单位代码

学号61231225

分类号

密级

****

毕业设计(论文)

基于Activiti的请假流程管理系统的设计与研究

|  |  |
| --- | --- |
| 院（系）名称 | 继续教育学院 |
| 专业名称 | 计算机科学与技术 |
| 学生姓名 | 秦飞 |
| 指导教师 |  |

2013年3月

本人声明

我声明，本论文及其研究工作是由本人在导师指导下独立完成的，在完成论文时所利用的一切资料均已在参考文献中列出。

作者：秦飞

签字：

时间：2013年月

基于Activiti的请假流程管理系统的设计与研究

学生：秦飞

指导教师：

摘要

随着企业规模的日益扩大，企业日常管理变得越来越复杂庞大，业务的流程时刻在发生着变化，企业管理管理工作面临这重复繁重的劳动。使用信息化技术来提高企业管理的质量和效率是企业事务发展的一种趋势，针对日常工作中具有固定程序的活动提出了工作流的概念，其目的是通过将工作分解为定义好的任务、角色、规则和过程来执行和监控的系统。

1993年工作流管理联盟(Workflow Management Coalition,简称WflVIC)对工作流管理系统的相关术语、体系结构及应用编程接口等方面制定了一系列标准。工作流管理联盟给出的工作流定义是:工作流是一类能够完全或者部分自动执行的经营过程,它根据一系列过程规则、文档、信息或任务能够在不同的执行者之间进行传递或执行。在实际情况中可以更广泛地把凡是由计算机软件系统(工作流管理系统)控制其执行的过程都称为工作流。工作流管理系统是指运行在一个或多个工作流引擎上的用于定义、实现和管理工作流运行的一套软件系统。工作流是通过工作流管理系统实现的。工作流引擎是为工作流管理系统在定义提供支持、同时在运行时提供解释和执行服务的一组数据模型和软件。

目前工作流商用应用软件产品比较成熟和丰富，其中比较著名的有IB M的MQSeriesW orkflow，A ction技术公司的Met，F ile N et公司VisualWorkFl, Jet Form公司的InTemp，Pa vone公司的Espress等。在工作流技术实现方面，基于JAVA的商用、开源工作流引擎很多包括Osworkflow，Jbpm，Activiti，OBE，Werkflow等。Activiti项目是一项新的基于Apache许可的开源BPM平台，从基础开始构建，旨在提供支持新的BPMN 2.0标准，包括支持对象管理组（OMG），面对新技术的机遇，诸如互操作性和云架构。Activiti5和jbpm4的api有这将近60%的相同，从而使Activiti能依托jbpm4的强大用户群，更加快速的应用到实际系统开发中。

本系统基于Activiti5工作流引擎进行功能扩充，包括独立的系统管理以及独立的流程管理。所进行的工作包括：

1.设计并实现了完整的基于企业的用户、部门、角色和权限的系统管理。

2.封装并重新设计了流程的设计、部署、启动、待办已办查询和流程的执行接口的改造，并针对自定义的系统管理进行流程节点上人员与角色的设定。

3.通过调用底层Activiti的api接口处理流程的跳转、会签、转发、动态设置参与者等工作流引擎未提供的功能。

系统完成后进行了系统的功能和性能测试，结果表明系统具有良好的可用性。在论文最后，对系统相关功能的改进和完善提出了一些设想。

关键词：工作流，Activiti，改造 ，请假

Based on the Activiti process for leave management system design and research

Author:Qin Fei

Tutor:

**Abstract**

Along with the expanding of enterprise scale, enterprise daily management is becoming more and more huge, complex business process time, there are changes in the enterprise management management facing this hard work again. Using information technology to improve the quality and efficiency of enterprise management is a trend in the development of corporate affairs, in daily work with fixed program activities, puts forward the concept of workflow, the purpose is to through will work into well-defined tasks, roles, rules and processes to implement and monitor system.

Workflow Management Coalition in 1993 (Workflow Management Coalition, WflVIC) the related terms of Workflow Management system, system structure and application programming interfaces (apis) developed a series of standards. Gives the workflow management coalition workflow definition, workflow is a kind of can fully or partially automated business process, it is according to the rules of a series of process, documents, information or tasks can be used to transfer between different performers or to perform. In actual situation can be more widely to all controlled by the computer software system (workflow management system) the process of its execution are referred to as a workflow. Workflow management system is running on one or more of the workflow engine is used to define, implement and manage workflow running a set of software system. The workflow is through the implementation of workflow management system. Workflow engine is to provide support for workflow management system has been defined, at the same time provide interpretation and execution services at run time, a set of data model and software.

Current Workflow commercial application software products mature and rich, which is famous with Lotus Domino, Notes a series of office automation software, enterprise Workflow products are BEA Weblogic IntegrationStudio, IBM's MQSeries Workflow, etc. In the aspect of workflow technology, commercial, open source workflow engine based on JAVA a lot including Osworkflow, Jbpm, Activiti, OBE, Werkflow, etc. Activiti projects is a new open source BPM platform, based on the Apache license to start building from the foundation, aims to provide new support BPMN 2.0 standard, including support for the object management group (OMG), in the face of new technology opportunities, such as interoperability and cloud architecture. Activiti5 and jbpm4 has the nearly 60% of the same API, so that the Activiti can relying on jbpm4 strong user base, applied to practical system development more quickly.

This system based on workflow engine Activiti5 function expansion, including independent of system management and process management. Of work include:

1. Complete was designed and implemented based on the enterprise user, department, roles and permissions system management.

2. Packaging and redesigned the process design, deployment, execution of startup, backlog has query and process interface modification, and focused on the system of the custom management on process node setting and role.

3. By calling the bottom jump Activiti API interface processing, sign and forward, dynamic set of workflow engine fails to provide functions such as participants.

System after the completion of the system function and performance test, the results show that the system has good availability. At the end of the paper, related to the system function improvement and perfection of some proposals are put forward.

**Keywords**：Workflow, Activiti, transform, ask for leave cene

**目录**

[Based on the Activiti process for leave management system design and research III](#_Toc355354255)

[**Abstract** III](#_Toc355354256)

[1 绪论 1](#_Toc355354257)

[1.1 课题背景及意义 1](#_Toc355354258)

[1.2 国内外研究现状 2](#_Toc355354259)

[1.2.1 工作流管理技术的发展 2](#_Toc355354260)

[1.2.2 工作流引擎研究 2](#_Toc355354261)

[1.2.3 BPMN2.0研究现状 5](#_Toc355354262)

[2 相关概念和技术 8](#_Toc355354263)

[2.1 工作流相关概念 8](#_Toc355354264)

[2.1.1 工作流(Workflow) 8](#_Toc355354265)

[2.1.2 业务流程（Business Process） 8](#_Toc355354266)

[2.1.3 过程定义（Process Definition） 8](#_Toc355354267)

[2.1.4 过程定义工具（Process Definition Tools） 8](#_Toc355354268)

[2.1.5 活动（Activity） 8](#_Toc355354269)

[2.1.6 过程/活动实例（Process/Activity Instances） 9](#_Toc355354270)

[**2.1.7** 工作流管理系统（Workflow Management System, WFMS） 9](#_Toc355354271)

[2.1.8 工作流引擎（Workflow Engine） 11](#_Toc355354272)

[2.1.9 工作项（Work Item） 11](#_Toc355354273)

[2.1.10 工作列表（Work Items List） 11](#_Toc355354274)

[2.2 Activiti工作流步骤 12](#_Toc355354275)

[2.2.1 加载（发布）流程定义 12](#_Toc355354276)

[2.2.2 启动流程 12](#_Toc355354277)

[2.2.3 处理任务 12](#_Toc355354278)

[2.2.4 记录流程的相关状态 12](#_Toc355354279)

[2.3 Activiti5 API应用 12](#_Toc355354280)

[2.3.1 RepositoryService： 13](#_Toc355354281)

[2.3.2 TaskService 13](#_Toc355354282)

[2.3.3 RuntimeService： 13](#_Toc355354283)

[2.3.4 HistoryService： 13](#_Toc355354284)

[2.3.5 IdentityService： 13](#_Toc355354285)

[2.3.6 FormService： 13](#_Toc355354286)

[2.3.7 ManagementService： 14](#_Toc355354287)

