|  |
| --- |
| doxt |
| **工作流管理系统详细设计** |
| Stage：2 |
|  |
| **wang.ligong** |
| **2009-2-19** |

|  |
| --- |
| 公司技术文档，请在内部交流使用。 |

# 目录

[1 目录 2](#_Toc263853189)

[1 系统运行环境 4](#_Toc263853190)

[2 开发技术 4](#_Toc263853191)

[3 系统组成和结构设计 4](#_Toc263853192)

[4 工作流建模工具 6](#_Toc263853193)

[4.1 建模工具选型 6](#_Toc263853194)

[4.2 流程文件的Schema 9](#_Toc263853195)

[4.2.1 活动(Activity) Schema 9](#_Toc263853196)

[4.2.2 转移(Transition)Schema 11](#_Toc263853197)

[4.2.3 角色定义 12](#_Toc263853198)

[4.2.4 业务控制数据 12](#_Toc263853199)

[4.2.5 流程(Process) Schema 13](#_Toc263853200)

[4.3 流程基本属性编辑 15](#_Toc263853201)

[4.4 流程文件的编译 15](#_Toc263853202)

[4.5 活动的定义 16](#_Toc263853203)

[5 工作流引擎服务 16](#_Toc263853204)

[5.1 工作流执行引擎 16](#_Toc263853205)

[5.1.1 工作流LINQ实体关系UML图 16](#_Toc263853206)

[5.1.2 工作流节点关系UML图 17](#_Toc263853207)

[5.1.3 重要算法 17](#_Toc263853208)

[5.2 工作流模式 18](#_Toc263853209)

[5.2.1 串行 19](#_Toc263853210)

[5.2.2 分支/合并 19](#_Toc263853211)

[5.2.3 串行/分支的图形元素示例 20](#_Toc263853212)

[5.2.4 多实例 21](#_Toc263853213)

[5.3 宿主程序开发 22](#_Toc263853214)

[5.4 运行时服务封装 22](#_Toc263853215)

[5.5 流程状态管理 22](#_Toc263853216)

[6 工作流权限管理 23](#_Toc263853217)

[6.1 基础数据维护 24](#_Toc263853218)

[6.1.1 角色数据维护 24](#_Toc263853219)

[6.1.2 用户拥有角色维护 25](#_Toc263853220)

[6.1.3 资源数据维护 25](#_Toc263853221)

[6.1.4 功能数据维护 26](#_Toc263853222)

[6.1.5 角色拥有功能数据维护 27](#_Toc263853223)

[6.2 权限访问控制策略 27](#_Toc263853224)

[6.3 数据库表结构 29](#_Toc263853225)

[6.3.1 角色表(WfRole) 29](#_Toc263853226)

[6.3.2 用户拥有角色表(WfUserRoles) 29](#_Toc263853227)

[6.3.3 资源类型表(WfResourceType) 29](#_Toc263853228)

[6.3.4 资源表(WfResource) 30](#_Toc263853229)

[6.3.5 功能表(WfFunction) 30](#_Toc263853230)

[6.3.6 角色拥有功能表(WfRoleFunctions) 31](#_Toc263853231)

[7 工作流管理工具 31](#_Toc263853232)

[7.1 流程模板管理 31](#_Toc263853233)

[7.2 流程模板状态 31](#_Toc263853234)

[7.2.1 状态转换图 31](#_Toc263853235)

[7.2.2 流程模板的状态枚举类型 32](#_Toc263853236)

[7.2.3 状态机图 32](#_Toc263853237)

[7.2.4 状态模式的类图 33](#_Toc263853238)

[7.2.5 流程模板状态管理的页面视图 34](#_Toc263853239)

[8 表单定制管理 35](#_Toc263853240)

[8.1 公司工作流程表单实现 35](#_Toc263853241)

[8.2 表单自定义工具选型 35](#_Toc263853242)

[9 工作流客户端 35](#_Toc263853243)

[9.1 个人任务列表 35](#_Toc263853244)

[9.1.1 8.1.1 界面显示字段 36](#_Toc263853245)

[9.1.2 8.1.2 数据结构 36](#_Toc263853246)

[10 消息通知系统 37](#_Toc263853247)

[10.1 站内短消息通知功能 37](#_Toc263853248)

[10.2 消息队列的配置和开发 37](#_Toc263853249)

[11 工作流发布服务接口 37](#_Toc263853250)

[11.1 WCF 服务用例 37](#_Toc263853251)

[12 工作流跟踪日志 38](#_Toc263853252)

[13 工作流异常处理 39](#_Toc263853253)

[13.1 异常处理的机制 39](#_Toc263853254)

[13.2 异常处理的分层 39](#_Toc263853255)

[13.2.1 数据层的异常处理 40](#_Toc263853256)

[13.2.2 业务层的异常处理 40](#_Toc263853257)

[13.2.3 表示层的异常处理 40](#_Toc263853258)

[13.3 异常查看 41](#_Toc263853259)

[14 工作流管理系统UI设计 41](#_Toc263853260)

[14.1 主界面 41](#_Toc263853261)

[14.2 流程基本数据 41](#_Toc263853262)

[14.3 流程建模图形化界面 41](#_Toc263853263)

[14.4 任务数据 41](#_Toc263853264)

[15 数据库设计 43](#_Toc263853265)

1. 系统运行环境

目前，开发和运行环境都是基于微软平台。对操作系统要求：Windows Xp, Windows2000, Windows2003, IE 6。应用程序服务器：IIS 6.0。数据库环境：SQL SERVER2005。

