安硕 AmarBPM 开发指南 V3.5

上海安硕信息技术股份有限公司

上海安硕信息技术股份有限公司授权:

THESE LISTED PEOPLE:

上海安硕信息技术股份有限公司

使用本文件

ARE AUTHORIZED BY AMARSOFT INFORMATION & TECHNOLOGY CO., LTD. TO USE THIS DOCUMENT

上海安硕信息技术股份有限公司 保密声明

本文件及其附件中的所有信息均应受到保密，受到法律的保护。本文件的信息仅限于指定使用者阅读。如果您并非指定的使用者，或者您只是本文件传递给指定使用者的转交人或机构，您不得使用、分发、复制本文件中的任何内容。如果本文件错误的转交给了您，请立即将其销毁并告知上海安硕信息技术股份有限公司。 联系电话：86-21-65106600，联系人：翟先生。谢谢您的合作。

AMARSOFT INFORMATION & TECHNOLOGY CO., LTD. Proprietary

The information contained in this document and its attachments (if any) is confidential and subject to legal privilege.The information is intended only for use of the individuals(s) to whom it is addressed. If you are not an intended recipient, or the agent or employee responsible to deliver it to an intended recipient, you are hereby notified that any use, distribution or copying of this communication is strictly prohibited.f you have received this document in error, please destroy it and immediately notify Amarsoft Information & Technology Co., Ltd. by calling 86-21-65106600, Mr. Zhai. Thank you.

上海安硕信息技术股份有限公司 特别申明

未经本公司书面同意，不得复制、翻译或摘录本文档。本公司明确依法保留有关版权的一切权利，保留更改权和解释权。

本文件为公司内部培训资料，限于在公司指定的培训场所使用。

文档信息

文档名称：	AmarBPM 开发指南
初稿作者：	张晴晴 qqzhang@amarsoft.com
初稿日期：	
内容概述：	

修订历史

版本	修订日期	修订人	复核日期	复核人	修改内容简述

发布历史

发布日期	提交人	质检人	发布说明

目　录

目　录 4
1. AmarBPM 原理 7
1. 流程页面 7
1.1.1 申请类型 7
1.1.2 按钮组 8
1.1.3 申请主树图 9
1.1.4 申请主树图节点 ..... 错误!未定义书签。
1.1.5 申请页面 10
1.1.6 审批类型 11
1.1.7 审批主树图 12
1.1.8 审批主树图节点 ..... 错误!未定义书签。
1.1.9 审批页面 13
1.2 流程模型 13
1.2.1 流程模型信息 13
1.2.2 流程模型节点信息 14
1.2.3 流程模型脚本 17
1.2.4 环境变量 18
1.2.5 外部函数 ..... 错误!未定义书签。
1.2.6 节点环境对象 20
1.2.7 意见权限查看控制 21
1.3 流程服务 23
1.3.1 流程实例表结构 23
1.3.2 流程对象表 23
1.3.3 流程任务表 25
1.3.4 流程意见表 27
1.3.5 流程服务 API 28
1.3.6 新增流程实例 29

1.3.7 取消流程 29
1.3.8 提交流程 29
1.3.9 撤回流程 29
1.3.10 退回任意阶段 29
1.3.11 其他流程操作 30
2. 常规流程开发 30
2.1 常规流程原理 30
2.1.1 流程模型 30
2.1.2 节点类型 30
2.1.3 流程实例 31
2.2 开发案例 31
2.2.1 新建流程 31
2.2.2 流程模型配置 35
2.2.3 配置申请类型及主树图 41
2.2.4 配置审批类型及主树图 44
2.2.5 编写申请、审批 JSP 页面 46
2.2.6 运行流程 49
2.2.7 高级特性开发 51
3. 并行流程开发 60
3.1 并行流程原理 60
3.1.1 并行流程模型 60
3.1.2 ActionScript 配置 61
3.1.3 PostScript 配置 61
3.1.4 并行流程实例 62
3.2 开发案例 62
3.2.1 新建流程 62
3.2.2 流程模型配置 62
3.2.3 配置申请类型及主树图 65
3.2.4 配置审批类型及主树图 66
3.2.5 运行流程 67

