业务流程柔性配置的研究和实现①

李耀芳[1], 彭慧卿[1], 李保清[2], 乜聚科[3]

[1](天津城建大学 计算中心, 天津 300384)
[2](天津普迅电力信息技术有限公司, 天津 300384)
[3](南开大学 信息技术科学学院, 天津 300071)

摘 要: 实现了一个柔性可配置轻量级工作流引擎系统. 系统使用工具 JaWE 进行可视化建模, 生成 XPDL 格式的流程定义文件; 并设计实现了基于 Token 的引擎执行控制机制, 使得引擎系统对其所在业务系统中的业务流程执行过程的控制准确而有效率. 采用适当的流程定义管理策略, 实现对流程运行时修改流程定义的支持; 同时引擎使用基于组件配置的方式形成业务系统, 使应用引擎的业务系统具有较大的柔性. 该引擎已成功应用在某制药企业的项目管理平台中, 取得了良好的应用效果.

关键词: 工作流引擎; 柔性; 可配置; XPDL; 流程

Research and Implementation of the Flexible Configuration in Business Process

LI Yao-Fang[1], PENG Hui-Qing[1], LI Bao-Qing[2], NIE Ju-Ke[3]

[1](Computer Center, Tianjin University of Urban Construction, Tianjin 300384, China)
[2](Tianjin Puxun Electric Power Information Technology Co. Ltd, Tianjin 300384, China)
[3](College of Information technology and Science, Nankai University, Tianjin 300071, China)

Abstract: This paper implements a lightweight workflow engine system which is flexible and can be configured. The system uses tools of JaWE for visual modeling, then it generats XPDL format process definition file. It designs an engine executive mechanism based on Token, which makes the control of engine system with the business process implementation more accurate and efficient. It adopts appropriate process definition management strategy that gives the modification definition support during process operation. At the same time, the engine forms business system by using the way based on components configuration, which makes application engine business system have great flexibility. The engine system has been successfully applied in a pharmaceutical enterprise project management platform, and has obtained a good application effect.

Key words: workflow engine; flexible; can be configured; XPDL; process

1 引言

随着信息化的快速发展, 任何简单的信息系统都要处理流程问题. 工作流[1]技术是一种旨在实现业务流程自动化管理的技术, 但是其研究还处于发展阶段, 有很多关键技术需要研究. 特别是随着市场竞争的全球化, 为了赢得市场竞争, 企业必须不断地调整自身的业务过程, 优化资源组合, 提升自己的核心竞争力. 这样就使得越来越多的工作流不再是静态的流程, 它们常常需要在运行的过程中进行实时修改以快速响应需求的变化. 传统的工作流管理系统缺乏柔性[2,3], 不能满足不断调整的业务过程管理需要. 因此近年来, 工作流的柔性技术成为人们研究的热点.

本文分析了一般的不同关键业务的基本特征, 基于关系数据库[4], 开发了业务流程可以柔性配置的轻量级工作流引擎. 利用此工作流引擎和流程活动处理单元, 经过简单的配置就可方便地构造出具有工作流特征的业务系统. 方法是: 用户先根据具体业务流程建立基于活动图的流程定义模型[5], 并为该流程定义模型中各个业务活动配置执行单元(完成组件设计), 然后把这些业务活动利用引擎形成协同控制的约束联

① 收稿时间:2014-10-29;收到修改稿时间:2014-11-28

系起来，以完成整个业务流程的执行过程. 因此，该引擎具有相当的柔性，能够通过对流程定义版本[6]. 的管理、配置执行单元的更改等方式灵活的应对业务流程的变更. 本文实现的引擎在某制药企业的项目管理系统的审批业务流程中得到应用.

2 工作流引擎系统的结构设计与开发

2.1 工作流引擎系统结构

主要实现一个可柔性配置的工作流引擎，工作主要有两块，一块是以图形化方式形成工作流，二是工作流在其他实际业务系统的应集成用. 利用引擎构建的业务系统的整体结构如图 1 所示.