1. 开发技术

服务器端数据处理采用Microsoft Visual Studio.Net 2008 的 C# 语言，.Net Framwork3.5。客户端数据处理采用Asp.Net、Ajax 和 SliverLight 技术。

在程序编码过程中，要用到的开发语言可能有：C#、vascript、DHTML、XML。在数据库开发过程中，要用的编程语言是：SQL、T-SQL。

1. 系统组成和结构设计

工作流管理系统主要划分为以下几个模块： 工作流建模工具，工作流引擎服务，工作流管理工具，表单定制工具，工作流Web Service 接口，工作流客户端。

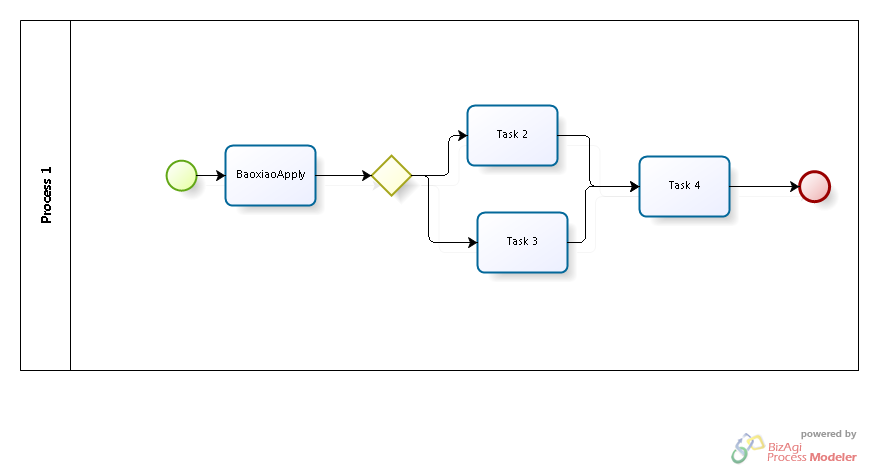


1. 工作流建模工具
   1. 建模工具选型

工作流建模工具的实现：ActiveX 技术, WF Designer 和 SliverLight 技术。这几个工具的选型主要考虑是，建模时候的图形处理能力，以及用户使用的友好性。

在目前实际的应用中，选用Microsoft SilverLight3.0作为开发工具进行开发。在实现基于Web的工作流图形化定义方面，由于SilverLight提供的rich web交互技术，使得图形化定义工作流变得更加方便。

一个图形化建模定义流程的UI如下图所示：



4.1.1 活动节点类图



4.1.2 线型类图



4.1.3 序列图

1> 工作流节点删除序列图



2> 工作流节点的连线删除序列图



3> 工作流节点拖动序列图



* 1. 流程文件的Schema
     1. 活动(Activity) Schema

<Activity name="Or分支节点" id="089f1d4f-01ff-422b-b0dd-28a280eb498e">

<ActivityType type="GatewayNode" gatewayDirection="OrSplit"/>

</Activity>

**其中: type 属性定义如下：**

**/// <summary>**

**/// 活动类型**

**/// </summary>**

**public enum NodeTypeEnum : int**

**{**

**/// <summary>**

**/// 开始节点**

**/// </summary>**

**StartNode = 1,**

**/// <summary>**

**/// 结束节点**

**/// </summary>**

**EndNode = 2,**

**/// <summary>**

**/// 活动节点**

**/// </summary>**

**TaskNode = 3,**

**/// <summary>**

**/// 网关节点**

**/// </summary>**

**GatewayNode = 4**

**}**

**其中:gatewayDirection 属性定义如下：**

**/// <summary>**

**/// 流程分支路由选择**

**/// </summary>**

**public enum GatewayDirectionEnum : int**

**{**

**/// <summary>**

**/// 未指定**

**/// </summary>**

**Unpecified = 0,**

**/// <summary>**

**/// 或分支**

**/// </summary>**

**OrSplit = 1,**

**/// <summary>**

**/// 异或分支**

**/// </summary>**

**XOrSplit = 2,**

**/// <summary>**

**/// 并行分支**

**/// </summary>**

**AndSplit = 3,**

**/// <summary>**

**/// 复杂分支**

**/// </summary>**

**ComplexSplit = 4,**

**/// <summary>**

**/// 或合并**

**/// </summary>**

**OrJoin = 5,**

**/// <summary>**

**/// 异或合并**

**/// </summary>**

**XOrJoin = 6,**

**/// <summary>**

**/// 并行合并**

**/// </summary>**

**AndJoin = 7,**

**/// <summary>**

**/// 复杂合并**

**/// </summary>**

**ComplexJoin = 8**

**}**

* + 1. 转移(Transition)Schema

<Transition name="T3" id="c22d96b6-534b-40b4-8258-f2b5faf8fe30" from="089f1d4f-01ff-422b-b0dd-28a280eb498e" to="8f6cb638-52b5-4eb2-b435-00dd24f55c97">

<Description/>

<Condition type="Expression">

<ConditionText>

<![CDATA[

askDays < 5 and leaveType != "shijia"