4. 待审会流程开发 68
4.1 待审会流程模型 69
4.2 待审会流程实例 69
4.3 开发案例 69
4.3.1 新建流程 69
4.3.2 配置流程模型 70
4.3.3 配置申请类型及主树图 72
4.3.4 配置审批类型及主树图 73
4.3.5 运行流程 74
5. 任务池流程开发 76
5.1 任务池流程原理 76
5.1.1 任务池流程模型 76
5.1.2 任务池流程实例 76
5.2 开发案例 77
5.2.1 新建流程 77
5.2.2 配置流程模型 77
5.2.3 配置申请类型及主树图 79
5.2.4 配置审批类型及主树图 80
5.2.5 运行流程 81
6. 分支任务池开发 82
6.1 分支任务池原理 82
6.1.1 分支任务池模型 83
6.1.2 分支任务池实例 83
6.2 开发案例 84
6.2.1 新建流程 84
6.2.2 配置流程模型 84
6.2.3 配置申请类型及主树图 85
6.2.4 配置审批类型及主树图 86
6.2.5 运行流程 87
7. 复杂模式流程参考 89

1. AmarBPM 原理

业务过程（Business process）是指为了实现某一业务、战略目标而组合在一起的一组过程或活动。而流程就是业务过程的自动化处理，在业务流程处理过程中，根据一组定义好的处理规则，可以把相关的信息、文档、任务或业务可以从一个参与者传递到另一个活动的参与者。

通常情况下 AmarBPM 是以人工在页面上提交的方式驱动流程运行的，所以 AmarBPM 的开发任务分为“页面”和“流程模型”两块。

1.1 流程页面

流程页面分为申请和审批页面，分别展示流程关联的业务对象在申请、审批阶段的视图。“申请类型”和“审批类型”分别描述流程的“申请要素”和“审批要素”，这些要素间接包括：页面要素（JSP 路径、按钮等）、流程关联业务对象描述（流程对象类型）等信息。当用户点击流程的申请、审批功能菜单，系统会根据菜单的 ApplyType、ApproveType 参数首先找到“申请类型”或“审批类型”，然后根据申请审批类型生成流程的申请、审批页面。

1.1.1 申请类型

申请类型描述流程的申请信息要素，主要包括对申请页面的描述和对业务对象的描述。申请类型存放在数据库的 Code_Library 表中，其中 CodeNo=’ApplyType’。其字段含义如下：

序号	属性	属性含义	说明
1	CodeNo	代码编号	始终为 ApplyType
2	ItemNo	申请类型编号	申请类型编号
3	ItemName	根节点名称	定义申请的名称（树图根节点名称和浏览器标题）
4	SortNo	排序编号	顺序

5	Isinuse	是否使用	该申请是否使用
6	ItemDescribe	树图代码	在 applyMain.jsp 引用的树图代码
7	ItemAttribute	流程对象类型	即 FLOW_OBJECT 的 ObjectType
8	Attribute2	审批流程基础模型编号 / 流程模型编号	即 Flow_Ctalog 表 BaseFlowNo / FlowNo
9	Attribute5	按钮组	申请使用的 ButtonSet

其中：

- ItemDescribe 是申请类型关联的“主树图”，代表了页面上将要展示的树图信息。
- ItemAttribute 是流程对象类型，是对业务对象在 AmarBPM 环境下的唯一标记，该字符串将会被记录到 Flow_Object 表的 ObjectType 字段，来唯一标记该业务对象（*关于 ObjectType 会在流程模型章节详细说明*）。
- Attribute2 是从申请类型到流程的反向关联。如果使用方法 `flowService.initFlow(applyType, objectNo, userID, tx)`进行流程初始化，会根据 attribute2 上配置当做 baseFlowNo 到 Flow_Catalog 表中找 isInUse 为 1 的流程(如果有多个，取 version 最大的那个流程)，如果没有找到流程，会把 attribute2 的值当做 FlowNo 到 Flow_Catalog 表找 isInUse 为 1 的流程，进行流程初始化。
- Attribute5 是关联主树图的按钮组，描述树图页面下的按钮信息。