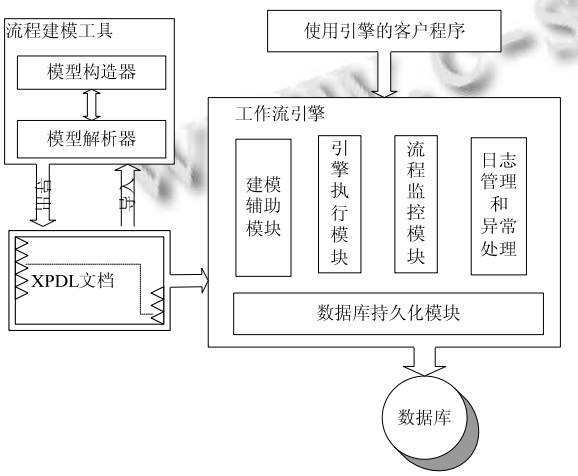

图 1 利用引擎构建的业务系统的整体结构图

2.2 工作流引擎执行模块

引擎系统的执行过程分成两个阶段，业务流程建模阶段和业务流程执行阶段. 通过对实际业务处理流程的分析，提取流程、参与者、工作流相关数据、活动和转移五个基本元素，使用 JaWE 建模工具进行可视化建模，最终生成 XPDL 格式流程定义文件. 然后使用 WorkflowDefParse 类对 XPDL 格式流程定义文件进行解析，从中解析出的流程、参与者、工作流相关数据、活动和转移信息分别存入引擎系统的数据库表，用于业务系统调用引擎实现对流程的执行和控制.

解析流程后，有一些字段还没有值. 引擎系统使用者通过流程细化配置页面为系统填入这几个缺少的值. 这样，引擎系统通过这里注册的组件与实际业务处理逻辑等紧密结合在一起，构成柔性可配置的业务处理系统.

引擎系统对活动逻辑执行、工作流相关数据获取、参与者指派等功能采用基于组件的实现方式. 引擎系统用到用户注册的组件时，通过查询数据得到流程定义表中以字符串的形式存储的组件名称，按照通常的编程方式不能得到组件对应的对象，进而调用组件的方法. 本文引擎系统是通过 IOC 容器[6]解决这一问题的.

系统使用.net 平台下开源的的 Castle 框架的 IOC 容器来执行组件.

下面以活动执行逻辑组件为例，给出利用 Castle 框架执行注册组件的配置和执行两个步骤.

1) 配置

引擎系统中添加了 IocConfig.xml 配置文件，用户需要将组件相关的信息添加到该配置文件.

2) 执行注册组件

引擎系统通过静态类 ComponetHelper 的全局静态方法 Execute 方法执行注册的活动逻辑组件，执行注册组件又分为四个步骤，分别是建立容器、加入组件、获取组件和使用组件. 执行活动逻辑组件的实例代码如下:

```
IWindsorContainer container = new WindsorContainer
(new XmlInterpreter(“configure.xml”));
// 建立容器
container.AddComponent(“activityExecuteComponent”,
typeof(IExecute),typeof(ActExecuteClass));  // 加入组件
IExecute execute=(IExecute)container[“activityExecute
Component”]; // 获取组件
execute.Execute(context); // 执行组件
```

系统采用的配置模型使引擎具有更大的柔性. 比如: 当流程的控制流转没有变动，而个别活动上的处理逻辑发生变化，而这些变化也不会影响后继活动的执行，这时可以不必重新定义流程，而仅仅更换注册的活动处理逻辑组件即可.

引擎执行模块是引擎系统的核心模块. 它负责启动流程实例，活动的执行和流程在活动间导航等. 活动的执行是需要用户来驱动的，用户只能通过 WorkflowEngine 类提供的接口与引擎交互，驱动活动执行. WorkflowEngine 类提供了所有用户和外部应用与引擎交互的调用接口.

在引擎系统中，用户通过两种方式驱动活动执行:

一种是用户启动流程，引擎开始执行，这是通过 WorkflowEngine 类的 StartFlow 方法实现的. StartFlow 方法创建流程实例，然后初始化第一个活动，之后开始执行活动. 该方法接口如下:

public static void StartFlow(int processId, string userId)

另一种是流程中某一项工作需要用户参与，引擎会生成工作项等待用户执行，用户获取该工作项并参与完成本身工作后，驱动引擎向下继续执行，这是通过 WorkflowEngine 的 RunFlow 方法实现的. RunFlow 方法驱动流程的运行，它与 StartFlow 方法执行有区别，它没有初始化工作需要做，直接开始执行活动. 该方法接口如下:

public static void RunFlow(EngineContextengine Context, AppContext appContext)

活动的执行是通过 Acitiviy 类的 Run 方法实现的.

重载一被 WorkflowEngine 类的 StartFlow 方法调用，它的执行过程如图 2 所示.