]]>

</ConditionText>

</Condition>

</Transition>

**其中，条件类型定义如下：**

**/// <summary>**

**/// 条件类型**

**/// </summary>**

**public enum ConditionTypeEnum**

**{**

**/// <summary>**

**/// 表达式**

**/// </summary>**

**Expression = 0,**

**/// <summary>**

**/// 规则**

**/// </summary>**

**Rule = 1**

**}**

* + 1. 角色定义

<Participants>

<Participant type="ROLE" id="3c917748-6752-4bc6-aea8-b85d40521888" name="Role\_HR\_Manager" outerId="102"/>

</Participants>

* + 1. 业务控制数据
    2. 流程(Process) Schema

**示例文件：**

<<?xml version="1.0" encoding="utf-8"?>

<Package>

<Participants>

<Participant type="ROLE" id="3c917748-6752-4bc6-aea8-b85d40521888" name="Role\_HR\_Manager" outerId="102"/>

</Participants>

<DataCollection>

<DataItems>

<DataItem type="Field" id="5p917748-6752-4bc6-aea8-b85d40521888" code="Fld\_Salary"/>

<DataItem type="Field" id="6x917748-6752-4bc6-aea8-b85d40521888" code="Fld\_AcceptedBy"/>

</DataItems>

<ParticipantPermissions>

<Participant id="3c917748-6752-4bc6-aea8-b85d40521888">

<DataItem id="5p917748-6752-4bc6-aea8-b85d40521888"/>

<DataItem id="6x917748-6752-4bc6-aea8-b85d40521888"/>

</Participant>

</ParticipantPermissions>

</DataCollection>

<WorkflowProcess>

<Process name="员工加班调休" id="7E960D99-6B05-4FEC-9198-F90471307DBE">

<Description>员工调休流程描述</Description>

<Activities>

<Activity name="开始" id="7E960D99-6B05-4FEC-9198-F90471307DBE">

<ActivityType type="StartNode"/>

</Activity>

<Activity name="结束" id="4d2b4913-69ee-41b2-b42d-3d3baf03ffdd">

<ActivityType type="EndNode"/>

</Activity>

<Activity name="申请调休" id="5e7dd6cb-395a-4192-9fc5-e9ce97c460e0">

<ActivityType type="TaskNode"/>

<Implement type="Auto"/>

<DataRequired>

<DataItem id="5p917748-6752-4bc6-aea8-b85d40521888"/>

<DataItem id="6x917748-6752-4bc6-aea8-b85d40521888"/>

</DataRequired>

</Activity>

<Activity name="Or分支节点" id="089f1d4f-01ff-422b-b0dd-28a280eb498e">

<ActivityType type="GatewayNode" gatewayDirection="OrSplit"/>

</Activity>

<Activity name="行政处理" id="8f6cb638-52b5-4eb2-b435-00dd24f55c97">

<ActivityType type="TaskNode"/>

<Impement type="Manual"/>

<Performers>

<Performer id="3c917748-6752-4bc6-aea8-b85d40521888"/>

<Performer id="8e54c07d-d424-4186-b86c-cec43019bed0"/>

</Performers>

</Activity>

<Activity name="中心总监审阅" id="89161725-b22c-4c2e-b6ea-eac5e0577d5a">

<ActivityType type="TaskNode"/>

</Activity>

<Activity name="Or 合并节点" id="c0ee3523-a4ea-4d46-be97-7c6f57a0166b">

<ActivityType type="GatewayNode" gatewayDirection="OrJoin"/>

</Activity>

<Activity name="总经理审批" id="c9b9416f-1581-41d1-92fc-4e6be14085d0">

<ActivityType type="TaskNode"/>

</Activity>

</Activities>

<Transitions>

<Transition name="T1" id="a5bd5c1c-f6b2-4091-9235-36c8f193f1a0" from="7E960D99-6B05-4FEC-9198-F90471307DBE" to="5e7dd6cb-395a-4192-9fc5-e9ce97c460e0" >

<Description/>

<Condition/>

</Transition>

<Transition name="T2" id="1ad6e341-3fb4-4860-80d2-011336f07a5c" from="5e7dd6cb-395a-4192-9fc5-e9ce97c460e0" to="089f1d4f-01ff-422b-b0dd-28a280eb498e">

<Description/>

<Condition/>

</Transition>

<Transition name="T3" id="c22d96b6-534b-40b4-8258-f2b5faf8fe30" from="089f1d4f-01ff-422b-b0dd-28a280eb498e" to="8f6cb638-52b5-4eb2-b435-00dd24f55c97">

<Description/>

<Condition type="Expression">

<ConditionText>

<![CDATA[

askDays < 5 and leaveType != "shijia"

]]>

</ConditionText>

</Condition>

</Transition>

<Transition name="T4" id="9d565899-4e63-4400-90bb-b345543175d3" from="089f1d4f-01ff-422b-b0dd-28a280eb498e" to="89161725-b22c-4c2e-b6ea-eac5e0577d5a">

<Description/>

<Condition type="Rule">

<ConditionText>

<![CDATA[

askDays >= 5 or leaveType == "shijia"

]]>

</ConditionText>

</Condition>

</Transition>

<Transition name="T5" id="21fe5ace-4700-49c9-b081-bd7cf8353e68" from="89161725-b22c-4c2e-b6ea-eac5e0577d5a" to="c0ee3523-a4ea-4d46-be97-7c6f57a0166b">

<Description/>

<Condition/>

</Transition>

<Transition name="T6" id="dcf0ce07-4dcc-404f-85af-ba3235cd05f9" from="8f6cb638-52b5-4eb2-b435-00dd24f55c97" to="c0ee3523-a4ea-4d46-be97-7c6f57a0166b">