1.1.2 按钮组

按钮组是对一组按钮的定义，存放在 Code_Catalog 和 Code_Library 中，其中 Code_Catalog 中存放按钮组的整体信息（CodeType2='按钮组'），CodeNo 为按钮组的唯一编号。Code_Library 存放按钮的详细定义信息，字段含义如下：

序号	属性	属性含义	设置方式
1	CodeNo	代码编号	设为：ButtonSetName
2	ItemNo	代码细项编号	自定义编码
3	ItemName	按钮文字	设定按钮上显示的文字
4	SortNo	排序编号	按钮显示的顺序

5	Isinuse	是否使用	该按钮是否使用
6	ItemDescribe	说明文字	对于该按钮的说明
7	RelativeCode	触发事件	对应页面中的 javascript 事件
8	Attribute2	类型	一般设定为：“Button”

- 申请类型的 Attribute5 即映射 Code_Catalog.CodeNo，按钮组决定了申请类型对应的申请主树图可以使用哪些按钮。

1.1.3 申请主树图

申请主树图是典型的 A3Web 树图定义，首先由申请类型. ItemDescribe 映射到 Code_Catalog.CodeNo 找到主树图，然后由 CodeNo 找到 Code_library 的主树图节点定义。申请主树图默认只有 CodeNo、CodeName、SortNo 有意义，分别代表主树图代码、主树图名称和排序号。

申请主树图节点存放在 Code_Library 表，申请主树图节点决定了申请页面的信息要素，其字段含义如下：

序号	属性	属性含义	设置方式
1	CodeNo	主树图代码	主树图代码
2	ItemNo	节点编号	和 Flow_Model.PhaseType 对应
3	ItemName	节点名称	定义申请的名称(树图根节点名称和浏览器标题)
4	SortNo	排序编号	顺序号，通常与 ItemNo 保持一致
5	Isinuse	是否使用	该树图是否使用
6	ItemDescribe	列表要显示的按钮	对应按钮组中的按钮名称
7	ItemAttribute	以什么视图查看对象详情	默认 001
8	Attribute1	申请列表显示模板的 Where 条件 1	会拼接到申请列表显示模板的 where 条件里
9	Attribute2	申请列表显示模板的 Where 条件 2	会拼接到申请列表显示模板的 where 条件里
10	Attribute4	申请列表的显示模板编号	申请列表显示模板编号
11	Attribute5	申请列表视图路径	申请列表视图路径

12	RelativeCode	授信申请详情 Tab	ALS 中用到

其中：

- ItemNo 为节点编号，映射 Flow_Model.PhaseType，映射函数在流程模型章节详细说明。
- ItemDescribe 按钮组，即该节点关联的 JSP 页面上的按钮，多个按钮代码以逗号分隔，这些按钮必需来自申请类型配置的按钮组。并且要求本节点关联的 JSP 页面中必需包含按钮对应的 JS 方法，否则点击按钮将会报错。
- Attribute4 显示模板编号，即该节点关联的 JSP 的 List 页面所使用的显示模板。申请页面的显示模板一般由主 JBO 对象 Flow_Object 对象关联业务对象组成，该显示模板必需包含 Flow_Object 的一些属性，以便配合 Attribute1 和 Attribute2 的条件完成对流程节点阶段的过滤。这些必需包含字段包括：OBJECTTYPE、OBJECTNO、SERIALNO、PHASETYPE、PHASENO、APPLYTYPE、FLOWNO、ORGID、USERID、INPUTDATE、FLOWSTATE。
- Attribute5 为 JSP 页面路径，即该节点关联的 JSP 页面路径。由于历史原因，A3Web3.5 之前路径后面必需用@符号跟上显示模板 ID。3.5 之后不必须。
- Attribute1 和 Attribute2 为显示模板过滤条件，会拼接到显示模板的 where 条件中。类似 and O.ObjectType = '#ObjectType' and O.PhaseType='#PhaseType' and UserID='#UserID' 其中#ObjectType 会被替换成申请类型的流程对象类型、#PhaseType 会被替换成树图节点编号、#UserID 被被替换成当前登录用户。这里的变量自动替换与流程配置的 AmarScript 中的自动替换类似。
- RelativeCode，在 ALS 中仅对授信申请有效，控制申请详情页面 Tab 的展现，Code_Library.ItemNo(其中 CodeNo= TaskViewTab)。

