

开发框架-审批 workflow 功能介绍说明书

V3.0

1. 引言

1.1. 背景

一讲到 workflow，很多人第一反应就是这个东西很深奥，有时候又觉得离我们较为遥远，确实完善的 workflow 涉及到很多方面，而正是由于需要兼顾很多方面，一般通用的 workflow 都难做到尽善尽美。微软也提供了几个版本的 workflow 框架支持，也有一些厂家是基于这个框架基础上开发的 workflow 应用。

以前由于项目的需要，参与过一些 workflow 的项目开发，其中有些是基于我简易 workflow 的原理上进行拓展的，包括一个广州市各区县使用的行业审批业务平台，由于基于自己的流程处理，界面设计、流程流转等方面可以很好符合客户需求，定制的弹性较好，缺点是不够通用，也需要编写表单部分代码。

后面由于业务的需要，workflow 方面的业务逐渐显得迫切，公司是想采用一个较为通用 workflow 框架来组织目前的业务，因此找了广州一家做 workflow 的公司，购买了他们的产品，虽然号称完全通过后台配置，零代码实现 workflow 业务表单的处理，但是由于客户对表单的设计要求比较多，有时候需要结合一些外部的数据接口，流程处理方面也有着进一步的需要，这样就可能就打破了他们原来的格局，导致无论在表单设计、流程配置等方面，都需要购买他们工程师的现场服务，来进一步完善整个项目的内容，导致整个项目进展缓慢，遭遇水土不服的处境。

因此感觉，一个 workflow 模块，号称再强大，如果不能很好结合项目应用，即使零代码的功能配置，也可能使你处于尴尬的境况之中，因为通过配置，可能在代码里面平常很容易实现的表单功能，要通过零代码配置，花费的时间更多更难掌握，因为零代码是有代价的，需要您很好利用他们的 API，他们的业务对象，有时候还需要很曲折的摸索参数，而这一切可能就是非常致命的弱点。

1.2. 编写目的

本文档介绍基于框架的基础上开发的 workflow 审批模块，主要介绍《workflow 模块》中的一些设计理念和功能特性，使得开发过程我们更加了解整个 workflow 的运行机制和相关功能。

1.3. 参考资料

序号	名称	版本/日期	来源
1	《Winform 开发框架-架构设计说明书.doc》		内部
2	《Winform 开发框架-数据库设计说明书.doc》		内部
3	《WCF 开发框架-系统部署使用说明书.doc》		内部
4	《开发框架-系统数据库还原操作说明.doc》		内部
5	《混合式开发框架主体框架项目说明.doc》		内部
6	《权限管理系统功能介绍白皮书.doc》		内部

1.4. 术语及缩写

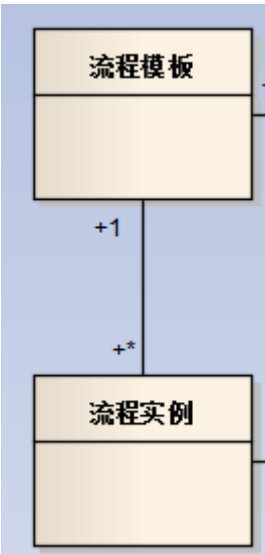
- 1 在本文件中出现的“开发框架”一词，除非特别说明，适用于《Winform开发框架》、《WCF开发框架》和《混合式开发框架》。
- 2 在本文安装.NET框架中，除非特别说明，均指.NET 4.0框架。
- 3 如无特别说明， workflow 模块是指本框架的 workflow 模块内容。

2. 工作流介绍

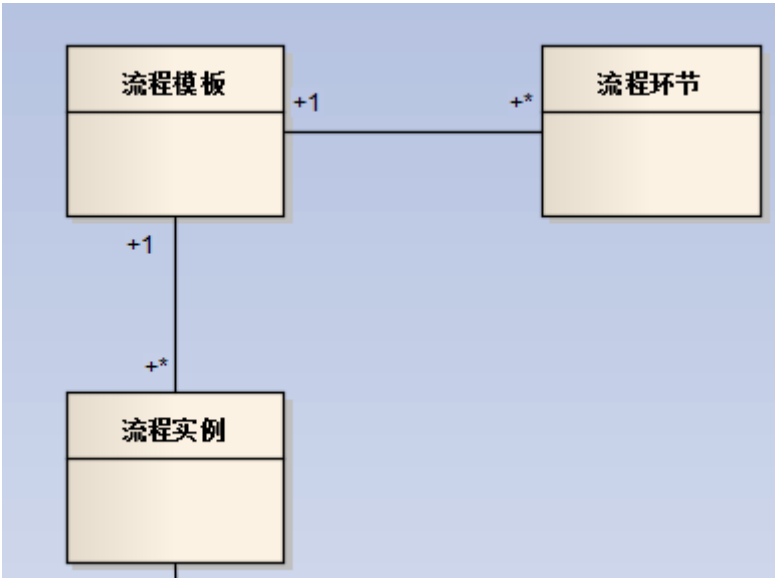
2.1. 简易工作流的设计模型

在没有第三方 workflow 模块的情况下，简易 workflow 就是利用数据库和业务对象之间的协作关系，构建的一个半模块化的流程引擎，它通过整合到项目代码中进行更好的融合以便实现 workflow 的相关功能。

首先我们知道，我们在 Office 里面创建任何文档，都有一个模板的概念，这样我们方便利用一些现成的数据和布局， workflow 也一样，有一个流程模板的概念，如下所示。

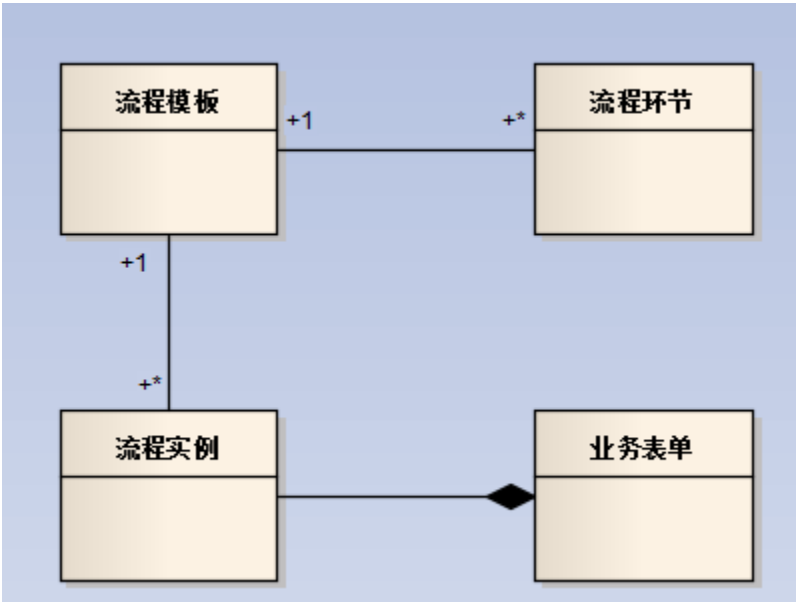


然后每个流程模板，本身会预定义了一系列的处理流程，以便在流程实例里面进行不同的处理，因此流程模板还包含了多个流程步骤对象，他们的关系构成如下。



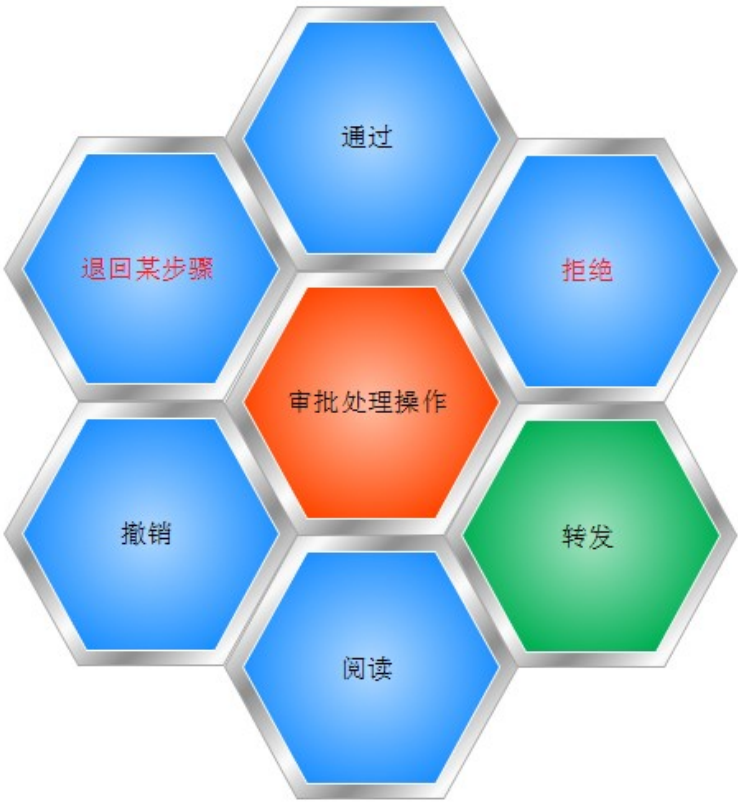
每个流程实例，除了他们自己的流程数据和字段信息外，它本身还有一个表单设计的问题，如费用审批，可能包含填写的费用清单数据等，所以流程实例还应该包含了流程的业务表单对象。

这样他们构成了一个完整的流程业务对象关系，如下所示。



2.2. 流程审批操作

对于一个流程处理操作，我们知道一般有审批通过、拒绝、退回到某步骤、转发到内部阅读、阅读，以及包括起草者能撤销表单呢等操作，当然如果还有一些具体的业务，可能还会有一些流程的处理才操作，不过基本上也可以归结为上面几种，只是他们每步处理的数据内容不同而已。因此审批的操作步骤分类如下所示。