[2.4 Activiti5设计模式 14](#_Toc355354288)

[3 基于Activiti的请假流程管理系统的设计 15](#_Toc355354289)

[3.1 系统需求分析与设计原则 16](#_Toc355354290)

[3.1.1 系统需求分析 16](#_Toc355354291)

[3.1.2 系统设计原则 17](#_Toc355354292)

[3.2 系统体系结构 17](#_Toc355354293)

[3.2.1 MVC分层 18](#_Toc355354294)

[3.2.2 模块化管理 18](#_Toc355354295)

[3.3 开发工具 20](#_Toc355354296)

[3.3.1 开发语言： 20](#_Toc355354297)

[3.3.2 开发工具 20](#_Toc355354298)

[3.4 系统架构 20](#_Toc355354299)

[3.4.1 数据库层配置 20](#_Toc355354300)

[3.4.2 依赖注入配置 20](#_Toc355354301)

[3.4.3 控制层配置 21](#_Toc355354302)

[3.5 模块的详细设计 21](#_Toc355354303)

[3.5.1 问题分析 21](#_Toc355354304)

[3.5.2 工作流系统的设计 21](#_Toc355354305)

[3.5.3 请假业务系统设计 22](#_Toc355354306)

[3.5.4 用户管理系统设计 22](#_Toc355354307)

[4 基于Activiti的请假流程管理系统的实现 24](#_Toc355354308)

[4.1 系统管理 24](#_Toc355354309)

[4.2 业务功能数据库设计 24](#_Toc355354310)

[4.3 流程设计 26](#_Toc355354311)

[4.4 代码实现 26](#_Toc355354312)

[4.4.1 流程文件上传模块 26](#_Toc355354313)

[4.4.2 业务流程绑定模块 27](#_Toc355354314)

[4.4.3 工作流模块 28](#_Toc355354315)

[5 系统实验设计与分析 30](#_Toc355354316)

[5.1 实验环境 30](#_Toc355354317)

[5.2 请假申请 31](#_Toc355354318)

[5.3 小结 32](#_Toc355354319)

[6 总结与展望 33](#_Toc355354320)

[6.1 工作总结 33](#_Toc355354321)

[6.2 工作展望 33](#_Toc355354322)

[参考文献 35](#_Toc355354323)

# 绪论

## 课题背景及意义

随着企业规模的日益扩大，企业日常管理变得越来越复杂庞大，业务的流程时刻在发生着变化，企业管理管理工作面临这重复繁重的劳动。使用信息化技术来提高企业管理的质量和效率是企业事务发展的一种趋势，针对日常工作中具有固定程序的活动提出了工作流的概念，其目的是通过将工作分解为定义好的任务、角色、规则和过程来执行和监控的系统。

目前大部分公司的请假业务还是停留在纸上或邮件发送请假申请给直接负责人，然后再有直接负责人一直向上提交申请，整个审核流程完成之后再次反向发送给申请人，整个流程的执行过程业务复杂，执行效率慢。而且一旦公司内部流程发生改变，整个流程将会随时可能在未知的环节上出现问题，所以基于无纸化办公和动态流程设计方案采用工作流引擎进行请假流程管理系统的设计与开发。基于公司内部人员与角色的管理系统以及独立的工作流管理系统的请假流程管理系统可以实现针对角色，针对个人进行流程的审批、发送、办理以及查询。同时由于进行了流程的跳转、会签、转发、动态设置参与者等功能的改造，系统完全可以进行任意流程跳转和人员指定，从而可以更加灵活的处理日常业务中的突发情况。

采用工作流管理系统的有点：

* 提高管理的规范化程度，便于公司内部的分工和管理；
* 降低业务过程的处理时间，通过系统的待办通知提醒和数据统计分析等提高工作人员的工作效率，减少在处理过程中的不必要等待和反复处理等问题。
* 降低管理成本，提高企业办事效率。
* 改进工作质量，避免因不必要的或重复的工作影响员工的日常工作。可以更好的进行工作量均衡、代理、信息查询等影响工作效率等工作。
* 针对企业内随时可能出现的业务变更及时进行流程更改。

## 国内外研究现状

### 工作流管理技术的发展

认识 阶 段 (1989-1992):人们对于工作流过程建模和过程的执行有初步的了解，开发出能够提供一定工作流功能的工具，并在实践的基础上得出一些理论和产品原型。

第 二 阶 段(1992- 1995):人们在第一阶段所获得的各种成功和失败的经验的基础上，发展了概念模型和系统体系结构。这一阶段明显区别于第一阶段的是独立的工作流引擎的出现，工作流过程模板的定义和应用程序中过程的执行，使工作流过程实现动态的改变成为可能。工作流过程的路由、系统开放性、交互性技术进一步成熟，使相关工作流产品不断涌现。具有标志性意义的是T-作 流管理联盟(WorkflowM anagementC oalition，WRAC)于1994年11月制定了工作流参考模型，使各种工作流系统有了统一的标准。不过在这一阶段，工作流系统的灵活性仍然在很大程度上受到过程定义和执行引擎的限制。

第三 阶 段 (1995至今):随着信息技术的发展，协同工作环境要求有更复杂、更有效、更灵的W RAS，动态自适应性成为这一领域的前沿课题。动态自适应性工作流要求工作流模板可以随着应用环境的需要而进行动态的调节，整个系统能够方便的移植不同的硬件和软件平台上，并易于系统的扩展，增加新的功能。Internet技术的发展与工作流技术相辅相承，基于web的工作流管理系统就是两者相结合的产物。

### 工作流引擎研究

工作流技术是近十年来的一个很重要的研究热点，国内外研究机构和企业开发并研究出了各种的工作流管理技术，包括IB M的MQSeriesW orkflow，A ction技术公司的Met，F ile N et公司VisualWorkFl, Jet Form公司的InTemp，Pa vone公司的Espress国外企业级工作流产品，西安协同、普元EOS、上海泛微等国内企业级工作流产品，以及Jbpm和Activiti等基于java的开源工作流管理系统。

工作流引擎介绍：

1. 基于纯XML技术

* XPDL（Xml Process Definition Language）

XPDL是至今工作流领域最为重要的一个标准,目前大多数工作流引擎是依据该标准设计开发的。

* BPML(Business Process Model Language)

WfMC和BPMI在2002年6月26日宣布将合作制定业务流程和工作流标准，即采用BPML来描述工作流过程，同时采用XPDL所定义的工作流模型。

* OMG的Workflow Management Facility

OMG的Workflow Management Facility联合XPDL的WfMC规范，定义如何将工作流向CORBA转换

* OFBiz

OFBiz提供了一整套的开发基于Java的web应用程序的组件和工具。其中包括实体引擎, 服务引擎, 消息引擎, 工作流引擎, 规则引擎等。

* OBE

OBE 是由Adrian Price主持开发的一个开放源码的Java工作流引擎，支持WfMC规范，基于J2EE实现。

* Shark

Shark是完全根据WFMC规范实施的，可扩展功能的工作流引擎，它利用xpdl来定义流程，同时还包括服务器端的用于活动节点执行的WFMC工具代理API。

1. 基于Web服务技术

* WSCI

2002年6月26日，BEA,Intalio,SAP,Sun四家公司提出了基于xml的WSCI规范，WSCI是第一个基于Web服务技术的规范。

* ebXML

ebXML是一组支持模块化电子商务框架的规范。ebXML支持一个全球化的电子市场，它使得任意规模的企业通过交换基于XML的信息，不受地域限制地接洽和处理生意。ebXML是联合国（UN/CEFACT，贸易促进和电子商务中心）和OASIS（结构化信息标准发展组织）共同倡导、全球参与开发和使用的规范。

* BPEL

2002年8月9日，Microsoft, BEA, IBM, SAP & Siebel联合提交发布了BPEL规范。 BPEL统合了各种技术（ XLANG, WSFL, BPML）。此规范描述如何处理输入的消息，它不是一个关于业务流程规格化定义的规范。

* OpenebXML

OpenebXML项目致力于提供一个ebXML框架，主要支持 UN/CEFACT和OASIS发布的ebXML规范2.0版。

* Bonita

Bonita是一个符合WfMC规范、灵活的协同工作流系统。Bonita基于浏览器、使用SOAP和XML数据绑定技术的Web Services封装了已有的工作流业务方法并将它们以基于J2EE的Web Service形式发布。

