

# 基于 J2EE 架构的工作流引擎的分析与设计

范延平<sup>1,2</sup> 曾建鹰<sup>2</sup>

(1 武汉大学资源与环境科学学院,武汉市珞喻路 129 号,430079)  
(2 国土资源部信息中心,北京市阜内大街 64 号,100812)

**摘 要:**基于 WFMC 的工作流管理系统标准模型以及 OMG 的工作流对象实现模型,进行了基于 J2EE 架构的工作流引擎的模块设计、数据库设计、对象设计及实现。  
**关键词:**工作流引擎;引擎对象;引擎服务  
**中图法分类号:**P208; TP302

## 1 工作流引擎的分析

工作流技术是实现企业业务过程建模、仿真优化分析、过程管理与集成,最终实现业务过程自动化的核心技术。工作流系统主要包括以下 3 个部分:① 工作流定制模块;② 工作流管理模块;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、逻辑的控制等,其主要作用体现在以下两个方面。

1) 工作流引擎是流程的状态转换机。工作流管理系统是一个以过程为中心的系统,所有的运行都是数据驱动。对于一个流程来说,每个逻辑单元都代表着一个流程实际业务的逻辑功能。在流程实例的生存期内,状态是重要的控制数据。它通过改变当前流程实例的状态,控制应用的工作状态。在工作流逻辑中,状态的转换按照一定的规则进行。工作流引擎的任务就是按照定义的规则控制实例的状态转换。

2) 工作流引擎是流程的路由控制器。一个流程对应着一条实际流转的业务,流程的流转路线受当前实例数据的控制。流程定义时就指定了流程的流转规则。在一般情况下,这些规则是根据流程实例数据制定的公式。引擎的作用是对这些公式规则进行解释,找出流程下一步的流向,进

行实例逻辑处理,使流程向下流转。  
通常,工作流引擎中主要包含机构模型、信息模型和控制模型 3 种模型<sup>[1]</sup>。

## 2 工作流引擎的设计

工作流代表了业务的办理,业务办理的实质就是对数据的处理。根据工作流引擎的 3 种模型及其工作流数据的处理原则,本文将数据划分为控制数据、相关数据、业务数据 3 类。

1) 控制数据。控制数据是由工作流管理系统或工作流引擎管理的数据。这种数据对应用程序来说是不可更改的。它代表了工作流系统、过程实例、节点实例、工作项的状态信息。

2) 业务数据。业务数据是应用专用的,工作流系统对其不进行处理,工作流流程定义可以引用,对工作流管理系统来说是不可视的。

3) 相关数据。相关数据是工作流管理系统和应用共同使用的数据。它是管理系统和应用进行能够相互影响流程的惟一途径。工作流管理系统负责了相关数据在流程流转过程中的传递,应用通过它实现对流程的参与和控制。从它应用的功能上可以分为系统保留的相关数据和用户自定义的相关数据两种类型。

对于管理系统来说相关数据有以下几个作用:① 决定流程的流向(参与条件流转);② 设置流程、节点的属性(特殊的数据,用来决定实例的

属性);③ 设置流程的下一步的办理步骤;④ 设置下一步的办理人;⑤ 决定实例的状态转换(参与了状态转换的条件)。

对于应用来说,相关数据是应用在上下节点之间建立联系的途径,如传递文档号。相关数据在定义时隶属于过程,过程创建后相关数据被实例化,应用可以对相关数据进行初始化。过程创建节点,将数据传递给节点,节点保留相关数据的副本进行相应的处理。各个节点将处理后的相关数据提交过程,过程进行处理后传递给下一个办理步骤。相关数据在整个流程的各个办理步骤之间流转,不断地被应用使用和修改,完成其功能。在工作流系统中,动态设置流程实例属性的方法有两种:① 通过管理工具进行修改;② 通过设置系统相关数据来修改流程已定义的属性。