这些操作我们都可以通过一些界面操作的封装实现，因为他们基本上都是通用的，我们传入一些流程 ID 等相关标识后，就能交给这些标准的操作界面完成了。

如审批界面如下所示，里面包含了通过、拒绝，跳回到某步骤，增加步骤等功能集合。

审批操作

常用意见

同意。

保存常用意见

处理意见

同意。

审批意见

☒ 批准申请

☐ 退回拟稿人重新处理

☐ 退回上一步处理

☒ 下一步流程审批

☐ 增加一步审批

下一处理人

发送

关闭

上面的界面是审批过程中，对于某一个流程处理人员实现的操作，而有时候，我们可能需要针对多个人进行某个步骤的处理，如会签处理，会签处理处理界面效果如下所示。

会签操作

常用意见

同意上报。

保存常用意见

处理意见

同意上报。

审批意见

☒ 批准申请☐ 退回拟稿人重新处理☐ 退回上一步处理

会签人员

待选人员

☒ 所在部门☐ 选择人员

包英浩
张焯
秀景琪
王珣
李陈佳
罗雪
张静春
梁爽
吕萍

>>>>
>>
<<
<<<<

已选人员

管理员
胡欣

注：会签处理，需要会签人员同时通过才算最终通过，否则任何审批人员可做退回处理。

发送

关闭

会签操作

常用意见

同意办理

保存常用意见

处理意见

同意办理

审批意见

☒ 批准申请☐ 不通过申请☐ 退回拟稿人重新处理☐ 退回上一步处理

会签人员意见

用户名	处理方式	意见
管理员	通过	同意。
广州管理员	未处理	

注：会签处理，需要会签人员同时通过才算最终通过，否则任何审批人员可做退回处理。

发送

关闭

以及传递给内部人员进行分阅操作，那么就应该选定多个人员进行处理，大概的处理界面效果如下所示。

第 5 页 共 44 页

批示分阅操作

常用意见

请内部传阅处理。

保存常用意见

处理意见

请内部传阅处理。

审批意见

☒ 批准申请

☐ 退回拟稿人重新处理

☐ 退回上一步处理

分阅人员

待选人员

☐ 所在部门

☒ 选择人员

>>>>

>>

<<

<<<<

已选人员

管理员
胡欣
吕萍
梁爽
张静春
罗雪
李陈佳
王珣
秀景琪
张烨
包英浩

发送

关闭

阅办处理

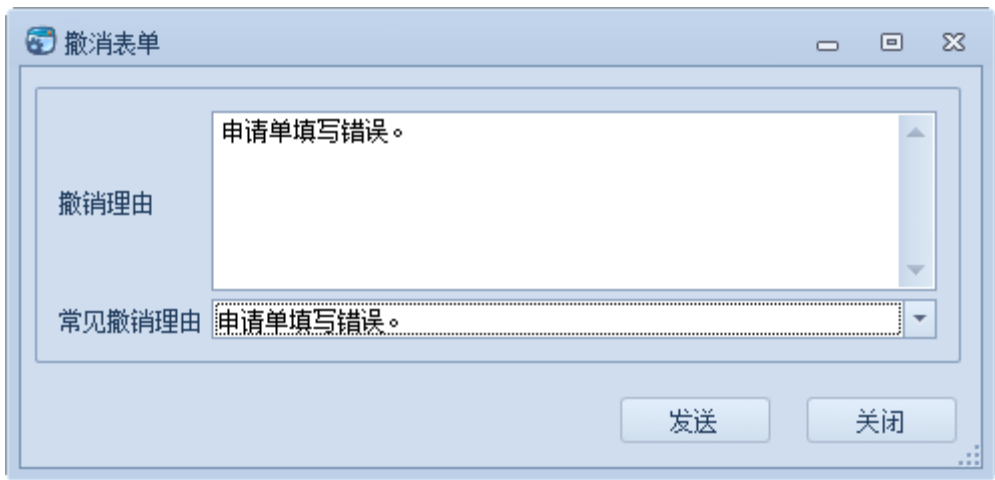
阅办意见

已阅

发送

关闭

当然，若申请人的申请单填写错误，需要撤销的话，那么也应该有这个操作，撤销表单后，就可以重新填写表单，然后再次提交进行流程。



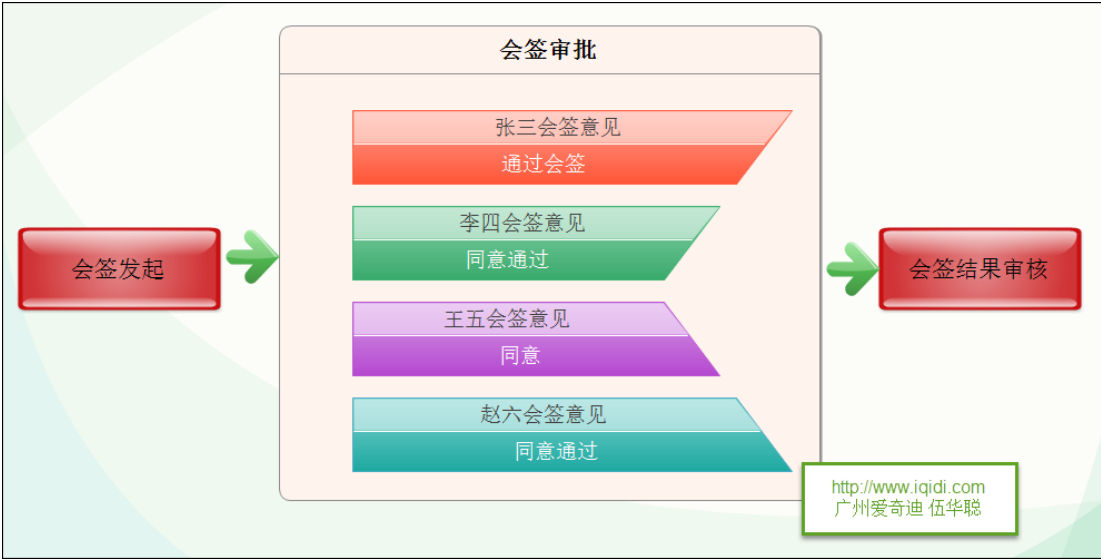
2.3. 流程会签操作

2.3.1. 会签流程定义

会签是指创建一个或多个子流程供相关人员进行审批，等待全部人员完成处理后再次回到主流程上，然后决定是否继续流转到下一个流程步骤上去，一般的申请单的主流程如下所示。



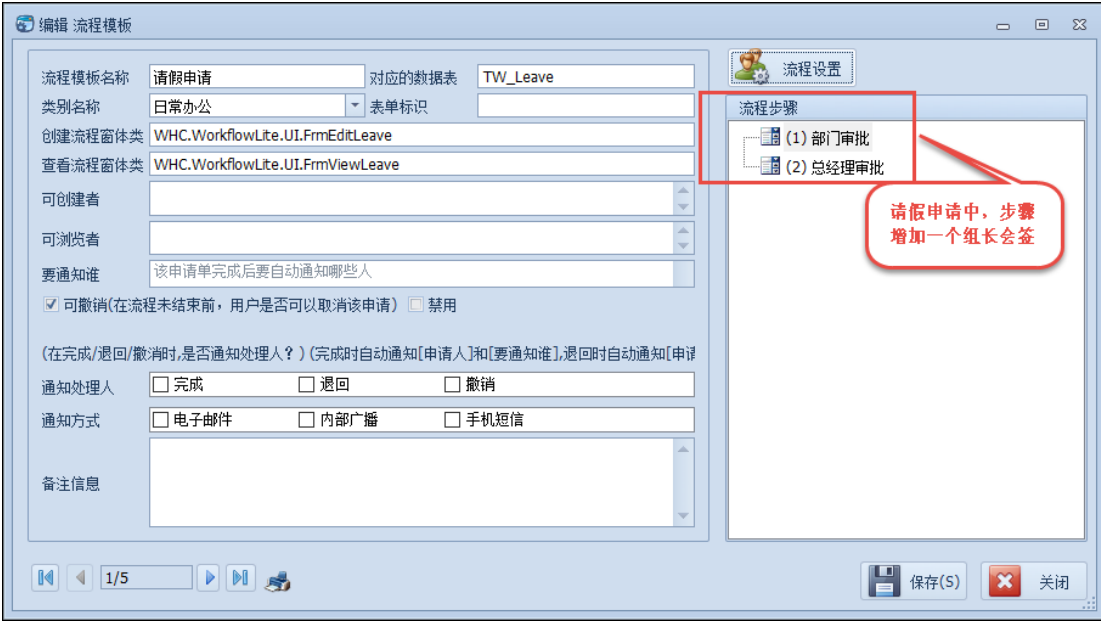
这里设置的会签处理就是其中一个步骤，一旦会签处理步骤发起会签，就会构建多个可供审批的子流程了，如下所示。



在会签发起的步骤，指定参与具体流程会签审批的人员，然后流程则会流转至不同人员进行相关的处理【待办事项】。

我在工作中定义会签完成后，由会签发起人审核（会签结果审核），决定是否进入下一步流程，在审核过程中决定如何处理这个申请单。

在流程定义里面，我们创建一个会签的流程步骤，我们以请假单为例，加入我们要求请假需要由各组长会签通过，然后在继续下面的部门审批、总经理审批步骤，如下所示。



增加会签后的流程步骤如下所示。

新建 表单流程步骤

步骤名称

组长会签

环节名称

会签

存在环节名称

部门审批

上一流程

适用对象（过滤）

过滤条件

流程处理人

管理员（admin） / 广州管理员（gzadmin） / 上海管理员（shadmin）

选择人员

选择处理人

申请人可选

增加流程环节

可以增加多级

通知方式

备注信息

+

添加(A)

保存(S)

×

关闭

完成后可以在流程步骤列表中看到会签的步骤了，如下所示。

流程设置

流程步骤

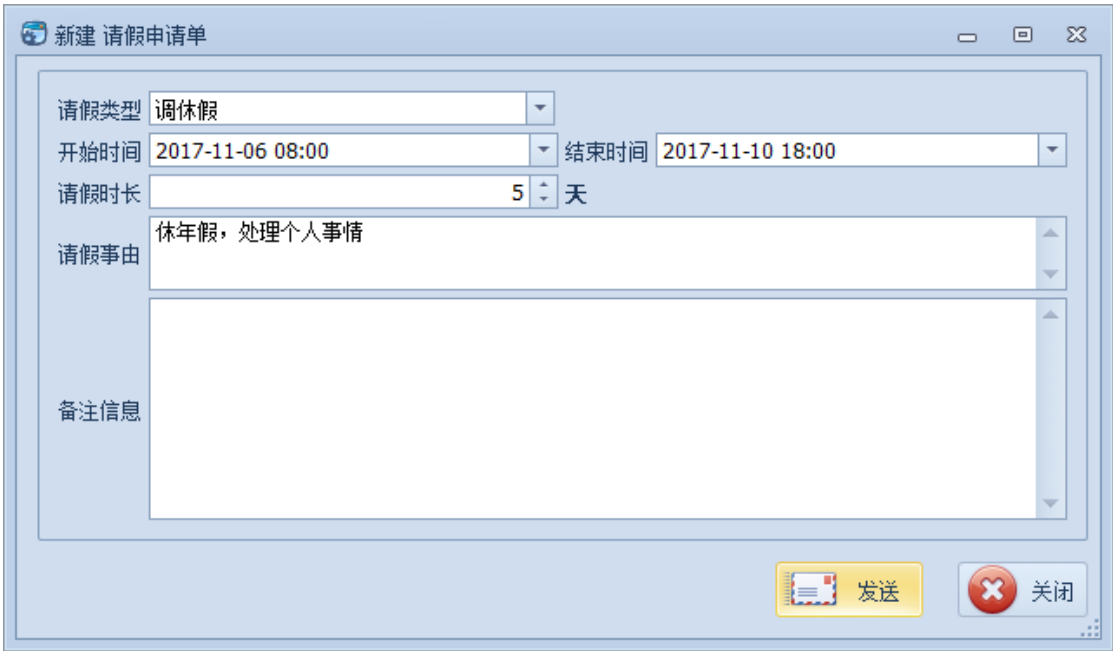
(1) 组长会签

(2) 部门审批

(3) 总经理审批

2.3.2. 会签流程处理

了解了会签的处理过程，并完成了上面的会签定义后，我们创建一个请假申请单，用来发起会签处理，介绍会签的步骤说明。



新建 请假申请单

请假类型: 调休假

开始时间: 2017-11-06 08:00 结束时间: 2017-11-10 18:00

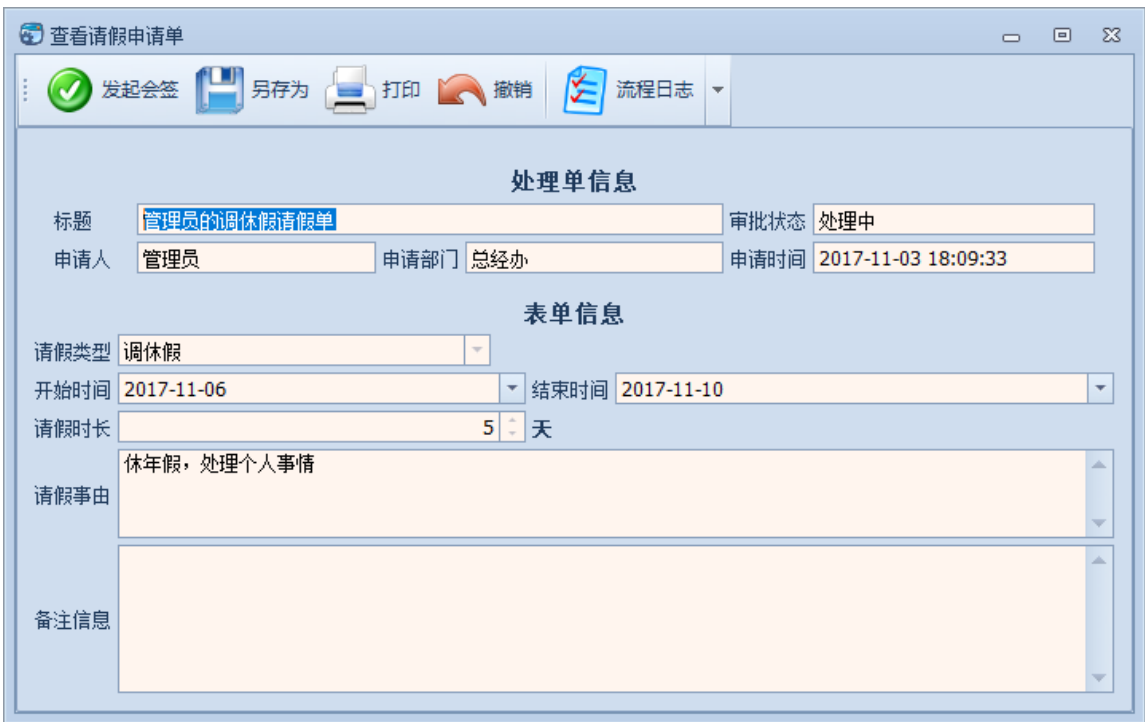
请假时长: 5 天

请假事由: 休年假, 处理个人事情

备注信息:

发送 关闭

完成请假单后提交给相关处理人, 处理人员在待办事项中查看申请单, 如下界面所示。



查看请假申请单

发起会签 另存为 打印 撤销 流程日志

处理单信息

标题: 管理员的调休假请假单 审批状态: 处理中

申请人: 管理员 申请部门: 总经办 申请时间: 2017-11-03 18:09:33

表单信息

请假类型: 调休假

开始时间: 2017-11-06 结束时间: 2017-11-10

请假时长: 5 天

请假事由: 休年假, 处理个人事情

备注信息:

那么他会发起【发起会签】的处理操作, 把相关的投票权发送给各个组长进行会签处理。

会签操作

常用意见

保存常用意见

处理意见

请各组长进行会签。

审批意见

☒ 批准申请

☐ 退回拟稿人重新处理

☐ 退回上一步处理

会签人员

待选人员

☐ 所在部门

☒ 选择人员

>>>>

>>

<<

<<<<

已选人员

北京管理员 (bjadmin)
上海管理员 (shadmin)
广州管理员 (gzadmin)

