

基于 J2EE 框架的工作流管理系统设计与实现

伍庆林, 王加阳

(中南大学 信息科学与工程学院, 湖南 长沙 410083)

摘要:随着应用需求的迅速增长, EJB 技术已经不能满足中小型企业工作流管理系统的开发要求, 而 J2EE 框架技术能更好地适应中小型企业级应用。在分析工作流管理联盟(WFMC)提出的工作流参考模型的基础上, 使用轻量级框架 Spring, 并结合 MVC 框架 Struts 和数据持久化框架 Hibernate, 设计与实现了一种高效、易用的工作流管理系统。文中重点介绍了系统的多层结构, 并对系统中的模型定义工具、用户组织工具、工作流引擎模块进行介绍与探讨。

关键词:工作流管理系统; J2EE 框架; 多层结构; spring; hibernate

中图分类号: TP311.52

文献标识码: A

文章编号: 1673-629X(2008)04-0107-04

Design and Implementation of J2EE Framework - Based Workflow Management System

WU Qing-lin, WANG Jia-yang

(College of Information Science & Engineering, Central South University, Changsha 410083, China)

Abstract: Along with the quick increase of application requirement, EJB technology has not satisfied the development requirement of the workflow management system in the middle and petty companies, while J2EE framework technology can adapt better to the small and medium enterprise applications. On the basis of analysis the workflow reference model of WFMC, design and implement an efficient workflow management system based on lightweight framework Spring in combination with MVC framework Struts and persistence framework Hibernate. The multi-layer structure of the system is mainly introduced in the paper. Simultaneously, the model definition tool, the user organization tool and the engine module of the system are described and discussed.

Key words: workflow management system; J2EE framework; Multi-layer structure; spring; hibernate

0 引言

根据工作流管理联盟(WfMC)的定义, 工作流管理系统是一个支持过程定义, 工作流建立、管理和监视的工具集^[1]。工作流技术是一种过程建模和过程管理的核心技术。以工作流技术为核心的业务管理系统具有高度的灵活性: 可以按照企业的具体需求而快速地生成应用系统; 业务流程可以在客户业务过程发生变化时迅速地重组。工作流技术的应用已经扩展到金融业、电信业、医疗业和办公自动化等诸多领域。

在现有工作流管理系统的开发过程中, 由于 J2EE 技术具有诸多优点和极大的跨平台优势而被大量运用。而在 J2EE 技术体系中, 与重量级 EJB 解决方案相比, 轻量级框架具有以下特点与优势^[2]: 它不会强迫业务对象遵循平台特定接口; 它所采用控制反转(IOC)

模式大大减少了程序量, 同时增加了代码的可测试性。开源框架 Spring 是应用最多的一个轻量级框架实现。使用 EJB 技术的应用无论规模大小如何, 它的复杂性都很高; Spring 不但为开发者提供 EJB 所提供有价值的服务, 而且使应用的复杂性与规模成比例^[3]。Hibernate 是一个优秀的 Java 对象/关系数据库映射框架, 使用它可以极大地减少 JDBC 编程工作量。Hibernate 封装特定厂商的 SQL 代码, 使应用在不同数据库平台之间具有高度的可移植性。

Spring、Hibernate 大大地降低了企业管理信息系统的开发应用难度, 成为当前企业级开发的流行技术。文中采用以上技术设计的分布式工作流管理系统, 能很好地结合当前流行技术, 满足企业业务过程网络化的需要。

1 系统构成

本系统根据 WfMC 提出的参考模型, 由以下三方面功能构成:

收稿日期: 2007-07-05

作者简介: 伍庆林(1982-), 男, 湖南新华人, 硕士研究生, 研究方向为工作流、J2EE 架构; 王加阳, 教授, 主要研究方向为智能计算、信息融合。

a. 建立期功能。流程模型定义工具负责定义和建模工作流程和活动,并将定义信息存入过程模型定义库中;用户组织工具在定义过程时需要引用组织机构人员信息,并将信息存于组织模型库中。

b. 运行期功能。工作流引擎是运行期主体,其对过程模型中定义的过程进行实例化、运行、进行监控。引擎由模型解释执行器、工作项列表管理器与消息服务、过程监控工具三大块构成。

c. 与外部的交互。用户客户端、管理员客户端、与其它系统的接口分别负责系统与普通用户、管理员以及外部系统的交互。系统采用三级结构:构数据库层+业务逻辑层+用户视图层,如图 1 所示。

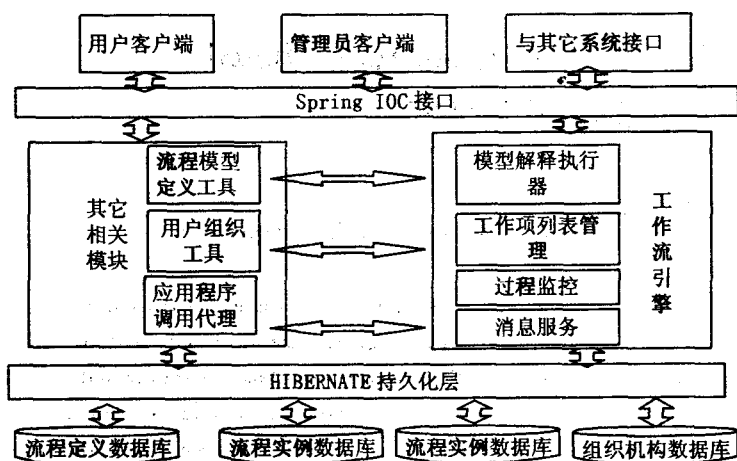


图 1 系统体系结构图

1.1 流程建模语言与模型定义工具

流程模型是业务过程计算机化的形式描述。它定义过程运行中涉及到的各种参数,如业务过程的开始和终止条件、各个工作环节及相互之间的控制与数据流动关系等。本系统选用 UML 活动图描述流程模型。UML 活动图以面向对象为基础,具有丰富的建模元素、强大的表现能力以及简洁直观的建模方法,可以比较好地对开放、交互的工作流进行描述^[4,5]。在 UML 活动图的基础上,结合 Flash 编程技术,开发了一种用户体验良好的 Web 图形化建模工具。主要功能有:图形界面方便用户轻松建模;提供工作流模型各类基本元素,比如简单活动、连接弧、子过程等;提供列表框罗列模型单元的各属性,方便用户修改;以 XML 文件与数据库两种方式对模型进行存取与解析;支持多层次建模及绑定外部的应用。

1.2 用户组织工具

首先,用户组织工具对组织机构、用户信息、角色信息以及用户的角色列表进行维护,包括增加、修改、删除和角色分配;其次,它为工作流管理系统的任务授权机制提供信息支持。这些信息包括:用户拥有的相

关权限,具有的特定角色等。用户组织工具根据需定义各种工作流用户角色,并设置角色拥有的访问权限,而用户根据其责任和权利被指派为不同的角色。这样整个访问控制过程就被分成了两部分,即访问权限与角色相关联,角色再与用户相关联,从而实现了用户与访问权限的逻辑分离^[6]。此举也使得用户组织从流程模型中分离出来,形成组织模型。这样,一个流程模型就可以被组织模型不同地使用,各自发生变化时的相互影响不大,提高了流程模型和组织模型的复用性,增加工作流管理系统柔性。

1.3 工作流引擎

工作流引擎(workflow engine)是工作流管理系统的核心,它为工作流实例提供运行环境,包括流程图的解释、资源的分配、逻辑的控制^[7]等,具体由模型解释执行器、工作项列表管理与消息服务器、过程监控器三大块构成。模型解释执行器负责对模型定义工具的定义的过程模型进行解析,并根据条件实例化流程模型;对流程模型的过程规则进行解析,根据流程实例的相关信息,计算流程的流向。工作项列表管理器与消息服务为用户提供与引擎交互的接口。消息组件采用可插入的形式,能够进行动态配置。过程监控器对流程执行、系统与外部的交互进行监控。

