

文章编号:1001-9081(2005)02-0469-03

一种基于 J2EE 的工作流引擎体系结构

徐建军,谭庆平,杨艳萍

(国防科技大学 计算机学院,湖南 长沙 410073)

(nudt_xjj@hotmail.com)

摘 要:工作流引擎是工作流管理系统的核心部件,它的结构是否合理对整个系统有着至关重要的影响。文中介绍一种基于 J2EE 实现的工作流引擎的体系结构。该引擎由六个部件构成:解析器、流程管理模块、执行器、任务分派模块、事件服务器、时间服务器和客户端接口。与同类研究相比,该引擎结构在可实现性、可扩展性、支持的交互模式和事务管理管理方面具有一定的先进性。

关键词:工作流引擎;体系结构;J2EE

中图分类号: TP311.131.1 **文献标识码:** A

Architecture of J2EE-based workflow engine

XU Jian-jun, TAN Qing-ping, YANG Yan-ping

(School of Computer, National University of Defense Technology, Changsha Hunan 410073, China)

Abstract: Workflow engine is the core component of workflow system, architecture of which is vital to the whole system. This paper addressed the architecture of J2EE-based workflow engine, which was consisted of six components: parser, process manager, executor, task assigner, event server, time server and client interface. By the comparisons with other similar research works, the advantages of this approach in terms of applicability, expansibility, interview mode and transaction management were illustrated.

Key words: workflow engine; architecture; J2EE

0 引言

在工作流管理联盟(WFMC)给出的工作流参考模型^[1]中,工作流引擎处于核心地位,是一个为工作流实例提供运行时期的执行环境的软件服务器。它是企业经营过程的任务调度器和企业资源的分配器,工作流引擎的效率和可靠性直接决定了整个系统的运行效率和性能。

目前市场上有很多工作流的产品,但是很少有关于这些商业工作流引擎体系结构的介绍。文献[2]将引擎划分为机构模型、信息模型和控制模型三种模型,从体系结构的角度对引擎进行讨论。文献[3]在传统关系数据库的基础上讨论一个引擎的具体设计原理与实现方法,描述了一些引擎用到的表结构。但是有关工作流调度机制、并发控制方法、流程的执行过程等方面很少被提及。

J2EE 提供了一个企业级的计算模型和运行环境,支持开发和部署多层体系结构的应用。它通过提供企业计算环境所必需的各种服务,使得部署在 J2EE 平台上的多层应用可以实现高可用性、安全性、可扩展性和可靠性。J2EE 技术的日益成熟使得在其基础上构建工作流系统成为很好的选择。

本文以一个金融业务平台为背景实例,介绍其中的核心部件——工作流引擎——的体系结构。该引擎以 J2EE 和关系数据库为基础,具有实现简单、可扩展性好、支持灵活地与客户端进行交互、支持事务等特点。

1 背景介绍

流程模板是工作流引擎的执行对象,尽管各个系统的建模语言差别较大,但是概括起来每个流程模板都由多个节点(或称为活动)和连接它们之间的路由边构成。节点具体定义业务逻辑的实现过程,路由边描述了节点的执行顺序。以金融领域中的一个信用证流程为例。其流程图如图 1。在这个流程中,信用证被开立后,流程处于等待外部操作“事件”状态,有效期内用户可以进行“改证”和“结清”操作。“改证”是可以重复多次的,而执行“结清”后,整个流程结束。信用证到期后,自动进行“结清”操作。这些操作分别通过路由边起来。

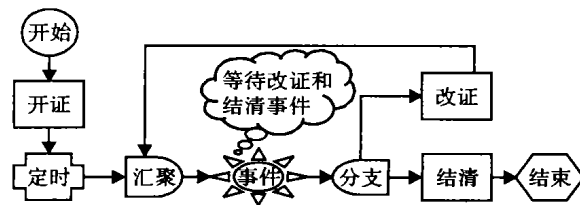


图 1 信用证流程

每个模板实例在执行过程中被分割成多个单独的执行片段(Fragment),执行片段是一个能够被计算机自动执行的工作流节点序列。就信用证流程而言,有四个执行片段:1)从“开始”一直到等待“事件”节点;2)从“事件”到“分支”;3)从“改证”到继续等待“事件”;4)从“结清”到“结束”。执行片

收稿日期:2004-07-19;修订日期:2004-10-12 基金项目:国家 863 计划项目(2003AA001023)

