基于关系数据库的工作流系统设计与实现

姚旭平

Version 0.201　building　051010

1、绪论

1.1、工作流技术的来源和发展

工作流的概念起源于生产组织和办公自动化领域，提出的目的是通过将工作分解成定义良好的任务、角色，按照一定的规则和过程来执行这些任务并对它们进行监控，达到提高工作效率、降低生产成本、提高企业生产经营管理水平和企业竞争力的目标。在计算机网络技术和分布式数据库技术迅速发展，多机协同工作技术日益成熟的基础上，于 20 世纪 80 年代中期发展起来的工作流技术为企业更好地实现这些经营目标提供了先进的手段。工作流技术一出现马上就得到广泛的重视和研究。至今工作流管理技术已成功地运用到图书馆、医院、保险公司、银行等行业，然而它更重要的应用还是在工业领域，特别是制造业领域。

1993 年，工作流管理联盟（Workflow Management Coalition，WfMC）的成立标志着工作流技术进入了相对成熟相对规范的阶段。为了实现不同工作流之间的相互操作，WfMC 在工作流相关术语、结构体系、应用程序接口、管理控制接口、过程语言描述等方面制定了一系列标准和规范。这些工作在很大程度上促进了工作流技术的发展和工作流管理系统在企业中的应用。

目前，在全球范围内，对工作流的技术研究以及相关的产品开发进入了更为繁荣的阶段，更多更新的技术被集成进来，文件管理系统、数据库、电子邮件、Internet 服务等都已被容纳到工作流管理系统之中。工作流产品的市场每年以两位数字的速度迅猛增长。市场上工作流产品发展迅速，据统计，1997 年工作流产品的增长率超过 35%。作为支持企业经营过程重组（Business Process Re-engineering, BPR）、经营过程自动化（Business Process Automation, BPA）的一种手段，工作流技术的研究应用日益受到学术界与企业界的重视。

1.2、本文结构安排

本文组织如下：

1. 绪论。介绍工作流技术的起源和发展。
2. 研究与分析工作流相关概念，参考模型，工作流技术的现状和发展等。
3. 工作流模型及工作流建模。
4. 工作流系统设计与实现，重点讲述设计的核心算法。
5. 工作流管理系统与通用业务系统的结合。
6. 本工作流管理系统总体使用说明
7. 后记，以及未来规划，交流方式。

2、工作流概述

2.1、工作流定义

根据 WfMC 的定义，工作流（Workflow）就是自动运作的业务过程部分或整体，表现为参与者对文件、信息或任务按照规程采取行动，并令其在参与者之间传递。简单地说，工作流就是一系列相互衔接、自动进行的业务活动或任务。如果将整个业务过程看作是一条河，其中流过的就是工作流。

从工作流定义中可以看出，工作流是经营过程的一个计算机实现，而工作流管理系统[1]则是这一实现的软件环境。使用工作流作为经营过程的实现技术首先要求工作流系统能够反映经营过程的以下几个问题，即经营过程是什么（由哪些活动、任务组成，也就是结构上的定义）、怎么做（活动间的执行条件、规则以及所交互的信息，也就是控制流与信息流的定义）、由谁来做（人或计算机应用程序，也就是组织角色的定义）、做得怎样（通过工作流管理系统对执行过程进行监控）。图 2-1 给出了一个称为工作流伞的示意图，反映了工作流覆盖的经营过程范围与对应的工作流研究领域。

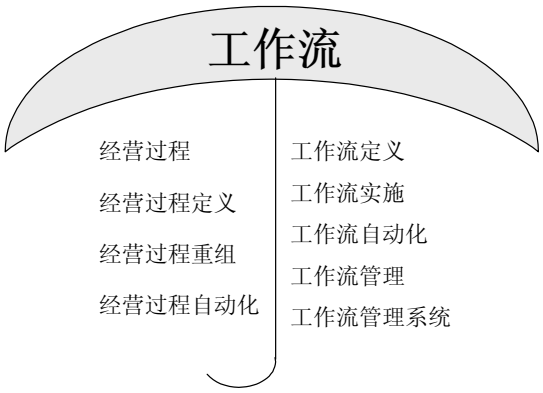


图 2-1 工作流伞

在企业应用中，工作流经常与经营过程重组相联系，完成对一个组织（或机构）中核心经营过程（或者称为关键经营过程）的建模、评价分析和操作的实施。虽然并非所有的 BPR 都需要采用工作流的方式实施，但是，工作流技术通常是实施 BPR 的一个较好方法，因为工作流提供了经营过程逻辑与其信息支撑系统的分离，并实现了应用逻辑和过程逻辑的分离，这种方式在企业进行实际应用时具有显著的优点。它可以在不修改具体功能模块实现方式（硬件环境、操作系统、数据库系统、编程语言、应用开发工具、用户界面等）的情况下，通过修改（重新定义）过程模型来改进系统性能，实现对生产经营过程部分或全部地集成管理，有效地把人、信息和应用工具合理地组织在一起，提高软件的重用率，发挥系统的最大效能。

2.2、工作流的技术优势和适用范围

工作流管理的目的是根据预定目标，找到合适的方法和手段来优化工作流程。其在企业的应用会给企业带来巨大的效益。采用工作流管理将使企业改变传统的按照功能来配置人员的组织结构，变成按照企业要实现的主要业务流程来配置组织结构，这样可以大大缩短主要业务过程的处理时间，提高对市场的响应能力。组织结构的改变将大大减少在企业内部不必要的物料、信息的传递时间。当然，整个企业组织结构的调整首先需要调整传统的以部门组织生产、人员从属于某个部门的做法，变成以项目组织生产和人员的工作方法。一个人可能同时从属于多个项目。

1、企业应用工作流管理系统主要可以取得以下优势：

(1) 提高企业管理的规范化程度。

(2) 更好的与上下游企业形成快速响应市场的供应链网络。

(3) 降低业务过程的整个处理时间，如在办公自动化环境中，通过更好地规划工作流程，并行执行相互独立的活动，减少文档传递过程中不必要的中间状态等方法来降低文档传递和临时储存时间。

(4) 降低管理成本，如避免不必要的重复工作，提高工作效率。

(5) 改进工作质量，如自动提供为完成某个任务所需要的相关信息。

(6) 在工作人员之间更好地均衡负荷，如在工作人员缺勤的情况下，自动地柔性分配潜伏人员。

(7) 通过在工作流模型中加入可预计的故障处理策略来提高系统的柔性。

(8) 在工作流的基础上改进控制策略，降低相应的控制成本，如通过监控工作流执行状态，利用分析和控制工具来进行优化控制。

(9) 通过对已经完成的工作流实例的分析，找出存在的不足，进而不断改进工作流程。

(10) 使工作内容更加丰富，并且提高工作人员的业务能力，减少工作人员进行单调乏味并且十分耗时的文档查找工作。

2、工作流技术可以应用于以下一些领域并发挥重要作用。

(1) 并行工程：工作流技术可以很好地用于产品开发过程的建模和管理，它也可以作为产品协同设计、产品设计中的冲突协调、产品数据管理与流程控制的支撑系统。在这一领域的应用中，需要增强工作流对产品数据及其相关集成文档的描述能力，并且需要在工作流技术融入 CSCW 的技术和方法。

(2) 敏捷制造：工作流管理系统可以作为企业间信息集成的使用工具，基于 WEB 和基于邮件方式的工作流管理系统可以为企业灵活地组建动态联盟和实现信息交换发挥重要作用。在这一应用领域中，要充分考虑广域网环境下系统之间信息传递的可靠性问题，以及不同工作流系统之间的互操作和重构问题。

(3) 供应链管理：工作流管理技术可以较好的用于实现供应链建模和管理功能，结合工作流仿真和优化技术，还可以用于进行企业分销体系的优化。

(4) 企业经营过程重组：这是工作流技术应用的主要领域。虽然工作流管理为系统的重构提供了必要的手段，但是要真正实现企业经营过程的快速重组，企业的应用系统需要按照组件的方式进行构建或改造，而且对应用组件的粒度要求应该与过程重组所需的灵活性相匹配。即灵活性要求较高，应用组件的粒度应该越小。

(5) 企业建模与系统集成：以工作流过程为核心，从功能、信息、组织与资源视图为辅助手段研究集成化的企业建模方法，并开发相应的集成化企业建模工具。在进行这方面的研究工作时，要重点解决不同视图模型之间的集成问题和模型的一致性问题，在此基础上，可以建立以工作流管理系统为基础的集成平台和集成框架软件，从而实现方便、快捷、灵活的应用系统集成。

工作流技术综合了计算机科学和管理科学中诸多研究领域的原理、方法和技术，如：数据库管理、面向对象技术、C/S 技术、汇编语言、图形化用户界面、系统集成、消息传递、文档管理、仿真等等。近年来，企业对过程建模、BPR 工具、敏捷制造、并行工程的需要为工作流技术的应用提供了一个广阔的市场，使工作流产品得以迅速发展。同时，工作流产品供应不断将信息技术、WEB 技术等研究中的最新研究成果应用于自己的产品开发中，促进了它的普及应用。虽然目前的工作流产品还存在很多问题，但是随着工作流技术的进一步发展，它必将在提高企业的效率和竞争力，使企业更好地适应市场变化等方面起到举足轻重的作用。

2.3、工作流参考模型

2.3.1、工作流管理系统功能

根据 WfMC 的定义，工作流管理系统（Workflow Management System, WFMS）是一个软件系统，它完成工作流的定义和管理，并按照在计算机中预先定义好的工作流逻辑推进工作流实例的执行。

通常，工作流管理系统是指运行在一个或多个工作流引擎上用于定义、实现和管理工作流运行的一套软件系统，它与工作流执行者（人、应用）交互，推进工作流实例的执行，并监控工作流的运行状态。

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，但它们具有很多共同的特性。概括地说，工作流管理系统提供了 3 种功能（见图 2-2）：

(1) 建立阶段的功能：主要考虑工作流过程和相关活动的定义和建模功能。

(2) 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。

(3) 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。

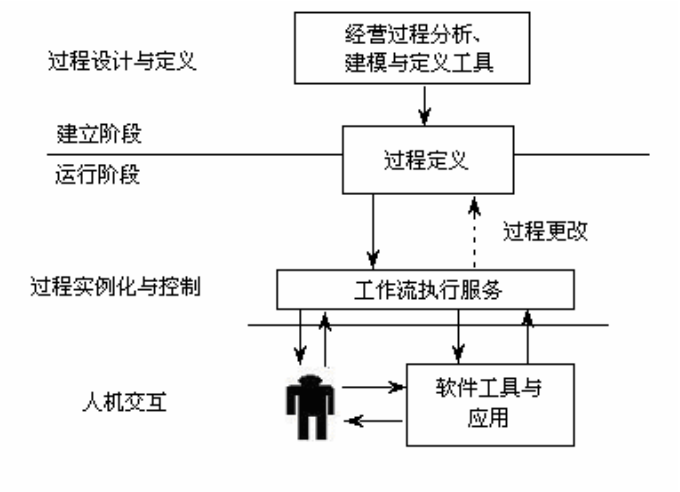


图 2-2 工作流管理系统的特性

2.3.2、工作流参考模型

图 2-3 给出了 WfMC 提出的工作流参考模型。参考模型描述了工作流管理系统结构中主要的模块以及模块之间的接口。工作流执行服务器周围的接口是 WAPI（Workflow APIs），通过这些接口可以访问工作流系统的服务，还可以与其他系统组件间进行交互。WfMC 定义的 5 类接口的功能：

接口 1：过程定义交换接口，定义了过程模型的互换格式和读写操作；

接口 2：客户端函数接口，约定所有客户应用与工作流服务之间的功能操作方式；

接口 3：应用程序接口，工作流机和直接调用的应用程序之间的直接接口；

接口 4：互操作接口，定义不同工作流管理系统之间的信息交互；

接口 5：系统管理与监控接口，实现对工作流的管理和监控。

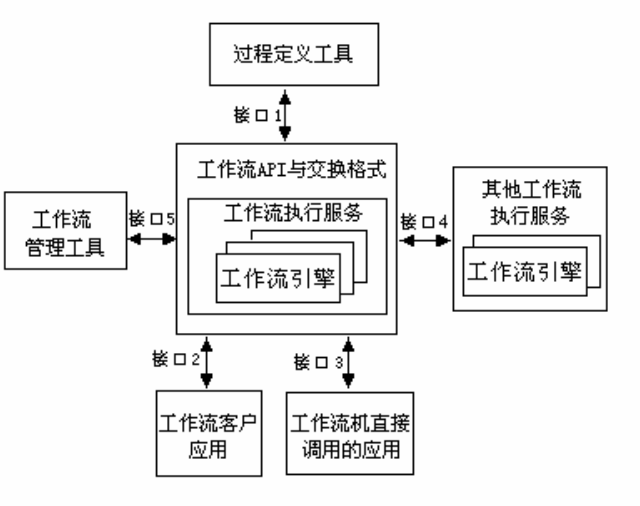


图 2-3 工作流参考模型

过程定义工具：以计算机能处理的形式进行过程定义，现在的大多数过程定义工具采用了图形方式，过程设计者通过绘图方式来创建过程模型，最后输出一个 XPDL 文件，有的过程定义工具还有分析、检测功能，帮助设计者设计出良好的过程模型。

工作流执行服务：由一个或多个工作流引擎组成，提供过程实例的执行，为活动进行导航，与外界资源交互完成各项活动，维护控制数据和相关数据等功能。

工作流客户端应用：提供用户操作工作流管理系统分配来的任务，由工作流任务表管理器和任务操作共同完成。工作流任务表管理器是一个软件模块，负责管理工作流的任务表，并完成与工作流参与者的交互操作。

工作流引擎直接调用应用：在工作流任务执行过程中，一些不需要人员参与的活动会直接调用一些应用。在简单的情况下，工作流引擎使用过程模型中定义的活动信息、应用程序类型，所需要的数据来激活外部应用程序；在复杂的情况下采用工具代理的方式。工具代理与工作流引擎之间通过专用集成接口来完成数据交换和消息传递。

系统管理和监控工具：对工作流在整个组织内的流动状况进行监控，并提供一系列管理功能，实现安全性、过程控制、授权等操作。典型的功能范围包括用户管理、角色管理、监控管理、资源管理、过程监控管理。具体如：过程模型的实例化，启动、挂起、恢复、终止过程实例；管理正在执行的过程实例等。

2.3.3、工作流实现模型

1、工作流管理系统的组成部分

图 2-4 为 WfMC 提出的工作流参考模型的体系结构图。这个参考模型的体系结构给出了抽象的工作流管理系统的功能组成部件和接口，它能够满足工作流管理系统和产品应该具有的主要功能，可为实现工作流产品之间的互操作提供公共的基础。从图 2-4 可以看出，工作流管理系统主要由三部分组成：