工作流引擎是工作流管理系统中为业务流程实例和节点实例提供运行环境的服务软件,是整个工作流管理系统的核心,它包括如下功能:① 支持工作流定制模块提供的全部特性的运行期实现;② 实现工作流定义解释,控制流程实例的创建、激活、挂起、终止等行为,控制实例生存期驱动工作流定制的业务流程的运行,在运行过程中记录、跟踪、监控、督办、查询和统计工作流处理的活动状态;③ 确定工作项正确分配给相应的用户,维护工作流相关数据,给/从应用程序或用户传递工作流相关数据等;④ 支持流程实例、节点实例、工作项三种对象的状态转换,支持在状态转换中对相应外部事件(如一个节点实例的完成)或指定由工作流引擎采取的控制决策(如导向流程实例的下一步行动)而改变;⑤ 支持分布式工作流引擎和多引擎协同工作机制,通过消息分发器对多引擎进行负载均衡;⑥ 提供完善的安全机制,系统通过对传送的数据包进行加密来实现数据一级的安全。

2.1 引擎的模块设计

2.1.1 层次上划分

系统从层次上分逻辑层和持久化层。逻辑层包括所有的数据对象和一些辅助的功能对象,完成引擎的所有逻辑功能的处理。持久化层负责实体对象的数据持久化工作,包括实体对象的创建、存取、更新和删除工作,屏蔽了逻辑层对底层数据的操作。持久化层的存在使逻辑对象脱离数据相关性的限制,当数据层发生变化时,尽可能小地影响逻辑层的变化。

2.1.2 逻辑功能划分

系统从逻辑功能上进行划分,包括过程定义、

实例、日志、消息和其他的辅助对象(包括解释器、日志、错误处理等),如图 1 所示。

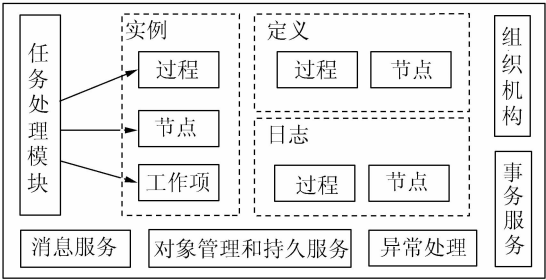


图 1 引擎模块图  
Fig. 1 Engine Module Map

2.2 引擎的数据库设计

工作流系统的实体对象均为持久化对象,对象的数据存储在数据库中。根据数据的存在时间将数据库中的表分为以下几类。

- 1) 定义表:存储流程全部的定义信息,它是流程运行的依据。
- 2) 实例表:存储流程运行的临时数据,具有临时性,运行后被删除。
- 3) 日志表:记录流程运行的历史数据,是统计的依据,支持一些特殊功能,如回退。
- 4) 组织机构表:它不依赖时间一直存在,被工作流实例所引用。

实体对象及其数据的生存期是一致的。在流程的整个生存期内,不是所有的对象和数据都存在。在各个时期,数据库保存的数据对象如下。

- 1) 定义表:过程、相关数据、节点、传输线、应用程序、参与者、子流程。
- 2) 实例表:副本、流程实例、流程相关数据、节点实例、节点相关数据、工作项、工作项相关数据、消息。
- 3) 日志表:流程实例、流程相关数据、节点实例、节点相关数据、工作项、消息。

对于支持动态修改流程图的流程,创建流程实例的同时也会创建一个定义副本,包含所有与该流程有关的流程定义信息。工作流引擎随之进行的操作也相应地依照流程定义副本进行,这样,当需要动态修改流程时,只会将影响限制在该流程实例的定义域内,而不会波及基于该流程定义的其他流程实例。