作者简介:徐建军(1980-),男,安徽休宁人,硕士研究生,主要研究方向:工作流技术;谭庆平(1965-),男,湖南衡阳人,教授,博士生导师,主要研究方向:软件工程;杨艳萍(1980-),女,湖南永州人,博士研究生,主要研究方向:软件体系结构。

段是引擎的基本运行单位。

工作流系统中会部署多个流程模板,每个模板可以被执行多次,也可以有多个有关这个模板的实例同时存在。如何维持执行这些模板实例是工作流引擎主要需解决的问题,也是引擎体系结构的主要区别之处。目前实现方案大致可以分为两种:队列调度^[4]和多线程。

队列调度是指引擎中维护一个(或多个)队列,所有来自客户端的请求先置于队列中,引擎有一个调度器对队列中各个请求调度执行。显然这种方案实现简单,系统可扩展性好,而且各个模块之间是松耦合。其缺陷在于需同步执行的操作被引擎异步处理,不能满足用户要求。多线程是指引擎中有一个线程池,针对每个请求创建一个线程,由该线程执行具体的请求任务。这种方案克服队列调度机制的缺点,但其不足之处在于系统实现复杂,系统状态难以维护,各种并发控制的实现对程序员要求较高。这两种方法各有其优缺点。

大量的业务逻辑、商业过程都要求事务的支持,特别是在贸易、金融等领域,许多相关的系列操作具有不可分性。工作流引擎应提供完整的事务支持以确保数据的完整性、一致性,过程的完整性、一致性。多种交互机制的支持、时间服务、引擎的可扩展性等是具体应用过程中提出的需求。

2 体系结构

在对比前面两种工作流引擎实现方案并仔细分析工作流引擎的需求的基础上,我们在 J2EE 平台和关系数据库之上设计并实现了一种工作流引擎。该引擎主要包括 6 个部件:解析器(Parser)、流程管理(Process Manager)、执行器(Executor)、任务分派(Task Assigner)、事件服务器(Event Server)、时间服务器(Time Server)和客户端接口(Client Interface)。整体体系结构可以用图 2 表示。

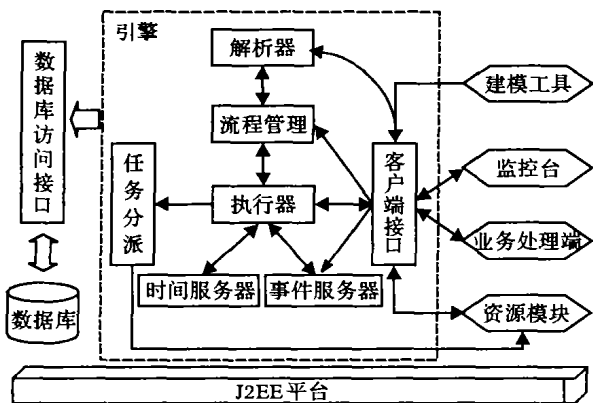


图2 工作流引擎体系结构

建模工具提供引擎执行的来源——XML 文件格式的工作流模板,在引擎运行时系统管理员可以通过监控台与引擎进行交互,整个运行过程中所需的人员等资源信息由资源模块提供,工作流用户具体处理引擎分派的任务是在业务处理端进行。引擎访问数据库是通过单独的数据库访问接口进行。

1) 解析器

解析器的作用是将描述工作流模板的 XML 文件解析成能够被引擎执行调用的模板对象并进行持久化。模板是以 XML 文件的格式保存于工作流模板库中,模板对象是一个复杂的 Java 对象,它是流程执行的基础。解析的结果最终通过

数据库被持久化存储。

2) 流程管理

模板实例化操作产生的模板实例称为流程。流程是工作流系统中一个主要概念,是引擎主要管理的对象。有的引擎针对每个流程创建一个单独的线程去维护其执行情况,显然这种线程的生命周期维护以及线程之间的通信较为复杂。这里通过一个 EJB 来负责流程管理。流程管理模块要负责流程的生命周期管理,包括对流程进行启动、挂起、恢复和停止操作。而且所有与流程相关的实例信息都由这个模块提供。

根据数据局部化原则,流程管理模块还维护一个 Cache,其中存放最近使用的工作流模板。模板相对于流程来说是静态信息,所以这个 Cache 中不会有数据不一致的问题。流程管理模块在构造实例信息就无需从数据库中读取,直接从内存中即可获得原始信息。