<Description/>

<Condition/>

</Transition>

<Transition name="T7" id="00200106-7510-11d9-86f2-000476a22f44" from="c0ee3523-a4ea-4d46-be97-7c6f57a0166b" to="c9b9416f-1581-41d1-92fc-4e6be14085d0">

<Description/>

<Condition/>

</Transition>

<Transition name="T8" id="a1dd27af-b362-4450-a325-c9c338bfa64d" from="c9b9416f-1581-41d1-92fc-4e6be14085d0" to="4d2b4913-69ee-41b2-b42d-3d3baf03ffdd">

<Description/>

<Condition/>

</Transition>

</Transitions>

</Process>

</WorkflowProcess>

</Package>

* 1. 流程基本属性编辑

对流程的编号，名称和描述等基本属性进行编辑。流程的基本信息也需要保存到数据库中去。数据库的对应表是：WorkflowDefinition。 流程定义内容要保存到流程模板数据表中的XmlContent 字段中。

流程的定义，包括流程的唯一标识（key）、中文标识（Name）,简要描述（Description）、 版本信息(Version)、所属的公司名称（CompanyName）。

流程属性也包含应用程序的Aspx页面(web应用)，数据库表中的**AppPage**字段用来存放页面文件的名称，一般是\*\*\*.aspx的格式。

流程包括若干个活动（Activity），若干个连接（Transition）, 及多个自定义变量(Variable)。

* 1. 流程文件的编译

流程生成的文件格式是XAML, 文件的扩展名是XOML。 目前WF引擎可以对基本功能的XML文件直接装载，但是对大部分XOML文件需要编译处理。具体是：当生成XOML文件之后，使用wfc.exe 对xoml  和 xoml.cs 文件编译，生成流程的dll 文件，然后再被工作流引擎装载。

* 1. 活动的定义

活动的基本属性包括：活动名称、活动描述。

1. 工作流引擎服务
   1. 工作流执行引擎
      1. 工作流LINQ实体关系UML图



* + 1. 工作流节点关系UML图



* + 1. 重要算法

1. 获取当前活动的下一步活动列表



* 1. 工作流模式

工作流引擎中已经提供了串行，分支等基本模式，目前需要做的工作是对这些模式进行抽象，为后续的图形化定义界面上提供代码支持。

* + 1. 串行

串行模式是指工作流流程中的任务是依照次序顺序执行，没有分支选择，循环等路由功能。



* + 1. 分支/合并

分支模式是指对工作流流程中的任务执行条件进行限定，在执行过程中，根据条件判断执行路径。



#### 5.2.2.1分支情况

AndSplit : 并行分支，所有连线都可以执行后续节点；

OrSplit：或分支，满足条件的连线，可行执行后续节点

XOrSplit: 异或分支，有第一个满足条件的连线，即可执行后续节点。

ComplexSplit：复杂情况的分支，未明确给定。

#### 5.2.2.2合并情况：

AndJoin：并行合并，所有连线上的条件必须都满足，才可以执行该Gateway节点。

OrJoin：或合并，有条件定义的连线，如果条件都满足，才可以执行该Gateway节点。

XOrJoin：异或合并，只要有第一个连线的条件满足，不用再做其他连线的条件判断，直接执行该Gateway节点。

ComplexJoin：复杂情况的合并，未明确给定。

* + 1. 串行/分支的图形元素示例

|  |  |  |  |
| --- | --- | --- | --- |
| **gatewaySplitJoinType** | **gatewayDirection** | **bPMN Graph** | **description** |
| Split(分支) | AndSplit  (并行分支) |  | Transition的属性包括:   1. Condition 2. parallelOption: (necessary, optional)   解析所有必需类型连线上的条件是否都满足，如果满足，则执行AndSplit节点的所有后续节点。 |
| OrSplit  (多选分支) |  | 解析满足条件的分支连线，如果条件为真，则执行这些连线上的后续节点。 |
| XOrSplit  (互斥分支) |  | Transition的属性包括:   1. Condition 2. priority: [1,..n]   根据优先级，解析出第一个满足条件的连线，然后执行该连线的后续节点，其它分支上的节点不再执行。 |
| Join(合并) | AndJoin  (并行合并) |  | Transition的属性包括:   1. Condition 2. parallelOption: (necessary, optional)   当所有必需连线上的条件都满足时候，才能执行AndJoin节点的后续节点。 |
| OrJoin  (选择合并) |  | 根据连线上的条件定义，解析条件，如果满足，则OrJoin节点之后的节点可以被执行；此时，其它前驱分支节点可以继续执行，但不再实例化XOrJoin节点。 |
| XOrJoin  (互斥合并) |  | Transition的属性包括:   1. Condition 2. priority: [1,..n]   根据优先级解析第一个满足条件的连线，如果有，XOrJoin节点的后续节点可以被执行，其它前驱分支节点不能再执行。 |

定义在Transtion上的XML节点实例：

<Transition name="T3" id="c22d96b6-534b-40b4-8258-f2b5faf8fe30" from="089f1d4f-01ff-422b-b0dd-28a280eb498e" to="8f6cb638-52b5-4eb2-b435-00dd24f55c97">

<Description/>

**<GroupBehaviour priority="1" parallelOption="Necessary"/>**

<Condition type="Expression">

<ConditionText>

<![CDATA[

askDays < 5 and leaveType != "shijia"