(1) 软件构件：完成工作流管理系统不同组成部分功能的实现，包括过程建模工具，工作流引擎，任务表管理器和用户界面；

(2) 系统控制数据：工作流管理系统中的一个或多个软件构件使用的数据，包括过程定义，组织/角色模型数据，工作流控制数据，工作流相关数据，任务表；

(3) 应用与应用数据：对于工作流管理系统来说，它们不是工作流管理系统的组成部分，而是属于外部系统和数据，它们被工作流系统调用来完成整个和部分工作流管理的功能，如被工作流管理系统调用的外部应用以及这些应用操作的数据。

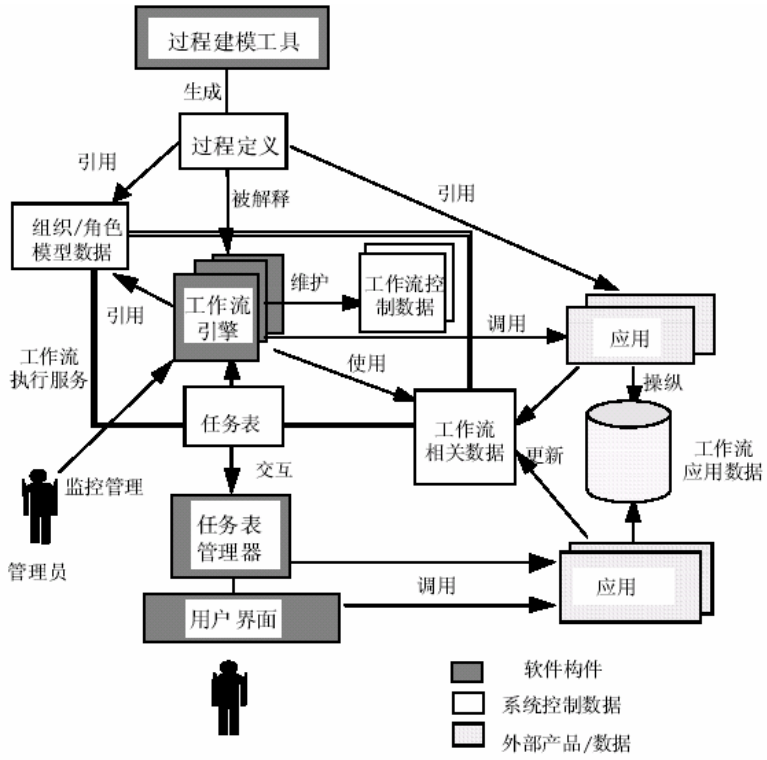


图 2-4 工作流管理系统的体系结构图

2、工作流实现模型各个模块说明

(1) 过程建模（定义）工具：过程建模工具（Process Definition Tool）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。

(2) 组织/角色模型：包含了组织结构和组织中角色的信息。这些信息往往与流程定义信息紧密相关。

(3) 工作流执行系统和工作流引擎：工作流执行系统（Workflow Execute System）也称为（业务）过程执行环境，包括一个或多个工作流引擎。工作流引擎（Workflow Engine）是 WM S 的核心软件。它的功能包括：解释过程定义；创建过程实例并控制其执行；调度各项活动；为用户工作表添加工作项；通过应用程序接口（API）调用应用程序；提供监督和管理功能等。工作流执行子系统可以包括多个工作流引擎，不同工作流引擎通过协作共同执行工作流。

(4) 工作流控制数据：被工作流执行系统和工作流引擎管理的系统数据，如工作流实例的状态信息、每一活动的状态信息等。

(5) 工作流相关数据：指与业务过程流相关的数据。WFMS 使用这些数据确定工作流实例的状态转移，例如过程调度决策数据、活动间的传输数据等。工作流相关数据既可以被工作流引擎使用，也可以被应用程序调用。

(6) 工作列表(Worklists)：流程执行中，当需要用户的交互时，工作流引擎便将工作项放置到由 worklist handler 管理的工作列表中，通过 worklist handler 实现与用户的交互。

(7) 应用程序和应用数据：应用程序可以直接被 WFMS 调用或通过应用程序代理被间接调用。通过应用程序调用，WFMS 部分或完全自动地完成一个活动，或者对业务参与者的工作提供支持。与工作流控制数据和相关数据不同，应用数据对应用程序来讲是局部数据，对 WFMS 的其他部件来说是不可见的。

3.1、工作流模型规范

3.1.1、工作流当前流行的模型规范

工作流标准还处于制定阶段，所以存在大量相互重叠的规范。在工作流领域第一个致力于标准化工作的是 Workflow Management Coalition (WfMC)，开始于 1993。 WfMC 发布的参考模型相对成熟完善，它定义了工作流管理系统和其他相关部分之间的接口。WfMC 的另一项成果是 XPDL 规范。 XPDL 定义了描述工作流声明部分（declarative part）的 XML 结构。因此目前 WfMC 推出的工作流参考模型和 XPDL 是工作流领域中最好的规范（纯个人认为。现在有不少人更推崇新的模型规范，这应该有他们的理由，主要我年级大了，学不动了，有机会还请多指教）。

在工作流模型规范中，目前处于一种百家争鸣的局面，主要有：

(1) WfMC's XPDL - WfMC 是由约 300 家成员参加的组织，基于参考模型定义了一系列的标准。参考模型用用例（use case）的形式描述了工作流系统和其他相关部分之间的关系。XPDL 是 WfMC 制定的描述业务流程控制流（control flow ）的 XML 格式规范。1994 年 11 月，WfMC 发布了工作流管理系统的参考模型。参考模型提出了五类接口，有关过程模型的定义则构成了接口 1 的核心内容。接口 1 早期的标准为 WPDL（Workflow Process Definition Language），后来,这一接口的规范变更为 XPDL。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。

(2) ebXML's BPSS - ebXML 是协同流程的相关标准集，主要关注不同公司流程之间的通讯。可以看作 EDI 的继承者。 ebXML 是由 OASIS 和 UN/CEFACT 联合发起。 BPSS 是 ebXML 的规范，其中的概念和本文阐述的很接近。

(3) BPMI's BPML & WSCI - (Intalio, Sun, SAP, ...)BPMI 定义的一个规范 (BPMN) ，描述如何将“可执行”业务流程可视化地表现。

(4) BPEL - Web 服务业务流程执行语言（BPEL）是一种编程语言，它明确定义了基于 Web 服务的业务流程，由一系列基于消息交换的规范（ XLANG, WSFL, BPML）产生。还有一个将此规范引入到 Java 的提案: BPELJ。

(5) UBL - The Universal Business Language (UBL)定义了用于不同组织间通讯的 XML 文档标准库。可以看作是对 ebXML 的补充，因为 ebXML 只定义了建立组织间流程的基础。此规范的竞争对手是 RosettaNet 标准中的一个子集。

工作流模型被描述成可被计算机处理的形式化表示为统成为“过程定义”，其描述语言称为“工作流过程定义语言”，是用来方便交互和在不同格式模型之间实现转换。不同的工作流实现模型有自己的规范描述语言，语法规范、标示符、关键字、文法规则。比如 WIDE 项目使用的是 WFDL（Workflow Description Language，工作流定义语言）；IBM FlowMark 使用的是 FDL（FlowMark Definition Language）；而 WfMC 使用的是 XPDL。

3.1.2、XPDL 模型规范

在 WfMC 提出的工作流参考模型中，以工作流服务（工作流引擎）为核心，定义了五种接口，其中有关过程定义的引入和导出构成了接口 1 的主要功能。工作流过程定义语言就属于这部分内容。

在企业应用中，使用者会用不同工具对工作流进行建模、分析、归档。工作流过程定义接口（接口 1）定义了一个公共的交换格式，使得不同产品的工作流定义可以实现模型交换。

接口 1 还实现了工作流过程的定义和执行两个不同阶段的分离，使得工作流的定义工具和执行引擎相互独立，极大地增加了工作流产品的灵活性和可配置性，为用户提供更好、更方便的选择和组合。图 3-4

体现了 XPDL 在不同工作流管理系统中所起到的桥梁作用。

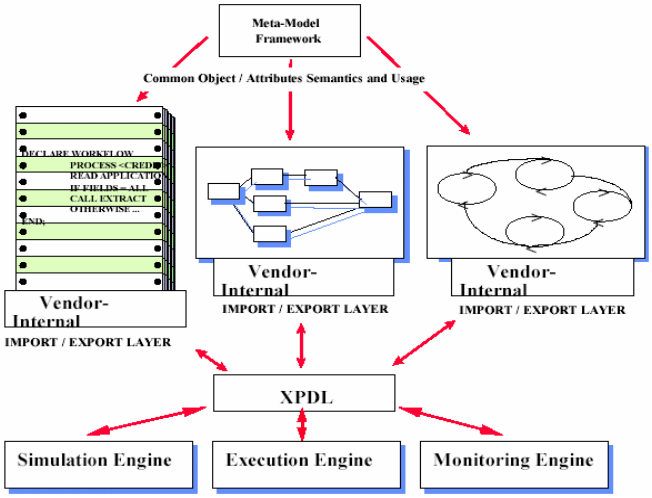


图 3-4　XPDL 的桥梁作用

在现实中，一项业务流程（如投诉、请假流程等）都能用工作流模型的过程模型表示出来。过程模型由一系列活动（Activity）按照一定的约束关系组成。这些活动在具体的执行过程中需要使用一些资源和角色（对应资源模型和组织模型）。工作流建模就是将这一系列的活动、活动之间的关系按照事务的需求定义出来，并对相应的活动安排活动的起止日期、活动要使用的角色、资源等，然后将模型上传给引擎，通过工作流引擎将任务项在“合适的时间发送到合适的人员”。

工作流建模就是通过对业务流程的分析，使用建模工具以计算机能处理的形式对工作流模型进行建模，并输出一个能被引擎解释并执行的过程定义。该过程定义可以使用 XPDL 描述。在工作流模型定义方面，WfMC 开展了三个方面的工作：

- 定义了一个元模型：所谓元模型指用来描述说明模型的模型，见图 3-5。这里的工作流模型的元模型用来描述工作流模型内在内在联系的模型，描述了工作流模型内部包含的各个对象、对象之间的关系以及它们的属性。该元模型有利于建立可相互交换信息的工作流模型。
- 定义了一套用于过程定义的标准交换接口。
- 基于元模型给出了工作流模型的具体设计。

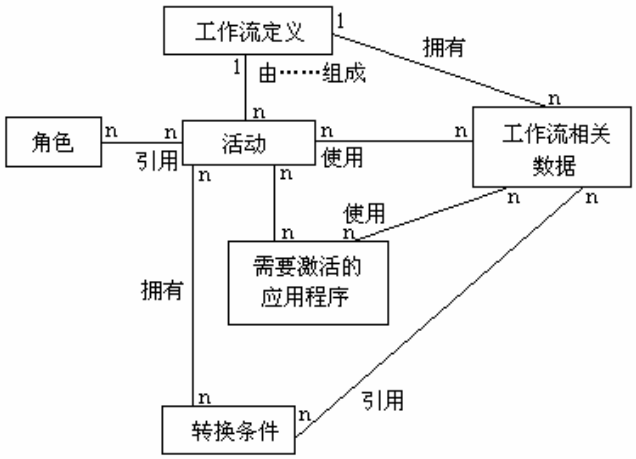


图 3-5 XPDL 的元模型实体

WfMC 的工作流过程语言包括 6 个基本实体，见图 3-5：

- 工作流定义：反映企业一个业务流程的目的。

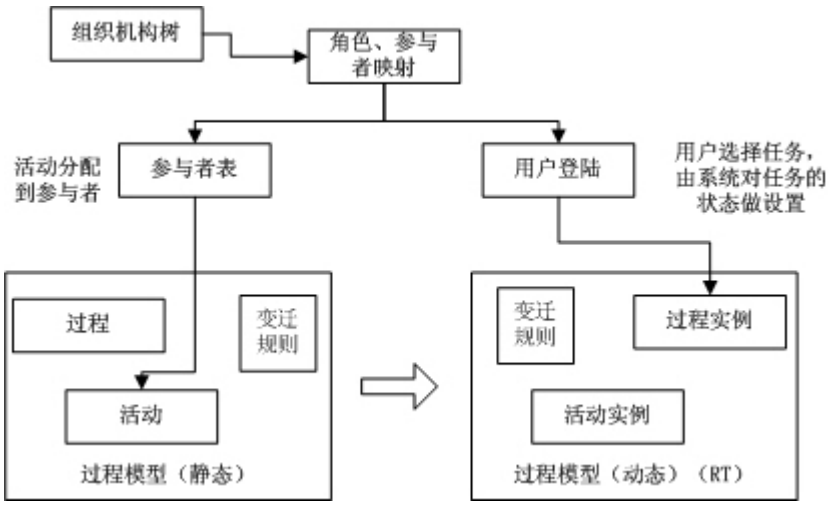
- 活动：对应业务流程中的任务，主要反映了完成该业务流程需要执行的哪些功能操作。
- 转换条件：负责为过程实例提供导航依据，对应于一个业务流程中的业务规则和操作顺序。
- 工作流相关数据：是引擎执行任务推进的依据之一，引擎根据相关数据和转换条件执行后续活动。
- 角色：角色或组织实体决定了参与某个活动的人员或组织单元，描述了业务流程中参与的操作人员和组织单位。
- 应用程序：描述了用于完成业务流程所采用的外部工具或工具。

活动是过程定义的核心部分。一个过程包括若干活动组成；特定的角色参与活动的执行；活动执行过程中使用工作流相关数据，激活特定外部应用程序；过程实例的推进是根据转换条件和工作流相关数据进行的。

过程元素定义：

	包	工作流过程	活动	条件转移	应用程序	相关数据	角色
通用属性元素	标识 名称 描述 扩展属性	标识 名称 描述 扩展属性	标识 名称 描述 扩展属性	标识 名称 描述 扩展属性	标识 名称 描述 扩展属性	标识 名称 描述 扩展属性	标识 名称 描述 扩展属性
特有属性	XPDL 版本 所有者 ID 创建日期 版本号 创建者 代码页 关键词 发布状态 一致类 优先权	创建日期 版本 创建者 代码页 关键词 公布状态 优先权 限制 有效开始日期 有效截止日期	活动模式 发散 汇聚 优先权 限制 开始模型 结束模型 最终期限			数据类型	角色类型
参考需要	依赖	参数 依赖	执行者 工具类型 子流程类型 活动集 真实参数	条件 来源 目的	参数	初始值	
信息描述	文档 图标	文档 图标	文档 图标				
过程模拟优化信息	单元	单元 持续时间 等待时间 工作时间	成本 持续时间 等待时间 工作时间				