注：会签处理，需要会签人员同时通过才算最终通过，否则任何审批人员可做退回处理。

发送

关闭

会签发起后，各个参与会签的人员在【待办事项】里面处理会签意见，如下所示。

会签操作

常用意见

保存常用意见

处理意见

通过会签

审批意见

☒ 批准申请

☐ 不通过申请

☐ 退回拟稿人重新处理

☐ 退回上一步处理

会签人员意见

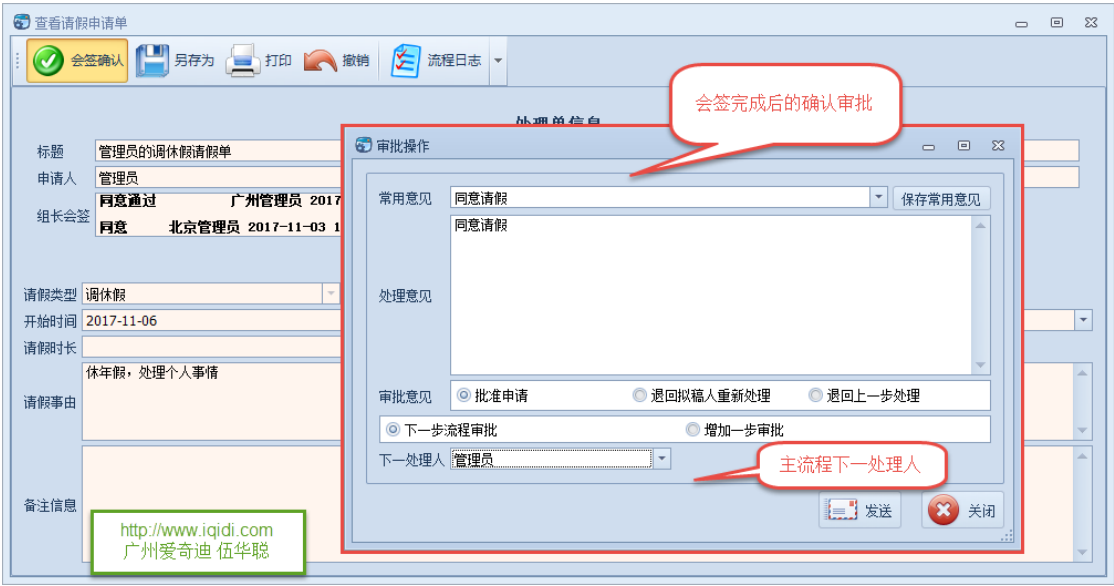
用户名	处理方式	意见
广州管理员	通过	同意通过
北京管理员	通过	同意
上海管理员	未处理	

注：会签处理，需要会签人员同时通过才算最终通过，否则任何审批人员可做退回处理。

发送

关闭

各个待审批的人员进行处理后，最后返回给会签发起人决定是否进入下一步流程，如下所示。



我们查看相关的流程日志，看到会签的流程步骤已经完成了，其他步骤还需要进行处理。

流程日志						
申请单处理流程		申请单处理历史信息		申请单系统日志信息		
	流程名称	流程处理人	是否处理	实际处理人	实际处理时间	处理意见
1	组长会签	管理员	已处理	管理员	2017-11-03 18:36:57	同意请假
2	部门审批	管理员	未处理			
3	总经理审批		未处理			

这样这个会签流程就算整体完成了，剩下的就是其他步骤的处理，按正常的审批处理即可。

2.4. 流程审批的表单处理

2.4.1. 信访投诉工作表单处理

在表单的动态设计和显示方面，如果倾向零代码的动态处理，则牺牲很多内在逻辑处理；如果每个表单开发过于繁琐，则降低开发效率。

因此我觉得把流程模块作为半模块化即可，把部分界面通过代码编写方式进行整合，因此把表单的填写，表单查看做到用户控件里面，然后在界面里面引用即可。

如下面的表单填写操作界面如下所示，对不同的流程表单，在项目中增加一个表单的填写界面和保存操作即可。

新建 信访投诉工作申请单

表格信息 正文内容 附件及备注

工作类别 信访投诉 紧急程度 ☒ 平件 ☐ 急件 ☐ 特急 过期日期 2014-02-28

标题 关于处理广场舞噪声问题的处理通知

内容摘要 关于处理广场舞噪声问题的处理通知，此处省略1000字。

拟办意见 交领导审批处理

发送 关闭

查看并处理的表单操作，我们可以先做一个查看表单信息的界面，然后整合流程的处理工具栏，组合成一个查看、处理操作一体化的流程操作。

查看申请单信息

领导审核 另存为 打印 撤销 流程日志

审批操作工具栏

表格 正文 附件

处理单信息

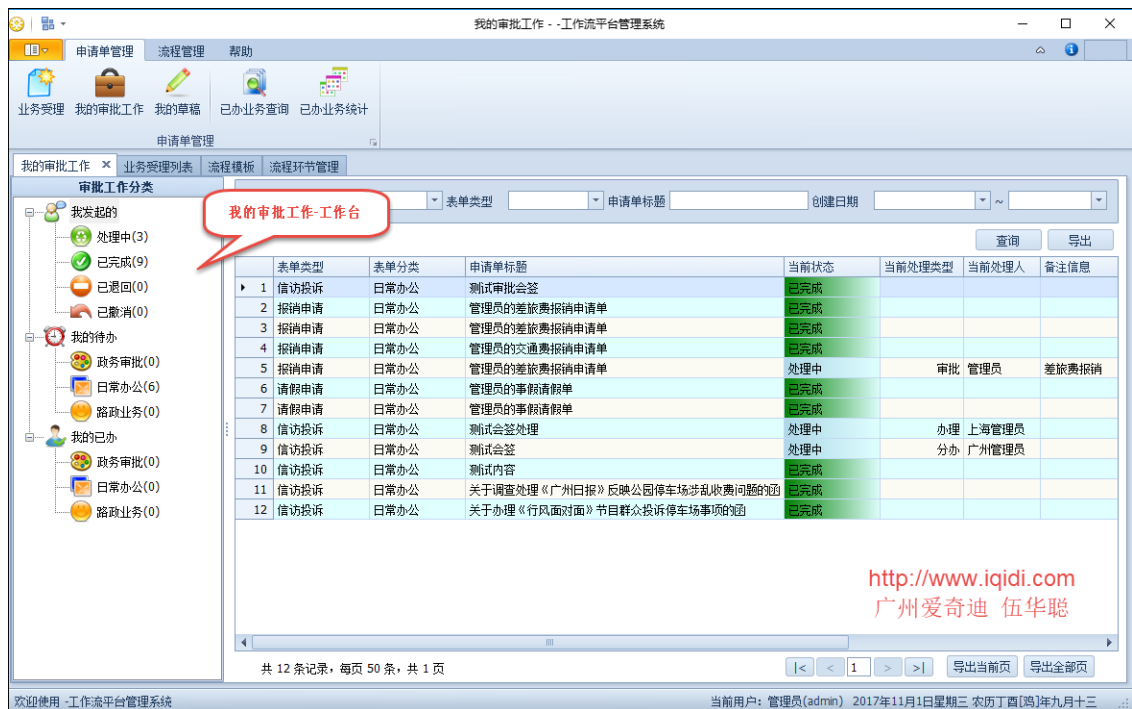
标题	关于处理广场舞噪声问题的处理通知	审批状态	处理中
申请人	管理员	申请部门	
工作类别	日常办公	紧急程度	平件
		过期日期	2014-02-28 00:00:00

内容摘要 关于处理广场舞噪声问题的处理通知，此处省略1000字。

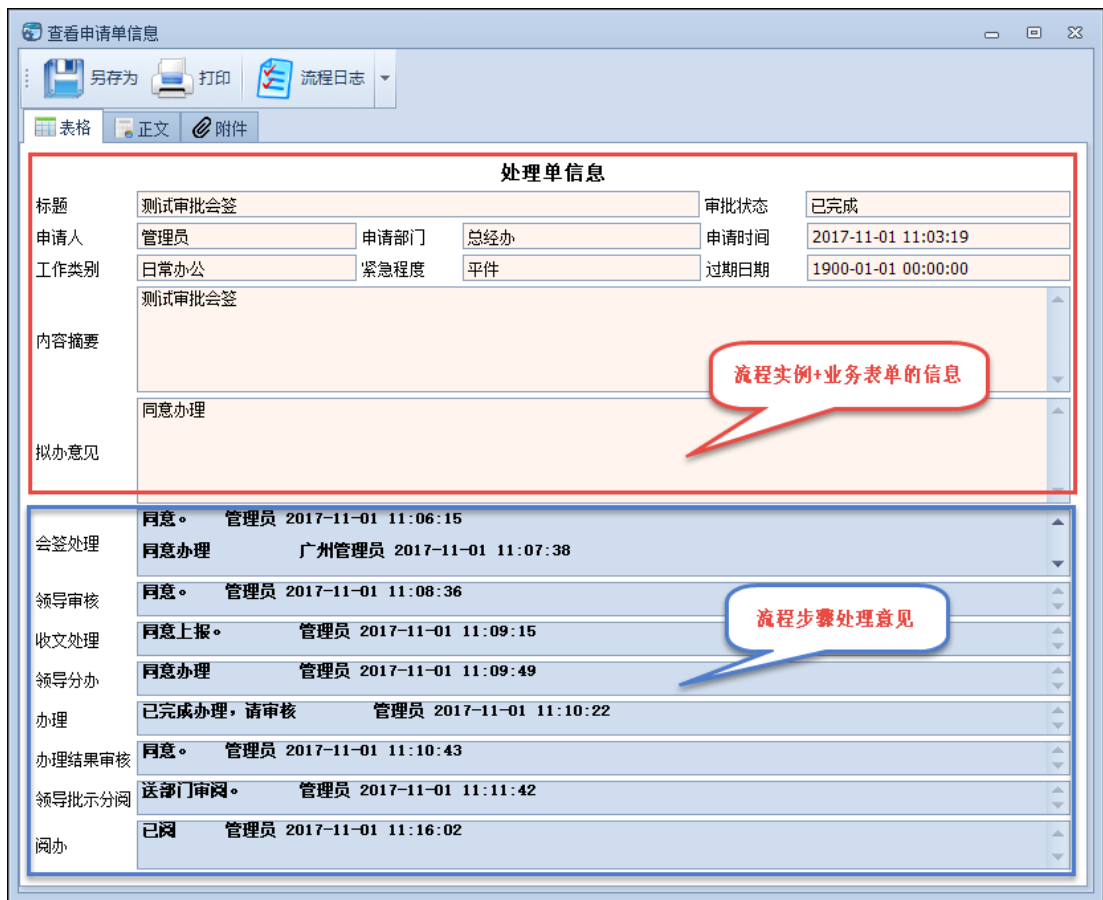
拟办意见 交领导审批处理

业务表单数据

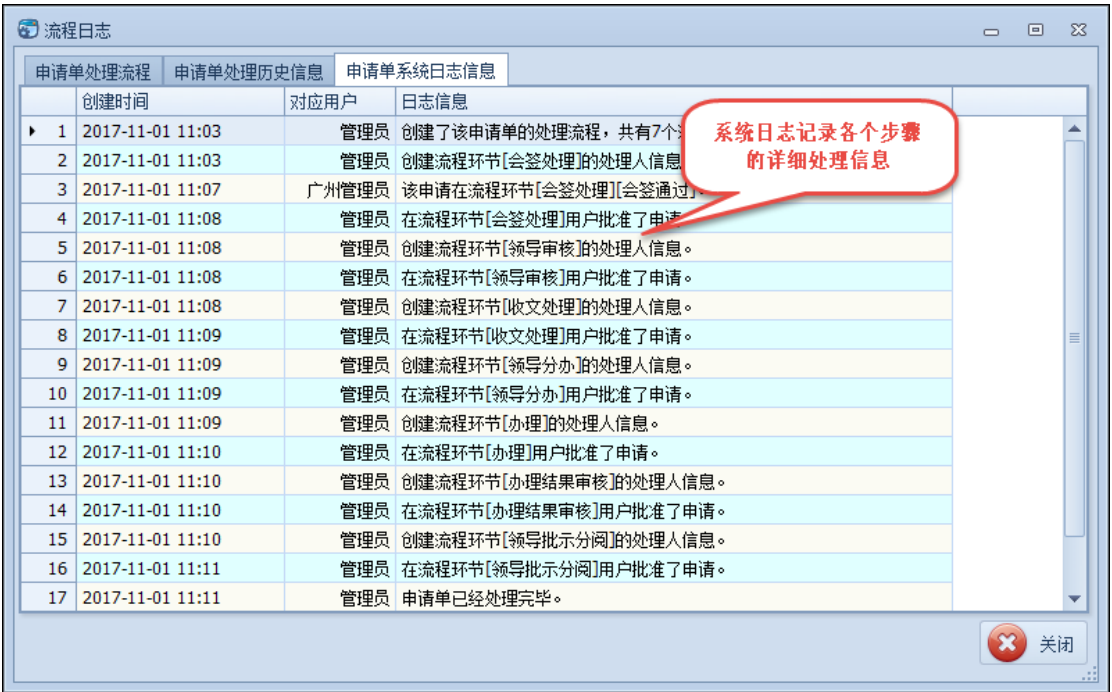
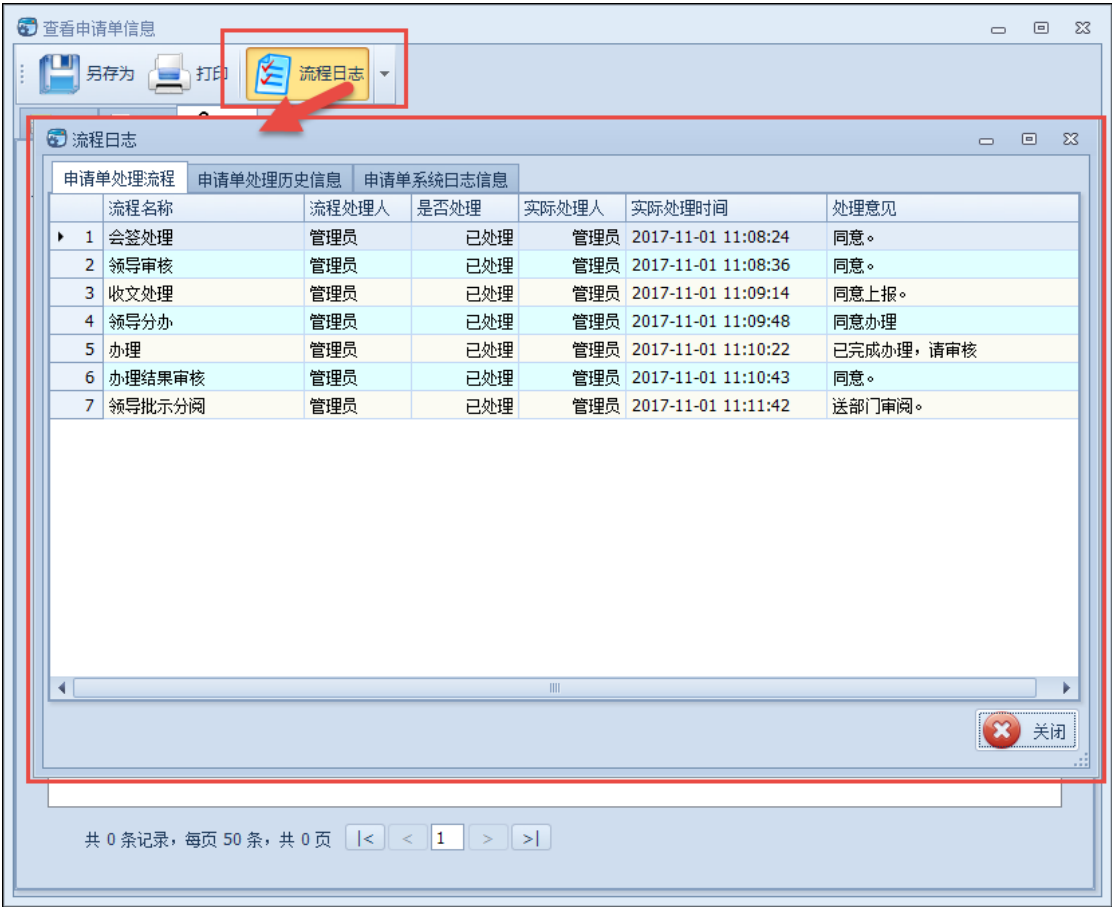
下面是具体表单的查看信息，包含了相关的处理步骤信息，以及相关的流程日志信息。