]]>

</ConditionText>

</Condition>

</Transition>

/// <summary>

/// 连线并行时的类型：必需、可选

/// </summary>

public enum ParallelOptionEnum

{

/// <summary>

/// 可选

/// </summary>

Optional = 0,

/// <summary>

/// 必需

/// </summary>

Necessary = 1

}

* + 1. 多实例

节点执行有多个实例后，才可以继续向下执行。

* 1. 宿主程序开发

工作流引擎服务的宿主程序可以是控制台应用程序、Windows服务和IIS 服务执行。作为应用程序服务器，IIS 多应用服务的支持上，有更好的扩展性。

在对IIS 6.0 和 IIS 7.0 的激活模式比较上：IIS 6.0 支持对Http协议上的激活模型，IIS 8\7.0 支持Http、TCP、Named Pipe 和 MSMQ 的激活模型。

从运行的扩展性上选择IIS，IIS 7.0 作为应用程序服务器的功能也更为强大。目前，IIS 7.0 的安装仅限于Windows Server 2008。

对宿主程序的开发，需要实现的功能有：

1> 配置文件的编写；

配置文件包含工作流运行时服务信息，协议类型等。

2> 读取配置文件；

通过宿主程序读取配置文件信息，并作相应处理。

* 1. 运行时服务封装

运行时服务的封装主要实现的功能：

1> 统一的数据访问接口；

2> 统一的服务加载功能，目前先实现跟踪服务，持久化服务的加载

3> 统一的事件绑定服务。

具体实现类有：WorkflowInstanceWrapper和 WorkflowRuntimeManager。对应文件：WorkflowInstanceWrapper.cs 和 WorkflowRuntimeManager.cs。



* 1. 流程状态管理

流程从开始运行到结束运行的状态变迁。工作流流程的状态，随着流程运行的不同阶段和工作流运行时服务提供的功能，有很多状态的变迁。

1. 工作流权限管理

从权限管理模型上看，目前实际应用中比较常见的的模型还是基于角色的权限管理模型，其基本思想是根据企业中的职能岗位划分出不同角色，将系统资源的访问权限附加到角色上，并未用户分配相应的角色，从而实现对用户访问资源的控制[5]。

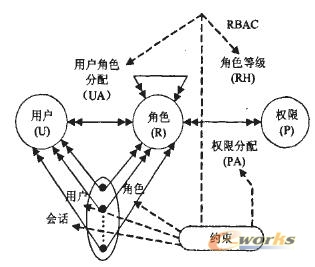


图6.1 基于角色的访问控制模型

基于角色的访问控制方法是目前公认的解决大型企业的统一资源访问控制的有效方法。其显著的两大特征是：

1> 减小授权管理的复杂性，降低管理开销；

2> 灵活地支持企业的安全策略，并对企业的变化有很大的伸缩性。

工作流权限模型在需求分析上也主要考虑RBAC模型的框架实现，对其包含的一些功能描述如下：

从权限管理的对象来看，通常包括功能权限设置、类权限设置、规则权限设置和属性权限设置。其中：

* **功能权限设置**：用于对系统功能进行控制；
* **类权限设置：**用于控制用户对业务数据的访问；
* **规则权限设置：**用于建立用户与类及其属性的约束关系，使得用户在访问系统数据时受到所设规则的限制；
* **属性权限设置：**用于控制对实体类的具体属性的访问。

从权限管理系统所提供的功能来看，主要包括授权和访问控制两不主要模块。它们具体实现的功能是：

* **授权模块：**实现权限的授予，授权模块主要用于对角色、用户和用户组进行管理，包括基本信息的维护，角色、用户和用户组之间关系的维护，以及分别为角色、用户和用户组授予各种类型的权限。
* **访问控制模块：**实现对权限的验证，访问控制模块实现对登录用户的身份认证，并在用户进行系统执行操作时，验证用户的操作权限。
  1. 基础数据维护
     1. 角色数据维护

角色数据主要维护角色名称、角色是否可用和角色描述的信息。其实现的基本是：

* 列表角色：将角色数据列表显示；
* 添加角色：添加新的角色记录到数据库中；
* 编辑角色：编辑角色的属性，并保存；
* 删除角色：从数据库中删除角色记录。
* 查询角色：按角色名称查询。



* + 1. 用户拥有角色维护

用户拥有角色是指当前用户被分配了那些角色，一个角色可以具有多个角色。要实现的功能有：

* 显示角色：显示该用户拥有的角色；
* 分配角色：给该用户分配角色；
* 移除角色：移除该用户拥有的角色。



* + 1. 资源数据维护

根据资源的关系描述，资源的主要功能操作有：

* 创建资源：创建新的资源实例
* 编辑资源：编辑资源实例
* 删除资源：删除资源实例



* + 1. 功能数据维护

功能是指某一个应用（或资源）下的功能，在功能表中有ResourceID来关联资源表中的资源数据。功能数据的维护主要完成功能表记录的增删改查。

* 新建功能：在某一个具体应用下新建功能记录项；
* 编辑功能：编辑某一个应用下的功能记录项；
* 删除资源：从某一个应用下删除功能记录项。



* + 1. 角色拥有功能数据维护

通常，拥有功能操作集合的对象成为角色。在系统维护模块中，在创建完角色之后，要把相应的功能权限分配给该角色。

* 分配功能：在该角色上，分配该角色关联应用下的某项功能；
* 删除功能：把已经分配到角色中的功能删除。



* 1. 权限访问控制策略

6.2.1 权限访问控制主流程

权限访问控制是实现对某个用户能否访问某个资源的权限验证。其主要的流程图如下



6.2.2 访问控制表生成(ACL)