3.2、XPDL 与关系数据库的映射



整体构架图

存在两大基本数据，模型库和运行库（RT 库）。模型库是用来模拟 XPDL 来存储过程信息的数据；RT 库是用来保留过程、活动运行期间的状态改变。

过程模型存放设计好的模型，用数据相互关系来表示，而工作流过程在运行状态由 Runtime 来表示，每个 Runtime 活动都有对应的角色分配，分配到具体的人，就是个人任务列表。

在往数据库中映射中，不会把所有的元素考虑进去，考虑主要的。模型库中主要包括：

过程数据；**活动数据**；**参与者信息**；**转化条件**；**相关数据**；外部应用程序信息（v2 版不考虑）

XPDL 包（xpdl package），

是一个容器，包括了全部工作流元素的说明，在这里舍弃，在当前的版本采用过程作为容器

Applications	工作流应用程序声明列表	不启用
ConformanceClass	定义针对过程定义的结构性约束	
DataFields	为该包定义的工作流相关数据列表	
ExtendedAttributes	外部扩展列表	
Id	标识	
Name	名称	
PackageHeader	包头信息	
Participants	参与者，	
RedefinableHeader	可重定义包头信息，由包和过程定义两者使用	
Script	标识在表达式中使用的脚本语言	
TypeDeclarations	在包中使用的数据类型列表	
WorkflowProcesses	工作流过程列表	

在以后的版本可能会添加该描述

过程（process）

定义了组成工作流的元素。包括了活动的定义、声明、可选、迁移、应用程序、过程相关数据等

Id	标识工作流过程	启用
Name	名称	启用
AccessLevel	过程的访问级别：public 和 private public 可由外部系统和应用程序调用 private 只能从实现类型为 subflow 的活动调用	不启用
Activities	工作流过程中的活动列表	通过外健与活动关联
ActivitySets	活动和迁移的自包含集合的列表	
Applications	应用程序声明列表	现在不考虑 apps，将来启用
DataFields	工作流相关数据列表	在相关参数表中是
ExtendedAttributes	扩展属性	不启用
FormalParameters	参数列表，可以传递给过程	通过外健与相关参数表关联
Participants	参与者列表，用于执行过程	和 DataFields 启用
ProcessHeader	过程头信息	部分启用
RedefinableHeader	可重定义头信息	部分启用
Transitions	连接过程的活动的迁移列表	通过外健与活动变迁表关联

活动（act）

Id	标记	启用
Name	工作流过程的活动名称	启用
BlockActivity	执行活动集的活动	不启用
Deadline	截止日期，如果达到截止日期， 则需要激活一个相应的活动	不启用
Description	活动的描述	启用
Documentation	活动的帮助文档的地址	不启用
ExtendedAttributes	可选，扩展属性	不启用
FinishMode	活动结束模式	放在活动类型中描述
Icon	活动图标的地址	不启用
Implementation	常规的活动，如果不是 route，则是必须的，	放在活动类型中描述
Limit	预计持续时间，用于时间管理，单位 DurationUnit	不启用
Performer	连接到实体工作流参与者，可能为表达式。	通过外健与参与者连接
Priority	活动开始的优先级，	启用
Route	“哑”活动	放在活动类型中描述
SimulationInformation	估计，用于活动的仿真	不启用
Startmode	活动的开始模式	放在活动类型中描述
TransitionRestrictions	提供迁移限制和上下文的语义描述	通过活动变迁表来描述，该表的外健和过程表相连，在活动表中不启动该元素

活动变迁（Transitions）

Id	标识迁移	启用
Name	名称	启用
Condition	迁移条件表达式（缺省为 true）	不启用
Description	描述	不启用
ExtendedAttributes	可选，扩展属性	不启用
From	前夕的来源（活动标识符）	启用
To	迁徙的目标（活动标识符）	启用

变迁条件（condition）

Type	定义了迁移的类型，有 condition ：条件满足，则实行 otherwise ：指示迁移为缺省迁移，如果没有条件被满足，则执行 exception ：存在一个异常，并异常条件满足，则执行 defaultexception ：指示迁移为缺省迁移，如果没有异常条件被满足，则执行
Xpression	使用 xml 标记的条件表达式

工作流相关数据

代表工作流过程或包定义的变量。一般用户维护决策数据（条件判断）或应用数据值（引用参数）

Id	标识	启用
Name	名称	启用
DateType	过程变量的数据类型	启用
Description	所定义数据的描述	启用
LsArray	是否为数组	不启用
Length	数据的长度	不启用
ExtendedAttributes	可选，扩展的属性	不启用

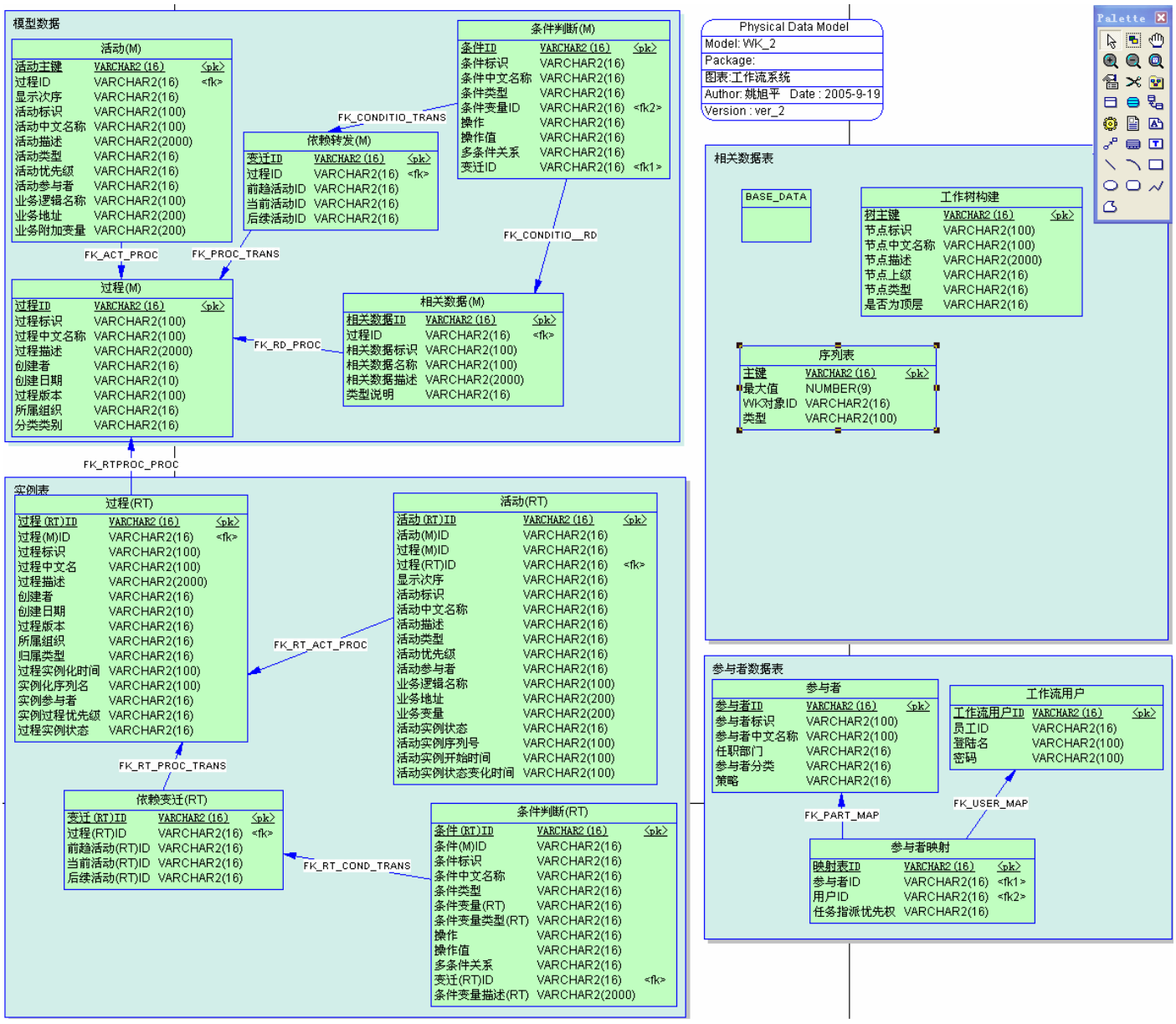
参与者

工作流参与者为如下类型：资源集、资源、组织单元、角色、人、系统。

角色和资源用作抽象的执行者，在运行时，被分配具体的人或程序

Id	标识	
Name	名称	
Descripation	描述	
ExternalReference	外部参与者的引用	
ExtendedAttributes	可选，扩展属性	
ParticipantType	工作流参与者实体类型的定义	

3.2.1、表结构


模型数据
活动(M)
活动主键 VARCHAR2(16) <pk>
过程ID VARCHAR2(16) <fk>
显示次序 VARCHAR2(16)
活动标识 VARCHAR2(100)
活动中文名称 VARCHAR2(100)
活动描述 VARCHAR2(2000)
活动类型 VARCHAR2(16)
活动优先级 VARCHAR2(16)
活动参与者 VARCHAR2(16)
业务逻辑名称 VARCHAR2(100)
业务地址 VARCHAR2(200)
业务附加变量 VARCHAR2(200)
条件判断(M)
条件ID VARCHAR2(16) <pk>
条件标识 VARCHAR2(16)
条件中文名称 VARCHAR2(16)
条件类型 VARCHAR2(16)
条件变量ID VARCHAR2(16) <fk2>
操作 VARCHAR2(16)
操作值 VARCHAR2(16)
多条件关系 VARCHAR2(16)
变迁ID VARCHAR2(16) <fk1>
依赖转发(M)
变迁ID VARCHAR2(16) <pk>
过程ID VARCHAR2(16) <fk>
前趋活动ID VARCHAR2(16)
当前活动ID VARCHAR2(16)
后续活动ID VARCHAR2(16)
过程(M)
过程ID VARCHAR2(16) <pk>
过程标识 VARCHAR2(100)
过程中文名称 VARCHAR2(100)
过程描述 VARCHAR2(2000)
创建者 VARCHAR2(16)
创建日期 VARCHAR2(10)
过程版本 VARCHAR2(100)
所属组织 VARCHAR2(16)
分类类别 VARCHAR2(16)
相关数据(M)
相关数据ID VARCHAR2(16) <pk>
过程ID VARCHAR2(16) <fk>
相关数据标识 VARCHAR2(100)
相关数据名称 VARCHAR2(100)
相关数据描述 VARCHAR2(2000)
类型说明 VARCHAR2(16)
FK_CONDITIO_TRANS
FK_ACT_PROC
FK_PROC_TRANS
FK_CONDITIO__RD
FK_RD_PROC
FK_RTPROC_PROC
实例表
过程(RT)
过程(RT)ID VARCHAR2(16) <pk>
过程(M)ID VARCHAR2(16) <fk>
过程标识 VARCHAR2(100)
过程中文名 VARCHAR2(100)
过程描述 VARCHAR2(2000)
创建者 VARCHAR2(16)
创建日期 VARCHAR2(10)
过程版本 VARCHAR2(16)
所属组织 VARCHAR2(16)
归属类型 VARCHAR2(16)
过程实例化时间 VARCHAR2(100)
实例化序列名 VARCHAR2(100)
实例参与者 VARCHAR2(16)
实例过程优先级 VARCHAR2(16)
过程实例状态 VARCHAR2(16)
活动(RT)
活动(RT)ID VARCHAR2(16) <pk>
活动(M)ID VARCHAR2(16)
过程(M)ID VARCHAR2(16)
过程(RT)ID VARCHAR2(16) <fk>
显示次序 VARCHAR2(16)
活动标识 VARCHAR2(16)
活动中文名称 VARCHAR2(16)
活动描述 VARCHAR2(16)
活动类型 VARCHAR2(16)
活动优先级 VARCHAR2(16)
活动参与者 VARCHAR2(16)
业务逻辑名称 VARCHAR2(100)
业务地址 VARCHAR2(200)
业务变量 VARCHAR2(200)
活动实例状态 VARCHAR2(16)
活动实例序列号 VARCHAR2(100)
活动实例开始时间 VARCHAR2(100)
活动实例状态变化时间 VARCHAR2(100)
FK_RT_ACT_PROC
FK_RT_PROC_TRANS
依赖变迁(RT)
变迁(RT)ID VARCHAR2(16) <pk>
过程(RT)ID VARCHAR2(16) <fk>
前趋活动(RT)ID VARCHAR2(16)
当前活动(RT)ID VARCHAR2(16)
后续活动(RT)ID VARCHAR2(16)
条件判断(RT)
条件(RT)ID VARCHAR2(16) <pk>
条件(M)ID VARCHAR2(16)
条件标识 VARCHAR2(16)
条件中文名称 VARCHAR2(16)
条件类型 VARCHAR2(16)
条件变量(RT) VARCHAR2(16)
条件变量类型(RT) VARCHAR2(16)
操作 VARCHAR2(16)
操作值 VARCHAR2(16)
多条件关系 VARCHAR2(16)
变迁(RT)ID VARCHAR2(16) <fk>
条件变量描述(RT) VARCHAR2(2000)
FK_RT_COND_TRANS
Physical Data Model
Model: WK_2
Package:
图表:工作流系统
Author: 姚旭平 Date : 2005-9-19
Version : ver_2
Palette
相关数据表
BASE_DATA
工作树构建
树主键 VARCHAR2(16) <pk>
节点标识 VARCHAR2(100)
节点中文名称 VARCHAR2(100)
节点描述 VARCHAR2(2000)
节点上级 VARCHAR2(16)
节点类型 VARCHAR2(16)
是否为顶层 VARCHAR2(16)
序列表
主键 VARCHAR2(16) <pk>
最大值 NUMBER(9)
WK对象ID VARCHAR2(16)
类型 VARCHAR2(100)
参与者数据表
参与者
参与者ID VARCHAR2(16) <pk>
参与者标识 VARCHAR2(100)
参与者中文名称 VARCHAR2(100)
任职部门 VARCHAR2(16)
参与者分类 VARCHAR2(16)
策略 VARCHAR2(16)
工作流用户
工作流用户ID VARCHAR2(16) <pk>
员工ID VARCHAR2(16)
登陆名 VARCHAR2(100)
密码 VARCHAR2(100)
FK_PART_MAP
FK_USER_MAP
参与者映射
映射表ID VARCHAR2(16) <pk>
参与者ID VARCHAR2(16) <fk1>
用户ID VARCHAR2(16) <fk2>
任务指派优先权 VARCHAR2(16)