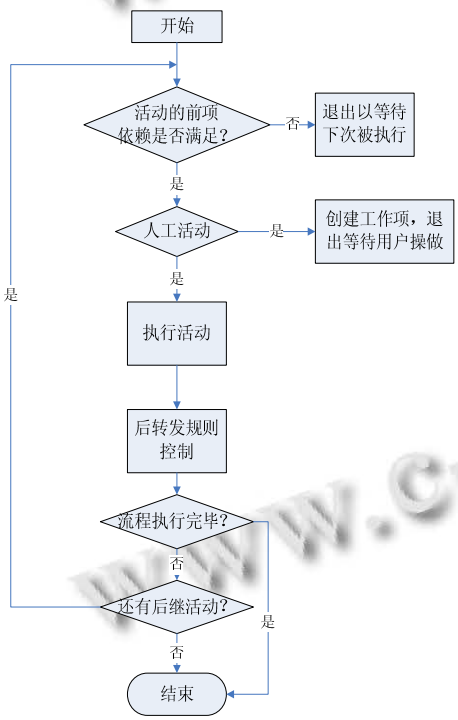

图 2　系统触发的活动执行过程

重载二被 WorkflowEngine 类的 RunFlow 方法调用. 它的执行过程与重载一得执行过程略有不同，执行过程如图 3 所示.

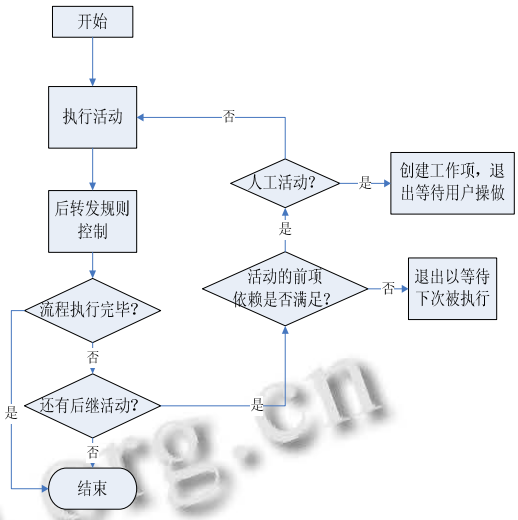

图 3　人工触发的活动执行过程

从两个图中可以看出，两种活动执行方式本质上是没有区别的，只是活动执行的起点不同，重载一方法的执行从检查前项依赖是否满足开始，重载二方法是人工参与流程中活动的执行，活动正处于等待执行状态，所以不必检查前项依赖条件即可执行.

在引擎执行控制过程中，Token 起着重要的作用，主要有两点:

一是，它可以表示流程实例当前的运行进展状况，Token 所在的节点就是当前流程的等待执行的位置. 当该节点处活动执行完毕，Token 被取消. 同时，下一个活动节点被设置 Token. 当流程实例中没有 Token 时，流程实例执行完毕.

二是, Token 做为检查活动执行条件是否满足的依据. 比如下图并发汇聚模式中的节点 D，当活动 A 执行完毕，并且其后的转移 A 上条件也满足，则活动 C 上会设置一个 Token. 活动 B 和活动 C 顺利执行后，活动 D 上会再设置两个 Token，这样活动 D 会有三个 Token，然后活动 D 才能执行. 推广而言，可以根据活动的前置条件(XOR 或 AND)，检查活动上 Token 的数目和活动的转入转移的数目之间关系来判断活动执行条件是否满足.

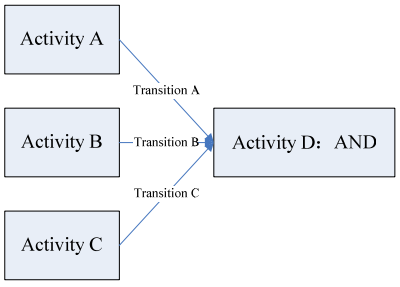

图 4　Token 做为检查活动可执行的条件示意图

流程执行过程中，引擎系统控制流程活动上 Token 的设置与取消，从而控制流程流转. 引擎的执行控制过程离不开 Token.

Token 的设置由 Token 类的 SetToken 方法负责，Token 的取消由 Token 类的 Cancel 方法负责，两个方法的接口如下:

public static void SetToken(int processId, int processCaseId, int actId)

public static void Cancel(EngineContext context)

在此总结一下在系统中 Token 的生命周期:

Token 的生成主要在两处: 一处是流程启动时，在初始化阶段为待执行的活动设置 Token; 另一处是一个活动的业务逻辑执行完毕后，在后转发规则控制中为获取的后续待执行活动设置 Token.

Token 的取消在活动前项依赖规则检查的结果满足，活动业务逻辑执行之前完成. 即先取消活动上的所有 Token，再执行活动业务逻辑.

在本文工作流引擎系统中，活动执行前后都需要规则管理，分别是活动的前依赖规则和活动的后转发规则. 活动的前依赖规则指明相应活动的启动条件，启动条件满足活动才能执行. 活动的后转发规则是指在当前活动执行结束时找到当前活动的后继待执行活动节点的规则，具体操作是为当前活动的后继待执行活动节点设置 Token.

2.3 流程监控模块

流程监控模块主要是给引擎维护人员提供管理的接口. 流程实例执行的整个过程中，都伴随着流程实例状态的改变. 流程实例包括以下几种状态: 初始态、运行态、完成态、终止态、挂起态.