* 1. 数据库表结构
     1. 角色表(WfRole)



约束要求：

* Unique Index (companyid, rolecode） 组合唯一
* rolecode, rolename 都不能为空
  + 1. 用户拥有角色表(WfUserRoles)



* + 1. 资源类型表(WfResourceType)



* + 1. 资源表(WfResource)



约束要求：

* Unique Index (companyid, resourcecode) 组合唯一
* resourcecode, resourcename, lastupdateddatetime 都不能为空
  + 1. 功能表(WfFunction)



约束要求：

* Unique Index (resource, functioncode) 组合唯一
* functioncode, functioname, lastupdatedatetime 都不能为空
  + 1. 角色拥有功能表(WfRoleFunctions)



约束要求：

* Unique Index (companyid, rolecode） 组合唯一
* rolecode, rolename 都不能为空

1. 工作流管理工具
   1. 流程模板管理

模板是对工作流流程模板数据进行维护。主要是对流程数据的增删改查操作。

* 1. 流程模板状态
     1. 状态转换图

通常在对流程模板管理的时候，需要设置更改流程模板的状态，流程模板处于运行和关闭状态时候，模板数据是不能进行删除操作的。

流程的模板状态有起草、测试、运行和关闭状态，工作流在启动时要检查流程模板是否处于可以运行状态，然后再启动。状体示意图如下：



流程模板状态转换时候，根据状态转移图进行判断，只有符合转移条件的操作，只能成功，否则给出提示信息。

* + 1. 流程模板的状态枚举类型

Public enum WorkflowDefinitionStatus

{

Drafting,

Testing,

Running,

Closed

}

流程模板的初始状态是：Drafting

* + 1. 状态机图

模板实例状态的转换如图所示。



* + 1. 状态模式的类图

在程序设计时候，增加流程模板的状态类，用来统一管理流程模板的状态。

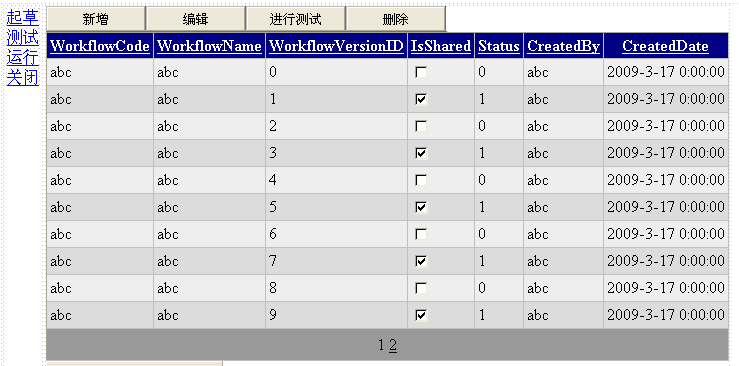


* + 1. 流程模板状态管理的页面视图

在流程模板的页面视图上，有起草、测试、运行和关闭状态的分类导航，在每一个类别下可以看到流程模板的实例数据，在数据列表的顶部，有可以对流程模板实例数据进行操作的按钮。

如：当流程处于起草状态，模板数据可以进行新增，编辑，删除的基本功能，还可以选择让它进入测试状态。

其它状态的流程模板实例数据的操作类似，对不同状态下的数据，提供不同的操作。在对页面视图的控制上，要有MVC的设计思想，尽可能地做到视图和控制的分离。



1. 表单定制管理
   1. 公司工作流程表单实现

公司目前的流程主要有：

1> 请假流程;

2> 出差申请流程;

3> 费用报销流程。

在第一阶段的开发上，需要先实现这3个流程的开发，在请假流程中，加入对请假天数的判断处理，条件分为3天之内，和超过3天的两种情况。

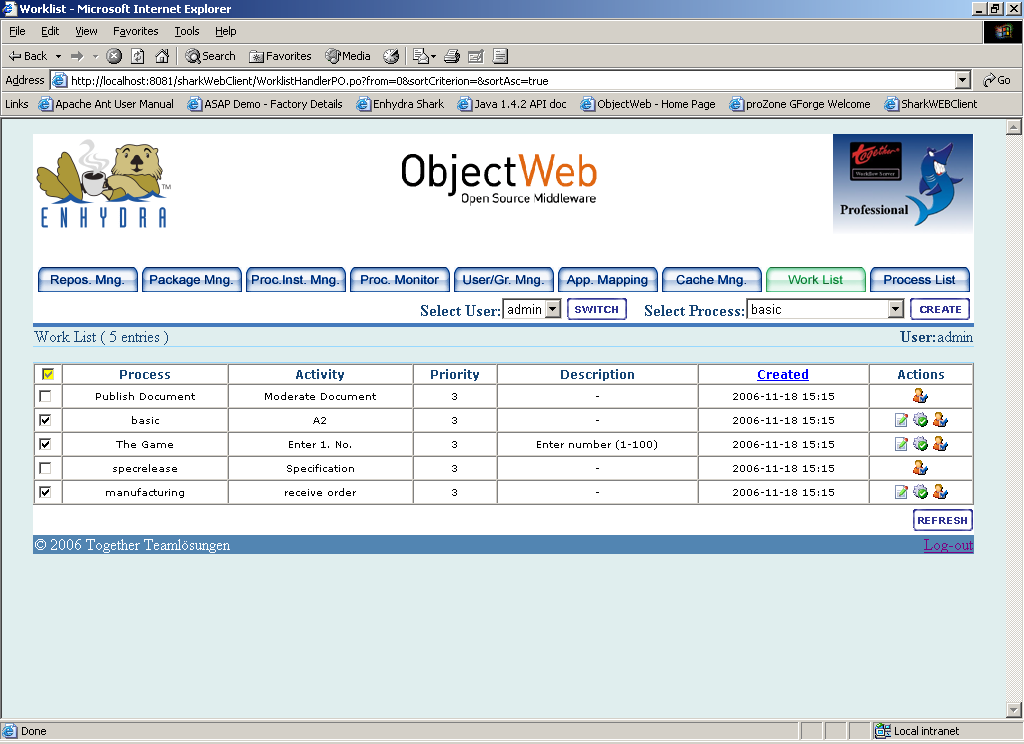
在未开发出表单自定义工具之前，可以先在Asp.Net 环境下，设计该页面，便于对流程流转过程的测试。