1.1.4 申请页面

所有流程的入口菜单都指向/Common/WorkFlow/ApplyMain.jsp，并且在菜单 Param 中加入 ApplyType=XXXApplyType 的参数，其中 XXXApplyType 为申请类型。

在 ApplyMain.jsp 中系统会获取 XXXApplyType 关联的主树图，将主树图信息加载到当前页面中。其中主树图节点的 Attribute5 配置了树图节点对应的 JSP 路径，该 JSP 称之为“流程申请页面”，在“流程申请页面中”需要定义按钮的 JS 方法，以便定义在主树图节点的 ItemDescribe 中的按钮的事件可以被正确响应。

理论上多个主树图节点也可配置不同的申请页面，但通常情况下我们会用同一个 JSP 页面，把多个节点用到的按钮方法都写在同一个 JSP 中。

1.1.5 审批类型

审批类型描述流程的审批信息要素，主要包括对申请页面的描述和对业务对象的描述。申请类型存放在数据库的 Code_Library 表中，其中 CodeNo='ApproveType'。其字段含义如下：

序号	属性	属性含义	设置方式
1	CodeNo	代码编号	ApproveType
2	ItemNo	审批类型编号	审批类型编号
3	ItemName	审批类型名称	定义审批的名称（树图根节点名称和浏览器标题）
4	SorNo	排序编号	顺序
5	Isinuse	是否使用	该审批是否使用
6	ItemDescribe	树图代码	主树图代码
7	ItemAttribute	流程对象类型	即 FLOW_OBJECT 的 ObjectType
8	Attribute2	审批流程模型编号	和对应的审批流程模型编号一致
9	Attribute5	按钮组	审批使用的 ButtonSet
10	Attribute8	审批列表页面路径	审批列表视图路径(A3WEB3.5 之前)

- ItemDescribe 是审批类型关联的“主树图”，代表了页面上将要展示的树图信息。
- ItemAttribute 是流程对象类型，必需要和申请类型的 ItemAttribute 一致。
- Attribute2 是从申请类型到流程的反向关联。如果需要关联多个流程需要用”,”分割。审批页面会以此字段值来计算一个或多个流程的待办工作和已完成工作。
- Attribute5 是关联主树图的按钮组，描述树图页面下的按钮信息。
- Attribute8 是审批列表视图路径（在 A3WEB3.5 之前），A3WEB3.5 之后的审批列表视图路径配置在审批主树图的 Attribute5 中。

1.1.6 审批主树图

审批主树图是典型的 A3Web 树图定义，首先由审批类型. ItemDescribe 映射到 Code_Catalog.CodeNo 找到主树图，然后由 CodeNo 找到 Code_library 的主树图节点定义。审批主树图默认只有 CodeNo、CodeName、SortNo 有意义，分别代表主树图代码、主树图名称和排序号。

审批主树图节点存放在 Code_Library 表，申请主树图节点决定了申请页面的信息要素，其字段含义如下：

序号	属性	属性含义	设置方式
1	CodeNo	代码编号	主树图代码
2	ItemNo	节点编号	Y/N
3	ItemName	节点名称	当前工作/已完成工作
4	SortNo	排序编号	顺序
5	Isinuse	是否使用	该树图是否使用
7	ItemAttribute	以什么视图查看对象详情	默认 001
8	Attribute1	申请列表显示模板的 Where 条件 1	会拼接到申请列表显示模板的 where 条件里
9	Attribute2	申请列表显示模板的 Where 条件 2	会拼接到申请列表显示模板的 where 条件里
10	Attribute4	申请列表的显示模板编号	申请列表显示模板编号
11	Attribute5	申请列表视图路径	申请列表视图路径