1.3.1 模型解释执行器

模型解释执行器的主要功能包括:

(1) 解释过程模型:流程模型是企业经营过程的一个模板,模型的多个实例可以同时运行^[8]。模型解释器解析过程模型,为模型的实例化以及流程实例的执行提供信息。这些信息包括:流程的构成环节,各环节之间的路由,各路由的执行条件等。

(2) 执行模型实例:根据流程定义与工作流相关数据,实例化一个过程模型或者终止一个模型实例的执行;执行流程实例的环节实例,根据实例数据与路由条件在环节实例之间进行路由。对环节中的任务生成活动实例,启用活动授权机制,将活动实例分配给授权用户(即添加到授权用户的工作项列表)。用户签收与执行后,引擎计算之后的流程环节并予以执行,直到整个流程实例按既定的定义执行完毕。

1.3.2 工作项列表管理器与消息服务

工作项列表管理器对工作项列表与活动实例进行管理,如:新的活动实例到工作项列表的加入,已经完成的实例的删除,未完成的活动实例的挂起。列表管理器对活动实例的相关信息维护,如:应用的

地址、名称、进入操作页面的 URL 链接等。

消息服务是 workflow 管理器将活动实例分配给用户时,对用户采用的消息通知机制。根据用户的需要,可以采用多种方式,如:短消息、邮件、系统消息等。并且,消息发送器根据工作项的结束时间规定其必需的签收时间,如果执行者在签收时间之前未签收属于自己处理的工作项,消息发送器将采用一定的方式进行通知。

1.3.3 过程监控器

过程监控器是 workflow 系统中非常重要的管理工具,对运行时的工作流系统进行管理和监控。主要的管理功能包括:

a. 监控与维护流程实例、环节实例、活动实例的状态;维护与之相关的工作流相关数据。

b. 统计和分析历史数据,以及时发现流程的瓶颈,改善流程以提高企业对业务处理的效率。

c. 归档与清理服务。导出已完成的流程实例、流程环节,以及其它流程运行时数据,进行归档。删除系统内的无效数据(如一些中间状态结果);根据一定的规则,恢复被意外遭到破坏的数据。

1.4 客户端模块

客户端模块是引擎与外部交互的中介,分为管理员客户端、用户客户端、与第三方系统接口三个部分。客户端模块由模块接口与接口的预定义实现两部分构成,可根据具体情况对预定义实现进行扩展,也可以对接口进行重新实现。

管理员客户端是系统的管理工具。通过管理员客户端,管理员能够:监控 workflow 实例及各执行工作流机的状态,并可对其进行人为干预;管理员工或角色的状态;管理流程角色的用户权限;对历史数据进行统计;对系统异常处理进行处理等。

通过用户客户端,普通用户能够:查询自己的任务列表;完成自己任务列表中的工作,并提交给执行引擎;查询自己当前正在处理的任務的状态与 workflow 实例的状态;统计已经完成的任務。

引擎与外部系统交互是预定义接口实现的:外部系统要调用引擎的功能必须通过相应的系统提供的对外接口;如果引擎要使用外部应用,则外部系统必须实现相应的引擎预定义的对内接口。

2 系统的分层设计

采用分层模式进行系统设计,具有以下优点^[9,10]:便于大型软件设计分解成较小且易管理的组件;封装了数据访问细节,降低了模块耦合度;各层之间具有可接插性。系统层次从后台到前台依次为:数据库层、数

据访问层、业务逻辑层、视图层。中间的两层都由层接口与层实现组成,上层的层实现依赖于下层的层接口。借助于 Spring IOC 技术,下层的实现在运行时注入。

2.1 数据库层与 DAO 模式

数据库层由多个库构成(如图 2 所示),数据库及功能说明如下:

(a) 流程定义数据库:存储建模结果,即 workflow 模型;

(b) 流程实例数据库:存储 workflow 运行过程中产生的各种实例,保存实例状态;

(c) 组织机构数据库:存储用户 ID、Password,用户任务表,组织机构与用户的当前状态,流程角色信息等;

(d) 业务逻辑数据库:存储具体业务逻辑数据或者其它 workflow 相关数据信息。

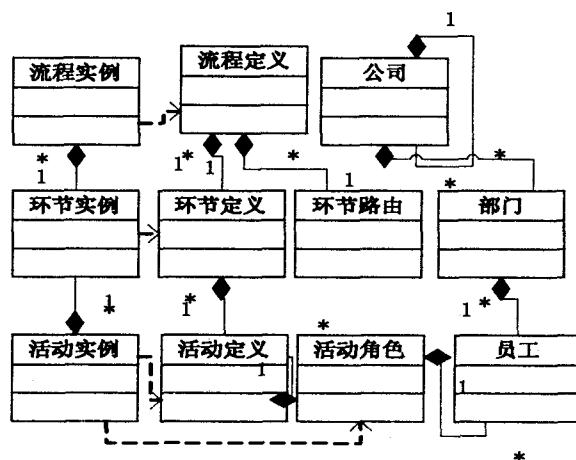


图 2 数据域模型图

数据访问采用 DAO(数据访问对象)模式^[11],隐藏数据源实现细节,为之上的业务逻辑层提供一致接口,适应现有企业级系统中数据源种类多样性的要求。选用成熟 Hibernate 框架进行数据持久化,避免采用 EMP(实体 Bean) 耗损性能,并免去重新实现一个持久性框架的复杂过程,也节省时间。系统的域模型分为过程模型域、实例域、组织机构域(见图 2)。过程模型域包括:流程定义、环节定义、活动定义、路由定义、活动角色。实例域包括:流程实例、环节实例、活动实例。组织机构域包括:公司、部门、员工。

2.2 业务逻辑层

业务逻辑层集中了 workflow 管理系统所提供的各项服务的逻辑实现。它调用数据访问层提供的接口访问底层数据,并以接口的形式向视图层提供服务。借助 Spring 容器管理各项服务。

各项服务以 POJO(Plain Ordinary Java Object)的形式提供,在 Spring 配置文件中注册。使用 Spring 申

明性事务,针对服务类中有的方法涉及多次数据操作配置的同一事务中,以确保事务的一致性与完整性。业务逻辑层包括的业务逻辑具体有以下几个方面:

(a)模型操作服务:实现 UML 流程模型的各种操作并为上层提供访问接口;调用数据访问层的接口将模型持久化;为流程引擎提供模型查找服务。

(b)用户组织服务:通过对数据访问层接口 API 的操作,实现对组织机构、用户信息、角色信息以及用户的角色的维护操作,并为上层提供访问接口;为流程引擎提供用户组织查找服务。

(c)工作流引擎服务:通过对数据访问层接口、模型操作服务接口、用户组织服务接口 API 的调用,实现工作流引擎所提供的功能(即流程模型的解释与执行、工作项列表的管理、消息服务、过程的监控服务),并为上层提供调用接口。它是系统的核心部分,控制着流程实例的运转、对各服务的监控、各服务状态的管理。

2.3 Web 客户端

较其它客户端形式而言,浏览器 Web 客户端不需安装客户端软件,可以实现客户端零维护,从而降低了维护费用,并方便系统的升级。本系统中,流程建模、系统管理人员操作客户端、普通用户操作客户端都采用 Struts 框架实现。客户端具体的处理逻辑在 Struts 提供的 Action 对象的子对象中实现。Action 子对象访问 Spring 容器,调用其管理的业务逻辑模块对数据予以处理,并将处理结果输出到 JSP 表单呈现给客户。