详细表单查看界面如下所示。



流程日志分为几个部分：申请单处理日志、申请单处理历史信息、申请单系统日志等几个部分



2.4.2. 请假申请表单

这里介绍的具体功能，是在业务表单已经完成好，具体的使用过程，我们以请假申请单为例，在我们创建对应的流程步骤如下所示后，

编辑 流程模板

流程模板名称: 请假申请 对应的数据表: TW_Leave

类别名称: 日常办公 表单标识:

创建流程窗体类: WHC.WorkflowLite.UI.FrmEditLeave

查看流程窗体类: WHC.WorkflowLite.UI.FrmViewLeave

可创建者:

可浏览者:

要通知谁: 该申请单完成后要自动通知哪些人

☒ 可撤销(在流程未结束前, 用户是否可以取消该申请) ☐ 禁用

(在完成/退回/撤销时,是否通知处理人?) (完成时自动通知[申请人]和[要通知处理人])

通知处理人: ☐ 完成 ☐ 退回 ☐ 撤销

通知方式: ☐ 电子邮件 ☐ 内部广播 ☐ 手机短信

备注信息:

流程步骤

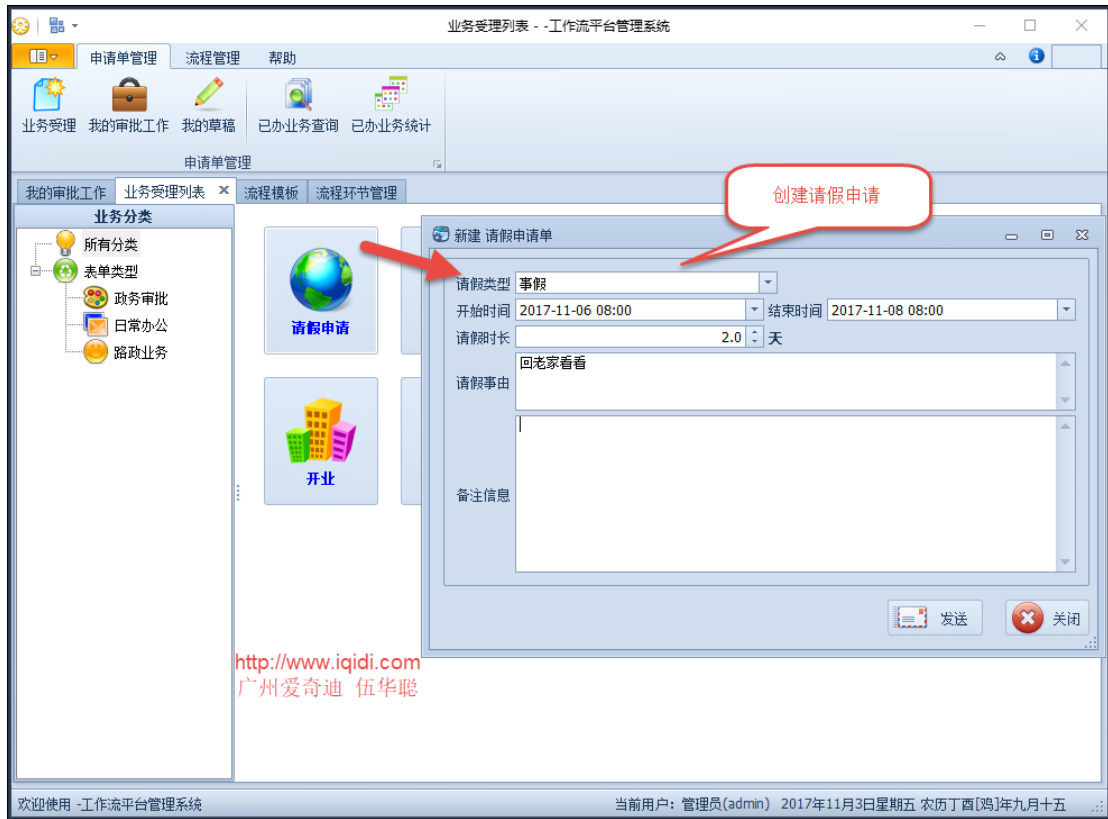
(1) 部门审批

(2) 总经理审批

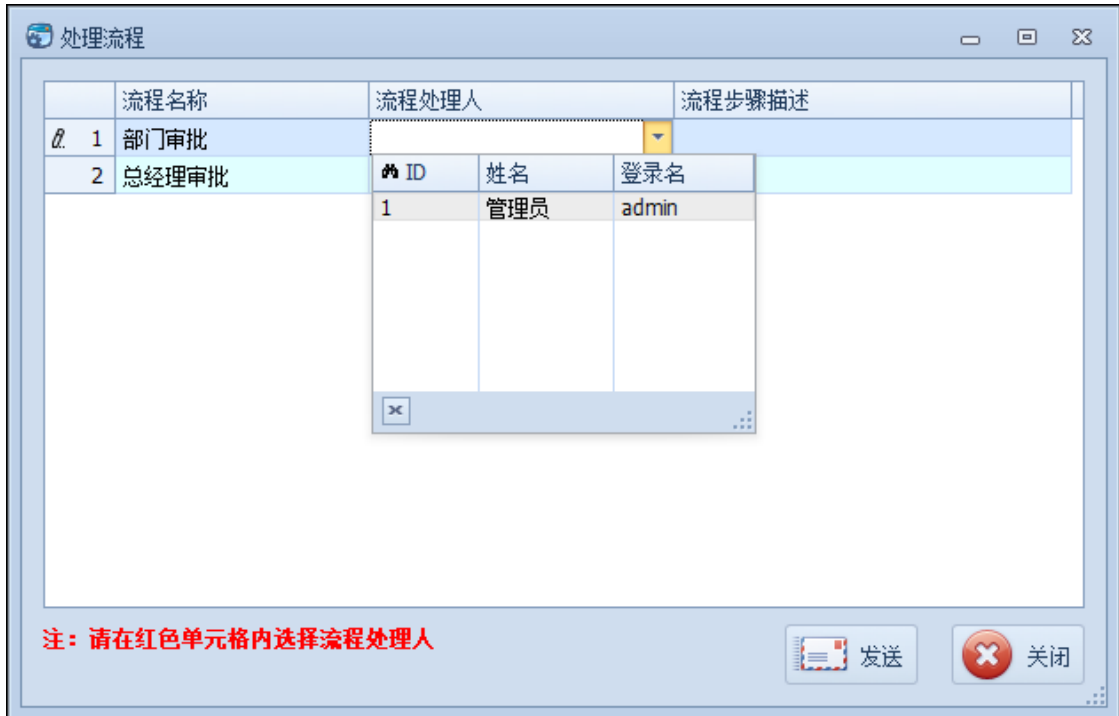
1/5

保存(S) 关闭

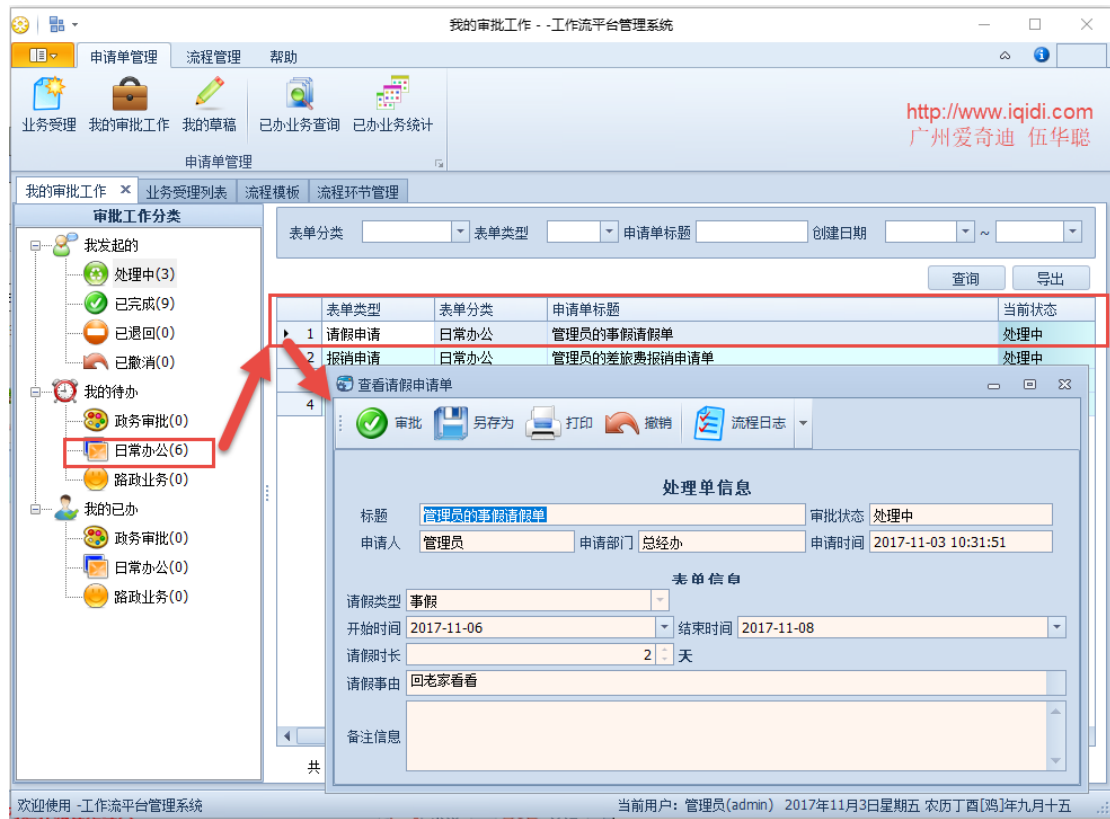
我们就可以开始创建具体的业务审批流程了，在业务受理列表里面选择对应的请假申请，弹出相应的创建业务流程表单窗体。