* Twister

Twister的目标是提供新一代、易集成、应用Java领域中最新成果、面向B2B的工作流解决方案。流程引擎基于BPEL业务流程规范和Web Service标准。

* ActiveBpel

Twister的目标是提供新一代、易集成、应用Java领域中最新成果、面向B2B的工作流解决方案。流程引擎基于BPEL业务流程规范和Web Service标准。

1. 基于其他技术

* OSWorkflow

一个开放源码的非常灵活的工作流引擎。

* OpenWFE

OpenWFE是一个开放源码的Java工作流引擎。 它的思想来源于 Scheme，包括可升级的三个组件：引擎、工作列表和Web界面。

* jBpm

jBpm是tom baeyens编写的一个灵活可扩展的工作流管理系统。Activiti 是由 jBPM 的创建者 Tom Baeyens 离开 JBoss 之后建立的项目，构建在开发 jBPM 版本 1 到 4 时积累的多年经验的基础之上，旨在创建下一代的 BPM 解决方案。Activiti5和jbpm4的api有这将近60%的相同，从而使Activiti5能依托jbpm4的强大用户群。本系统采取的即为最新版的Activiti5.11进行的开发。Activiti项目是一项新的基于Apache许可的开源BPM平台，旨在提供支持新的BPMN 2.0标准。下图为Activiti5和Jbpm5的对比：

|  |  |  |  |
| --- | --- | --- | --- |
| 序号 | 技术组成 | Activiti5 | jBPM5 |
| 1 | 数据库持久层ORM | MyBatis3 | Hibernate3 |
| 2 | 持久化标准 | 无 | JPA规范 |
| 3 | 事务管理 | MyBatis机制/Spring事务控制 | Bitronix，基于JTA事务管理 |
| 4 | 数据库连接方式 | Jdbc/DataSource | Jdbc/DataSource |
| 5 | 支持数据库 | Oracle、SQL Server、MySQL等多数数据库 | Oracle、SQL Server、MySQL等多数数据库 |
| 6 | 设计模式 | Command模式、观察者模式等 |  |
| 7 | 内部服务通讯 | Service间通过API调用 | 基于Apache Mina异步通讯 |
| 8 | 集成接口 | SOAP、Mule、RESTful | 消息通讯 |
| 9 | 支持的流程格式 | BPMN2、xPDL、jPDL等 | 目前仅只支持BPMN2 xml |
| 10 | 引擎核心 | PVM（流程虚拟机） | Drools |
| 11 | 技术前身 | jBPM3、jBPM4 | Drools Flow |
| 12 | 所属公司 | Alfresco | jBoss.org |

### BPMN2.0研究现状

#### 1、BPMN定义

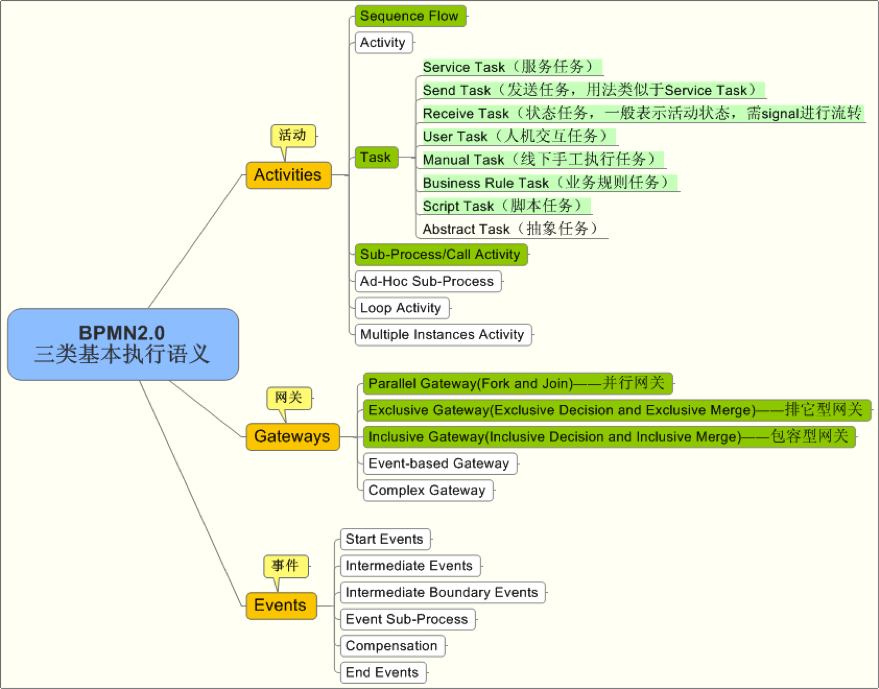
“通过建模、自动化、管理和优化流程，打破跨部门跨系统业务过程依赖，提高业务效率和效果”。2011年新成立的WFMC联盟发布了BPMN2.0新规范，为各工作流产品互容互通提供了统一的标准

#### 2、BPMN2.0规范

BPMN2.0对流程执 行语义定义了三类基本要素：

* Activities（活动）——在工作流中所有具备生命周期状态的都可以称之为“活动”，如原子级的任务（Task）、流向（Sequence Flow），以及子流程（Sub-Process）等。
* Gateways（网关）——顾名思义，所谓“网关”就是用来决定流程流转指向的，可能会被用作条件分支或聚合，也可以被用作并行执行或基于事件的排它性条件判断。
* Events（事件）--在BPMN2.0执行语义中也是一个非常重要的概念，像启动、结束、边界条件以及每个活动的创建、开始、流转等都是流程事件，利用事件机制，可以通过事件控制器为系统增加辅助功能，如其它业务系统集成、活动预警等。

这三类执行语义的定义涵盖了业务流程常用的Sequence Flow（流程转向）、Task（任务）、Sub-Process（子流程）、Parallel Gateway（并行执行网关）、ExclusiveGateway（排它型网关）、InclusiveGateway（包容型网关）等常用图元，如图



# 相关概念和技术

用活动及活动之间变化的过程表示业务流程就是工作流，工作流的概念起源于生产组织和办公自动化领域，它是针对日常工作中具有固定程序的活动而提出的一个概念，目的是通过将工作分解为定义良好的任务、角色，按照一定的规则和过程来执行这些任务并对他们进行监控来提高办事效率、降低生产成本、提高企业生产经营管理水平和企业竞争力。

## 工作流相关概念

### 工作流(Workflow)

按照 WfMC的定义，工作流是一类能够完全或者部分自动执行的业务过程，根据一系列过程规则，文档、信息或任务能够在不同的执行者之间传递、执行。

### 业务流程（Business Process）

在功能确定的组织结构中，能够实现业务目标和策略的相互连接的过程和活动集。

### 过程定义（Process Definition）

业务过程的形式化描述，用来支持系统建模和运行过程的自动化。过程可分解为一系列的子过程和活动，其中定义包括描述过程起始、终止的活动关系网络，以及一些关于个体行为的信息，具体而言，即构成过程的活动以及各活动的关系、组织成员的角色、应用中的数据结构等。

### 过程定义工具（Process Definition Tools）

过程定义工具被用来创建计算机可处理的业务过程描述。它可以是形式化的过程定义语言或对象关系模型，也可以是简单地规定用户间信息传输的一组路由命令。

### 活动（Activity）

业务过程的一个执行阶段，由执行者完成。执行者可以是人、软件系统或二者的集合。活动是过程执行中可被工作调度的最小工作单元，要求有人或机器参与。

### 过程/活动实例（Process/Activity Instances）

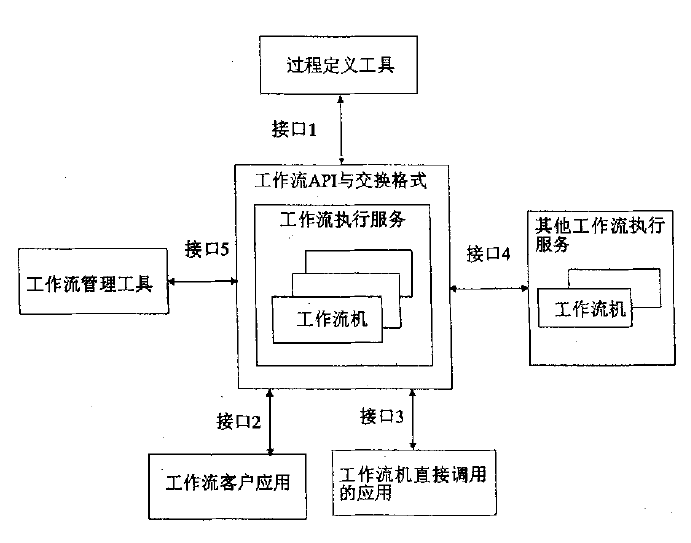
指的是一个工作流过程的具体执行。在过程实例的执行中，工作流引擎将负责解释对应的过程（和它包含的活动）定义，动态生成活动实例，并根据过程定义中的规则控制协调这些活动实例之间的执行顺序，同时完成活动之间的数据传递。

