关键因素之一。“word is cheap，show me the code”是程序员届流传很广的一句名言，只有多动手编程，多参与实际项目开发，多在实际中运用自己所学习的知识，才能让自己更好的理解所学习的知识。面对“软件工程”这样一个实际运用很强的科目，多走出象牙塔去接触真实的开发，是必须的，也是我们之后职业生涯的一个良好铺垫。

同时对于“软件工程”这门学科，既然是“工程”，那么往往就不只是一个人能够轻易解决的。所以学会团队合作，学会从一个团队宏观的角度去思考问题，学会时时刻刻明白自己在团队中的位置已经责任，是我们从事这个行业必不可少的素养。

#### 2.8.3 部分核心代码

下面代码提供数据接口核心业务类，调用对应函数分别实现查询服务器状态、查询任务号状态、上传数据、清除上传数据、发起审核、查询审核结果、强制上传数据，强制删除数据功能。

代码 1 **AccessRequest.java**

|  |
| --- |
| public class AccessRequest {  Util<AccessRequest> util=new Util<AccessRequest>();  InsertDAO insertDAO=new InsertDAO();  QueryDAO queryDAO=new QueryDAO();  //public static Connection conn=null;  /\*  \* status  \*/  public String status(String auditno, String data, String authcode) throws SQLException{  String outputbody=util.createNullOutputbody();  if(queryDAO.queryIsWorking()){  return util.returnXMLMessage("1","ok",outputbody);  }else{  return util.returnXMLMessage("0","no",outputbody);  }  }  public String getjobno(String auditno, String data, String authcode,Connection conn) throws SQLException{  String outputbody=null;  Jobno jobno=new Jobno();  String jobnoS=jobno.getJobno();  outputbody=util.createOutputbody(jobnoS, Constant.WSDL);  \* beginaudit  \*/  public String beginaudit(String auditno, String data, String authcode,Connection conn) throws UnsupportedEncodingException, SQLException, DocumentException, ClassNotFoundException, InterruptedException{  SAXReader saxReader = new SAXReader();  Document document=null;  String outputbody=util.createNullOutputbody();  StringBuffer returnMsg=new StringBuffer();  document=saxReader.read(new ByteArrayInputStream(data.getBytes("UTF-8")));  Element root=document.getRootElement();  String jobno=root.elementText("jobno");  String group\_id=root.elementText("group\_id");  //System.out.println("------------------------------------------group\_id:"+group\_id);  String multithread=root.elementText("multithread");  String result=null;  String output=null;  boolean isSetReturnStatus=false;  int count=0;  /\*  \* \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  \*/    /\*  \* queryjobstatus  \*/  public String queryjobstatus(String auditno, String data, String authcode, Connection conn) throws UnsupportedEncodingException, SQLException, DocumentException{  SAXReader saxReader = new SAXReader();  Document document=null;  StringBuffer returnMsg=new StringBuffer();  QueryDAO queryDAO=new QueryDAO();  document = null;  Element root=null;  String jobno=null;  String result=null;  String output=null;  document=saxReader.read(new ByteArrayInputStream(data.getBytes("UTF-8")));  root=document.getRootElement();  jobno=root.elementText("jobno");  /\*  \* \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  \*/  result=queryDAO.queryJobnoStatus(jobno,conn);  output=util.createOutputbody(result);  return util.returnXMLMessage("1","ok"+"\n"+returnMsg, output);  }  /\*  \* queryjobresult  \*/  public String queryjobresult(String auditno, String data, String authcode, Connection conn) throws UnsupportedEncodingException, SQLException, DocumentException, ClassNotFoundException{  SAXReader saxReader = new SAXReader();  Document document=null;  StringBuffer returnMsg=new StringBuffer();  QueryDAO queryDAO=new QueryDAO();  document = null;  Element root=null;  String jobno=null;  String result=null;  String output=null;  document=saxReader.read(new ByteArrayInputStream(data.getBytes("UTF-8")));  root=document.getRootElement();  jobno=root.elementText("jobno");  /\*  \* \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  \*/  result=queryDAO.queryJobnoStatus(jobno,conn);  if(result.equals(Constant.FINISHED)){  output=queryDAO.queryOutput("output", jobno,conn);  return util.returnXMLMessage("1",""+"\n"+"ok", output);  }else{  Return util.returnXMLMessage("0",result, util.createNullOutputbody());  }  //String output=util.createOutputbody(jobno, wsdl)  //return util.returnXMLMessage("1","ok"+"\n"+returnMsg, output);  }  /\*  \* clearjobdata  \*/  public String clearjobdata(String auditno, String data, String authcode,Connection conn) throws UnsupportedEncodingException, DocumentException, SQLException{  SAXReader saxReader = new SAXReader();  Document document=null;  StringBuffer returnMsg=new StringBuffer();  DeleteDAO deleteDAO=new DeleteDAO();  QueryDAO queryDAO=new QueryDAO();  Element root=null;  String jobno=null;  document = saxReader.read(new ByteArrayInputStream(data.getBytes("UTF-8")));  root=document.getRootElement();  jobno=root.elementText("jobno");  String result=null;  String output=null;  result=queryDAO.queryJobnoStatus(jobno,conn);  if(result.equals(Constant.NODATA) || result.equals(Constant.LOADING) || result.equals(Constant.NO\_JONNO)){  if(deleteDAO.deleteJobData(jobno,conn))  return util.returnXMLMessage("1",Constant.SUCCESS,util.createNullOutputbody());  else{  return util.returnXMLMessage("0",Constant.FAILURE+"删除失败",util.createNullOutputbody());  }  }  else{  return util.returnXMLMessage("0",""+jobno+"当前状态:"+result+",不能删除",util.createNullOutputbody());  }  // return null;  } |

## 第3章 结束语

这大半年的实习是自己大学生涯中非常重要、有意义的一段经历。他让我走出了象牙塔去接触社会，让我不再停留在纸上谈兵。关于我这次实习的心得体会，可以总结为两点：

（1）注重能力的培养。由于互联网行业的特殊性，技术更新换代迅速，所以如果我们仅仅停留在“用轮子”的层面上，不但费时，比如需要不断的去学习新的技术，而且当年龄上涨时，也不允许我们有年轻时的一样多的精力去学习。所以掌握“造轮子”的能力十分重要，自己在今后的职业生涯中会花更多的精力去训练自己“造轮子”的能力，希望自己的职业生涯进入一定阶段后，能够沉淀出自己的东西。

（2）学会团队合作。与人相处有时候比编程更难，因为代码是死的，但人却是活的。但凡一个有一定规模的项目，就不可能凭一己之力完成，所以在一个团队中学会和其他成员去沟通、交流是尤为重要的。一个优秀的团队能很好的促进甚至决定一个人的成长。

## 参考文献

[1] 关于我们：GE医疗在中国[OL]. http://www3.gehealthcare.cn/zh-cn/about\_gehc\_china/about\_us

[2] 刘思伟，李彬，田联房，陈萍.计算机工程与应用[M]. 华北计算技术研究所，2009，45(22) 191

## 致谢

在我的实习过程中，成都莲合创想科技股份有限公司的楚金老师对我细心指导、照顾，教会了我许多为人处世的道理。信软学院的李晓瑜老师关心我的学习生活、耐心指导我进行科研探索。实验中心的肖堃老师耐心指导我报告书写，对我认真、负责。对以上老师我表示由衷的感谢。最后更要感谢学院为我们学生提供的实习机会，对我们悉心的栽培。