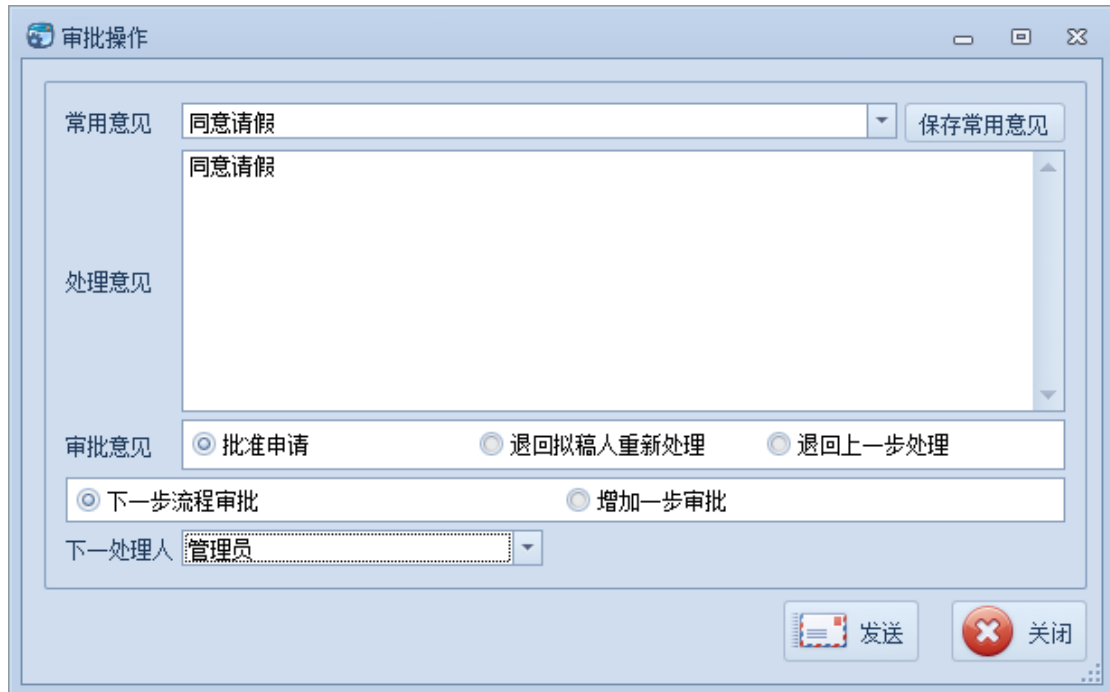
单击【发送】按钮，会要求选择下一步的审批人，如下所示。



这样对应的审批人员进去后，在我的待办业务列表里面，就可以看到刚才的表单了，双击可以进行查看，以及相关的审批处理工作。



单击表单左上角的工具栏按钮【审批】，可以进行流程的审批处理操作了。



在处理完成相关的审批操作后，流程就会自动流转到了表单定义的下一流程人了，这里为了方便选择了管理员的角色来处理而已。

流程人登陆后查看具体业务申请单后，可以看到前面的审批意见等信息，如下所示。

查看请假申请单

审批

另存为

打印

撤销

流程日志

处理单信息

标题

管理员的事假请假单

审批状态

处理中

申请人

管理员

申请部门

总经办

申请时间

2017-11-03 10:31:51

部门审批

同意请假

管理员

2017-11-03 10:39:53

表单信息

请假类型

事假

开始时间

2017-11-06

结束时间

2017-11-08

请假时长

2

天

请假事由

回老家看看

备注信息

前面流程环节的审批意见

携带的表单数据

继续【审批】处理环节，这样两个环节的流程就处理完成了。

查看请假申请单

审批

另存为

打印

撤销

流程日志

处理单信息

标题

管理员的事假请假单

审批状态

处理中

申请人

管理员

申请部门

总经办

申请时间

2017-11-03 10:31:51

部门审批

同意请假

管理员

2017-11-03 10:39:53

请假类型

事假

开始时间

2017-11-06

请假时长

请假事由

回老家看看

备注信息

日常办公(0)

路政业务(0)

审批操作

常用意见

同意请假

保存常用意见

处理意见

审批意见

批准申请

退回拟稿人重新处理

退回上一步处理

下一步流程审批

增加一步审批

继续审批处理

发送

关闭

最后我们查看完成的业务申请单，可以看到相关的审批信息了。

查看请假申请单

另存为

打印

流程日志

处理单信息

标题

管理员的事假请假单

审批状态

已完成

申请人

管理员

申请部门

总经办

申请时间

2017-11-03 10:31:51

部门审批

同意请假 管理员 2017-11-03 10:39:53

总经理审批

同意请假 管理员 2017-11-03 10:43:45

表单信息

请假类型

事假

开始时间

2017-11-06

结束时间

2017-11-08

请假时长

2

天

请假事由

回老家看看

备注信息

当然也可以查看具体的业务流程日志。

流程日志

申请单处理流程

申请单处理历史信息

申请单系统日志信息

	流程名称	流程处理人	是否处理	实际处理人	实际处理时间	处理意见
1	部门审批	管理员	已处理	管理员	2017-11-03 10:39:53	同意请假
2	总经理审批	管理员	已处理	管理员	2017-11-03 10:43:45	同意请假

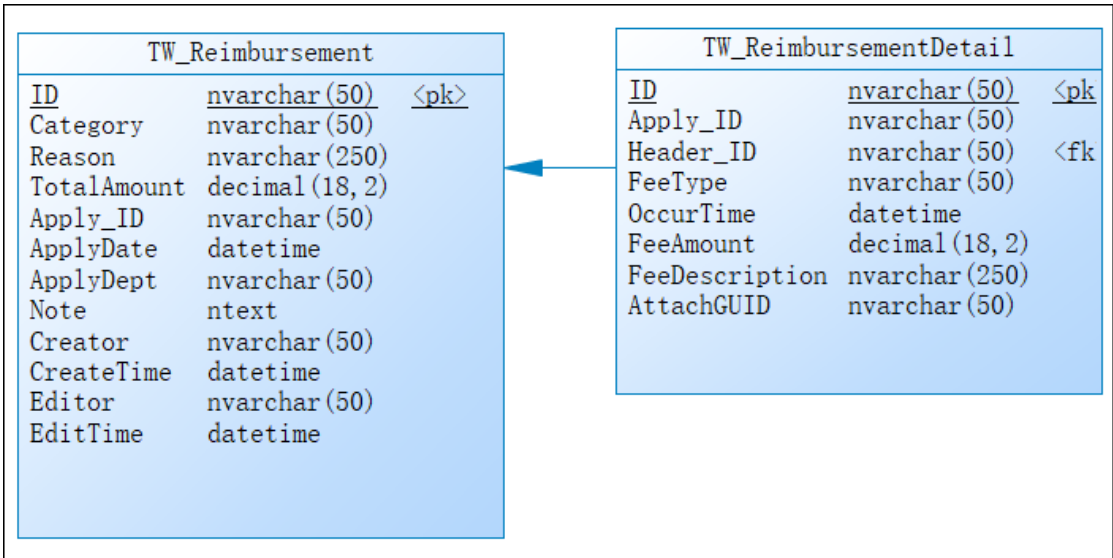
关闭

整个流程步骤全部完成了。

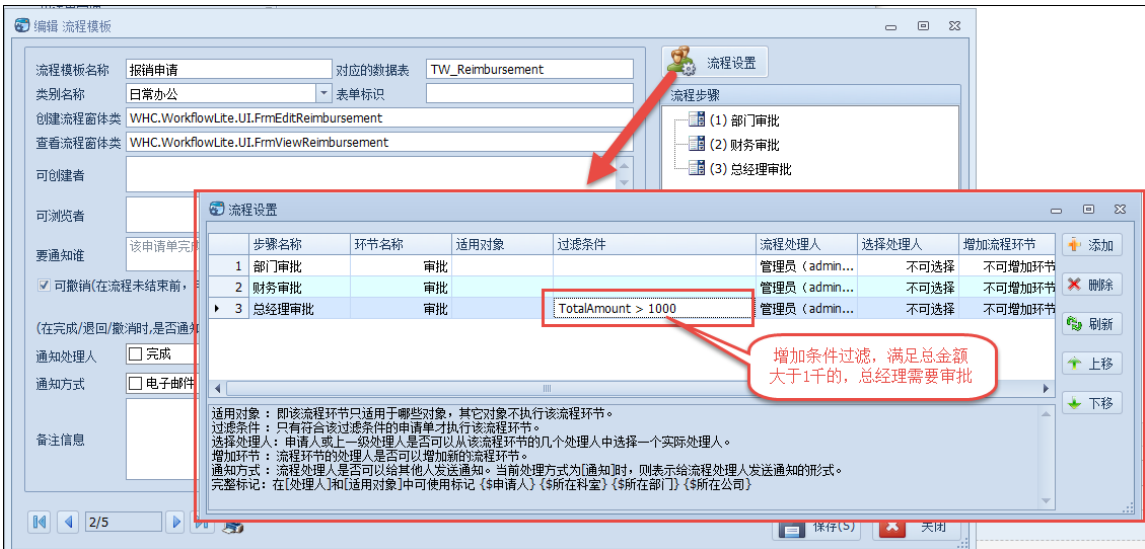
2.4.3. 报销申请单

为了增加复杂一点的表单我们引入一个含有主从表的业务表单，明细表包括报销的具体列表内容。

具体的业务表单设计如下所示。



首先我们定义好业务流程，如下所示。



创建一个业务表单，如下所示。

新建 报销申请单

报销类型 差旅费

报销事由 差旅费

总金额 300 元

备注信息

报销申请明细清单

	费用类型	发生时间	费用金额	费用说明
1	飞机票	2017-11-03 10:56	1000	
2	餐饮费	2017-11-03 10:57	1000	
3	住宿费	2017-11-03 10:57	1000	
*				

发送

关闭

其中几个流程的审批和上面的处理操作差不多，最后完成表单后，那么这个总共参与的步骤是三个步骤，包括了总经理审批环节了，和下面的表单过程类似，界面效果如下所示。

查看报销申请单

另存为

打印

流程日志

处理单信息

标题

管理员的差旅费报销申请单

审批状态

已完成

申请人

管理员

申请部门

总经办

申请时间

2017-10-29 17:09:40

部门审批

同意申报

管理员 2017-10-29 17:09:40

财务审批

同意申报

管理员 2017-10-29 17:10:00

总经理审批

同意申报

管理员 2017-10-29 17:10:11

处理步骤意见

表单信息

报销类型

差旅费

报销事由

差旅费

总金额

3000.00

元

备注信息

报销申请明细清单

	费用类型	发生时间	费用金额	费用说明
▶ 1	餐饮费	2017-10-29 16:58	1000.00	
2	住宿费	2017-10-29 16:58	1000.00	
3	飞机票	2017-10-29 16:58	1000.00	

如果是报销费用少于条件值（3000）的，那么审批的环节就不包括总经理审批环节了，如下表单所示。

查看报销申请单

另存为

打印

流程日志

处理单信息

标题

管理员的差旅费报销申请单

审批状态

已完成

申请人

管理员

申请部门

总经办

申请时间

2017-10-29 17:24:25

部门审批

同意申报

管理员

2017-10-29 17:24:44

财务审批

同意申报

管理员

2017-10-29 17:25:07

表单信息

报销类型

差旅费

报销事由

差旅费

总金额

900.00

元

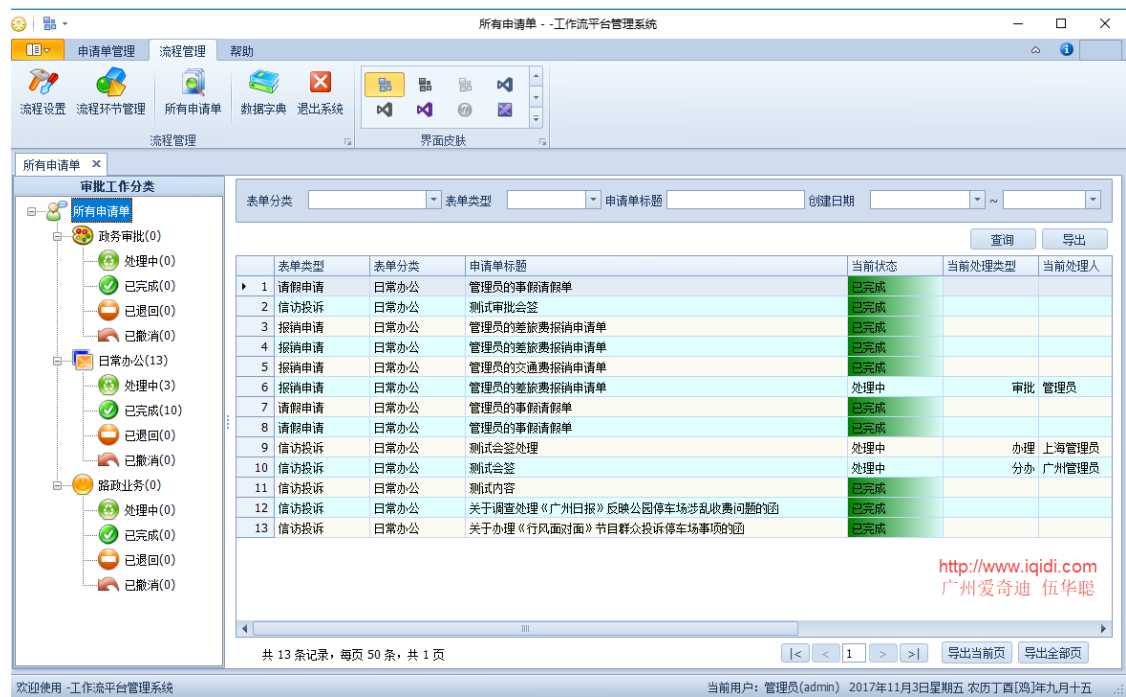
备注信息

报销申请明细清单

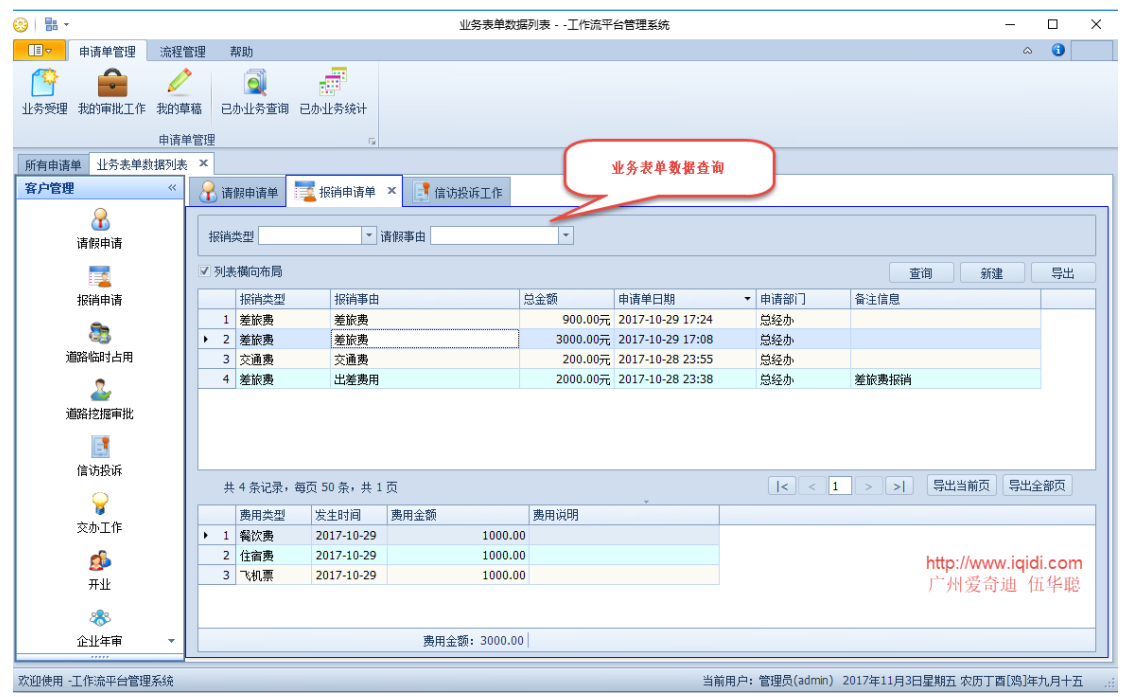
	费用类型	发生时间	费用金额	费用说明
▶ 1	飞机票	2017-10-29 17:15	900.00	