3.2.2、系统基础数据

类别	名称		内容
过程状态	初始化	inactive	
	运行中	running	
	挂起	suspended	
	终止	terminated	
	结束	completed	
活动状态	没有启动	not_started	
	初始化	inactive	
	就绪等待	waiting	
	运行中	running	
	挂起	suspended	
	与汇聚同步	pending	
	终止	terminated	
	完成	completed	
活动类型	启动活动	start	过程的起始，不作实质内容
	结束活动	end	过程的结束，不作实质内容
	人工活动	human	与参与者进行交互，通过 url 地址得到业务逻辑页面，通过“点击完成”来明确通知引擎执行下一个活动。
	自动处理活动	tool	可以调用一个外部的应用，
	与发散	and_split	
	与汇聚	and_join	“waiting”，等待其他路径活动汇合
	或发散	or_split	判断，选择哪个路径
	或汇聚	or_join	先来先走，类似串行过程
	哑活动	dummy	哑活动，用户丰富过程的表示能力
优先权	很高	very_hight	
	高	hight	
	正常	normal	
	低	low	
	很低	very_low	
参与者类型	资源集	resource_set	资源的集合
	资源代理	resource	特定的资源代理
	角色	role	该类型允许使用角色或技能集合表示执行者。在该上下文中，角色是某人在组织内所拥有的职责。由于职责不必是唯一的，（出于管理的目的或以备万一进行异常处理）可以定义等价物以及与角色相关的人员的列表。
	组织单位	organization_unit	位于组织模型内的部门或任何其它的单元
	人工	human	通过应用程序与系统交互的人，该应用程序代表针对参与者的用户接口

	自动化代理	system	
条件操作	等于	=	两个设定为 string 类型
	不等于	!=	两边设定为 string 类型
	小于	<	设定为 number 型
	大于	>	设定为 number 型
条件类型	前依赖规则	before	
	后转发规则	after	
相关数据类型	字符串	String	
	日期	Date	
	时间戳	Datatime	
	整形	Int	
	浮点形	Float	
	布尔	Boolean	
工作项指派策略	唯一对应	Only	一个参与者只对应一个用户（缺省）
	先进先出	Fifo	一个参与者对应 n 个用户，哪个用户先登陆，把任务分配给哪个用户
	任务少优先	Load	比较该参与者的所有用户，当前任务最少的得到该任务指派
	优先权大优先	priority	比较参与者的各个用户优先权，优先权大的得到该任务指派。优先权在映射表中额外一个字段保存
节点类型	最高节点	Top	
	模型构建模块	Model	用于用户在构建工作流模型时，左边所出现的树内容
	运行监控	Monitor	在过程已经运行，用户处于监控状态中，左边出现的树内容
条件间关系	多条件与	and	
	多条件或	or	

3.3、工作流建模

业务流程建模为是为了获得一个业务流程的工作流模型描述，不同的建模方法有其不同的适用范围。目前比较典型的工作流建模方法主要有三种：

(1) 基于活动网络的建模方法：以活动与活动之间的关系作为基础建立工作流模型。特点是界面友好，容易理解，缺点是缺乏严格的形式化语义，不立于模型验证。

(2) 基于状态图和活动图建模的方法：活动直接对应了工作流参考模型中的活动概念，反映了系统功能上的分解。活动图是一种有向图，采用标注数据信息的有向弧表示活动间的数据流；而状态图反映了系统的动态行为，表述活动间的控制流程。该方法已经被 OMG 组织接纳为 UML 描述系统动态行为的建模方法。

(3) 基于 Petri 网的建模方法:在 Petri 网的基础上，提出了工作流网的概念，以及如何把把工作流模式映射到 Petri 网作了研究。Petri 网兼顾了严格的语义和图形语言，也是一种基于状态的建模方法，并拥有强有力的分析方法。

比较而言，基于 Petri 网的过程建模有着其他方法不能比拟的优势，将来的主流趋势是采用 Petri 网建模技术和建模工具，所以在后续软件的开发和改进中，也会要采用基于 Petri 网技术的建模工具。目前因为时间关系和自身条件的限制，本论文中主要利用 Petri 理论对工作流过程进行分析，不涉及独立设计 Petri 建模工具，在设计工作流过程模型时采用 JAWE 开源工作流建模器，该工具属于活动网络的建模方法，产生的是符合 WfMC 规范的 XPDL 文件。

3.3.1、基于 Petri 网的建模方法

3.3.1.1、基于 Petri 网的工作流描述

Petri 网是一个图形化的数学建模工具。一方面可以利用图形化的方式来描述工作流过程，令一个方面可以通过形式化的分析技术检查工作流模型的正确与否，甚至对其进行性能分析。

Petri 网定义成三元组，PN=(P,T,F),其中：

- P={p1,p2,p3…pm }是库所的有限非空集;
- T={t1,t2,t3…tn}是变迁的有限非空集;
- F= P×T∪T×P 是有向弧的集合（×表示笛卡尔集合）,P 和 T 还满足 P∩T=ø 且 P∪T≠ø;

Petri 网由库所和变迁组成，圆圈表示库所，矩形表示变迁，库所和变迁用有向弧连接。Petri 网的动态行为用托肯（token）的分配来描述，用(•)表示，Petri 网结构是固定的，而库所中的托肯的分布是可以变化的。变迁出发的条件是它每个输出库所中含有的托肯数目要多于从该库所到变迁的有向弧数。

变迁是 Petri 网中的主动元素，通过实施变迁，过程从一个状态转移到另一个状态。变迁经常表示为事件、操作、转换、传输；库所是 Petri 网中的被动元素，不能改变网的状态，通常表示为媒介、缓冲器、位置、阶段、条件；托肯表示为对象，可以表示为一个特定的事务或抽象的信息。

通过扩展 Petri 网模型定义，满足如下两个条件：条件 1，使得工作流网必须具有一个起始点和一个终止点，进入起始库所的托肯代表一个过程实例的开始；而进入终止库所的托肯代表一个过程实例的结束；条件 2，使得工作流网中不存在处于孤立状态的活动与条件，所有的活动与条件都位于起始点到终止点的通路上，该 Petri 网就能被称为工作流网，其数学定义为：

- N 有两个特殊的库所 i 和 o 。i 是一个起始库所，即*i = ø；o 是一个终止库所，即 o* = ø。
- 如果在 N 中加入一个新变迁 t’，使*t’={o}且 t’*={i}，则得到的 N’是一个强连通的 Petri 网。

在建模过程中，如果使用条件和任务的概念，则库所表示条件，变迁表示任务。一个变迁（任务）有一定数量的输入和输出库所，分别表示任务的前置条件和后置条件。库所中的托肯表示可以使用的资源或数据。被建模系统的状态可以用每个库所中的托肯表示状态的变化由变迁的触发引起，变迁触发的结果是每个连到该变迁的库所，在触发后，所有的输入库所减少一个托肯，而所有的输出库所增加一个托肯状态的变化代表了流程的演进过程。

3.3.1.2、工作流模式与工作流网的映射

在工作流系统应用范畴内，用工作流过程表示企业的一个业务流程，用库所表示条件，其中包含一个开始库所和一个结束库所分别对应了过程的开始和结束，用变迁表示任务，任务的执行方式由路由决定。根据 WfMC 的定义，工作流应该包括 4 种基本的路由结构，分别是顺序、分支、并行、循环，由它们能构造出复杂的流程。

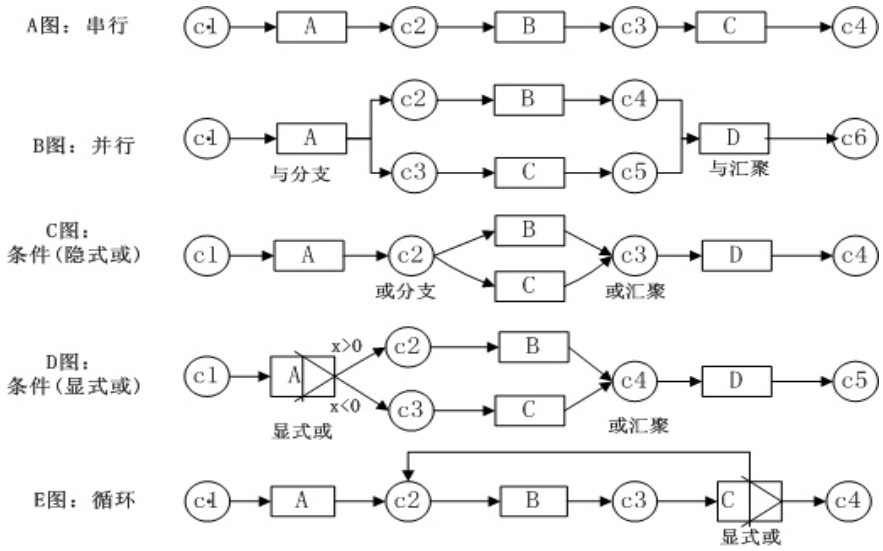


图 3-6 用 Petri 网描述工作流的四种基本模式

顺序路由 用于表达任务间的因果关系。见图 3-6 (a 图)。只有 A 执行完后 B 才能开始。C1 模拟任务 A 的后置条件和任务 B 的前置条件。活动的前置条件指明相应活动的启动条件，启动条件是通过相应活动的直接前趋活动以及相应的状态标志来表示的；活动的后置条件是当前活动所对应的任务结束后该启动哪些后继活动。在 Petri 网中，通过在两个任务间添加一个库所进行链接的方式来建模。

并行路由 如图 3-6 (b 图) 所示，任务 B 和 C 可按任意顺序执行。为了表达这种关系，在 Petri 网中添加 And-Split 任务和 And-Join 任务。任务 A 表达了 B 与 C 并行的关系，而 D 表达了同步的思想，只有 B 与 C 所在的活动都执行结束后 D 才可以开始。

选择路由（隐式或分支）当 c2 含有托肯后，就出现了 B 和 C 任务同时就绪的情况，但哪个能够真正被执行，从图中无法得知（所以称为隐式或分支），这取决于 B 和 C 被触发的结果。如果 B 先于 C 触发，则 B 执行，C 被禁止，反之则 C 执行，B 被禁止。这里 B 与 C 的选择结果与 A 无关。触发机制有 4 种不同类型，见图 3-7。

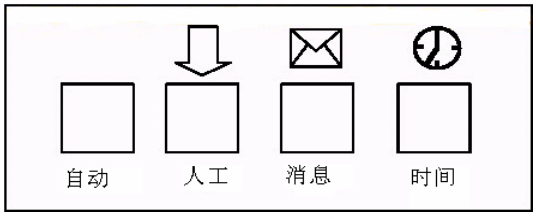


图 3-7 触发的类型

- 自动触发：任务就绪后就能被触发，用于自动型的任务。
- 人工触发：任务的执行是通过执行者从工作流任务列表中选择来触发。
- 消息触发：来自系统外部的消息触发任务的执行。
- 时间触发：由控制时间的定时器触发就绪的任务。

选择路由（显式或分支）见图 3-6（d 图），任务 A 具有两个输出库所 c2 与 c3，但这里与“与分支”不同，A 在这里表示“或分支”，A 只能根据其某个活动属性 x 的值来决定向哪个库所输出托肯。比如，若 x>0，c2 获得托肯，任务 B 被执行；若 x<0，则 c3 获得托肯，任务 C 被执行。

循环路由 见图 3-6 (d 图)所示，B 是被反复执行的任务，C 可以理解为一个起控制作用的任务，用来检验 B 的执行结果，来决定是把托肯转移到 c4 还是移回 c2，如果托肯被移到 c4，B 不再被执行，而是继续推进流程的执行；如果托肯被移回 c2，B 将反复的执行。

用工作流术语来描述：工作项是准备执行任务的组合；活动是一个工作项的实际执行；工作项被实际执行，就转化为活动。用 Petri 网的术语来描述：工作项对应了一个就绪的变迁；任务对应了一个或多个变迁；活动对应了一个变迁的实施。在对某个业务流程进行分析时，构造块（如 AND-split、AND-join、OR-split、OR-join 被用来 建模顺序、条件、并行、循环路由，工作项由变迁来构建，因果依赖使用库所和有向弧来构建。在 Petri 网与工作流过程映射中，一个库所对应一个条件，能用作某个任务的前转发条件和/或后依赖条件；一个 AND-split 对应一个或多个输出库所的变迁，一个 AND-join 对应着一个或多个输入库所的变迁。OR-split/OR-join 对应一个或多个输出/输入弧的库所。

3.3.1.3、工作流建模实例

图 3-8 介绍了一个实际使用的工作流过程实例。这是软件开发中常见的过程，一个项目被发起，初审通过后进入开发阶段，角色有程序员，数据库管理员，测试小组。程序员和数据库管理员的工作可以并行工作，软件的一个版本出来后，由测试小组进行测试，通过了则提交给用户测试，否则版本迭代开发阶段。作为用户在大多数情况下对软件产品都能提出自己的看法和意见，这就要求开发小组的成员根据用户的意思做进一步的修改，再经过不同层面的测试，最终到达用户手里是一个成熟的软件产品。

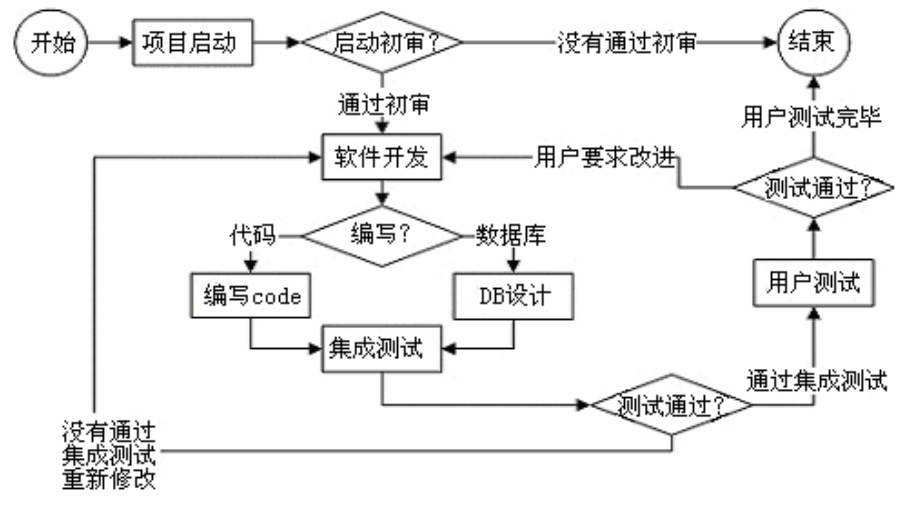


图 3-8 软件开发的基本流程

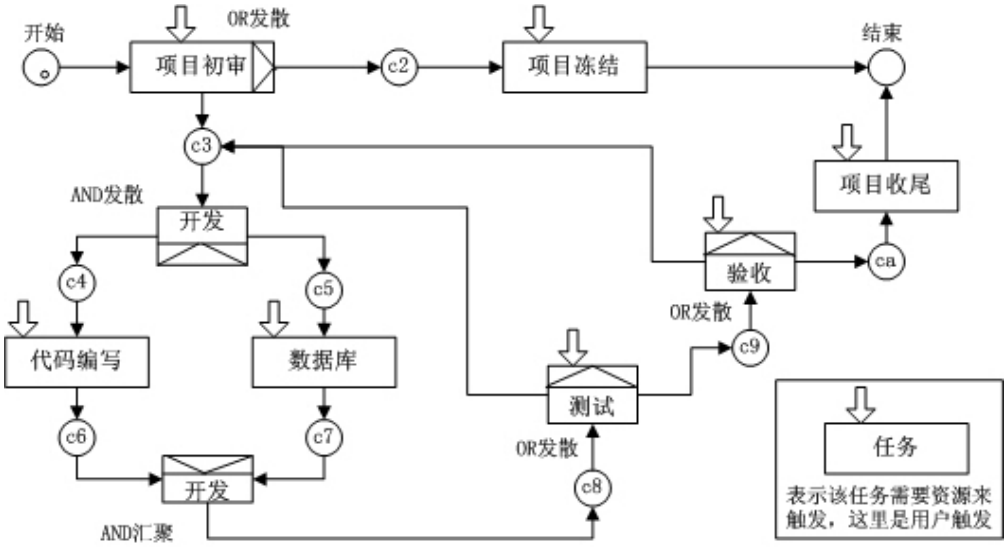


图 3-9 用 Petri 网来设计《软件开发》的过程

(1) 图 3-9 中包括作为开始条件的起始库所和作为结束的终止库所，起始库所中包括了一个托肯，起始库所中的托肯代表一个开发流程，由此激活整个过程；当有托肯输出到终止库所中时，整个流程进入结束的稳定状态。