其中：

- ItemNo 为节点编号，Y 代表已完成的工作，N 代表未完成的工作。
- Attribute4 显示模板编号，即该节点关联的 JSP 的 List 页面所使用的显示模板。
- Attribute5 为 JSP 页面路径(A3Web3.5 之后)，在 A3Web3.5 之前审批页面的 JSP 路径是放在审批类型的 Attribute8 中的。
- Attribute1 和 Attribute2 为显示模板过滤条件，会拼接到显示模板的 where 条件中。

注意：审批树图节点没有“按钮组”的定义，审批页面的按钮是定义在 Flow_Model 的 Attribute1 和 Attribute2 中的，因为审批页面按钮时根据流程所进行的当前进度决定的，所以审批按钮配置在流程模型中。

1.1.7 审批页面

所有流程的入口菜单都指向/Common/WorkFlow/ApproveMain.jsp，并且在菜单 Param 中加入 ApproveType=XXXApproveType 的参数，其中 XXXApproveType 为审批类型。

在 ApproveMain.jsp 中系统会获取 XXXApproveType 关联的主树图，将主树图信息加载到当前页面中。其中主树图节点的 Attribute5 配置了树图节点对应的 JSP 路径，该 JSP 称之为“流程审批页面”，在“流程审批页面中”需要定义按钮的 JS 方法，以便定义 Flow_Model 中的 Attribute1、Attribute2 按钮的事件可以被正确相应。

理论上多个主树图节点也可配置不同的审批页面，但通常情况下我们会用同一个 JSP 页面，把多个节点用到的按钮方法都写在同一个 JSP 中。

1.2 流程模型

流程模型描述流程的工作流转模型，存放在 Flow_Catalog 和 Flow_Model 表中。Flow_Catalog 存放流程基本信息，FlowModel 存放流程节点流转信息。另外还有一张表 Flow_Swimlane 存放 ADE 设计时的流程泳道信息，没有运行时含义。

流程模型被实例化数据放在 Flow_Object 表中，流程每走一步会在 Flow_Task 表生成一条代表流程节点实例的任务信息，而流程意见记录存放在 Flow_Opinion 表中。

1.2.1 流程模型信息

流程基本信息存放在 Flow_Catalog 中，描述流程体系信息，字段含义如下：

序号	属性	属性含义	描述
1.	FLOWNO	流程编号	唯一的流程编号
2.	FLOWNAME	流程名称	
3.	FLOWTYPE	流程类型	暂未使用
4.	GROUPTITLES	流程图分组标题	不在使用
5.	VIEWFILELENGTH	流程图描述长度	不在使用
6.	FLOWDESCRIBE	流程描述	
7.	INITPHASE	初始节点	流程实例默认从哪个阶段开始走

8.	AAENABLED	是否启用授权系统	不在使用
9.	AAPOLICY	授权方案	不在使用
10.	METAFLOWNO	元版本号	不在使用
11.	ISINUSE	是否有效	
12.	VERSION	版本	
13.	BASEFLOWNO	基础流程号	

- INITPHASE 初始节点，即流程从哪个节点开始走，流程实例被初始化后所处的默认开始流转节点。

1.2.2 流程模型节点信息

流程模型的节点信息存放在 Flow_Model 中，包括节点基本信息、流程走到该节点如何被初始化，该节点到下一节点的提交人、如何被流转等信息，具体字段含义如下：

序号	属性	属性含义	描述
1.	FLOWNO	流程编号	流程编号
2.	PHASENO	节点号	流程节点编号
3.	PHASETYPE	阶段类型	流程实例所归属到申请页面的树图节点编号
4.	PHASENAME	节点名称	
5.	PHASEDESCRIBE	节点描述	没加 NodeType 字段前存放流程节点类型
6.	PHASEATTRIBUTE	节点属性	
7.	PRESCRIPT	前沿执行SCRIPT	流程实例走到该节点时首先执行的脚本
8.	INITSCRIPT	相关承办人SCRIPT	承办人数组，数组个数决定了当前节点被实例化的个数
9.	CHOICEDESCRIBE	意见描述	
10.	CHOICESCRIPT	意见生成SCRIPT	用什么脚本来生成意见
11.	ACTIONDESCRIBE	动作描述	
12.	ACTIONSCRIPT	动作生成SCRIPT	用什么脚本来生成提交人信息
13.	POSTSCRIPT	后续节点SCRIPIT	用什么脚本来决定下一节点