以上就是两类表单，一个是普通的单表表单、一个是包含明细表的主从表表单的流程处理，以及步骤中包含相关的过滤条件，满足条件的则包含该条件的审批处理，否则系统自动跳过这个步骤，直接跳转到下一步或者完成整个申请单的处理。

有时候为了方便查看对应的业务表单，一般提供一个所有表单的入口给管理查看，方便进行维护管理，如下所示。

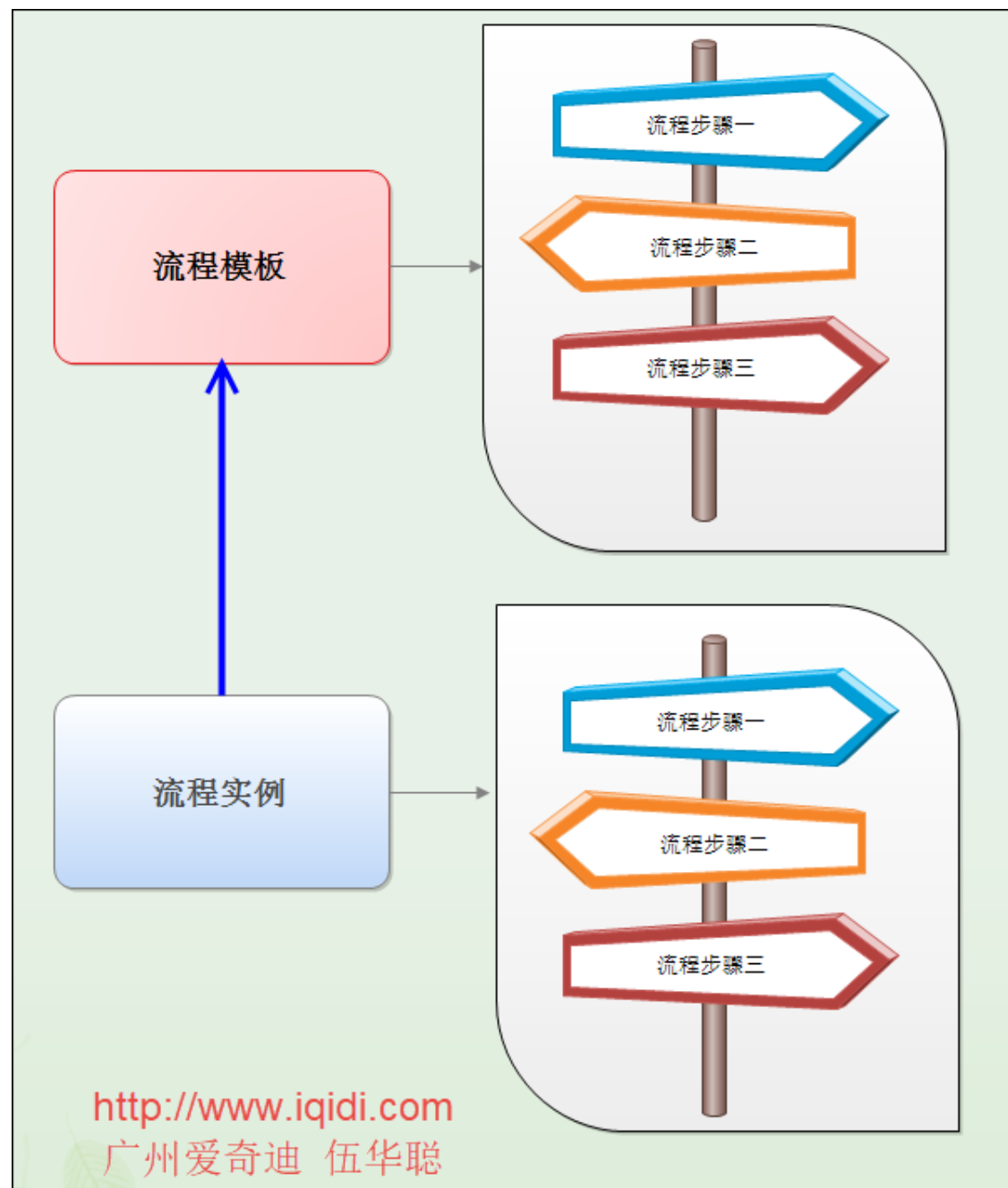


管理员可以在这里对业务表单进行删除、撤回等特殊处理, 另外, 我们也可以提供一个业务数据的入口方便查询统计, 如下所示的界面。

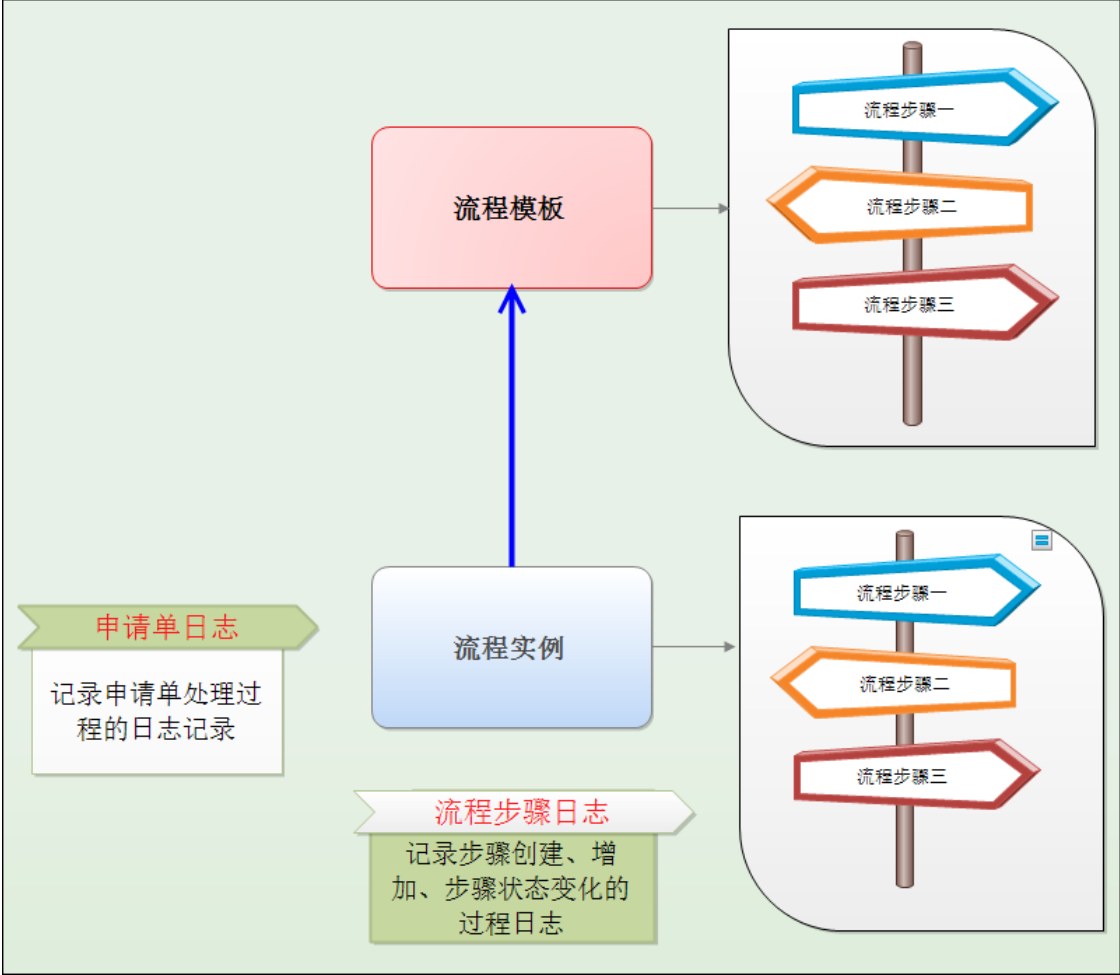


3. workflow 模块的表设计分析

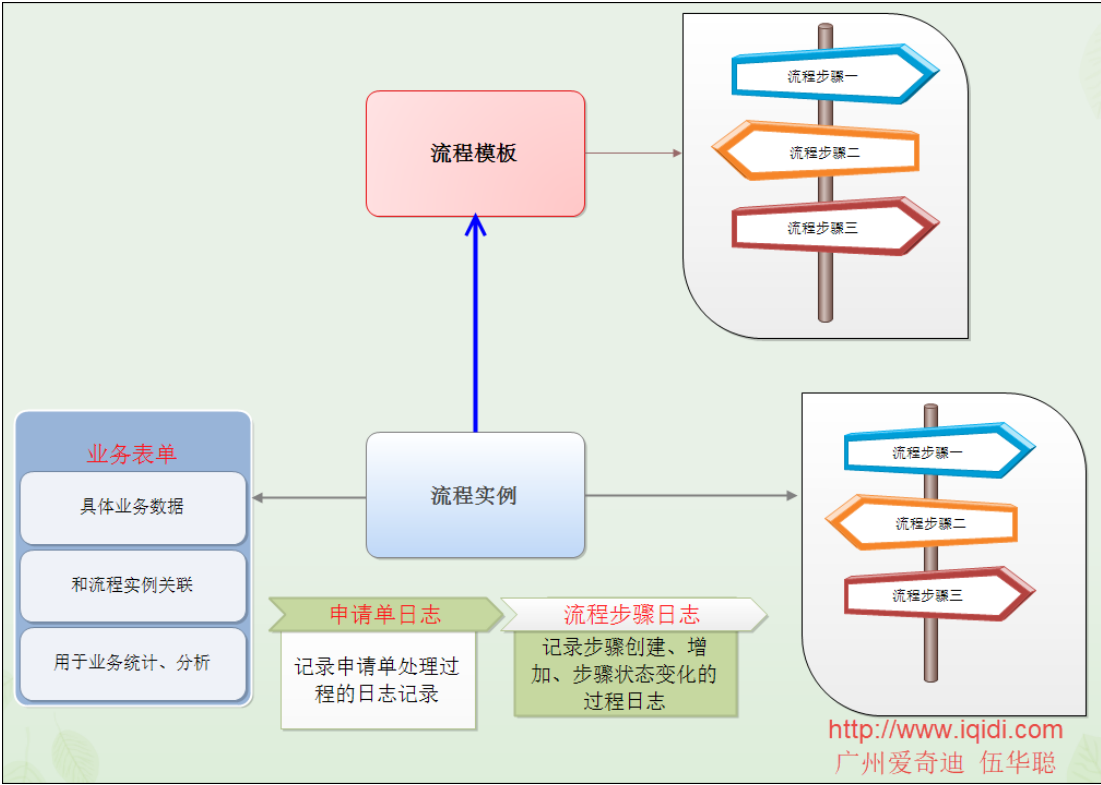
在 workflow 处理表中，首先我们区分流程模板和流程实例两个部分，这个其实就是类似模板和具体文档的概念，我们一份模板可以创建很多个类似的文档，文档样式结构类似的。同理，流程模板实例为流程实例后，就是具体的一个流程表单信息了，其中流程模板和流程实例表单都包括了各个流程步骤。在流程实例的层次上，我们运行的时候，需要记录一些日志方便跟踪，如流程步骤的处理日志，流程实例表单的处理日志等这些信息。



当然实际的流程实例里面需要记录很多信息，其中流程步骤日志、申请单处理日志等信息是必须要记录的，方便我们跟踪相关的处理记录。因此 workflow 业务表包含多两个日志记录的表，如下所示。