### 工作流管理系统（Workflow Management System, WFMS）

定义、创建、执行工作流的软件系统。在最高层次上，WFMS应能提供以下三个方面的功能支持：建造功能，对工作流过程及其组成活动进行定义和建模；运行控制功能，在运行环境中管理工作流过程，对工作流过程中的活动进行调度；运行交互功能，指在工作流运行中，WFMS与用户（业务工作的参与者或控制者）及外部应用程序交互的功能。大部分的工作流产品都实现了WFMC工作流参考模型的接口1、接口2、接口3和接口5：

* 接口1，流程设计器：包括了两种类型的设计器，一种是基于Web的设计器，实现技术包括了Swing和Flex，一种是基于Eclipse插件的本地应用实现。除去普元之外，大部分工作流产品都选择实现了一种类型的设计器。Web设计器的好处在于对最终用户友好，基于Eclipse的设计器的好处在于对开发人员友好，能够比较容易的进行单元测试和流程测试，缺点则是基本上隔绝了最终用户对工作流的使用，将工作流死死限制在开发者的层次上。
* 接口2，工作项客户端接口：通过API暴露调用和交互接口，完成工作项的列表展现、拾取、退回和提交。
* 接口3，外部应用调用接口：基本上都没有对主流ERP、企业管理软件和财务软件进行集成的专有支持，这和国内工作流产品应用的场景有关系，工作流多作为支持单个应用的嵌入式使用，在这一点上天翎提供有与SAP的集成接口。大部分通过支持Web服务调用进行支持。
* 接口5，管理控制台：包括两部分，一部分是对运行中的案例进行监控和干预，包括了案例的中止、挂起与恢复，任务的中止、跳过、挂起与恢复，参与者的重新指定和催办，工作流变量的修改查看等；一部分是对案例的统计与分析，包括了针对案例、任务的时间统计，针对参与者的任务效率统计等。

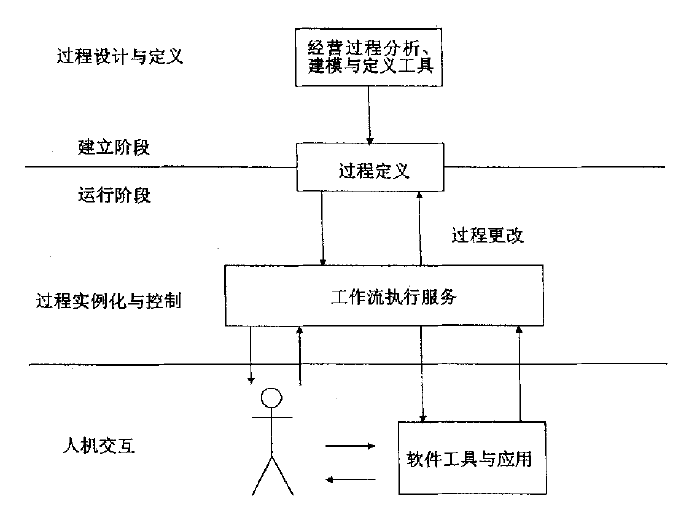
下图为WFMC提出的工作流参考模型：



虽然不同的工作流管理系统具有不同的应用范围和不同的实施方式，但他们还是具有许多共同的特性。从比较高的层次上来抽象地考察工作流管理系统，可以发现所有的工作流管理系统都提供了三种功能：

* 建立阶段功能:主要考虑工作流过程和相关活动的定义和建模功能。
* 运行阶段的控制功能:在一定的运行环境下，执行工作流过程，并完成每个过程中活动的排序和调度功能。
* 运行阶段的人机交互功能:实现各种活动执行过程中用户与IT应用工具之间的交互

下图给出了工作流管理系统的三个主要功能之间的关系：



### 工作流引擎（Workflow Engine）

工作流管理和应用系统的核心软件部分，为过程实例提供运行时环境和服务，可以形象地比作过程执行的发动机。工作流引擎接口向外部提供了支持、控制过程（实例）执行的功能。这些功能包括：解释过程定义；创建和管理过程实例，控制流程的运行状态，如开始、停止、挂起、重启等。

### 工作项（Work Item）

过程实例中要被参与者执行的工作，它与活动定义与活动实例相关。工作项一般需要人工的参与，它的完成将推动过程的进展。

### 工作列表（Work Items List）

与业务过程的某个参与者相关的一系列工作项的集合。工作列表的生成需要工作流引擎和工作流列表处理程序共同参与。

## Activiti工作流步骤

### 加载（发布）流程定义

通过jbpm的designer插件，或者是用其他工具，制定出processDefinition，然后将其加载到应用中的过程。这个加载可以是写入内存中，或者是直接写入数据库等。

### 启动流程

创建流程实例的过程。具体创建实例的方法有多种，可根据自己的需要自行选择。

### 处理任务

在流程流转的过程中，JBPM引擎会为我们生成任务的实例，我们就需要针对这些任务实例来进行处理，然后结束这些任务实例，并推动流程的流转。

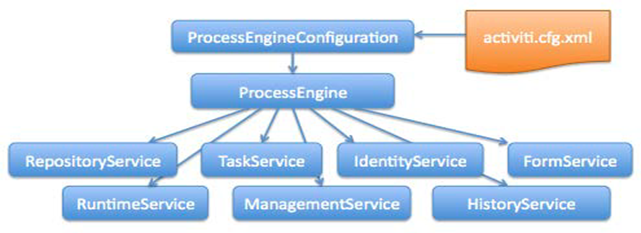
### 记录流程的相关状态

记录流程状态这点包括且不限于以下内容：

1. 流程实例的开启
2. 任务实例的创建
3. 任务实例的开始执行
4. 任务实例的结束
5. 流程实例的结束

## Activiti5 API应用

ProcessEngine是Activiti系统的核心接口，七类基础服务接口通过ProcessEngine获取，均采用链式API方式。



### RepositoryService：

流程资源服务的接口，主要用于对流程定义的部署、查询和删除操。

### TaskService

任务领取（claiming）、任务完成（completing）和任务指派（assigning），还包括对任务的创建、查询、保存、删除等。

### RuntimeService：

运行时服务主要用于启动或查询流程实例，以及流程变量、当前激活状态活动的查询、流程实例的删除等。

### HistoryService：

流程历史的服务接口。提供对历史流程实例、历史任务的查询和删除操作。

### IdentityService：

用户、组管理服务接口，用于管理Group、User的增删改查，并维护Membership。

### FormService：

表单服务用于访问表单数据以及在启动新的流程实例时或完成任务时所需的渲染后的表单，提供UI界面辅助用户填写相关值以保存至流程变量。

### ManagementService：

提供流程管理和控制操作的接口服务，和业务流程的运行没有关联关系，比如查询数据库本身的内容、Activiti的版本及序列生成ID规则等。

## Activiti5设计模式

命令模式能将命令的发出与执行分开，委派给不同的对象，每一个命令都代表一个指令，其最大的好处是提供了一个公共接口，使得用户可以用同一种方式调用所有的事务，同时也易于添加新事务以扩展系统。

Activiti5大量采用了命令模式，在流程运行期间，所有的指令执行（比如流程部署、流程流转、获取任务等）都使用此模式实现， 其中涉及到四个重要的概念：

Command:Activiti5的命令定义接口，仅有一个execute方法，所有运行期要执行的指令都要实现该接口，定义要执行的具体行为。

CommandContext:命令执行的上下文环境，每个Command的执行都依赖其上下文环境，CommandContext创建了命令执行 期间的引擎会话与数据库会话，每个CommandContext都是一个单独的ThreadLocal，执行期间不会受其它线程干预，是线程安全的。