14.	ATTRIBUTE1	审批页面当前工作按钮	逗号分隔，按钮代码必需在审批类型的按钮组中
15.	ATTRIBUTE2	审批页面已完成工作按钮	逗号分隔，按钮代码必需在审批类型的按钮组中
16.	ATTRIBUTE3	意见查看权限方式	none_except（白名单）all_except（黑名单）2 种，仅对 Attribute4 起作用
17.	ATTRIBUTE4	意见权限节点	格式 FlowName.PhaseNo,逗号分隔
18.	ATTRIBUTE5	意见查看特权角色	哪些角色直接可以看本节点意见
19.	ATTRIBUTE6	仅查看自己签署意见所对应的节点	暂未使用
20.	ATTRIBUTE7	ATTRIBUTE7	暂未使用
21.	ATTRIBUTE8	ATTRIBUTE8	暂未使用
22.	ATTRIBUTE9	ATTRIBUTE9	暂未使用
23.	ATTRIBUTE10	ATTRIBUTE10	暂未使用
24.	AAENABLED	是否启用授权	不在使用
25.	AAPOINTINITSCRIPT	授权点初始化SCRIPT	不在使用
26.	AAPOINTCOMP	授权点组件	不在使用
27.	AAPOINTCOMPURL	授权点组件URL	不在使用
28.	STANDARDTIME1	(NEW)审批时间标准1(分钟)	不在使用
29.	STANDARDTIME2	(NEW)审批时间标准2(分钟)	不在使用
30.	COSTLOB	(NEW)审批成本归属	不在使用
31.	STRIPS	(NEW)节点STRIP[逗号分隔]	不在使用
32.	CHECKLIST	(NEW)节点CHECKLIST	不在使用
33.	DECISIONSCRIPT	(NEW)自动审批规则	不在使用
34.	RISKSCANRULE	(NEW)风险探测规则组	不在使用
35.	BUTTONSET2	(NEW)STRIP按钮	不在使用

36.	INPUTUSER	录入人	
37.	INPUTTIME	录入时间	
38.	UPDATEUSER	更新人	
39.	UPDATETIME	更新时间	
40.	DISTRIBUTERULE	分发方式	不在使用
41.	ID	图形节点编号	ADE 工具使用
42.	TYPE	图形节点类型	ADE 工具使用
43.	NAME	图形节点名称	ADE 工具使用
44.	XCOORDINATE	图形节点X坐标	ADE 工具使用
45.	YCOORDINATE	图形节点Y坐标	ADE 工具使用
46.	SWIMLANE	泳道编号	ADE 工具使用
47.	WIDTH	图形节点宽度	ADE 工具使用
48.	HEIGHT	图形节点高度	ADE 工具使用
49.	VERSION	版本	
50.	NODETYPE	流程节点类型	
51.	FLOWPHASECONTEXT	流程节点上下文	上下文变量定义 JSON 串

- PhaseType 阶段类型即申请页面所属树图节点，映射申请主树图节点的 ItemNo。当流程实例走到该节点时，会被展示到申请主树图对应的树图节点下，而 PhaseType 就是映射主树图编号。
- PRESCRIPT 前沿脚本是流程走到该节点时所执行的前沿脚本，该脚本会在本节点被实例化时执行（通常是上个节点提交时）。
- INITSCRIPT 承办人初始化脚本，用本脚本获取本节点的承办人数组。开始节点为 {#UserID} 即当前登录人员，其他普通节点为 toStringArray('#PhaseAction',",," ",1)，即从上一节点的 PhaseAction 中提取出承办人的 UserID 数组，有几个 UserID 就生成几个任务。
- CHOICESCRIPT 意见生成脚本将产生提交页面的“意见项”，脚本返回值是一个字符串数组，代表最终显示的若干意见。
- ACTIONSCRIPT 动作生成脚本将产生提交页面的“提交人”，返回值是一个字符串数组，