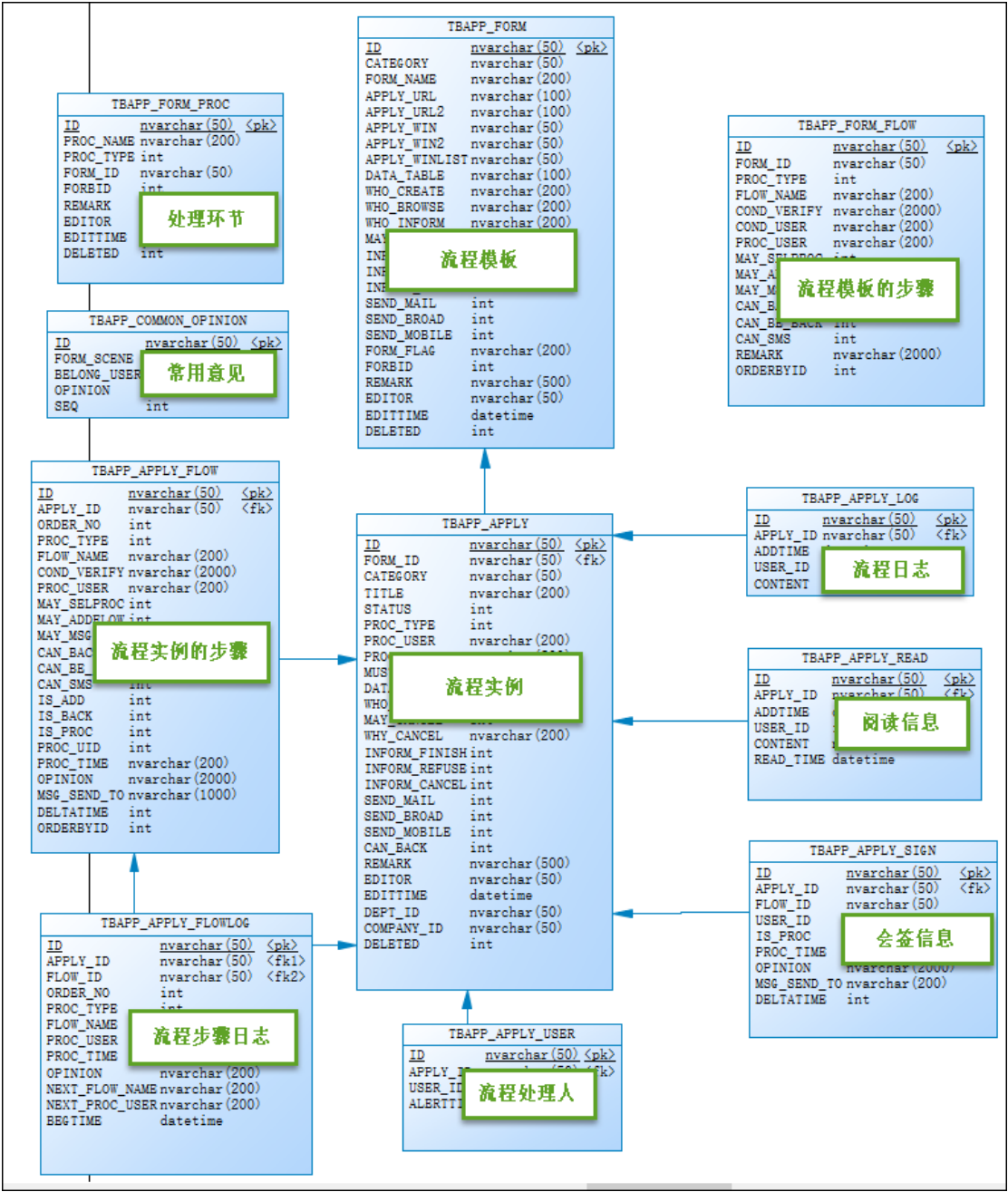


一旦流程实例根据模板创建后，流程先根据模板初始化后，在处理过程还可以动态增加一些审批步骤，使得我们的处理更加弹性化。



当然，为了更好的处理流程的相关信息，还需要记录流程处理人，流程会签人、流程阅办人，以及常用审批意见等相关辅助表，以便对流程的各个处理信息进行合理处理和展示。

整个工作流的核心基础表如下图所示。

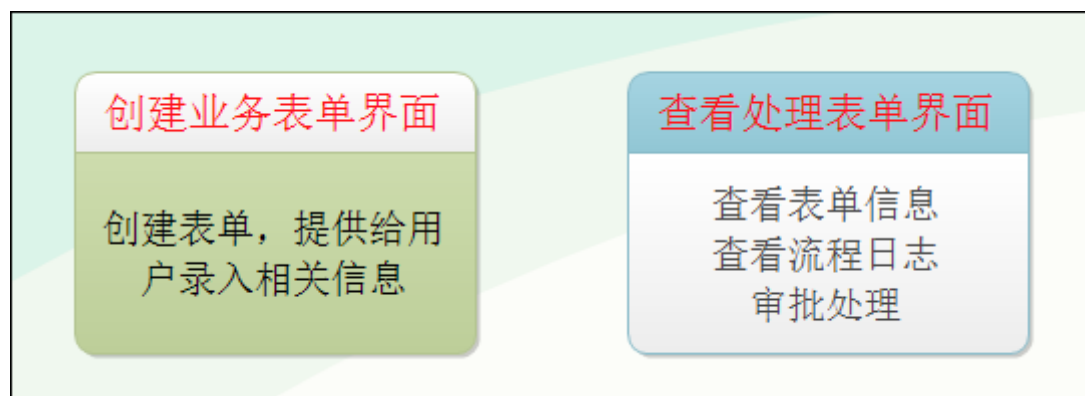


4. workflow 模块的业务表单开发

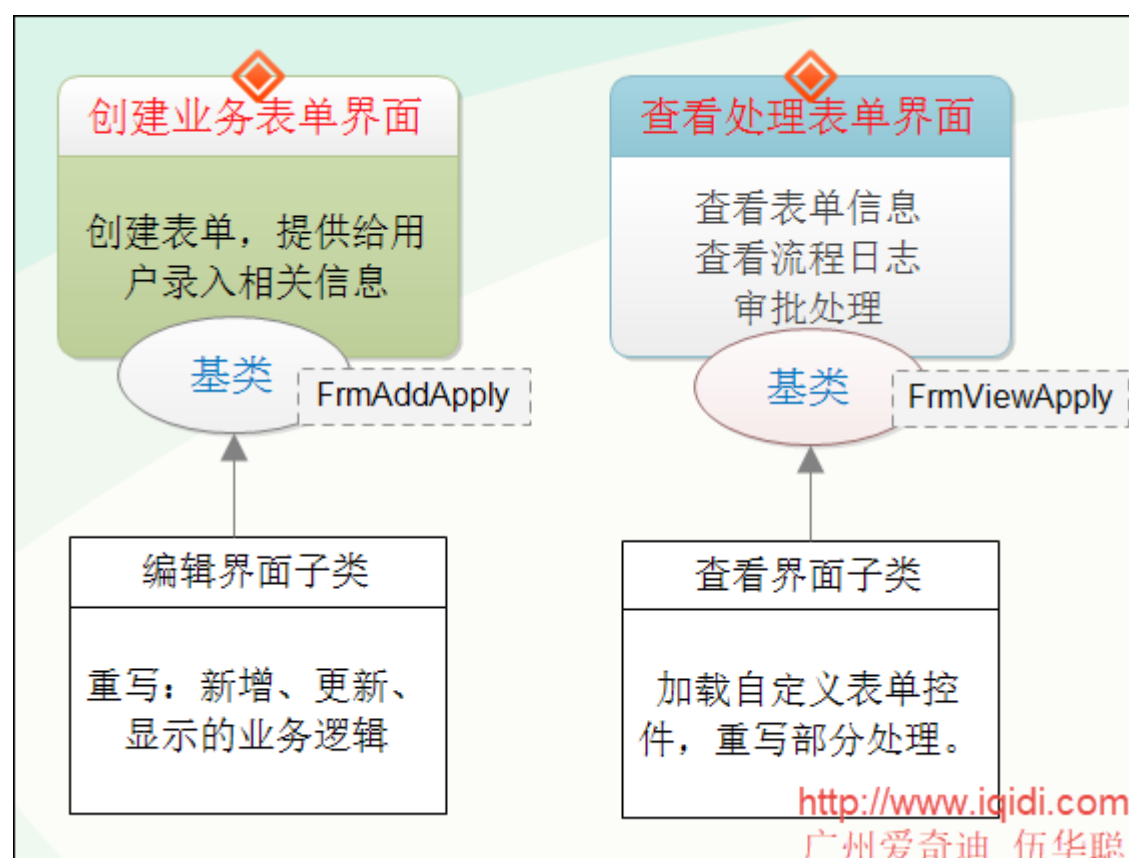
在我们开发 workflow 的时候，往往需要设计到具体业务表单信息的编辑，有些是采用动态编辑的，有些则是在开发过程中处理的，各有各的优点，动态编辑的则方便维护各种各样的表单，但是数据的绑定及处理则比较麻烦，而自定义开发的，则数据弹性很大，方便修改调整。本篇随笔基于表单的开发设计过程，介绍在 workflow 中如何新增一个业务表单，以便快速的实现审批业务的上线处理。

4.1. 业务表单的基类继承

首先我们来了解一下业务表单的对应关系，一般创建一个业务流程处理，都需要有一个具体的创建业务表单的界面，以及一个查看处理表单的界面。



为了方便，我们尽可能减少代码编写，我们需要把大多数的逻辑处理放在基类实现，这样我们在新增一个业务表单的时候就可以减少很多代码编写及维护了。



如对于 FrmAddApply 类定义如下，我们定义一些抽象接口用于下面的业务表单实现


```
/// <summary>
/// 创建申请单的窗体基类
/// </summary>
public partial class FrmAddApply : BaseForm
{
    /// <summary>
    /// 表单ID
    /// </summary>
    public string FormID { get; set; }

    /// <summary>
    /// 申请单ID
    /// </summary>
    public string ApplyId { get; set; }

    public FrmAddApply()
    {
        InitializeComponent();
    }

    /// <summary>
    /// 显示数据的函数(子类必须实现)
    /// </summary>
    public virtual void DisplayData() { }

    /// <summary>
    /// 实现控件输入检查的函数(子类必须实现)
    /// </summary>
    /// <returns></returns>
    public virtual bool CheckInput() { return true; }

    /// <summary>
    /// 编辑状态下的数据保存(子类必须实现)
    /// </summary>
    /// <returns></returns>
    public virtual bool SaveUpdated() { return true; }

    /// <summary>
    /// 新增状态下的数据保存(子类必须实现)
    /// </summary>
    /// <returns></returns>
    public virtual bool SaveAddNew() { return true; }

    .....
}
```

这样我们创建一个新的业务表单，只需要利用代码生成工具，生成所需要的各层框架代码，然后再生成 Winform 窗体代码，复制部分界面处理代码过来这个业务表单的子类即可。

下面是一个请假申请的业务表单设计，如下所示。

The screenshot shows a Windows Form titled "请假申请单" (Leave Application Form). The form has a title bar with standard Windows controls. The main content area includes several input fields: a dropdown for "请假类型" (Leave Type), two date/time dropdowns for "开始时间" (Start Time) and "结束时间" (End Time), a numeric spinner for "请假时长" (Leave Duration) currently set to 0 with the unit "天" (days), a text area for "请假事由" (Reason for Leave), and another text area for "备注信息" (Remarks). At the bottom of the form, there is a red URL "http://www.iqidi.com" and the text "广州爱奇迪 伍华聪". To the right of this text are two buttons: "发送" (Send) and "关闭" (Close).

我们看到这个表单可以使用代码生成工具 Database2Sharp 快速生成后进行一定调整的，而这个编辑表单的界面，我们只需要使用自动生成的部分代码即可。

相关代码只需要复制上面的新增、更新、显示数据的代码即可。

```
/// <summary>
/// 请假申请单的编辑界面
/// </summary>
7 个引用
public partial class FrmEditLeave : FrmAddApply
{
    /// <summary>
    /// 创建一个临时对象，方便在附件管理中获取存在的GUID
    /// </summary>
    private LeaveInfo tempInfo = new LeaveInfo();

    2 个引用
    public FrmEditLeave()
    {
        InitializeComponent();
    }

    /// <summary>
    /// 实现控件输入检查的函数
    /// </summary>
    /// <returns></returns>
    4 个引用
    public override bool CheckInput()...

    /// <summary>
    /// 初始化数据字典
    /// </summary>
    1 个引用
    private void InitDictItem()...

    /// <summary>
    /// 更新界面控件数据显示
    /// </summary>
    4 个引用
    public override void DisplayData()...
```

对于查看申请单的基类 FrmViewApply 类，我们更加简单，我们需要把它的自定义界面控件加载出来即可。

下面是查看申请单的基类，封装了相关的处理逻辑。

```
/// <summary>
/// 本窗体是通用的查看申请单界面基类。
/// 为减少开发相关页面的工作量，只需要创建一个新窗体，并继承本窗体，然后在子窗体Form_Load函数里面，初始化对应的申请单显示控件即可。
/// </summary>
public partial class FrmViewApply : BaseDock
{
    /// <summary>
    /// 申请单ID
    /// </summary>
    public string ApplyId { get; set; }

    /// <summary>
    /// 申请单自定义控件
    /// </summary>
    public BaseUserControl ApplyControl { get; set; }

    /// <summary>
    /// 默认构造函数
    /// </summary>
    public FrmViewApply()
    {
        InitializeComponent();
    }

    private void FrmViewApply_Load(object sender, EventArgs e)
    {
        if (!this.DesignMode)
        {
            InitToolBar();
        }
    }

    /// <summary>
    /// 初始化申请单控件
    /// </summary>
    protected virtual void InitApplyControl(BaseUserControl control)
    {
        if (control != null)
        {
            this.ApplyControl = control;
            this.ApplyControl.Dock = DockStyle.Fill;
            this.Controls.Add(control);
        }
    }
}
```

```
/// <summary>
/// 打印申请单控件内容 (默认调用窗体打印)
/// </summary>
protected virtual void PrintApplyControl()
{
    if(this.ApplyControl != null)
    {
        PrintFormHelper.Print(this.ApplyControl, false);
    }
}

/// <summary>
/// 表单另存为
/// </summary>
protected virtual void ApplySaveAs()
{
}

/// <summary>
/// 初始化工具栏的按钮和状态
/// </summary>
protected virtual void InitToolBar()
{
    .....//基类实现，控制什么时候该做什么审批处理，以及一些常见按钮
}

.....
```