* 1. 表单自定义工具选型

自定义表单的实现方式的可能性有：自己开发页面定制工具，或者集成InfoPath 页面。目前，需要对这两种方式进行比较。

1. 工作流客户端
   1. 个人任务列表

展现个人任务列表，包括代办、已办记录。在流程实例的列表界面上，提供链接到流程办理的界面。任务列表的界面如下：



* + 1. 界面显示字段

流程名称、活动名称、优先级、描述、创建日期、操作

* + 1. 数据结构

1> 活动优先级

public enum ActivityPriority

{

Lowest = 0,

Low = 1,

Normal = 2,

High = 3,

Highest = 4

}

2> 流程实例，活动实例表的关联

列表中的流程数据由流程实例表获取，活动数据由活动实例表获取，创建日期是指活动实例的创建日期。

3> 操作

操作是指对活动实例的操作，在该栏先预留”办理”操作的位置。

1. 消息通知系统
   1. 站内短消息通知功能

工作流有新的任务产生时，提供短消息通知功能。消息体有链接指向办理页面。在数据库中，要创建未发送的消息表，和已经发送的历史消息表。

* 1. 消息队列的配置和开发

消息队列是为了保证系统的可靠性和伸缩性。作为将来的企业级工作流平台，需要消息队列来支撑构建，以满足应用对消息的处理需求。

在第一阶段，先了解消息队列的开发技术，开发出测试用例。

1. 工作流发布服务接口
   1. WCF 服务用例

工作流发布的服务按照 WCF 标准设计和开发。目前要开发的服务接口：

**1> 流程的启动；**

外部用户调用流程启动的Web Service 来启动流程，调用者需要提供流程的基本信息，和跟调用者的身份信息，只有验证为合法的用户，流程的公司代码匹配，才可以启动流程。

方法体：Int StartWorkflow(string companyCode, string processCode, string userInfo)

**2> 流程的暂停；**

外部用户调用流程暂停的Web Service 来暂停流程，提供和验证信息同上。

方法体： Int SuspendWorkflow(string companyCode, string processCode, string userInfo)

**3> 流程的恢复；**

外部用户调用流程暂停的Web Service 来恢复流程，提供和验证信息同上。

方法体： Int SuspendWorkflow(string companyCode, string processCode, string userName)

**4> 流程的结束；**

外部用户调用流程暂停的Web Service 来恢复流程，提供和验证信息同上。

方法体： Int SuspendWorkflow(string companyCode, string processCode, string userInfo)

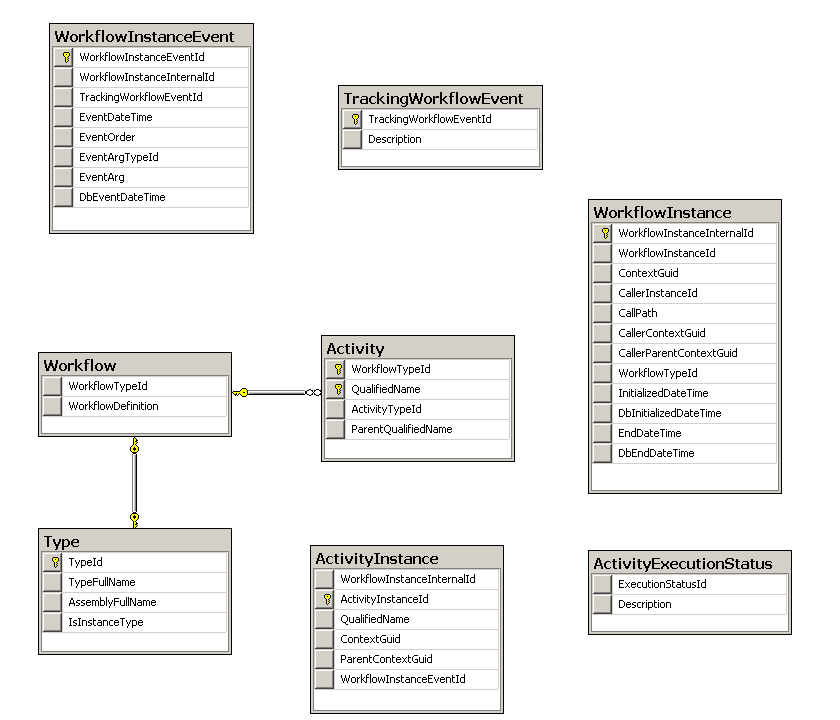
1. 工作流跟踪日志

工作流实例执行过程，所有状态发生的变化、事件和异常信息等，都要被记录在数据库中，在WF 框架中，提供了WorkflowTrackingService 的功能，在现阶段的处理上，把该服务添加到工作流运行时服务，跟踪日志的记录也是被保存到跟踪数据库中。