代表最终显示的若干提交人信息。系统自带的类及方法中有“当前机构角色人员列表”、“所有机构角色人员列表”等，用来返回对应的操作人员。

- POSTSCRIPT 后续脚本将产生提交页面的“下一阶段”，即本节点的下一流转节点，返回值是一个字符串，映射下一节点的 PhaseNo。

- NODETYPE 节点类型，目前系统有如下几种节点类型：

序号	类型	含义	描述
1.	START	开始节点	同时该节点应该是流程的 InitPhase
2.	TASK	一般节点	
3.	SUPPLY	退回补充资料节点	A3Web3.0 之后不再使用
4.	END	流程结束节点	批准或否决
5.	FORK	并行分支节点	
6.	JOIN	并行汇聚节点	
7.	DISTRIBUTE	待审会分发节点	
8.	MEETING	待审会汇总节点	
9.	POOL	任务池节点	
10.	FORK&POOL	并行任务池节点	该节点既是任务池又是并行分支节点

1.2.3 流程模型脚本

流程模型中的脚本（包括节点初始化脚本，承办人初始化脚本，意见脚本，动作脚本，后续事件脚本）是 AmarScript 脚本，在脚本中可调用外部函数（Class_Method），详细的 AmarScript 语法参见《程序员手册_[Amarscript].doc》。

流程模型中的脚本的解释执行的过程：

1. 如果流程模型阶段上配置有环境对象(见 1.2.5)，把配置信息中#ObjectNo,#ObjectType 等环境变量(见 1.2.4)自动替换为对应的数据，解析配置信息，得到对象；

2. 把流程模型脚本中#ObjectNo,#ObjectType 等环境变量(见 1.2.4)替换为对应的数据；

3. 根据第 1 步中得到的对象,把流程模型脚本中${obj.attr}自动替换为对应的数据；

4. 经过以上的变量替换和对象属性替换，得到可解释执行的 AmarScript 脚本。在遇到类方法时，执行对应的类方法得到对应的数据，最后根据 IF 判断以及其它运算得到最终结果。

意见脚本、动作脚本、后续脚本将产生提交界面的“意见”、“提交人”和“一下阶段”，如下图：

1.2.4 环境变量

流程中的流程环境变量是可以在流程脚本中直接使用的变量，如#ObjectNo、#UserID 等，这些变量通过井号加变量名引用，在脚本被执行时会被替换成运行时实参。流程环境变量实际是从当前流程实例和任务的字段中获取的，包括：

序号	变量名	取值	描述

1.	#ObjectNo	当前流程实例.ObjectNo	
2.	#ObjectType	当前流程实例.ObjectType	
3.	#ApplyType	当前流程实例.ApplyType	
4.	#PhaseName	当前任务.PhaseName	
5.	#PhaseNo	当前任务.PhaseNo	
6.	#PhaseType	当前任务.PhaseType	
7.	#FlowNo	当前任务.FlowNo	
8.	#FlowName	上个任务.FlowName	
9.	#LastBeginTime	上个任务.BeginTime	
10.	#LastEndTime	上个任务.EndTime	
11.	#LastFlowName	上个任务.FlowName	
12.	#LastFlowNo	上个任务.FlowNo	
13.	#LastOrgID	上个任务.OrgID	
14.	#LastOrgName	上个任务.OrgName	
15.	#LastPhaseAction	上个任务.PhaseAction	
16.	#LastPhaseChoice	上个任务.PhaseChoice	
17.	#LastPhaseName	上个任务.PhaseName	
18.	#LastPhaseNo	上个任务.PhaseNo	
19.	#LastPhaseOpinion1	上个任务.PhaseOpinion1	
20.	#LastPhaseType	上个任务.PhaseType	
21.	#LastUserID	上个任务.UserID	
22.	#LastUserName	上个任务.UserName	
23.	#OrgID	上个任务.OrgID	
24.	#OrgName	上个任务.OrgName	
25.	#PhaseAction	上个任务.PhaseAction	
26.	#PhaseOpinion1	上个任务.PhaseOpinion1	
27.	#UserID	上个任务.UserID	
28.	#UserName	上个任务.UserName	