查看请假申请单的窗口就是继承这个 FrmViewApply 即可，如下所示。

```
/// <summary>
/// 查看请假申请单的窗体
/// </summary>
public partial class FrmViewLeave : FrmViewApply
{
    private LeaveControl control = null;

    public FrmViewLeave()
    {
        InitializeComponent();
    }

    private void FrmViewLeave_Load(object sender, EventArgs e)
    {
        //初始化控件并展示在基类窗体里面
        control = new LeaveControl();
        control.ApplyId = this.ApplyId;
        control.DisplayData();

        base.InitApplyControl(control);
    }
}
```

这个就是全部的窗体源码了,主要的内容我们看到是在 **LeaveControl** 这个用户控件类里面的了,

而这个控件主要就是上面编辑请假申请单的界面设计,并复制相关的显示数据代码即可。

LeaveControl.cs [设计]

处理单信息

标题

审批状态

申请人

申请部门

申请时间

表单信息

请假类型

开始时间

结束时间

请假时长

0

天

请假事由

备注信息

相关界面代码如下所示。

```
/// <summary>
/// 查看请假申请单的内容显示控件
/// </summary>
public partial class LeaveControl : BaseUserControl
{
    /// <summary>
    /// 申请单ID
    /// </summary>
    public string ApplyId { get; set; }

    public LeaveControl()
    {
        InitializeComponent();

        SetReadOnly();
    }

    /// <summary>
    /// 设置整个窗体布局为只读并设置只读的背景颜色
    /// </summary>
    private void SetReadOnly()
    {
        this.layoutControl1.OptionsView.IsReadOnly = DevExpress.Utils.DefaultBoolean.True;
        this.layoutControl1.Appearance.ControlReadOnly.BackColor = Color.SeaShell;
    }

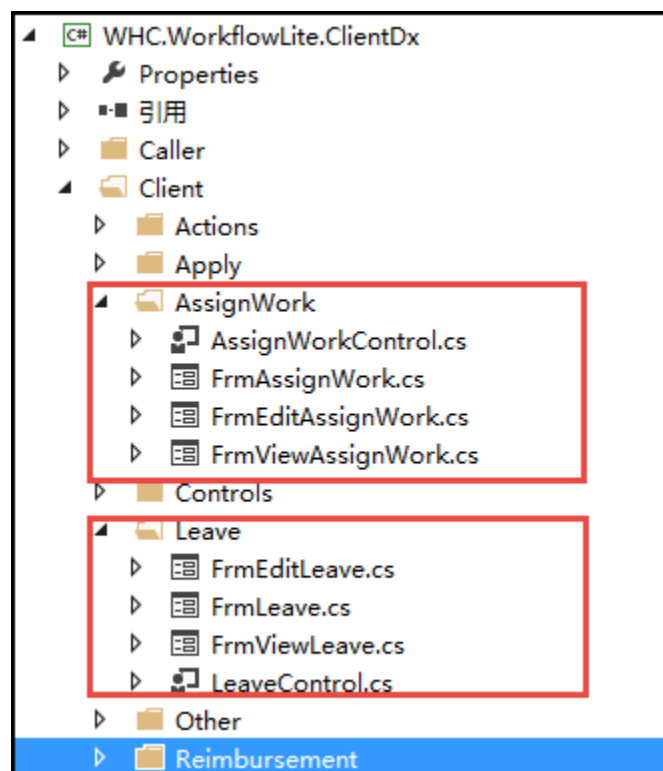
    private void LeaveControl_Load(object sender, EventArgs e)
    {
        this.applyInfoControl1.ApplyId = this.ApplyId;
        this.applyInfoControl1.BindData();
    }

    /// <summary>
    /// 初始化数据字典
    /// </summary>
    private void InitDictItem()
    {
        //初始化代码
    }

    /// <summary>
    /// 数据显示的函数
    /// </summary>
    public void DisplayData()
    {
        InitDictItem(); //数据字典加载 (公用)
    }
}
```

通过上面定义的对对应表单的窗体基类，可以减少我们重复编码的需要，我们只需要利用最有效率的生成界面，然后复制代码后调整即可快速生成我们所需要的不同表单界面。

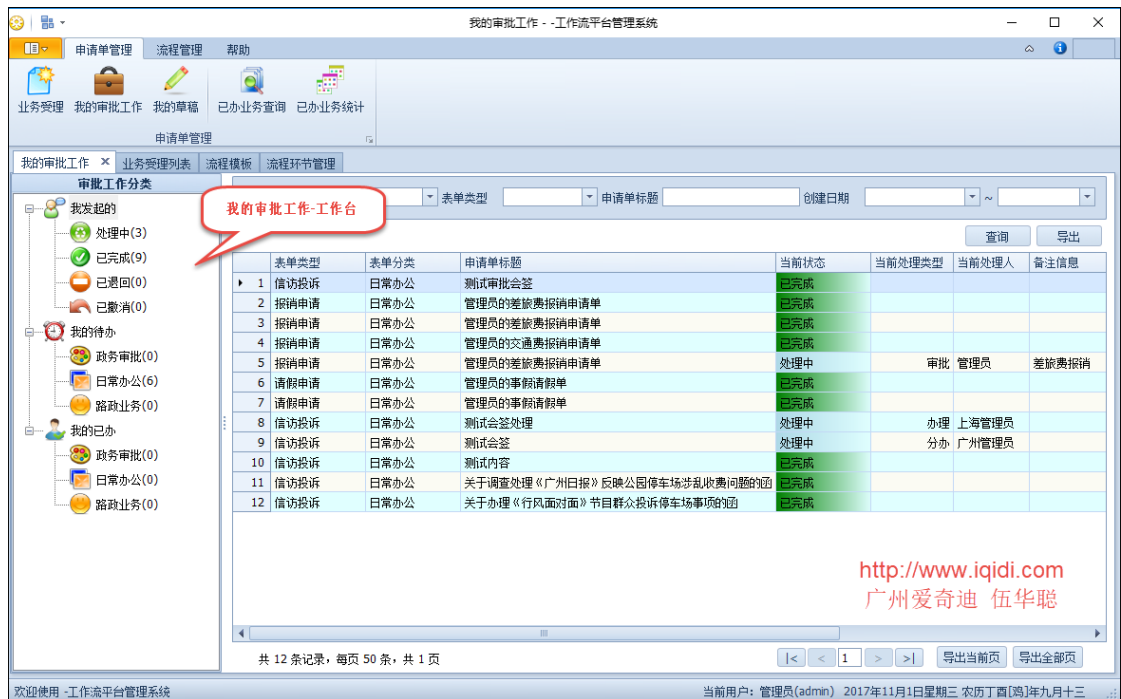
每个表单我们放在一个目录上，这样我们就可以很好管理它们了。



4.2. 业务表单的动态展示处理

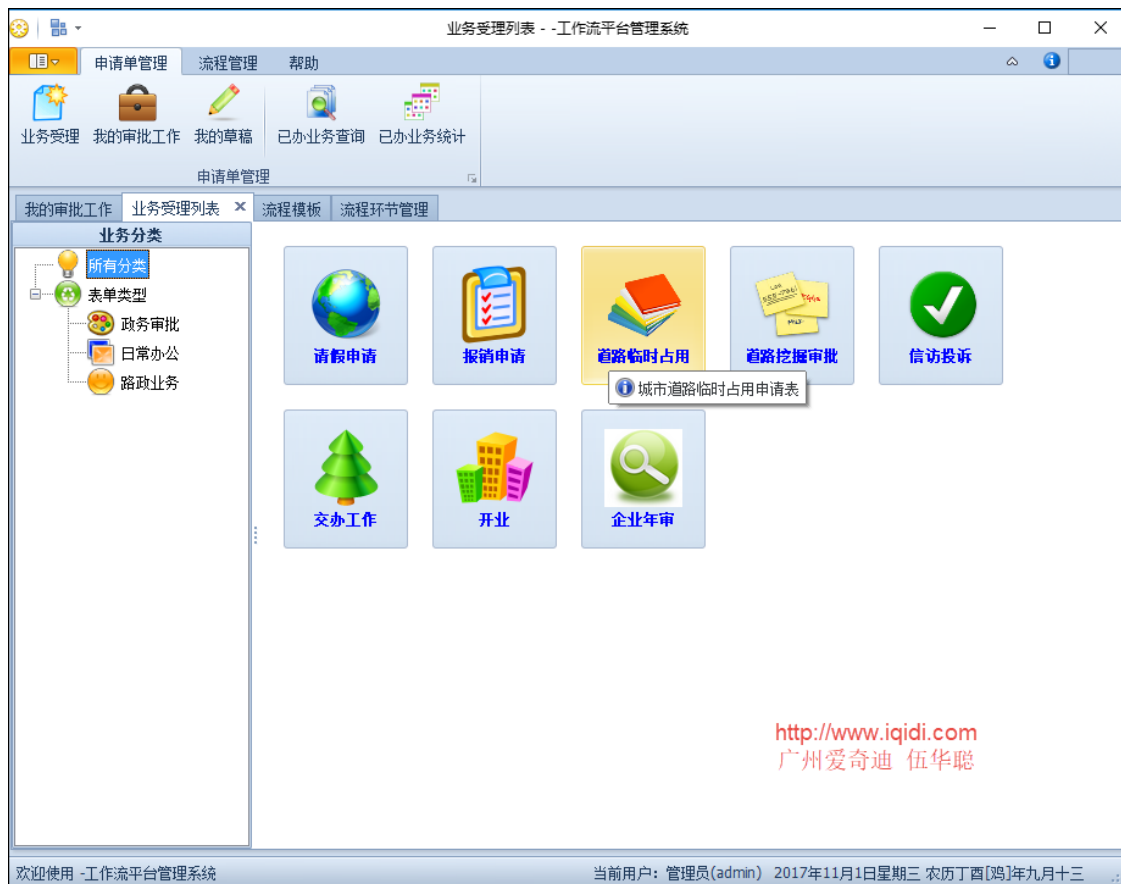
上面介绍了业务表单的填写、查看两个不同的窗口，我们在申请单的审批界面里面，统一显示不同的表单，以及创建不同的业务表单界面，这种动态的处理可以实现不同业务表单的创建及显示界面。

如我的审批工作中，表单的显示界面如下所示，查看具体表单后，可以动态展示不同的业务窗口界面。



另外我们在创建业务表单的时候, 根据数据库的配置信息, 动态展示所有可以展示的创建入口, 单击相关的按钮, 可以动态调用创建对应的表单界面。

创建流程业务表单的入口如下所示。



在我的审批工作界面，动态创建对应的查看表单窗体代码如下所示。

```
/// <summary>
/// 分页控件编辑项操作
/// </summary>
private void winGridViewPager1_OnEditSelected(object sender, EventArgs e)
{
    //获取记录ID和表单ID
    string ID = this.winGridViewPager1.gridView1.GetFocusedRowCellDisplayText("ID");
    string FormId = string.Concat(this.winGridViewPager1.gridView1.GetFocusedRowCellValue("FormId"));

    if (!string.IsNullOrEmpty(ID) && !string.IsNullOrEmpty(FormId))
    {
        var formInfo = BLLFactory<BLL.Form>.Instance.FindByID(FormId);
        if (formInfo != null && !string.IsNullOrEmpty(formInfo.ApplyWin2))
        {
            try
            {
                //根据配置的查看窗体，动态构建查看申请单对象
                FrmViewApply dlg = Assembly.GetExecutingAssembly().CreateInstance(formInfo.ApplyWin2) as FrmViewApply;
                if (dlg != null)
                {
                    dlg.ApplyId = ID;
                    dlg.OnDataSaved += new EventHandler(dlg_OnDataSaved);

                    if (DialogResult.OK == dlg.ShowDialog())
                    {
                        BindData();
                    }
                }
            }
            catch (Exception ex)
            {
                LogHelper.Error(ex);
                MessageDxUtil.ShowError(ex.Message);
            }
        }
    }
}
```

这个代码替代了需要手动创建不同对象的处理。

```
var dlg = new FrmViewAssignWork();
dlg.ApplyId = ID;
dlg.OnDataSaved += new EventHandler(dlg_OnDataSaved);

if (DialogResult.OK == dlg.ShowDialog())
{
    BindData();
}
```

同理，对于创建编辑界面，我们也可以同样的方法动态创建相关的编辑表单界面，如下代码所示。

```
/// <summary>
/// 单击某个动态生成的按钮，触发的申请表单创建界面
/// </summary>
1 个引用
void button_Click(object sender, EventArgs e)
{
    SimpleButton button = sender as SimpleButton;
    if (button != null)
    {
        var formId = button.Tag.ToString();
        var formInfo = BLLFactory<BLL.Form>.Instance.FindByID(formId);
        if (formInfo != null && !string.IsNullOrEmpty(formInfo.ApplyWin))
        {
            try
            {
                //动态构建创建申请单的界面窗体并赋值
                var dlg = Assembly.GetExecutingAssembly().CreateInstance(formInfo.ApplyWin) as FrmAddApply;
                dlg.FormID = button.Tag.ToString();
                dlg.ShowDialog();
            }
            catch (Exception ex)
            {
                LogHelper.Error(ex);
                MessageDxUtil.ShowError(ex.Message);
            }
        }
        else
        {
            MessageDxUtil.ShowTips(button.Text + "暂未开通");
        }
    }
}
```

<http://www.iqidi.com>
广州爱奇迪 伍华聪