2.3 引擎的对象设计及引擎服务

2.3.1 工作流引擎对象设计

工作流引擎中主要的对象有过程、节点、工作项及其他的辅助对象,它们的所有活动决定了引擎的逻辑和实现。下面是引擎的主要对象类别

表,类之间关系图如图 2 所示。

- 1) 过程类
- 过程实例类:WEProcessInstance;
  - 子过程实例类:WESubProcessInstance;
  - 过程定义类:WDProcess。
- 2) 节点类
- (通用)节点实例类:WEActivityInstance;
  - 手动节点实例类:WEManualActivityInstance;
  - 自动节点实例类:WEAutoActivityInstance;
  - 子过程节点实例类:WESubProcessActivityInstance;
  - 路由节点实例类:WERouteActivityInstance;
  - 节点定义类:WDActivity。
- 3) 工作项类
- 工作项类:WEWorkItemInstance;
  - 子工作项类:WESubWorkItemInstance。
- 4) 对象管理类
- 对象管理器:WEWorkflowObjectManager。
- 5) 持久化类
- 过程持久化类:WEProcessInsPersist;
  - 节点持久化类:WEActivityInsPersist;
  - 工作项持久化类:WEWorkItemInsPersist。

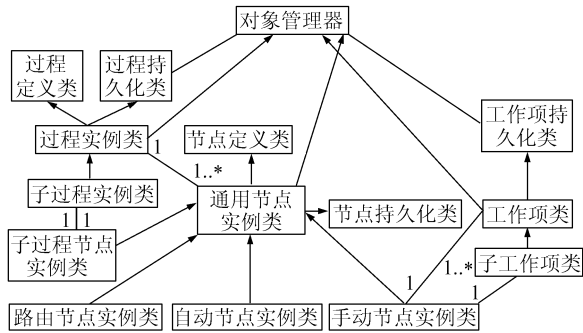


图 2 引擎中类关系图

Fig. 2 Class Relationship Map in the Engine

类的设计采用了面向对象的继承方法,在一定程度上屏蔽了节点类型之间的差别。如一个节点挂起的操作,在调用过程中不必区分是什么类型的节点对象,只要调用通用的接口就可以了。对象管理器在负责装配对象时需要对不同的类型创建不同的实例。

在工作流运行的对象生存周期中,当流程被创建后,流程便从静态的定义状态变为动态可流转状态。流程定义存在两种逻辑单元:过程和节

点的定义,以及过程的数据变量。当流程成为可流转的过程时,生成过程实例,而节点是否能成为实例状态,则要由过程流转的路径来决定。而可流转的实例对象也有活跃的内存状态和钝化的数据库状态两种状态。以节点为例,当前节点完成后,过程根据定义路由到下一个节点,这时需要将该节点从定义态转成为可流转的状态,过程向 Object Manager 发出创建节点的请求, Object Manager 根据请求类型生成相应空的对象实例,对象实例根据定义标识进行原始数据初始化,并获得过程传递来的相关数据,接着将所有数据通过持久化对象写入数据库,并启动逻辑功能服务,创建工作项或进行路由。当前任务完成后,如果节点完成则将自己从运行表中删除,并退出内存状态,否则直接退出内存状态。仅退出内存的节点,当过程再次需要时,向 Object Manager 发出 load 对象的请求。Object Manager 通过持久化层将数据从数据库中取出,创建对象并装配数据,使其成为内存对象。对于完成的节点,在被删除前会将完成的信息记录日志。过程和过程相关数据、节点和节点相关数据、工作项都会被记入日志。

2.3.2 工作流服务

结合以上类设计构建工作流引擎服务,具体包括以下几个方面。

1) 工作流执行服务(workflow execution service)。工作流执行服务是系统的核心部分,主要负责流程实例的运转和对象的状态转换,支持群集运算。

2) 管理/监控服务(admin service)。负责对各服务的监控,管理各服务的状态,可自动对异常状况进行报告。

3) 定时服务(scheduling service)。定时服务根据系统的配置,周期性地启动或调度相应的系统进程来完成某些特殊的任务。如间隔一定时间,检测流程中是否有已经超时的节点、更新应用服务的注册信息、与业务系统交换数据等等。