(2) 首先活动《项目初审》被执行，这是“显式或分支”的人工触发活动，通过得到参与者设置的工作流相关数据 proj_status 的值来决定后续的活动。该活动有 2 个可能的输出：1、项目初审没有通过，在执行了《项目冻结》活动后，托肯经过 c2 被送到结束；2、项目初审通过，托肯放入 c3。

(3) 出于路由的考虑，添加了一个“与分支”和“与汇聚”活动，该活动没有实质的内容，在图中对应的就是 And 发散和 And 汇聚的活动，该路由活动使得《代码编写》和《数据库》人工触发活动能并发执行，两个活动处理完毕，分别通知工作流引擎，并在 And 汇聚活动中汇聚。之后，托肯被存放到 c8 中。

(4) 《测试》是一个 OR 类型的发散活动，也是“显式或分支”类型的活动，由工作流相关数据 develop_status 的值决定后续活动，并与 c3 和 c9 组成了一个 OR-split，还形成了一个小循环。当测试合格，托肯存放在 c9 中，否则就存放在 c3 中，进入循环开发阶段的活动；

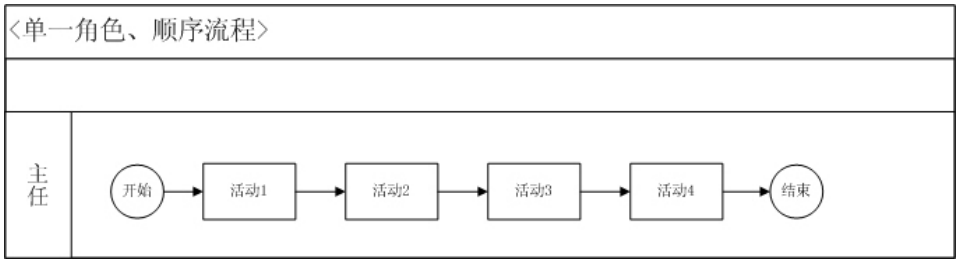
(5) 《验收》同《测试》一样，通过判断工作流相关数据 soft_status 的值来决定后续活动，与 c3 和 ca 也组成了一个 OR-split，并与 c3 形成一个大循环。当《验收》合格，执行一些收尾工作，托肯经过 ca，被存放到了结束，这样一个工作流的过程就执行完毕。这里没有使用到选择路由（隐式或分支）的部分。

一般说来，用工作流建模工具（作者使用的是 JAWE1.4）所描绘的活动（理解为任务的实例化）由 Petri 网所定义的一个库所和一个变迁以及它们之间的有向弧组成，某个活动所以就包含了条件（可以由工作流相关数据不同的值来表示），触发类型（人工、时间、消息），以及所代表的工作内容；活动之间的流程方向由库所与变迁之间的有向弧定义；另外，为了描述工作流模型的方便，引入了路由活动的概念，可以理解为在 Petri 网的模型中加入了一个库所和一个内容为空的变迁，该变迁不需要触发类型（属于自动执行的活动类型）。在本工作流过程中用来实现循环过程的入口。

3.3.2、基于关系数据库建模

基于关系数据库的工作流建模，没有太多的理论和道理讲述。我们还是通过几个实例来说明一下工作流模型如何在关系数据库中表示。

3.3.2.1、单一角色、顺序流程



一个角色有 4 个业务活动，（开始、结束也算活动类型之一，但这里为符合日常习惯，特意避开了，下同）

1. 设计一个过程
2. 设计 6 个活动，start 类型 1 个；end 类型 1 个；人工类型 4 个，都把活动参与者映射给“主任”
3. 不需要设计过程相关参数
4. 为每个活动设计活动变迁，这样，活动间的前趋，后续的关系就建立了。
5. 不需要为活动变迁设定变迁条件。

<table>
<tr>
<td> 
</td>
<td>过程 1 的建模界面
这里不需要设计相关数据</td>
</tr>
<tr>
<td> 
</td>
<td>Start 活动类型,只有后续,无前驱

‘活动 1’,前驱活动是 start;后续活动是’活动 2’

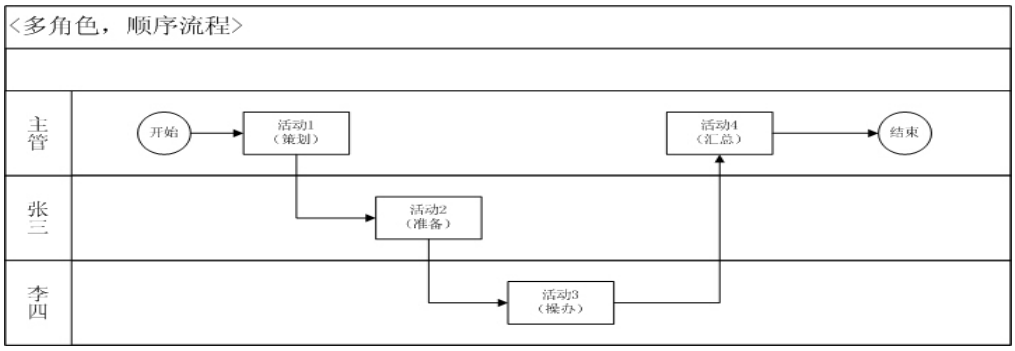
活动 3

活动 4

活动 5

End 活动,没有后续,只有前驱</td>
</tr>
</table>

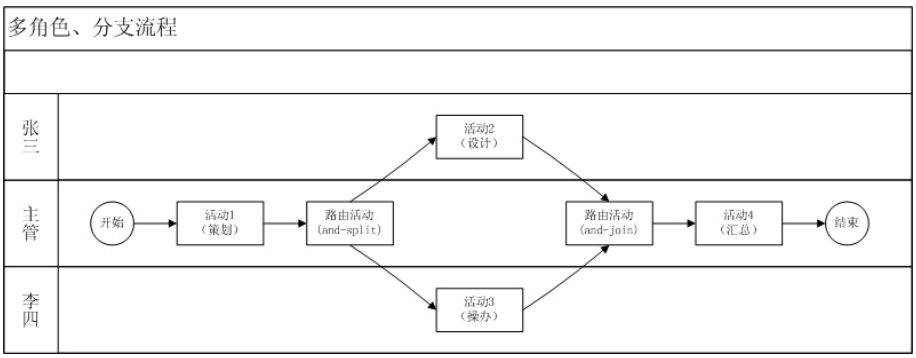
3.3.2.2、多角色、顺序流程



1、 设计一个过程
2、 设计 6 个活动，start 类型 1 个；end 类型 1 个；人工类型 4 个，活动参与者分别映射给不同的角色
3、 不需要设计过程相关参数
4、 为每个活动设计活动变迁。
5、 不需要为活动变迁设定变迁条件。

过程2,多角色参与的顺序流程
变迁路径和上面的图一样,只是参与者是多个角色

3.3.2.3、and 类型分支流程



1. 设计一个过程
2. 设计 8 个活动，start 类型 1 个；end 类型 1 个；and-split 活动 1 个；and-join 活动 1 个；人工类型 4 个，活动参与者分别映射给不同的角色

3. 不需要设计过程相关参数
4. 为每个活动设计活动变迁。
5. 不需要为活动变迁设定变迁条件。

在 and-join 路由活动中，应该会出现某个参与者等待的情况，活动 4 的执行必须等待 and-join 的完成；而 and-join 的完成，需要活动 2 和活动 3 都为“完成状态”

过程 3,多角色参与的 and 发散的流程
其中该流程还带有具体的业务逻辑地址,在个人工资列表中,可以直接启动该业务逻辑页面

Start 活动,无前驱

‘活动 1’

‘And_split 活动’有 2 个后续,一个前驱,在变迁表中由 2 条记录表示

‘活动 2’,分支的 1 个

‘活动 3’,分支的另外 1 条

‘and_join’活动,2 个前驱,1 个后续,用 2 条记录来表示.这里的活动都能通过下拉框来选择。在变迁的下面是变迁条件，在需要条件判断的路径中设置

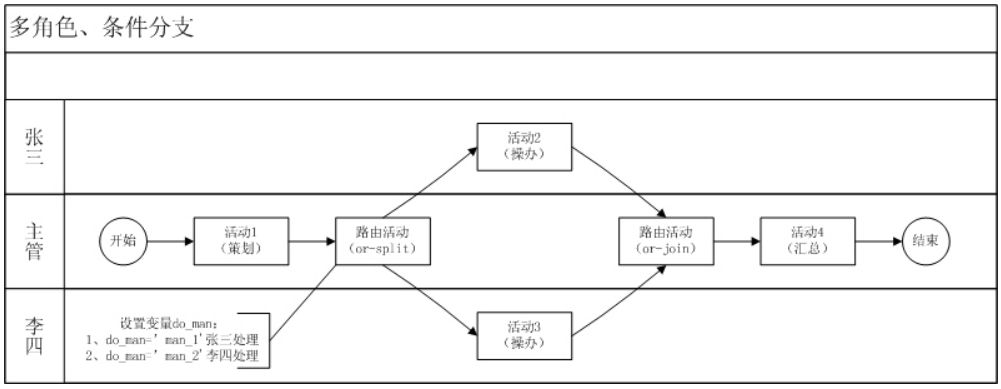
	当前序号	当前活动(只读)	前趋序号	前趋活动	后续序号	后续活动
▸	7	act4_汇总	6	and_join	8	结束

	当前序号	当前活动(只读)	前趋序号	前趋活动
▸	8	结束	7	act4_汇总

‘活动 4’

‘end 活动’

3.3.2.4、or 类型分支流程



1. 设计一个过程
2. 设计 8 个活动，start 类型 1 个；end 类型 1 个；and-split 活动 1 个；and-join 活动 1 个；人工类型 4 个，活动参与者分别映射给不同的角色
3. 设计过程相关参数 do_man；string 类型 ；操作符号为＝；表示必须相等，才能执行后续任务
4. 为每个活动设计活动变迁。
5. 不需要为活动变迁设定变迁条件。

在 or-split 路由活动中，弹出一个模态窗口，等待用户的交互，系统根据用户的输入来决定后续活动的路径。该路由活动状态由“not_start”->“running”；如果决定了后续活动；设置该路由活动状态为“完成”，同时把后续活动状态的“not_start”->“waiting”

过程 4,多角色参与的 or 类型路程
因为在 or_split 节点需要作条件判断，所以在相关活动变迁上添加变迁条件关系

当前序号	当前活动(只读)	后续序号	后续活动
1	开始	2	活动1_策划

当前序号	当前活动(只读)	前趋序号	前趋活动	后续序号	后续活动
2	活动1_策划	1	开始	3	活动_or_split

Start 类型活动

‘活动 1’

过程标识	过程中文名称	过程描述	过程创建者	创建时间	过程版本
proc_1	单一顺序流程	一个参与者，顺序过程，4个人工活动	流程设计者	2005-10-01	1
proc_2	多角色顺序流程	3个参与者，顺序流程，4个人工活动	程序员	2005-10-01	1
proc_3	发散流程1	3个参与者，and_split类型发散	流程设计者	2005-10-01	1
proc_4	发散流程2	3个参与者，or_split类型发散，需要过程相关数据	流程设计者	2005-10-04	1

相关数据标识	名称	描述	*数据类型
do_man	处理人	根据输入的值，来决定谁来处理。man_1：张3来处理；man...	字符串

Or_split 节点，这里需要由用户输入一个值作为后续变迁的判断。
相关数据在这里设定

活动变迁示意表

当前序号	当前活动(只读)	前趋序号	前趋活动	后续序号	后续活动
3	活动_or_split	2	活动1_策划	4	活动2_操办
3	活动_or_split	2	活动1_策划	5	活动3_协助操办

变迁 1，

变迁条件设置

条件标识	条件名称	条件类型(只读)	相关变量标识	类型(只读)	符号	值	条件间关系
man_1	张3处理	后转发规则	do_man	字符串	等于	man_1	

相关数据的设置

活动变迁示意表

当前序号	当前活动(只读)	前趋序号	前趋活动	后续序号	后续活动
3	活动_or_split	2	活动1_策划	4	活动2_操办
3	活动_or_split	2	活动1_策划	5	活动3_协助操办

变迁 2

变迁条件设置

条件标识	条件名称	条件类型(只读)	相关变量标识	类型(只读)	符号	值	条件间关系
man_2	李4处理	后转发规则	do_man	字符串	等于	man_2	

相关数据的设定

当前序号	当前活动(只读)	前趋序号	前趋活动	后续序号	后续活动
4	活动2_操办	3	活动_or_split	6	活动_or_join

活动 2

当前序号	当前活动(只读)	前趋序号	前趋活动	后续序号	后续活动
5	活动3_协助操办	3	活动_or_split	6	活动_or_join

活动 3

活动变迁示意表

当前序号	当前活动(只读)	前趋序号	前趋活动	后续序号	后续活动
6	活动_or_join	4	活动2_操办	7	活动_整理汇总
6	活动_or_join	5	活动3_协助操办	7	活动_整理汇总