CommandExecutor:命令执行器，负责执行所有的运行时Command。引擎中各项指令的执行（即命令的产生者可能来源于多种对象）都托CommandExecutor处理，仅有一个接口方法：execute(Command command)。 ActivityBehavior:活动行为定义，用于定义BPMN2.0执行语义层的各图元在流程引擎的行为，或称之为所具备的图元特征。与 Command的概念类似，仅仅描述“待执行”的指令是什么，会发生什么样的行为，但真正要执行时则由引擎负责驱动。

人机交互任务是业务流程应用中最常用的业务类型，以BPMN2.0中定义的“Task”这个典型元素说明一下命令模式在Activiti5中的应用：

Activiti5针对BPMN2.0的Task Element定义了Task接口，并依据Semantic.xsd执行语义定义了相关任务元素所具有的行为特性，此行为特性通过 setActivityBehavior方法进行行为与元素的绑定，这些Behavior在流程引擎驱动流转到活动节点时将被触发，通过 execute(ActivityExecution execution)执行ActivityBehavior中指定的操作；

每个活动有若干个Command与之对应，比如ClaimTaskCmd、CompleteTaskCmd、DelegateTaskCmd、 SaveTaskCmd、DeleteTaskCmd等，分别表示任务的领取、完成、转交、保存、删除等，这些操作指令的执行结果通过命令执行上下文 （CommandContext）得到DAO层的TaskManager将任务对象的变更持久化到数据库中；

引擎不关心要执行什么，凡是实现了Command接口的类都可以通过CommandExecutor执行，除了引擎提供的这些原生的任务指令外，如 果业务系统有额外的特性化操作，也可以自定义一组Command，在Command.execute()中自由调用外部服务、发送手机短信、附加任务属 性、调用DAO操作数据库等，封装完毕后交由引擎去执行，即可得到希望的结果。同样，如果在业务系统中需要自定义BPMN元素或属性，仅需同步增加 ActivityBehavior接口的实现，在解析流程定义文件时将自定义的行为实现与元素（属性）帮定，并缓存之，待引擎驱动到达节点时自动执行。在 ActivityBehavior. execute()中依然可以调用各种各样的API已实现特定的业务目的。

此处需要注意的是，Activiti5的CommandContext是包含事务处理的，在每次关闭上下文环境时，会执行事务的提交，但在实际业务 系统中，业务事务、引擎事务以及数据库事务应该是被统一到一个事务中去管理，这就需要将Activiti5的事务与业务系统的事务合并。 Activiti5通过Spring注入提供了该方式的可行性，引擎内部的事务控制可以委托给业务层去处理，在初始化引擎配置时，将业务系统中定义的 DataSource和TransactionManager传递给流程配置的dataSource、transactionManager属性 后，Activiti5内部会使用Spring提供的TransactionAwareDataSourceProxy来封装传进来的 DataSource，并利用外部的事务管理来接管Activiti5的事务控制，确保了从该DataSource获取的数据库连接与 Spring 定义的事务能够完美地结合，从而实现业务系统与引擎系统事务的集成。

# 系统设计

本章从基于请假管理系统的功能需求出发，设计一个基于Activiti的请假流程管理系统。系统的基本目标是利用开源性代码的Activiti流程引擎来构建一个通用的、易于扩展的并且独立于应用系统的流程管理系统模型。该模型主要应用于企业内员工的请假审批流程，该系统模型可以达到以下几个设计目标：

* 基于Activiti开发一套流程的部署、启动、执行、查询和查看功能的系统。
* 通过调用Activiti未开放的底层api实现流程的跳转、会签、转发、动态设置参与者。
* 使用 UEL 作为表达式获取权限管理系统中的用户角色信息中制定的信息。
* 是否能够独立于应用系统之外进行流程的启动、执行和查询。
* 开发一套流程管理系统，能够支持定义好的流程文件的上传、部署、启动，通过简单的配置即可针对相应的业务系统进行流程的执行。

下面首先从基于Activiti的流程管理系统的应用需求进行分析，接着进行系统整体结构的设计，最后对主要功能模块进行分析设计。

## 系统需求分析与设计原则

### 系统需求分析

为了实现基于Activiti的流程管理系统的功能，系统的整体设计应该满足以下几点需求：

* 统一接口

基于Activiti，方便进行接口的调用，封装出符合系统需要的调用接口。并实现Activiti未提供的流程的跳转、会签、转发、动态设置参与者等接口。

* 可重用性

该系统应该具备的可重用性，系统通过配置功能，可方便地为业务功能增加流程支持，使其能方便的为用户提供流程管理服务。

* 文件上传校验

根据系统提供的流程文件的上传功能进行流程文件的上传，针对预定义的错误项进行正确提示。。

* 处理多种格式的流程文件

系统可以为方便用户的使用，提供了支持BNPN和BPEL两种流程定义文件的部署。

* 流程查看

系统提供相应的流程图和流程定义文件的查看，以及查看办理历史和办理意见等信息。

* 响应速度

由于存在调用Activiti底层未开放的Api方法，不影响系统功能的情况下提高系统的相应速度。

* 易用性

简单明了的操作提示，完整的待办已办信息查询，方便的请假信息查询。

基于Activiti请假流程管理系统能够为系统管理员提供完整的流程定义、管理以及查询功能，能够为企业员工提供方便的请假审批操作。这就需要本系统能完成以下功能：

* 请假流程文件的创建、修改和上传，并能实现流程的部署、启动和流转。
* 系统用户、部门、角色和功能的管理，方便企业用户信息的安全和管理。为流程定义提供人员选择接口。
* 能够进行流程的提交、回收及选择办理人员等。
* 能够进行待办、已办信息的查询，并能针对未办事宜进行提醒。
* 能够查看流程的执行情况，各级办理人员的审批意见等信息。

### 系统设计原则

模块化原则：为了使最终的请假流程管理系统达到设计和实现上的可重用性，本系统将采用模块化方式，将工作流系统和业务系统进行划分，从而使工作流系统可以实现相对独立的功能。同时管理系统、业务系统和工作流系统能够实现通过简单配置即可进行系统开发要求。

透明性原则：系统需要在用户提交到下一环节的时候获取可提交人员列表信息，选择该部门可办理工作人员的信息，并直接默认将该办理信息进行签收。

## 系统体系结构

本系统通过MVC分层开发和模块化管理的方式，将系统需求分解，从而将复杂的问题分割成为几个子问题，分别限制在不同的模块中完成；每个模块各司其职，使系统结构设计更加清晰，同时每个功能模块得到了简化，提高了系统整体的灵活性和健壮性。

### MVC分层

系统采用Struts2、Spring、Hibernate、Activiti等技术进行搭建，使用了当今JavaEE流行的MVC开发模式，即模型层（Model）、视图层（View）和控制器层（Controller）三个部分：

* 模型（Model）：负责封装应用的状态，并实现应用的功能。通常分为数据模型和业务逻辑模型，数据模型用来存放业务数据，比如请假信息、用户信息等，本系统采用hibernate实现数据库表与实体对象的映射；而业务逻辑模型包含应用的业务操作，比如请假单的添加、修改和删除等业务逻辑操作。
* 视图（View）：用来将模型的内容展现给用户，用户可以通过视图来请求模型进行更新。视图从模型获得要展示的数据，然后用自己的方式展现给用户，相当于提供界面来与用户进行人机交互；用户在界面上操作或者填写完成后，会点击提交按钮或是以其它触发事件的方式，来向控制器发出请求。Struts2提供了丰富的标签库来实现复杂的表单组件。
* 控制器（Controller）：使用基于拦截器的Struts2来控制应用程序的流程和处理视图所发出的请求。当控制器接收到用户的请求后，会将用户的数据和模型的更新相映射，也就是调用模型来实现用户请求的功能；然后控制器会选择用于响应的视图，把模型更新后的数据展示给用户。