初始态是指，一个过程实例刚创建，还没有执行，在这个时候，工作流实例一般都放置在内存或数据库中，等待参与者的执行指令; 运行态是指流程实例正在执行; 挂起状态是指流程实例继续执行的一些条件并未满足而由用户暂时的停止又或是在执行流程实例的过程中出现异常而由工作流引擎将本流程实例暂停执行; 完成状态是指流程实例已经正常的完成; 终止状态是指过程实例被非正常的终止掉.

流程实例的状态变迁如图 5 所示.

WorkflowEngine 类的 Suspend、Resume、Terminate 方法[7]实现对流程状态的管理. 另外, StartFlow 方法执行时也会影响流程状态，新建的流程实例处于初始态，起始活动开始执行后，流程实例被置成运行态. 活动执行也会影响流程的状态，在 Activity 类的 Run 方法中，后转发规则控制之后，会检查流程实例是否执行完毕，如果执行完毕，流程会被置为完成态.

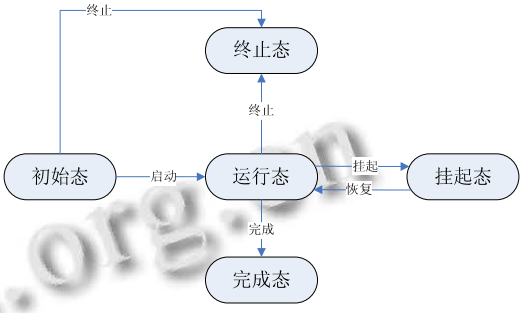

图 5 流程实例的状态变迁图

2.4 日志管理和异常处理模块实现

日志管理模块

日志管理模块用来记录引擎系统的运行过程信息，通过这些信息，系统管理员可以查看引擎运行状态、过程.

异常处理模块

在工作流管理系统中，异常事件出现是不可避免的[8].

异常处理措施可以分成几种基本策略: 忽略、取消、替代和补偿，表示成 S={Ignore, Abort, Alternate, Compensate}.

系统中定义了若干可预知的异常类. 并为各个异常定义了不同的 Si 值，根据 Si 值各有不同的处理方式，如挂起流程实例、终止流程实例等.

引擎系统通过 EngineException 类继承系统异常类，该类定义了各种异常情形，而在 Try-Catch 结构中给出相应的异常处理逻辑的实现.

3 工作流引擎的应用

以某制药企业采用项目化管理为例，引擎负责处理平台中所有流程控制相关的任务，以 A 级立项论证报告审批流程为例来做说明引擎在平台中的应用. 利用 JaWE 建模工具可视化建模后，将模型定义文件上传上到系统中，经过流程配置细化后，流程配置页面如图 6 所示.

图 6 流程细化配置页

当配置完成后，整个业务流程的配置过程就完成了，工作流引擎与执行单元很好的结合在一起组成业务系统了．普通用户可以启动该流程进行具体的工作任务了．业务流程会在引擎的控制下，正确的流转．

4 总结

针对企业中普遍面临的业务流程处理问题，分析业务流程的一般特征，进行可视化业务流程建模，在此基础上，实现了一个业务流程可以柔性配置工作流引擎系统．

引擎的实现为业务流程系统的开发提供了一种新的实现模式，用户先对业务流程进行建模，并分别实现各个执行单元，然后做简单的配置，就可以利用工作流引擎组合出一个工作流管理系统，实现对具体业务的处理．采用这种模式，可以方便地应用引擎开发新的业务系统，也可以方便地将引擎集成到已有的业务系统．

参考文献

1 范玉顺.工作流管理技术基础.北京:清华大学出版社,2001.
2 周建涛,史美林,叶新铭.柔性工作流技术研究的现状与趋势.计算机集成制造系统,2005,11(11):1501–1509.
3 于勇,彭岩.工作流管理系统柔性机制.计算机工程,2008,34(24):40–42.
4 苏鹏,王文,刘学理,等.基于关系数据库的轻量级工作流引擎.计算机应用与软件,2008,25(3):117–119.
5 覃海宁,徐东.工作流访问控制模型研究综述.玉林师范学院学报(自然科学),2009,30(3):135–137.
6 李竞杰,王维平,杨峰.柔性工作流理论方法综述.计算机集成制造系统,2010,16(8):1569–1577.
7 金达文.基于 jBPM 工作流引擎的 OA 系统中的应用[学位论文].北京:北京邮电大学,2007.
8 Brambilla M, Ceri S, Comai S, et al. Exception handling in workflow-driven web applications. Proc. of the 14th Int. Conf. on World Wide Web. May, 2005. 170–179