Or_join 类型节点

变迁条件设置

条件标识	条件名称	条件类型(只读)	相关变量标识	类型(只读)	符号	值	条件间关系

活动_整理汇总

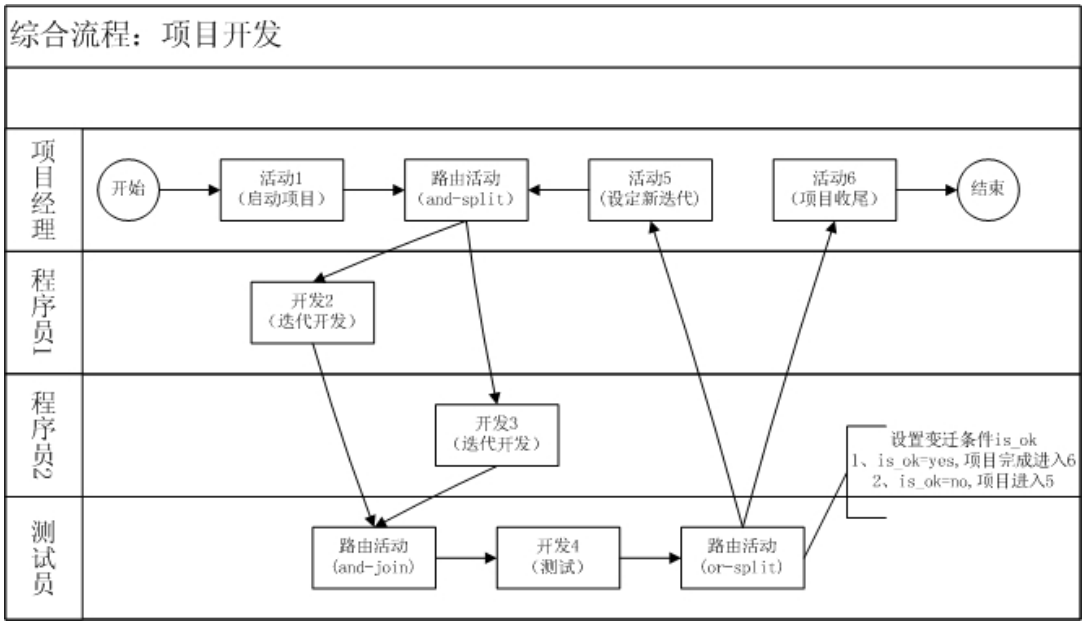
当前序号	当前活动(只读)	前趋序号	前趋活动	后续序号	后续活动
7	活动_整理汇总	6	活动_or_join	8	结束

End 类型活动

当前序号	当前活动(只读)	前趋序号	前趋活动
8	结束	7	活动_整理汇总

3.3.2.5、综合流程（项目开发）

根据前面用 petri 网分析过的项目开发流程，作了点简化，但还是能体现稍微复杂的流程，其中包括 and 类型发散，1 个 or 类型发散，它们又构成了循环。



根据一个开发流程简化而来，该过程包括了 1 个 and 发散和 1 个 or 发散。在实现上，or 发散可以模拟出循环的效果。

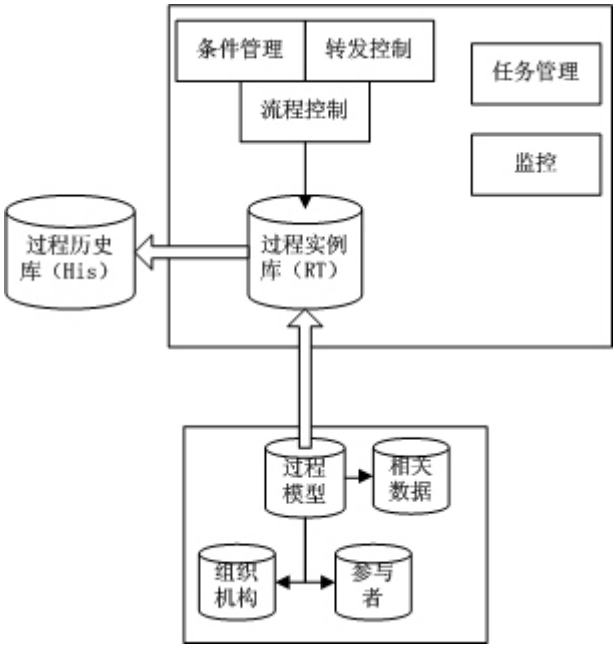
1. 一个过程
2. 4 个参与者
3. 1 个开始活动；1 个 end 活动；3 个路由活动；6 个人工活动
4. 1 个过程相关变量 is_ok ；string； 等于操作符
5. 1 个条件变迁

在 and-split 活动后，同时启动开发 2，开发 3 活动； 到了 and-join 活动，系统会在这里等待，直到两个活动都执行完毕。完毕后启动开发 4；在 or-split 出，系统弹出一个对话框，等待用户输入，根据用户的输入来决定后续的活动，是完成该开发过程还是继续迭代。

就是前面顺序、and 类型、or 类型节点的组合

4、工作流管理系统的设计

4.1、总体设计

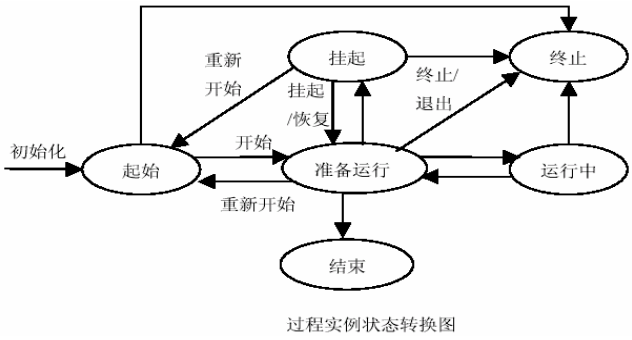


引擎结构图

4.2、过程状态转变图

工作流机应该提供控制过程实例和活动实例的状态转化。

- 初始（inactive）：一个过程实例已经生成，但该过程实例并没有满足开始执行的条件；
- 准备运行（active）：该过程实例已经开始执行，但是还不满足开始执行第一个活动并生成一个任务项的条件；
- 运行中（running）：一个或多个活动已经开始执行（已经生成一个工作项并分配给了合适的活动实例）
- 挂起(suspended)：该过程实例正在运行，但处于静止状态，除非有一个“重启”的命令使该过程实例回到准备运行状态，否则所有的活动都不会执行；
- 结束(completed)：该过程实例满足结束的条件，工作流管理系统将执行过程实例结束后的操作（如统计），并删除该过程实例。
- 终止（terminated）：该过程实例在正常结束前被迫终止，工作流管理系统将执行补救措施，并删除该过程实例。



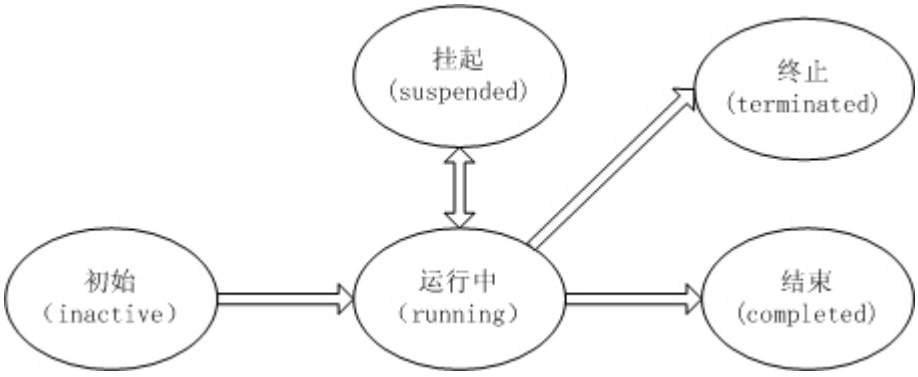
过程实例状态转换图

WFMC 提议图

在本系统设计中，过程的状态转化将按照下图来执行。

过程在设计中，不涉及到状态，就是说保存在模型库中的过程表，不需要设置状态字段。

- ✓ 一旦过程被实例化，该过程的状态被设置为“初始（inactive）”，并保存在数据库中。
- ✓ 如果用户此时进入自己的工作列表，启动了第一个活动实例，状态由“初始（inactive）-> 运行中（running）”
- ✓ 可以在 monitor 中把过程实例的状态设置从“运行中（running） ->挂起(suspended)”，那与该过程实例相关联的活动实例都要设置为“挂起(suspended)”，活动实例并被设置为不能使用，直到过程实例的状态被恢复成“运行中（running）”。
- ✓ 可以在 monitor 中把过程实例的状态强迫设置从“运行中（running） ->终止（terminated）”，那与该过程实例相关联的活动实例都要作不可恢复的删除；过程实例不删除（保存在数据库中），但设置“终止（terminated）”状态。
- ✓ 过程实例下所有的活动实例运行完毕，判断 is_end_act（）为 true，设置过程实例的状态为：“运行中（running）->结束(completed)”。过程实例不删除（保存在数据库中），但设置“结束(completed)”状态。

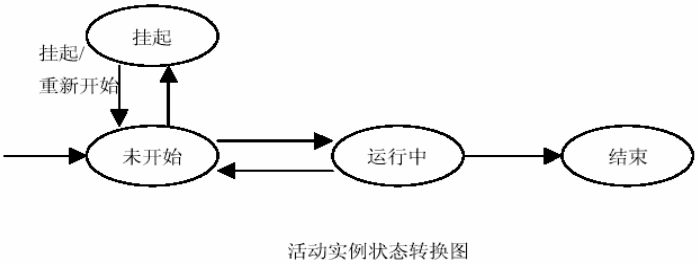


过程状态转化（本系统使用）

4.3、活动状态转化图

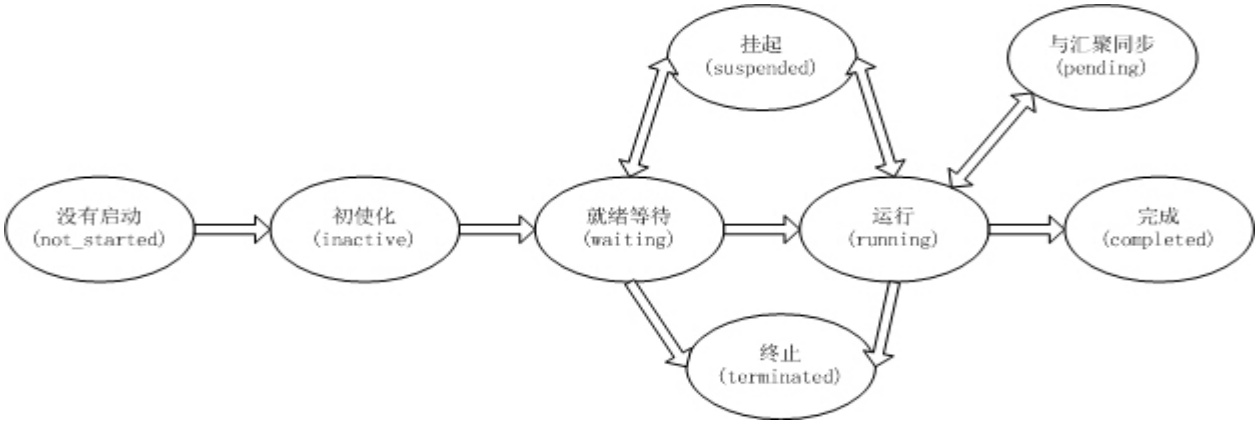
- Inactive（就绪）：活动实例已经生成，但没有执行。
- Active（运行）：活动正在执行。
- Suspended（挂起）：活动实例处于被阻塞状态，多半是资源不满足的条件下。
- Completed（完成）：该活动实例执行完毕，进入资源回收阶段。

- Inactive -> suspended：刚建立的活动实例因为资源得不到满足而被阻塞。人工活动中，因为参与者没有选择该任务，该任务就被阻塞；对于自动类型的活动，外部应用没有响应也会造成活动实例被阻塞。
- Suspended -> inactive：和上述情况相反，需要的资源得到了响应，工作流参与者对该任务进行了选择。
- Inactive -> active： 活动正在执行中。
- Active -> inactive： 活动在运行中出现异常，重新回到就绪状态
- active -> completed：活动顺利执行



WFMC 提议图提议图

活动状态的转化就作了较大的扩展，以期表达更多的状态。



活动状态变迁图（本系统使用）

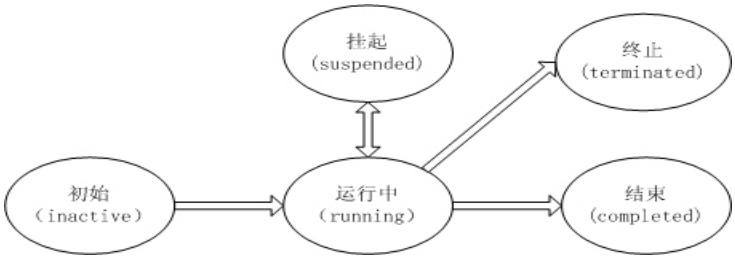
1、 在过程实例启动时，设置过程实例状态为“初始（inactive）”，与之相关的所有活动实例的状态都设置为“没有启动（not_started）”。
2、 找到过程实例第一个活动（start 类型），并把第一个活动设置为“完成（completed）”，找到下一个活动实例，设置为“就绪等待（waiting）”。
3、 用户登陆自己任务管理器，得到属于自己的“waiting”任务，选择任务处置，完成后点击接受，该活动实例状态从“waiting”直接越过“running”->“完成（completed）”
4、 可以 monitor 页面设置某个活动挂起，或终止，则该活动实例不再出现在用户的任务列表上。挂起为可恢复过程，而终止为不可逆过程。
5、 当前活动实例为“and-jion类型”，而标识位属于路径没有全部到达时，把当前活动实例设置为“pending”；当标志位为全部到达时，状态由“pending”直接越过“running”，到达“completed”

4.4、流程处理说明

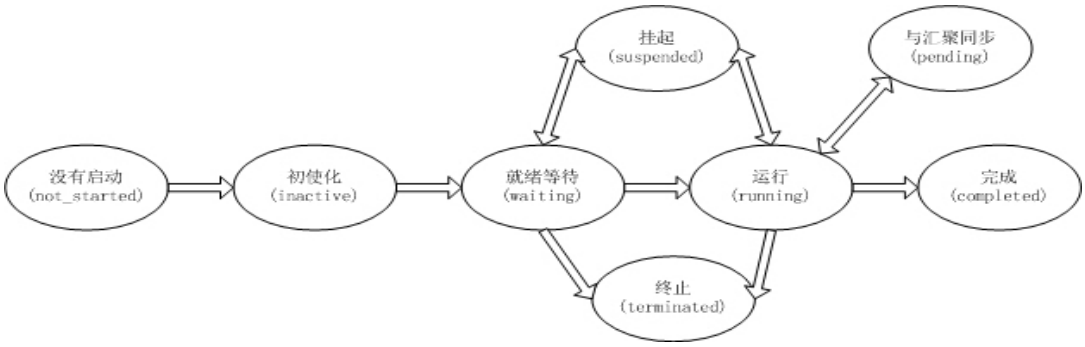
业务规则

- 业务规则分解为前依赖规则，后转发规则。
- 前依赖指某个活动的启动依赖于直接前趋活动的状态标志，包括：and_join 这个活动；
- 后依赖只某个活动结束后要启动哪个后续活动，包括：顺序、and_split、or_split。