### 模块化管理

本系统采用模块化管理方案，将系统划分为系统管理、请假信息管理和流程管理三部分，能够实现三个模块的松耦合，提高系统功能的易用性和可重用性：

* 用户管理系统

系统用户，部门，角色，功能，系统字典等功能的管理。

* 业务管理系统

提供请假单业务功能的管理。

* 流程管理系统

提供流程文件的上传、部署、与请假信息的绑定管理、流程的执行、待办已办信息的查询、流程图和流程文件的查看等功能。本系统设计的流程存在以下几种基本状态：

文件状态：使用Eclipse提供的Activiti插件创建流程文件，流程文件创建完成后通过系统提供的上传接口，上传到服务器制定目录下。

流程初始化状态：通过调用Activiti提供的流程部署方法创建过程实例。

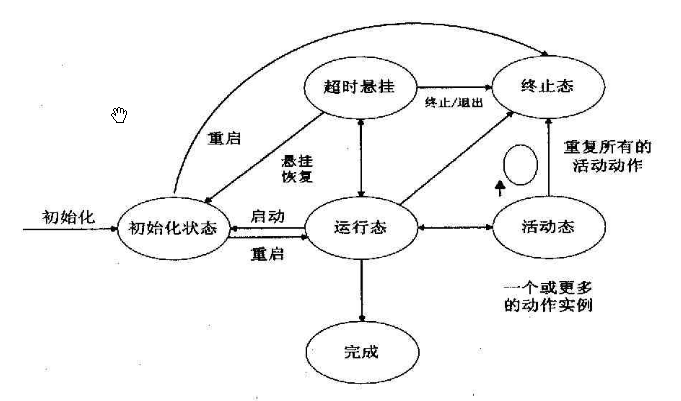
流程运行态：已经启动的流程实例可以进行流程的流转、流程相关业务信息、流程图信息以及流程文件信息的查询。Activiti在流程部署的时候会创建流程图文件并且通过流程定义的文件可以获取流程节点的坐标信息和办理相关信息，从而动态模拟出流程图信息界面。

流程的挂起状态：提供流程的挂起，即流程实例处于静止的状态。流程暂时无法继续进行流转操作。

流程完成状态：按照流程设计执行完流程后，该流程实例进入完成状态。此时流程执行完成。

流程终止状态：在任务正常完成之前，流程实例的执行已经停止。

下图为流程实例状态转换示意图：



## 开发工具

### 开发语言：

本系统选用Java语言。Java具有跨平台、面向对象、高可靠性的特点。同时Java有丰富的基础类库支持，以及大量支持服务器开发的类库。再者，Java语言的流行使得许多工具都基于Java或提供了Java接口，如本系统要使用的Lucene搜索引擎工具包的源码就是使用Java编写的，使用Java语言调用Lucene提供的方法非常方便。这些类库及工具的支持大大简化了开发过程，提高了开发效率。

### 开发工具

本系统开发采用Mycelipse for spring 8.6版本，该集成了spring的所有特性，数据库采用mysql5.1.3版本，操作系统是window7，数据库客户端采用SQLyog\_Enterprise，项目设计采用Microsoft Visio 2010，数据库设计采用Powerdesigner v15.1版本，项目原形使用Axure RPAxure5。

## 系统架构

本系统采用现在java流行的Struts2、Spring、Hibernate的java EE轻量级企业解决方案的框架进行开发。

### 数据库层配置

本系统通过Hibernate进行数据库操作，在与spring整合的配置文件中使用DBCP进行数据库连接配置，主要配置参数包括数据库连接驱动、数据库链接地址、用户名、密码、连接池初始连接数、连接池最大值、连接超时时间等参数。Hibernate的sessionFactory配置实体类扫描路径、数据库连接方言、控制台是否显示sql等参数，实体类中通过@Entity标注进行配置。声明通过Transactional的全标注的方式进行事务管理，在需要使用事务的方法前增加标注@Transactional来进行事务控制。

### 依赖注入配置

在spring的配置文件中通过component-scan标签进行依赖注入扫描路径配置，通过使用通配符的方式制定需要注入的Dao层和Service层类，在需要依赖注入的类前只需要加入@Component , @Controller , @Service , @Repository等标注即可。在需要使用注入类的地方通过@Autowired标注进行注入。

### 控制层配置

通过在项目的入口web.xml中配置Struts2的拦截过滤器，拦截所有需要Struts2处理的请求路径。在struts.xml配置文件中配置Struts2自定义访问包，通过通配符方式指定访问路径对应的处理Action，从而将客户端请求提交给服务器后台处理。使用json-plugin插件进行java对象到json对象的自动转换，该包继承自json-default包。

## 模块的详细设计

### 问题分析

Activiti是一套基于java的开源工作流项目，通过该工作流引擎我们可以进行流程文件的定义，流程的部署、启动，通过businessKey参数与业务系统进行绑定，实现了流程的正常流转、分支、并行、多实例、子流程等日常流程逻辑，但是该流程引擎无法支持流程的跳转、会签、转发、动态设置参与者等特殊的业务功能，完成这些功能需要自定义工作流接口通过调用Activiti未开放出来的接口。

* 回退：通过Activiti的API中提供的动态创建流向的方法，创建上一个流程节点的流向，删除原流向，然后执行流程的提交后恢复原流程的流向即可实现流程的回退功能，需要扩展一个操作历史记录表。
* 会签：通过Activiti的多实例方案，动态设置多人到流程会签节点上，通过并行或者串行的流程执行方式执行会签操作。支持串行会签和并行会签的处理方式。
* 转发：通过Activiti的API中提供的设置人员的方法进行任务的转发。
* 动态设置参与者：通过UEL表达式中的方法表达式，通过使用流程变量和动态参数的方式进行动态的参与者设置。流程节点的处理参与者可以指定为特定组织机构、特定部门/子部门、特定角色、特定用户。

### 工作流系统的设计

工作流引擎室工作流系统的核心部分，它保证了整个系统的自动运行，各种过程实例的执行也是由它负责，在本系统中将会对Activiti提供的流程运行方式进行部分改造：

* 流程文件上传：提供流程文件选择界面，将上传的流程文件保存在系统临时目录下，然后调用Activiti的流程部署方法将流程文件发布到Activiti的系统表中。在已上传的列表中查询已经部署的流程，并提供流程图的查看以及流程文件的查看功能。
* 流程与业务绑定：首先提供业务功能的注册功能，将业务实体类名称、对应中文名称、类文件的路径信息进行录入到业务系统表中。录入后的业务功能可以选择已经部署的流程文件并选择绑定的业务实体类属性和值以方便在业务系统录入数据后可以获取到相应绑定的工作流程。
* 流程启动：在绑定的业务系统请假单界面判断是否有绑定流程，如果有绑定流程则显示流程相关按钮，否则用户看到的只是普通的请假单查看页面。点击流程启动后可以创建流程实例并绑定请假单业务id。
* 待办信息查询：根据用户id获取Activiti系统表中的待办信息列表。
* 任务办理：根据绑定的业务id可以查看业务表单信息，通过Activiti提供的流程实例操作相关api获取下一节点的可办理人员，选择相应的办理人员并填写完审批意见和信息后将流程实例id和审批信息保存到自定义审批表中即可实现整个请假流程的流转。