3 结束语

文中在 WfMC 提出的工作流参考模型的基础上,结合 Spring IOC 及 Hibernate 持久化技术,设计并实现了一种多层体系结构的轻量级工作流管理系统。项目

开发的实践证明,此工作流管理系统具有低耦合、可扩展性强、构件的可重用性高等特点。而且,在本系统基础上进行信息系统的开发,能节省开发时间,提高开发效率和效益。

参考文献:

- [1] Workflow Management Coalition. The Workflow Reference Model[M]. TC00-1003 Issue 1.1. Hampshire, UK: [s. n.], 1995.
- [2] 牟军,吕立.使用轻量级框架进行 J2EE 应用开发[J]. 小型微型计算机系统,2006,27(6):1149-1152.
- [3] Walls C, Breidenbach R. Spring in Action[M]. Greenwich, CT: Manning Publications Co, 2005: 6-8.
- [4] OMG. Unified modeling language specification version 1.5 [EB/OL]. Object Management Group, 2003. <http://www.omg.org>.
- [5] 汪涛,吴耿锋,黄力芹,等.工作流管理的过去、现在和未来[J]. 上海大学学报:自然科学版,1999,5(s):1-7.
- [6] 谭支鹏.基于角色的工作流模型及其应用[J]. 小型微型计算机系统,2003,24(6):1064-1066.
- [7] 范延平,曾建鹰.基于 J2EE 架构的工作流引擎的分析与设计[J]. 武汉大学学报:信息科学版,2005,30(12):1107-1110.
- [8] 范玉顺.工作流管理技术基础[M]. 北京:清华大学出版社,2002:58-60.
- [9] Nock C. 数据访问模式:面向对象应用中的数据库交互[M]. 鄢爱兰,王安鹏译. 北京:中国电力出版社,2004:120-124.
- [10] 王卫平,王松涛,王名茗.一种基于 J2EE、Spring 和 Hibernate 的轻量级 EAI 构架[J]. 计算机系统应用,2005(11):38-41.
- [11] 贾昆,甘切初,高慧颖.数据访问对象模式在企业应用集成中的应用[J]. 计算机工程与设计,2006,27(3):373-375.

(上接第 106 页)

加灵活地调整 ACD 的坐席路由策略,提高系统在实际使用过程中的可用性。

参考文献:

- [1] 朱翼隼.呼叫中心排队模型的研究[J]. 镇江高专学报,2006,19(4):1-2.
- [2] 李洪荣.建立市政系统呼叫中心[J]. 中国科技信息,2007(1):1-2.
- [3] 朱仁祥,朱翼隼,方基奎.重试反馈 M/M/s/k 排队的呼叫中心性能分析[J]. 系统工程学报,2006,21(6):2-3.
- [4] 段云峰.呼叫中心系统的构建[R/OL]. 2000-09. <http://www.ctiforum.com/forum/2000/09/f0910.htm>.
- [5] 那塔利·卡尔弗特. Gower handbook of call and contact centre

management[M]. 北京:电子工业出版社,2006.

- [6] 李跃.呼叫中心分布模式[M]. 北京:人民邮电出版社,2007.
- [7] 袁道唯,田淑红.呼叫中心的视野与格局[M]. 北京:清华大学出版社,2005.
- [8] 段云峰.自动呼叫分配 ACD 系统帮助路由[R/OL]. 2003-06-19. <http://www.ctiforum.com/callcenter.html>.
- [9] Shalloway A, Trott J R. Design patterns explained: a new perspective on object-oriented design[M]. Beijing: China Machine Press, 2006.
- [10] Klein M. Windows programmer's guide to DLLs and memory management[M]. 北京:清华大学出版社,1995.
- [11] Lippman S B, Lajoie J. C++ primer[M]. Beijing: Posts & Telecom Press, 2005.