在 start，end 的活动类型，是不需要前依赖、后转发的规则，直接到下一个活动，可以理解为它们后面有个不带条件的后转发规则



过程状态转变



活动状态转变

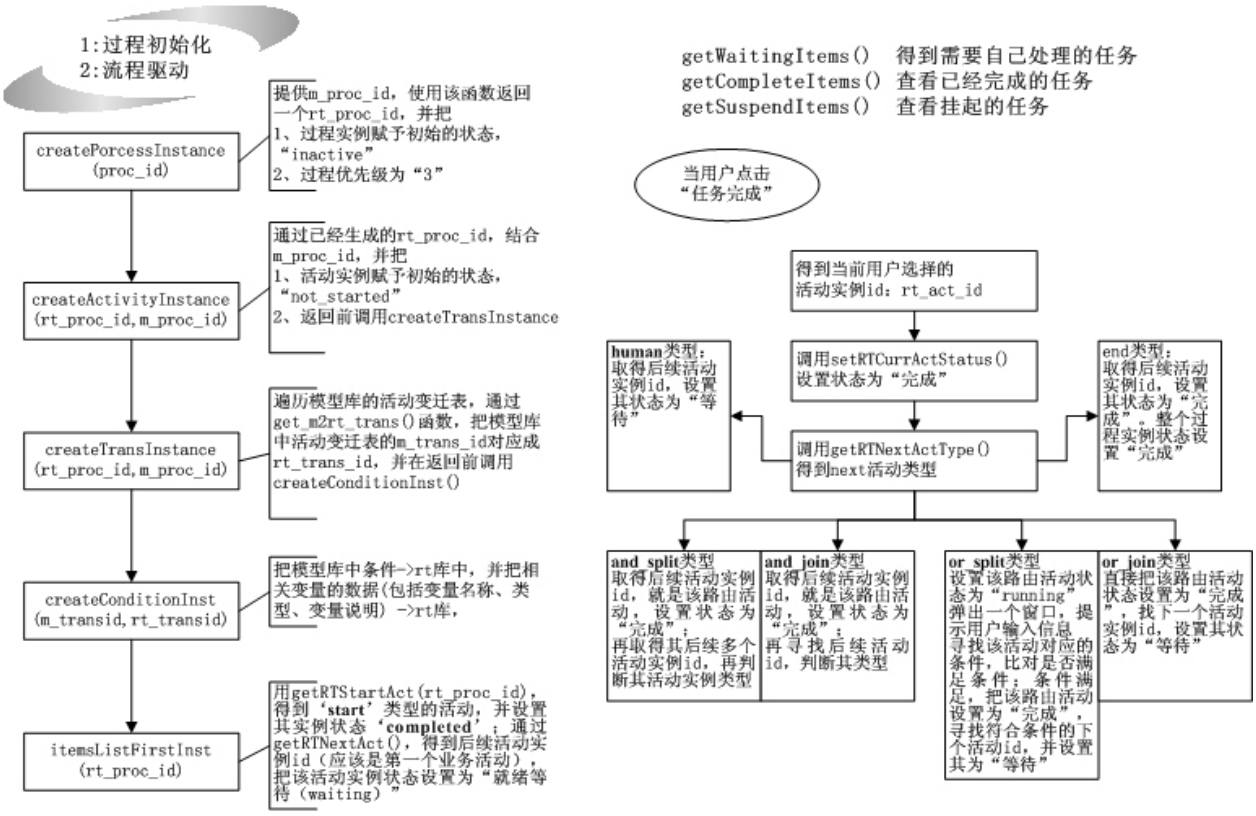
业务逻辑关系：

过程包括一组活动，也包括可以在该过程中使用的相关变量。活动之间的关系通过活动变迁表来表述。当活动变迁有条件的化，则需要变迁条件来说明，条件所使用的相关变量在前面已经定义

1、 当过程（m）实例化后，过程(m)->过程(rt)；把过程的状态设置为“inactive”；
   a、 把与该过程相关的活动实例都设置为“not_started”类型。
   b、 找该过程实例的开始活动（start）类型，把开始活动（start 类型）设置为“完成（completed）”；遍历依赖变迁（rt）表，得到后续活动实例 id（应该是第一个业务活动），把该活动实例状态设置为“就绪等待（waiting）”。
2、 用户点击“代办事宜”；
   a、 遍历活动（rt）表中某个角色、并且活动实例状态为“waiting”，显示在任务列表器；
   b、 选择任务（目前活动应该都是人工活动），完成后点击接受；根据当前活动实例 id，遍历变迁依赖（rt）表；
      - 如果有条件判断（rt），设置活动实例状态为“running”，根据 id 主键查询，找到条件变迁，提供条件变量，操作，比对操作值，从而控制流程的走向；如果条件满足；设置活动状态为“完成”；
      - 如果没有条件变迁，该活动实例状态从“waiting”直接越过“running”->“完成（completed）”；

   c、 根据当前活动实例 id，遍历依赖变迁（rt）得到后续活动实例 id，查活动（rt）表，得到活动类型；
      - 如果是“and_join”，则需要检查另外分支的活动状态是否为“完成”，否则在任务列表中显示该路由活动，状态为“pending”；
      - 如果是其他类型，直接把当前活动状态设置为“完成”；
   d、 在这里用户不能更改活动实例的状态，要改，只能到 monitor

4.5、关键流程算法



辅助函数

getRTStartAct(rt_proc_id)
返回该过程实例的start活动id

getRTEndAct(rt_proc_id)
返回该过程实例的end活动id

isRTStartAct
(rt_proc_id,rt_curr_act_id)
所给活动id是否为该过程实例的start活动，返回true|false

isRTEndAct
(rt_proc_id,rt_curr_act_id)
所给活动id是否为该过程实例的end活动，返回true|false

setRTCurrActStatus
(rt_curr_actid, setStatus)
设置当前活动状态的状态

setRTCurrActPriority
(rt_curr_actid, setPriority)
设置当前活动优先级

setRTCurrProcPriority
(rt_curr_procid, setPriority)
设置当前过程实例的优先级

getRTNextAct
(rt_proc_id,rt_curr_act_id)

getRTNextActType
(rt_proc_id,rt_curr_act_id)

getRTNextActStatus
(rt_proc_id,rt_curr_act_id)

getRTNextActNumber
(rt_proc_id,rt_curr_act_id)
返回后续活动的个数

getRTCurrActType(rt_curr_actid)
得到当前活动类型

getRTCurrTrans(rt_curr_actid)
得到当前活动对应的变迁id

getRTCurrTransVO(rt_curr_transid)
得到给定transid的条件表达式，string类型

compareRTTrans(VO,value,type)
分析比较，返回true，false

getRTPrevAct
(rt_proc_id,rt_curr_act_id)

getRTPrevActType
(rt_proc_id,rt_curr_act_id)

getRTPrevActStatus
(rt_proc_id,rt_curr_act_id)

getRTPrevActNumber
(rt_proc_id,rt_curr_act_id)
返回前续活动的个数

5、工作流管理系统与业务系统的结合

工作流过程包括一组活动以及活动之间的逻辑关系。活动对应着业务过程中的任务，逻辑关系决定了活动的执行顺序和活动之间的数据流动关系。工作流参与者负责人工活动（No 类型活动）的执行，在任务列表中选取要执行的任务，执行完毕通知引擎，该任务完成，并依据过程的预选设定，赋予相关的参数。

工作流管理系统本身和业务系统建立同一个平台上。工作流管理系统是业务过程逻辑的管理者，它帮助建立整个企业的业务过程，并且通过管理和监视业务过程的执行来完成业务系统的一部分功能。工作流中相应活动的功能由软件逻辑单元来实现，这些软件逻辑单元可以处理某项任务，比如填写表单页面、通知送达等，而这些软件逻辑单元由工作流管理系统根据过程规则进行触发。工作流参与者通过工作流客户端进入系统，在工作流执行服务的支持下，完成系统中业务过程的运转。

工作流管理系统与业务系统共同存在，工作流管理系统通过调用相关子系统中的软件组件来实现相应的功能，同时可以通过访问企业数据库、或参数传递来进行数据交换。通过工作流系统来管理其他子系统中的业务过程，增强业务系统对业务过程的处理能力。

在业务过程中，人工活动由人来控制执行，执行时需要调用相关的应用程序和 ERP 中的软件组件,而自动活动由计算机程序来执行，可以通过引擎直接调用，或通过工具代理来启动外部应用。这些外部应用程序是活动的功能设计的，并且在模型中建立活动与这些外部应用程序之间的映射关系，在工作流执行时由工作流引擎来调用它们以实现过程的运转。

5.1、与业务流程的通用结合方法

1．整合的方法

工作流的过程定义是描述过程的执行步骤和顺序，活动是过程处理的具体环节，而用户界面是针对人工活动（No 类型）提供了用户如何激活任务，以及引擎如何从用户界面中得到数据的人机接口。

Web 应用设计如下：

JSP 页面是用户界面，提供与用户交互的接口。

MIS 系统的业务逻辑另外形成一个处理层，不与工作流引擎发生关系，以便于保持业务的独立性。

通过流控制技术把用户请求的控制权转交给到其它 Web 页面。

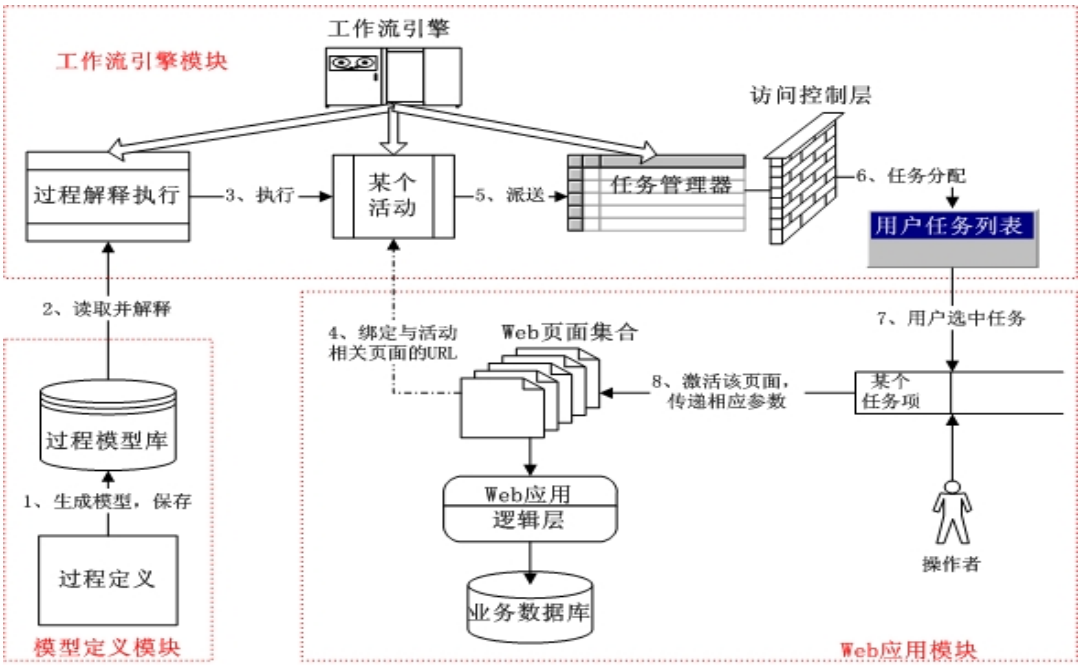


图 5-1 工作流与 web 业务系统的结合

工作流引擎只负责处理与流程运转相关事宜，处理过程的解释执行、流转规则，控制任务管理器。架构在工作流引擎之上的 Web 应用的具体业务处理另外编写，以保持工作流引擎的独立性和简洁性。

2．大致流程：

(1) 工作流管理人员设计工作流过程模型，并保存到模型库中。

(2) 该工作流过程模型被调用，被工作流引擎解释执行。

(3) 引擎根据过程模型中定义的规则，调用某个活动（该活动是人工活动，需要操作者来执行该活动）。

(4) 该活动是一个人工活动，比如是《员工出差申请表》。该页面已经制作完毕，上面有相应的栏目需要员工填写，并有提交的按键。该页面的逻辑处理自行处理，可以是不同的 jsp 页面，也可以是 struts 等框架设计，对于引擎来说，它只需要 url 地址和附加的参数。该页面的 URL 存储在活动表中的“业务地址”的字段中，附加的参数存储在活动表中的被该活动的“业务附加变量”。整个业务名称由“业务逻辑名称”来表示。需要人工完成该活动时，由工作流引擎把与活动绑定的页面地址分配给某个用户，用户直接点击该 URL（该 URL 由页面地址和附加在后面的参数组成），就可以出现该页面，用户就可以在上面作相应的处理，处理完毕，通知工作流引擎。

(5) 与该活动绑定的页面地址由工作流引擎发到任务管理器，在页面地址后附加需要的参数，工作流引擎希望通过 URL 来传递变量。

(6) 通过组织模式来确定该活动具体有哪个用户来执行，并经过访问控制模块的检查和过滤，最后发到特定人员的任务列表中。

(7) 该用户点击该活动的连接。

(8) 与该活动绑定的页面出现，从 URL 接受传递来的变量，用户在该页面进行操作，完成后，用户通知工作流引擎，该活动完成。

(9) 工作流引擎根据过程模型执行后续的活动。

在具体处理中，比如填写物品申购表，员工请假单，这些界面应该都是规定好的。过程执行到某个需要人工处理的环节，直接调用与该环节绑的页面，调用与之相关的数据和处理逻辑，相关的结果反馈给工作流引擎。