### 请假业务系统设计

将设计好的请假业务系统录入到系统业务系统表中，提供请假单相关的请假标题、请假内容和请假日期等表单信息的录入、修改、删除和查看等功能。

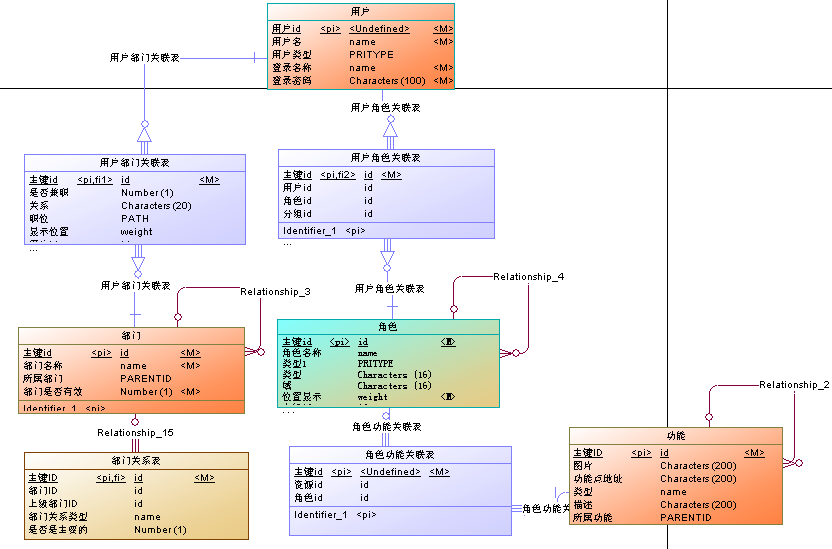
### 用户管理系统设计

* 系统功能管理模块：提供系统功能的添加、修改和删除的功能，从而可以通过权限配置获取用户可以使用的功能。
* 用户管理：提供新用户的添加、修改和删除等功能，方便用户信息的录入和修改，为系统登录提供用户名和密码验证服务。
* 部门管理：提供企业分级部门管理，通过树形方式展示部门组织机构信息，提供部门信息的添加、修改和删除功能。
* 角色管理：提供用户角色的添加、修改、删除和用户信息的关联等功能。
* 角色授权管理：为角色制定可使用的功能菜单信息，在用户登录后可以通过用户信息获取用户可以操作的功能菜单信息，从而打开相应权限的界面。
* 系统字典管理：为系统内的性别、学历、类别、证件类型等基本固定内容提供字典支持。该功能包含两个子功能，分别为字典主项管理和字典子项管理，提供了字典大类和细类的管理。

详细设计根据前一章对系统需求的讨论，对系统实现目标的确立以及对系统功能的设计，本章将首先对系统实现中的开发环境和技术基础进行说明和介绍，并重点介绍基于基于Activiti的请假流程管理系统主要模块的实现，详细阐述各个核心功能模块的实现细节，给出关键流程的实现。

## 系统管理

系统管理的逻辑模型：



## 业务功能数据库设计

提供主要业务逻辑功能的数据库设计，涉及到的表包括请假信息相关表和流程与业务整合关联相关表：

病假表(sick)

|  |  |  |
| --- | --- | --- |
| 名称 | 类型 | 描述 |
| Id | Varchar2(32) | 主键id |
| sickTitle | Varchar2(50) | 病假标题 |
| sickContent | Varchar2(200) | 病假内容 |
| fromDate | Date | 请假开始时间 |
| endDate | Date | 请假结束时间 |
| sickTime | Number | 请假时长 |
| createDate | Date | 请假日期 |
| processInstanceId | Varchar(32) | 流程实例id |

审批表(sickApp)

|  |  |  |
| --- | --- | --- |
| 名称 | 类型 | 描述 |
| Id | Varchar2(32) | 主键id |
| userId | Varchar2(32) | 用户id |
| Ispass | Char(1) | 是否通过 |
| Result | Varchar2(50) | 审批意见 |
| CreateDate | Date | 审批时间 |

业务表(busTable)

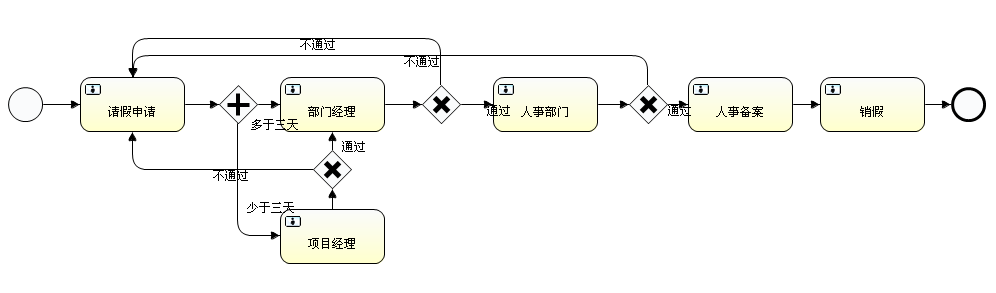
|  |  |  |
| --- | --- | --- |
| 名称 | 类型 | 描述 |
| Id | Varchar2(32) | 主键id |
| className | Varchar2(32) | 实体类英文名 |
| Name | Varchar2(32) | 实体类中文名 |
| classPath | Varchar2(32) | 实体类包名 |
| CreateDate | Date | 创建时间 |

业务流程绑定表

|  |  |  |
| --- | --- | --- |
| 名称 | 类型 | 描述 |
| Id | Varchar2(32) | 主键id |
| Key | Varchar2(32) | 流程定义key |
| busId | Varchar2(32) | 业务表主键id |
| busPro | Varchar2(32) | 绑定的属性名称 |
| busValue | Varchar2(32) | 绑定的属性值 |
| CreateDate | Date | 创建时间 |

## 流程设计

使用eclipse的Activiti插件进行基于BPMN2.0的流程图设计：



## 具体实现

### 流程文件上传模块

使用jquery GooUploader组件进行多文件异步上传，该上传组件式基于swf插件来实现的文件上传，后台需要通过File对象进行接收并保存到工程的deployments目录下。Activiti支持三种类型流程文件的部署，包括xml、zip、bar格式的文件。通过判断文件的扩展名进行判断调用Activiti的哪个api方法进行流程部署

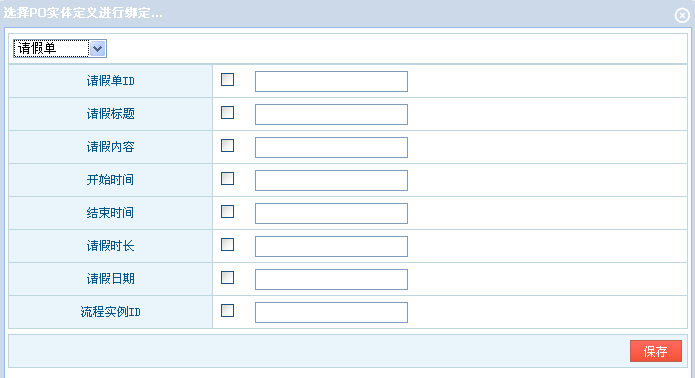
|  |
| --- |
| //截取文件的扩展名进行判断  String ext=bpmnPath.substring(bpmnPath.lastIndexOf(".")).toLowerCase();  if(".xml".equals(ext)){  //部署xml格式的流程文件  return repositoryService.createDeployment().addClasspathResource(bpmnPath).  deploy();  }else if(".zip".equals(ext)){  //部署zip保存的流程文件和流程图片  ResourceLoader resourceLoader = new DefaultResourceLoader();  Resource resource = resourceLoader.getResource(zipPath);  InputStream inputStream = resource.getInputStream();  if (inputStream == null) {  throw new BusinessAppRunTimeException("流程部署文件不存在");  }else {  ZipInputStream zis = new ZipInputStream(inputStream);  Return repositoryService.createDeployment().addZipInputStream(zis).deploy();  }  }else if(".bar".equals(ext)){  //部署bar格式压缩包的流程文件  ResourceLoader resourceLoader = new DefaultResourceLoader();  Resource resource = resourceLoader.getResource(barPath);  InputStream inputStream = null;  try {  inputStream = resource.getInputStream();  if (null == inputStream) {  logger.info("发布的文件为空 "+barPath);  return null;  }else {  ZipInputStream zis = new ZipInputStream(inputStream);  return repositoryService.createDeployment().addZipInputStream(zis).  deploy();  }  }else{  throw new BusinessAppRunTimeException();  } |

### 业务流程绑定模块

查询流程部署信息列表：

|  |
| --- |
| //获取所有已经部署的流程信息列表  repositoryService.createProcessDefinitionQuery().orderByDeploymentId().desc().list; |

在流程定义的查看页面提供流程的自定义主键key值进行业务功能绑定，通过选择相应的请假业务表，并制定绑定的属性和值。



### 工作流模块

在请假单的新增和修改页面中判断该业务功能是否已经绑定了流程，如果绑定了则显示提交按钮，可以讲业务表单提交流程进行流转操作，否则只有保存按钮。

如果用户只点击了保存按钮，则该请假单只是作为普通的业务功能进行保存，不提交流程进行审批，当用户点击提交按钮的时候会把请假单ID传递到流程管理系统中，在流程管理系统中会通过业务功能名称获取绑定的流程key，通过调用Activiti的Api流程启动方法进行流程启动：

|  |
| --- |
| //用于将参数保存到流程系统的流程变量中  Map<String,Object> variables=new HashMap<String,Object>();  //将启动流程的用户id作为流程变量保存到流程系统中  variables.put("userId", userId);  //key为流程定义id，bussnessKey为业务表单id  runtimeService.startProcessInstanceByKey(key,businessKey,variables); |