3) 执行器

执行器是引擎的核心部分,负责流程中节点实例的具体执行。它通过解析节点中定义逻辑,与外部资源交互完成具体业务,解析路由规则为流程和节点的执行进行导航。并且节点可指定不同的事务属性,执行器中有一个 EJB 定义了对应的 J2EE 事务属性的节点执行方法,实现对事务的支持。

执行器中有一个待执行的执行片段构成的队列,当前执行片段在并发分支等情况下会派生新的执行片段,新的片段置于队列中被队列监控器调度执行,如图 3 所示。这个队列实际上解决了多线程并发的

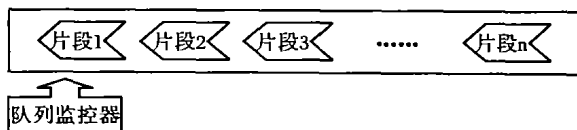


图3 执行片段队列

4) 任务分派

任务分派模块是为需要人工执行的任务节点分派合适的执行者,这是工作流的一个主要功能体现。如果执行器是解析具体的企业业务规则,那么任务分派则是解析企业的人力资源分派规则。任务分派所用的人员组织信息是由引擎外部的资源模块提供,引擎中不包含这些信息是因为实际企业的组织信息复杂多变,引擎无法抽象出统一的模型。企业根据需求定制其组织结构,引擎只需根据定义的接口获取其所需信息。

5) 事件服务器

异步的事件是引擎与外界交互的一个重要途径。通过事件服务器,引擎可以发事件给外部系统,以及接收外部发给引擎的事件,这是通过 JMS 完成的。事件服务器收到事件,首先在数据库中根据事件属性查找是否有引擎其他模块已经注册了等待该事件。若存在相关的注册信息,则根据注册信息中事件目标通知引擎其他模块。如果不存在则通知事件发出者事件没有被引擎成功处理。

6) 时间服务器

时间服务器为整个系统提供定时服务和日历管理功能。引擎维持一个定时列表,其中是引擎其他模块注册的定时信息。服务器监测该列表,时间点到达则根据定时的目标通知引擎的其他模块。日历管理是实际企业应用提出,实际企业时间会有上班时间、下班时间和休假时间等区分。时间服务

器为用户提供定制企业工作日历的接口。在正常工作时间内,引擎正常运行。工作时间之外,时间服务器会使得工作流引擎的各个模块处于挂起状态。

7) 客户端接口

客户端接口是引擎与外部系统交互的中介,在WFMC定义的工作流参考模型中,工作流引擎通过5个接口为整个工作流管理系统服务。在这里,引擎与外部系统交互是通过客户端接口模块中的一些EJB和预定义接口实现的。外部系统要调用引擎的功能必须通过相应EJB提供的API,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接口。

3 运行过程

用户可以通过客户端接口启动流程。流程启动时如果对应的模板不在Cache中,流程管理模块则将模板从数据库中读取进来,创建新的流程实例,初始化各种相关数据,取得模板的开始节点实例化后请求执行器执行,执行到结束节点后流程管理模块停止整个流程。

对于第2部分的信用证流程,第1个片段执行过程时,流程执行“开证”节点,向时间服务器注册信用证的有效时间,然后流程状态被保存于数据库,流程挂起并等待外部通过事件服务器发送信用证操作事件。操作事件到达后流程恢复执行第2个片段,根据路由规则选择执行后续执行片段(“改证”或是“结清”),把后续片段加入到执行器的执行片段队列中。信用证有效期到达,时间服务器发送“结清”事件给事件服务器。第4个片段结束后,流程管理模块终止这个信用证流程。