4) 归档服务(archive service)。归档服务是将已完成的流程实例及其日志记录从当前运行库导出的过程。用户也可以选择导出某时刻以前启动的所有流程实例。归档服务通过保证引擎数据库工作在一个健康的记录数内,来保障数据安全和引擎的运行效率。

5) 网关服务(gateway service)。网关服务是对消息传递系统的补充。众所周知,不是所有发生在工作流管理系统和其他外部系统之间的通信

都可以借助于消息队列来完成,对于某些如电子邮件这样的系统间数据,需要有相应的服务来将其“翻译”成 workflow 管理系统内部可以传递和处理的格式,网关服务就是用来完成这一功能的。

6) 清理服务 (cleanup server)。功能与归档服务类似,不同点主要体现在以下两个方面:一方面是处理的结果,归档服务将系统数据导出成持久化文件或转移到别的数据库中,以便进行数据备份和统计,而清理服务只是将一些系统内的无效数据(如一些中间状态结果)直接删除;另一方面,清理服务还要根据一定的规则,对一些被意外遭到破坏的数据进行恢复,以保证 WFMS 的正常运行。

7) 日志服务 (log server)。日志是 workflow 运行时相关信息的记录,记录信息包括对象日志和事件日志两部分。

2.4 引擎的异常处理机制

在 Java 的编程思想中,系统的错误是通过捕获异常来实现的。workflow 服务器系统使用 Java 编写,采用了 Java 的异常处理思想,抛出的异常统一通过异常类 WEEException 来描述。

WEEException 类继承了 Java 的异常类 Exception,并覆盖了 Exception 类的 getMessage() 方法。其中 WEEException 类中保存了可识别的异常对照表。在 Java 的异常处理中,在出错的地方将异常抛出,不进行处理。异常被抛到更高的层次,直到某个层次能够进行这种异常的处理。workflow 的错误包含两类错误:① 流程错误,如节

点的办理人不存在,节点的应用程序定义错误;② 系统的错误,如网络不通,数据库系统异常。对于引擎来说,前类异常是属于无法处理的错误,只能进行错误的通知;后者是引擎可以处理的错误,如数据库异常或连接临时中断等等,引擎可以进行容错处理,例如,在数据库恢复后自动重新建立连接。引擎对可能出现的错误进行了编码,并附有对应的描述信息。

参 考 文 献

1 吴朝晖,潘云鹤. workflow 管理技术. 计算机世界,1999 (18)

2 Wu S G, Shi M L. Support Environment for CSCW Research-design and Implementation of a Desktop Computer Conferencing System. International Conference on Communication Technology, Beijing, 1996

3 Steinar C. Action Port model: A Mixed Paradigm Conceptual Workflow Modeling Language. The 3rd IFCIS International Conference on Cooperative Information Systems, New York, 1998

4 Mohan C. Recent Trends in Workflow Management Products, Standards and Research. NATO Advanced Study Institute (ASI) on Workflow Management Systems and Interoperability, Istanbul, 1997

第一作者简介:范延平,博士生,助理研究员。主要研究方向为分布式组件和地理信息系统的相关理论及有关应用。  
E-mail: ypfan@infomail. mlr. gov. cn

Analysis and Design of the Workflow Engine Based on J2EE

FAN Yanping<sup>1,2</sup> ZENG Jianying<sup>2</sup>

(1 School of Resource and Environment Science, Wuhan University, 129 Luoyu Road, Wuhan 430079, China)  
(2 Information Center of Ministry of Land and Resources P. R. C. ,64 Funei Street, Beijing 100812,China)

**Abstract:** On the basis of the workflow management system’s standard model of WFMC and the workflow object implementary model of OMG, this paper develops the design and implementation of the workflow engine module, the workflow engine database and the workflow engine object based on J2EE.

**Key words:** workflow engine; workflow engine object; workflow engine server

About the first author: FAN Yanping, Ph. D candidate, assistant researcher, majors in the theories and applications of the distributing Component and GIS.  
E-mail: ypfan@infomail. mlr. gov. cn