根据系统中的用户id查询用户的待办信息列表，再通过流程引擎提供的业务表单id补充相应的业务信息后，提醒用户进行待办信息办理

|  |
| --- |
| //根据传入的用户id，分页的起始和结束号进行待办列表查询  List<Task>list=taskService.createTaskQuery().taskAssignee(userId).orderByTaskPriority().desc().orderByTaskCreateTime().desc().listPage(firstResult, maxResults)；  //获取流程文件中指定的待办查看页面的路径  String url=formService.getTaskFormData(taskId).getFormKey()； |

打开获取到的待办信息查看url的页面后，可以查看待办详细信息，并提供流程办理按钮，点击办理按钮获取当前流程所在节点，以及下一个节点和可选办理人员信息，选择完成后提交流程：

|  |
| --- |
| //根据当前流程节点的定义id获取获取流程节点定义相关信息  ActivityImpl activityImpl = ((ProcessDefinitionImpl) processDefinition).findActivity(currentTask.getTaskDefinitionKey());  //获取流程节点的所有流向  List<PvmTransition> pvmTransitions = activityImpl.getOutgoingTransitions();  //遍历所有的流程走向  for (PvmTransition pvmTransition:pvmTransitions) {  //获取下一级可选择的流程节点  ActivityImpl activityImpDes = (ActivityImpl)pvmTransition.getDestination();  //获取流程节点的行为属性，判断流程节点的类型，如果是排他分支则提供用户进行单//选，如果为并行分支或者普通的流程节点则不需要选择流向，如果是包容分支则//可以多选。  ActivityBehavior activityBehavior = activityImpDes.getActivityBehavior();  //获取流程线上的属性信息  Object objectValue = pvmTransition.getProperty(BpmnParse.PROPERTYNAME\_CONDITION\_TEXT);  //获取线上表达式的值，提供给排他分支和包容分支的走向的判断条件  String sequenceFlowConditionExpression = (String)objectValue; //得到线上表达式的值  } |

通过流程节点定义对象可以获取流程节点上设置的UEL表达式，并通过调用Activiti内置的执行方法获取UEL表达式指定的系统接口返回的人员列表，从而可以实现办理人员的选择功能：

表4.6 TestTokenizer类中的方法及其功能

|  |
| --- |
| //获取activiti:candidateUsers（进行任务的指派）的属性值（获取UEL表达式）  Set<Expression> expressions = taskDefinition.getCandidateUserIdExpressions();    if (expressions!=null&&expressions.size()!=0) {  expression = expressions.iterator().next();  //防止报空指针异常  Context.setProcessEngineConfiguration(processEngineConfiguration);  Context.setCommandContext(new CommandContext(null, processEngineConfiguration));  //获取UEL表达式的值  Object expressionValue = expression.getValue((ExecutionEntity) processInstance);  //移出“防止报空指针异常”设置的参数  Context.removeProcessEngineConfiguration();  Context.removeCommandContext();  } |

流程走向和办理人员选择完成后就可以通过taskService的complete方法提交给流程的下一个办理人员进行处理，从而继续循环从待办信息到办理的过程直至流程执行完。

# 系统实验设计与分析

在第三章中，对基于Activiti的请假流程管理系统做了详细的分析；第四章接着给出了系统中核心功能模块的实现。本章将对本系统的功能和性能进行测试和分析。首先介绍实验环境，然后分别针对系统的功能和性能设计具体的实验，实施实验并分析测试结果，验证本系统的可用性。

## 实验环境

实验在本地计算机上进行，搭建Web应用服务器，本地浏览器访问。其硬/软件环境明确如下：

1.硬件环境：

CPU：Intel(R) Core(TM) i5-2410QM @2.30GHz

内存：4g

硬盘：500GB

2.软件环境：

操作系统：Windows 7

Java运行环境：JDK6、MyEclipse 8.6

Web服务器：Tomcat6

## 请假申请

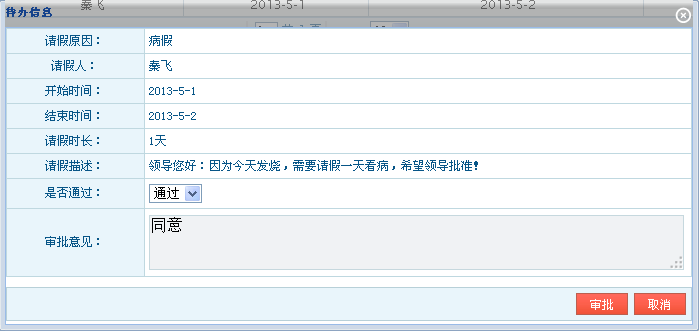
创建请假单：



获取待办列表：



填写审核结果和审批意见：



选择流程流向和办理人员：



经测试该系统可以实现正常的请假申请，待办信息查询，意见的审批，流程流向和可办理人员的选择等功能，符合最初设计要求。

## 小结

本章在系统设计与实现的基础上，通过模拟请假申请验证了本系统的可用性。该系统解决了Activiti提供的api中无法选择流程流向和办理人员的指定问题，实现了Activiti流程与请假业务系统的松耦合式绑定，方便了用户的使用和系统的管理。

# 总结与展望

## 工作总结

随着计算机产业的迅速发展以及企业规模的扩张，传统的企业事务管理将会变得越来越繁琐和臃肿，无纸化和信息化将是未来的趋势。大量的重复的、复杂的审查审核流程将会更多依赖于工作流为基础的流程管理系统来处理，这正是本论文重点需要讨论的内容。

本论文的主要工作成果包括如下：

1. 从课题的研究背景出发，分析了当前主要的工作流引擎，了解了工作流在流程软件中的重要作用，以及未来工作流的发展趋势，特别是基于java的开源工作流平台Activiti将会有更好的发展；
2. 基于课题背景的研究，指定了本系统的设计目标与原则，设计了基于Activiti的请假流程管理系统。从分析本文所述系统的基本功能需求入手，明确系统功能要求，根据系统的设计目标、原则和方法，将整体系统分解为模块化结构，对拆分完成的系统子模块需求进行具体分析，并讨论其实现方案和使用技术。基于模块化的设计原则，使得系统具有灵活性、易用性、可移植性等优势；
3. 基于系统的需求设计，讨论了基于Activiti的请假流程管理系统的具体技术细节，并最终进行了系统实现。该系统大大的提高了企业请假审批速度，提高了企业的办事效率；
4. 通过具体的实验场景，通过实际的测试用例对系统的功能和性能进行测试，验证了该系统在实际应用中良好的可用性和性能表现。系统可以完全符合企业的流程审核、流程的提交、任务的回收等功能。

## 工作展望

本文设计并实现的基于Activiti的请假流程管理系统已具备一定的可用性和稳定性，但由于时间有限，仍然有很多方面需要改进和进一步地研究：

1. 未使用Activiti自带的权限管理功能，而是采用了UEL表达式的方式直接通过自定义权限系统进行流程的流转，不完全符合Activiti的设计初衷。
2. 调用了Activiti未发布的底层api接口来实现任务的回收，流程的任意流转，在处理上需要更加优化。
3. 系统提供的请假流程还是很简单的，没有考虑更多可能存在的复杂业务逻辑。

参考文献

1. 范玉顺等. 《工作流管理技术基础》. 第1版，清华大学出版社，2001
2. 范玉顺，吴澄 《工作流管理技术研究与产品现状及发展趋势》. 计算机集成制造系统CIMS，2000-6
3. Rademakers, Tijs. 《Activiti in Action》、Manning社区，2012
4. Tom Baeyens.《Activiti 5.11 用户指南》. Activiti社区，2012
5. [Rod Johnson 等](http://search.china-pub.com/s/?key1=Rod+Johnson+%b5%c8) .《Spring框架高级编程》. [机械工业出版社](http://www.china-pub.com/love/chubanshe/chubanshe.asp?newid=101)，2006
6. 存在问题：
7. 1、论文已经比第一版有了很大的进步
8. 2、论文有些章节内容还是把握不准确，建议再次认真推敲和修改