跟踪服务提供的功能主要如下：

1. 工作流流程实例日志
2. 工作流活动实例日志
3. 工作流流程实例的事件日志

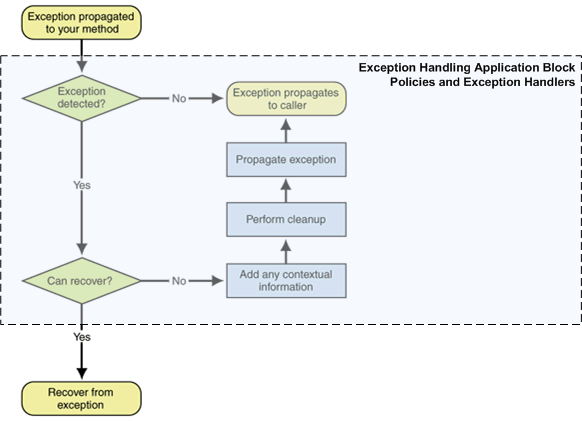
系统的表结构如下：



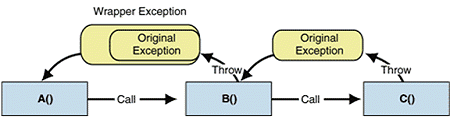
1. 工作流异常处理
   1. 异常处理的机制

异常处理的基本步骤

只有在满足

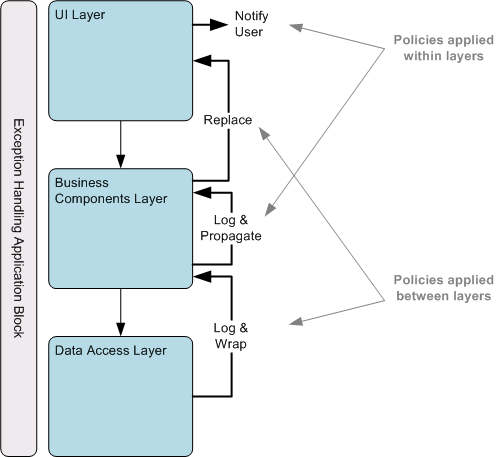


异常的抛出



* 1. 异常处理的分层

工作流管理系统的系统结构是按照数据访问层、业务层和表现层来划分的。在每一个处理层上都会涉及到异常抛出的问题。异常如何从数据层传递到表现层，最终让用户理解详细的异常信息，这需要一套完善的统一处理方式。



* + 1. 数据层的异常处理

数据访问层在发生异常时，通常处理的要求有：

* 记录日志：将发生的异常信息，记录到系统的日志中去。
* 异常包装：异常包装之后，发送到业务层。
  + 1. 业务层的异常处理

业务层在发生异常时，通常处理的要求有：

* 记录日志：将发生的异常信息，记录到系统的日志中去。
* 异常推出：将异常信息通过替换后，发送到表现层。替换的要求是，使用用户能够理解的语言去描述异常的详细信息。
  + 1. 表示层的异常处理

表示层的异常处理是，将业务层传出的异常通知给用户，具体通知的方式：

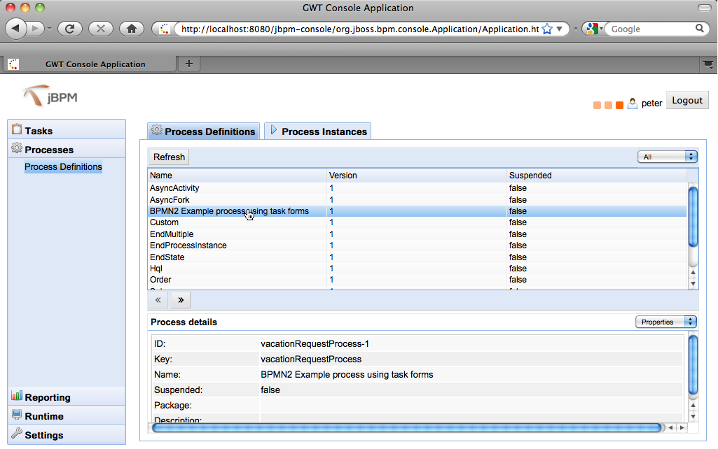
1> 直接的界面显示；

2> 如果有其他载体的订阅，则根据订阅方式，发送。

* 1. 异常查看

根据日志表中记录的异常信息，用UI显示出来。

1. 工作流管理系统UI设计
   1. 主界面



* 1. 流程基本数据

维护流程定义的基本数据。

1. 列表
2. 流程基本属性编辑
3. 链接到图形化定义页面
   1. 流程建模图形化界面
   2. 任务数据

展现用户需要办理的任务数据。

1. 根据任务表中角色信息判断用户的任务列表；
2. 根据任务的完成状态判断是否是待办任务、或已经完成任务；
3. 根据任务的完成人的信息，给出可查询的该用户任务列表。
4. 任务列表UI



应用类别为枚举类型:

1. 用户账户管理
2. 平台产品
3. 任务查询页面:



1. 数据库设计