5.2、结合的实例

以过程 3 为例，在该界面出输入你的业务地址和必要的参数

相关数据设定 | 活动内容设定 | 过程图例显示

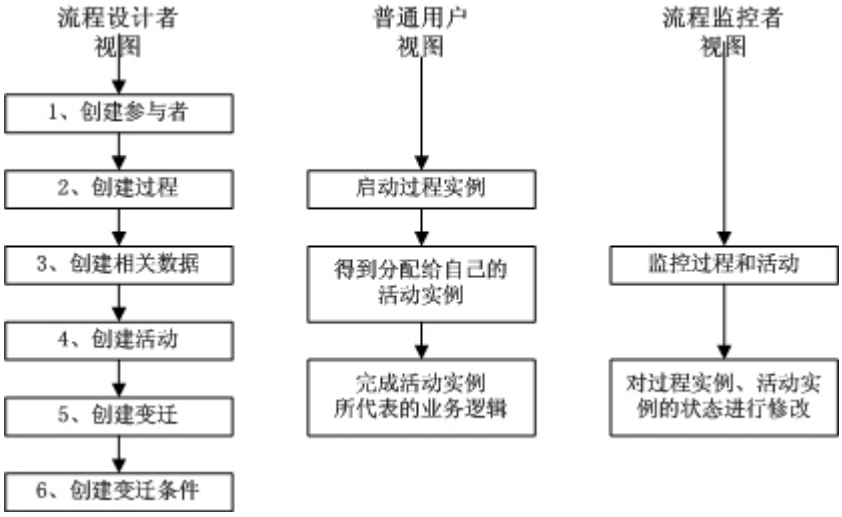
标识	*活动中文名称	*活动类型	*优先级	*参与者	业务名称	业务地址	业务变量
t	启动	启动活动	普通	院长			
	act1_策划	人工活动	普通	院长	策划，执行(emp_plan)	/wsoft/workflow_0.2/test_logic...	
_split	and_split	与发散	普通	院长	and split 路由		
	act2_设计	人工活动	普通	人资项目经理	设计，执行(zp_qudao)	/wsoft/workflow_0.2/test_logic...	
	act3_操办	人工活动	普通	程序员	操办，执行(addUsers)	/wsoft/workflow_0.2/test_logic...	
_join	and_join	与汇聚	普通	院长	and join 路由		
	act4_汇总	人工活动	普通	院长	汇总整理!(执行workday)	/wsoft/wqhr/workday/workday.js...	?edit_type=edit&yg_id=20041024...
	结束	结束活动	普通	院长			

然后登陆工作列表器，得到输入自己的任务（工作），点击“工作业务名称”，
系统会弹出新的窗口，等待用户处理，当用户处理完毕，完毕该窗口，在工作列表器上选中刚才已经完成的任务，点击“任务接受”的按键，整个客户端用户需要完成个工作就结束了。

6、 本工作流管理系统的使用说明

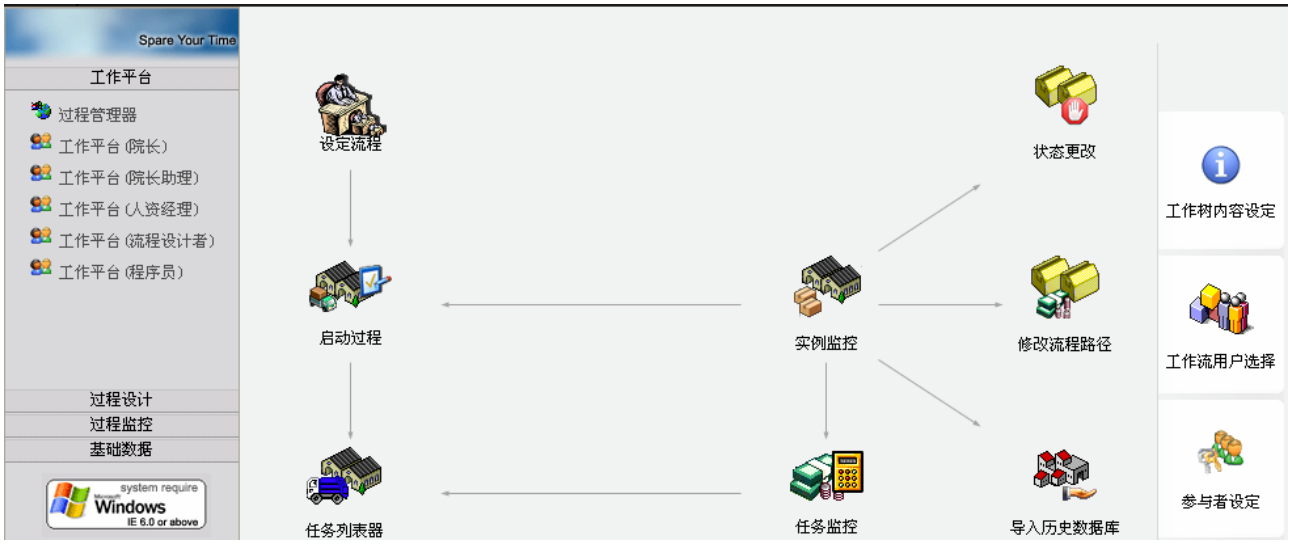
系统的总体视图，该工作流管理系统大致有 3 类用户使用：

- 流程设计者：用来设计具体的业务流程
- 普通用户　：参与流程，完成活动实例所代表的具体业务逻辑，在完成任务后，点击“接受活动”，表示该活动完成。
- 流程监控者：对过程、活动进行监控，在必要的时候可以修改活动的状态，改变流程的走向



本工作流管理系统的各个涉众视图

在具体情况下，有些步骤可以省略。比如有些流程是简单的顺序流程，不需要变迁条件、相关数据等，那在设计的时候可以忽略。普通用户登陆的时候，更多的情况下是直接察看自己的“代办事宜”，当有必要启动新的过程实例时，才去申请。



工作流管理系统的首页

左边的树把一部分参与者罗列出来是为了测试方便。

6.1、流程设计者

1、创建参与者

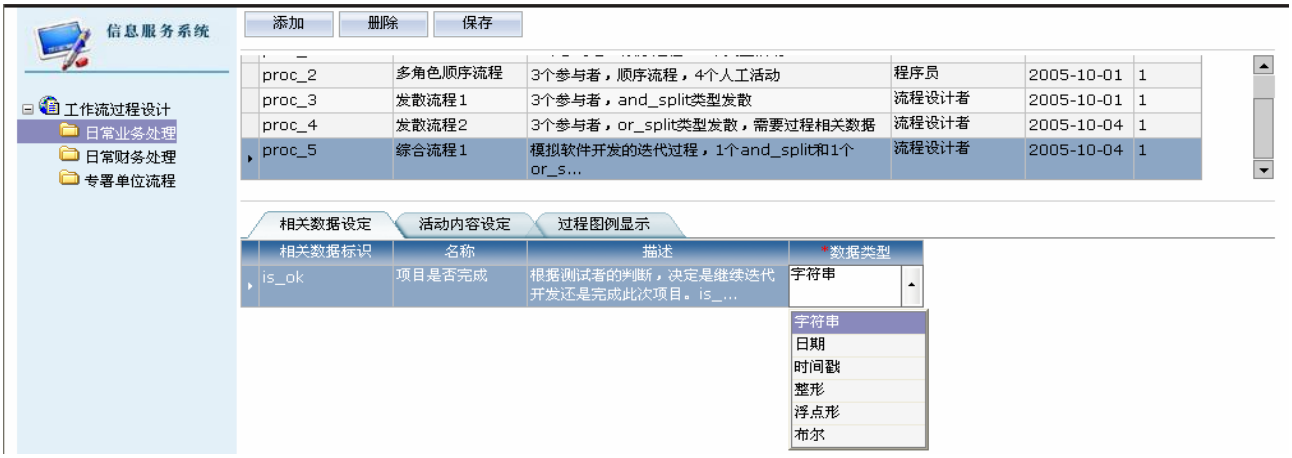
工作流参与者列表，参与工作流活动的实例分配。具体参与者由谁（用户）来承担，点击“用户列表”。

用户选择界面，把系统的用户赋予某个参与者。

2、 创建过程

首先创建过程，过程创建完毕，才能创建与之相关的相关数据，活动等

3、 创建相关数据（如果没有，就不需要创建，直接到创建活动的步骤）



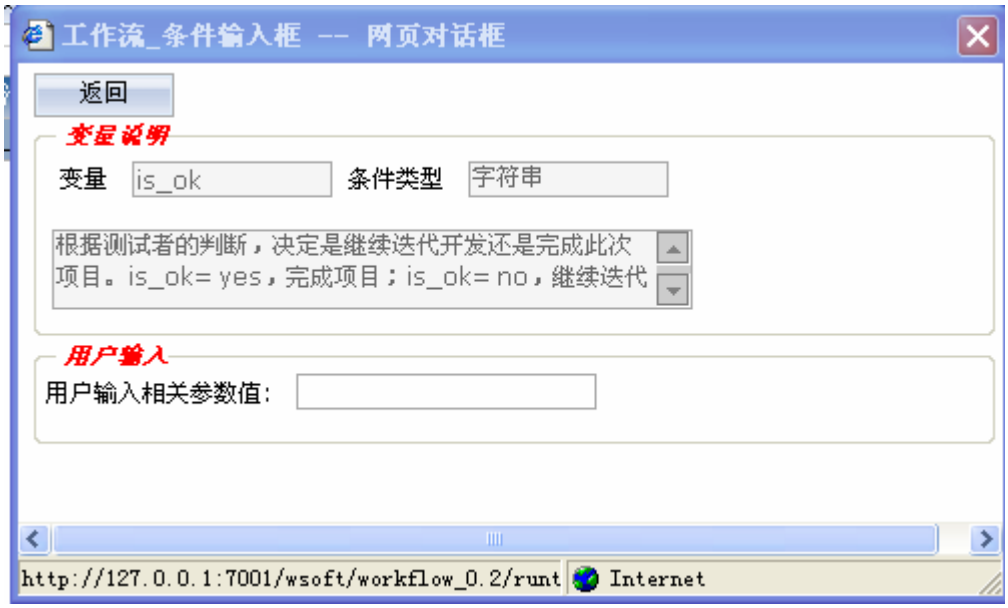
相关数据是在活动变迁条件中用到。

4、 创建活动

创建完活动，才能创建活动变迁和变迁条件。其中变迁条件的变量来自相关数据中的设定，

5、 创建活动变迁、以及变迁条件

该节点是 or_split，需要设置条件，系统会根据用户的输入，比对系统中的数据，来给出后续活动的路径

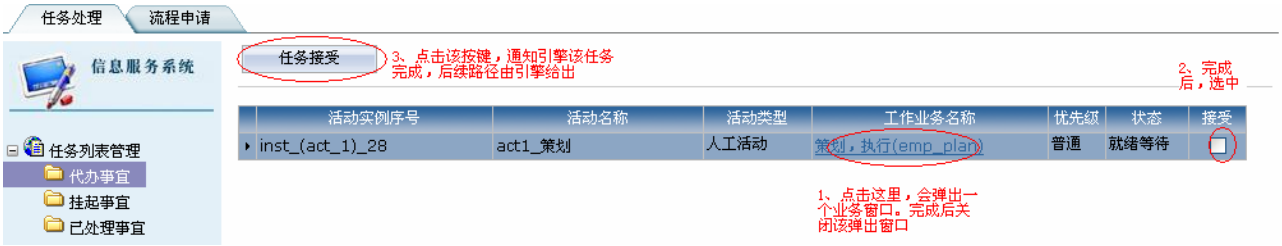


在点击“接受任务”的时候，系统会弹出一个提示框，用户可以根据提示，根据自己工作任务完成的情况，填入正确的值

6.2、普通参与者

用户界面由 2 部分组成，“申请流程”部分，和“任务处理”。

该部分提供了给参与者启动某个过程实例的途径。由谁，什么时候启动的流程，可以在监控台看到。选择你需要启动的流程，点击“申请启动流程”。在下面是该流程的活动列表，用户提示用户



得到分配给自己的活动实例

察看已经完成的任务

该界面是普通参与者用得最多的地方。参与者得到分配给自己的任务，进行处理。

6.3、流程监控者

过程唯一标识是唯一的，每次启动一次实例，序号＋1，下面是该过程的活动实例列表，从列表中可以清楚的看到流程的状态，以及目前所在的位置等。

通过在监控界面对状态的修改（过程能修改状态、优先级；活动能修改状态、优先级），从而改变流程的走向，比如跳过某个活动，执行某个分支，以及以后版本的回退等。

7、后记

本文主要设计理论来自《基于关系结构的轻量级工作流引擎》，同时还引用了老婆的几篇关于工作流的期刊论文以及毕业论文，一块表示感谢。

因为该工作流管理系统是架构在商业框架之上，纯粹的开放源代码没有任何意义，所以我在文章叙述的是思想和伪代码。建议在设计类似的工作流管理系统，最好和你的业务系统的框架一致，比如业务代码是用 struts 写的，那你们在设计基于此文的工作流管理系统时，也最好用 struts；用 spring 的业务系统，就用 spring 开发数据库；tapestry 也是如此。。。

关于本工作流管理系统的后续开发，我这里有个简略的开发计划，目前释放出来的是 0.2 版本，也希望对工作流感兴趣的、或者是正在作工作流的朋友，提些建议，我会考虑加在后续的版本中。

version_0.1	1、工作流的基本框架设计完毕 2、能跑基本的顺序流程（分不同的参与者），顺序流程 3、有简单的过程建模工具、用户客户端以及基本的数据维护模块 4、可用的用户、参与者映射工具
version_0.2	1、完善 xpdl -> 关系数据库映射的关系，让其能体现更多的 xpdl 内容 目前包括：过程、活动、参与者、活动变迁、变迁条件 2、完善工作流模式:支持顺序；and 发散、汇聚；or 发散、汇聚 3、完善 basedata 的输入与维护，基础数据通过维护界面来操作 4、参与者映射工具表字段修正一下，避免出错误信息
version_0.3	1、增加历史数据库的功能，把某些已经完成的过程记录 -> 历史库 2、增加删除过程的功能。某些过程确实不必要保存在历史库，也不需要保存在 rt 库，删除 3、优化过程实例化功能、删除过程功能、转移到历史库功能，采用更高效的方法。 4、完善条件判断中各种数据类型的支持。
version_0.4	1、增加回退功能，不是所有的步骤都能回退，前驱活动是人工的，并且是单一变迁 2、增加日志功能，把过程、活动状态的改变就记录在日志表中，这也是为回退提供依据
version_0.5	1、 根据数据库对过程模型的描述，能用图形方式显示处理

该文档我会继续随的基于关系数据库的工作流管理系统的软件升级而升级，新的文档我首先会发布在 http://www.wf800.com/ 中国工作流论坛。

如果朋友们有什么好的建议和意见，可以和我联系：

Email： timeson@gmail.com

还有几个 QQ 群，推荐给大家：

工作流联盟（14455633）：hongsoft 的群主，致力于工作流在国内的推广。

Shark 工作流（11577828）：这里有几个高手，一休，上帝等。

系统架构师（14558205）：本人的群主，考系分的或考架构的朋友，我们一起交流