引擎中最基本的执行对象是节点实例。以执行人工任务的节点为例,图4通过UML顺序图的形式描述了工作流引擎中节点一个典型的执行过程。

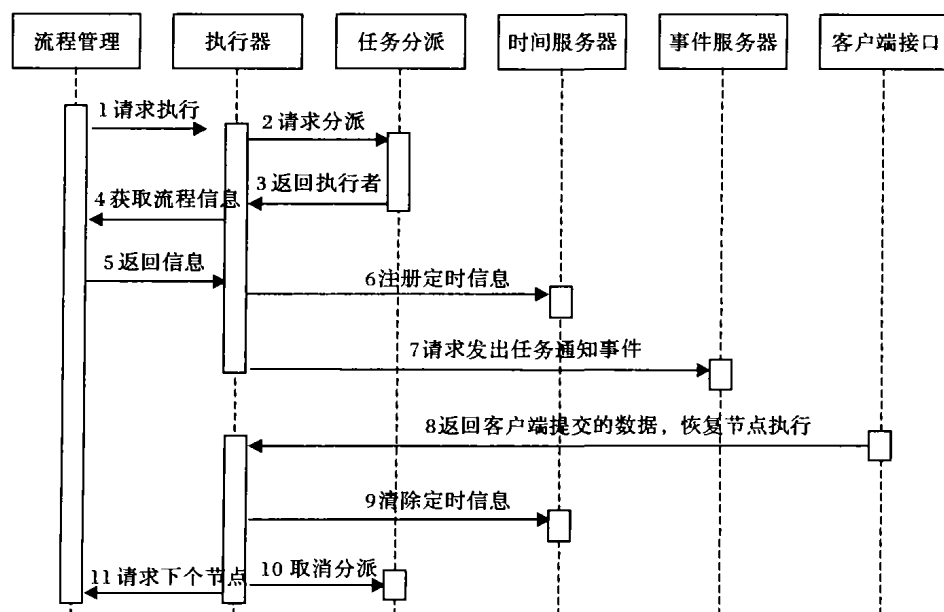


图4 引擎中一个的典型执行过程

首先,流程管理模块把要执行的节点实例传给执行器,执行器根据节点事务属性调用对应的方法执行该节点。在第2步中执行器请求任务分派模块为该任务节点分派合适的执行

者,任务分派模块综合各种要素后在步骤3返回合适的执行者。在节点执行过程中执行器会向流程管理模块请求工作流相关数据(4、5)。第6步表示任务节点执行时需要向时间服务器管理模块注册定时信息,以便在有效时间内仍未能做完任务情况下可以通知引擎。若任务执行需要与用户进行交互,则第7步中执行器以发送JMS消息的方式通知业务处理端,然后节点的当前执行状态被保存到数据库中。用户做完任务后,会调用客户端接口中的EJB提供的API提交任务给引擎,第8步客户端接口将用户提交的数据返回执行器,任务节点从数据库中恢复上次执行状态继续向下执行,第9、10步分别表示执行器请求时间服务器清除定时信息和取消任务分派。节点执行结束后,第11步执行器根据节点后继路由向流程管理模块请求下一个要执行的节点。

4 结语

通过上述对这种工作流引擎体系结构的介绍,可以了解该引擎以执行片段为基本的运行单位,以节点实例为执行对象。用户可以通过调用客户端接口的API与引擎同步交互,也可以通过发送JMS事件异步地与引擎交互。实际上该引擎结合了队列调度和多线程两种方案,客户端接口中EJB可以同步处理并发的用户请求,但这种多线程是J2EE平台实现的。执行器中的执行片段队列则将流程并发控制等问题转化为队列调度解决,降低了实现的复杂度。引擎在J2EE事务模型的基础上实现了对事务的支持,允许通过定义节点的不同事务属性以定制事务范围。由于引擎内部采用模块化的结构设计,实现松耦合,可以扩展为分布式的工作流引擎。Cache机制的使用进一步增强了系统的性能。

目前基于这种体系结构的工作流引擎已被应用于一个金融业务平台的工作流系统中,通过实际应用验证发现,该引擎能够很好地满足金融业务的实际需求。

参考文献:

- [1] HOLLINGSWORTH D. Workflow Management Coalition. The Workflow Reference Model [S]. Document Number WfMC-TC00-1003, Brussels, 1995, 22-23.
- [2] Karl RPH Leung, et al. The Liaison Workflow Engine Architecture [EB/OL]. In: Proc of the 32nd Hawaii Int'l Conf on System Sciences, Hawaii, Jan. 1999, <http://www.computer.org/proceedings/Hiccs2/>, 2004.
- [3] 何清发, 李国杰, 焦丽梅, 等. 基于关系结构的轻量级工作流引擎[J]. 计算机研究与发展, 2001: 2(38).
- [4] 卢本捷, 魏守平, 刘俊. 基于消息队列中间件的总线式工作流管理系统设计[J]. 计算机应用研究, 2004, 1: 159-161.