1.2.5 节点环境对象

在流程脚本中外部函数的每次调用都会将外部函数重新执行一遍，这样可能导致性能下降，为了解决这个问题，A3Web3.1 引入了环境对象概念。即可以预先在 Flow_Model 的 FlowPhaseContext 字段使用 JSON 定义一些对象，这些对象会被一次性加载，之后便可以在流程脚本中多次使用。环境对象只对本节点可见。

1.2.5.1.1 对象配置

环境对象分 2 种，分别是 BizObject 类型和 POJO 类型。其 BizObject 类型为使用 JBOQL 语句查出来的 JBO 对象，定义 BizObject 有以下几个维度：

序号	维度	JSON 字段	描述
1.	名称	Name	环境对象名称
2.	类型	Type	固定为 BizObject
3.	JBO 类	MainClass	从哪个 JBO 类查数据
4.	查询语句	Query	JBOQL

定义 POJO 类型对象有以下一个维度：

序号	维度	JSON 字段	描述
1.	名称	Name	环境对象名称
2.	类型	Type	固定为 POJO
3.	POJO 类	Class	从哪个 POJO 类获取数据
4.	构造参数	Paras	Java 类的构造参数，逗号分隔，不包括最后一个参数 jboTrans

例如定义一个名称为 FO 的环境对象，使用 JBOQL 查询结果，同时定义一个名为 FO2 的环境对象，其值取自 POJO 类 demo.POJO，那么其配置的 JSON 串为：

[{"Name":"FO","Type":"BizObject","MainClass":"jbo.sys.FLOW_OBJECT","Query":"select * from O where objectType='#ObjectType' and objectNo='#ObjectNo'"},{"Name":"FO2","Type":"POJO","Class":"demo.POJO","Paras":"#ObjectType,#ObjectNo"}]

直接编写 JSON 串是比较复杂的，我们一般使用 ADE 工具来配置环境对象（以后章节会介绍）。

1.2.5.1.2 环境对象使用

增加上下文配置的目的是可以减少类及方法的使用，使用**${对象.属性}**得到上下文配置中的对象的属性值。对于 jbo 对象，取得 jbo 对象中对应属性的值；对于普通 java 对象，优先使用对应属性的 get 方法的返回值，如果没有找到对应属性的 get 方法，则使用 getAttribute(属性名)方法的返回值。例如${FO.UserID}和${FO2.BusinessSum} 。

1.2.6 外部函数

动作脚本常用外部函数有：

1. !审批流程.所有机构角色人员列表 (#RoleID)，返回所有机构拥有#RoleID 角色的所有人员字符数组，返回字符串格式为：*UserID 空格 UserName 空格 RoleID*。
2. !审批流程.当前机构角色人员列表(#OrgID, #UserID,#RoleID)，返回本机构（#OrgID）拥有#RoleID 角色的所有用户，但不包括自己（#UserID）。
3. !审批流程. 所有机构角色人员列表并行(#PhaseNo,#RoleID)，返回字符串数组，其中字符串格式为：*PhaseNo|UserID 空格 UserName 空格 RoleID 空格 RoleName* 其中 PhaseNo 用来标记一个动作（提交人）信息属于哪个分支流向。
4. !审批流程.当前机构角色人员列表并行 JBOQL(#PhaseNo,#OrgID,#UserID,#RoleID)，返回本机构（#OrgID）拥有#RoleID 的所有用户，但不包括自己（#UserID），并在返回字串前加上#PhaseNo|。

以上所有方法都有对应的 JBOQL 方法，使用 JBOQL 实现，解决表名大小写敏感等问题。

1.2.7 意见权限查看控制

查看意见的 JSP 为 ViewFlowOpinions.jsp，其在 Flow_Model 中的控制字段为：

1.	ATTRIBUTE3	意见查看权限方式	none_except（白名单）all_except（黑

			名单）2 种，仅对 Attribute4 起作用
2.	ATTRIBUTE4	意见权限节点	格式 FlowName.PhaseNo，逗号分隔
3.	ATTRIBUTE5	意见查看特权角色	哪些角色直接可以看本节点意见
4.	ATTRIBUTE6	仅查看自己签署意见所对应的节点	暂未使用

1.2.7.1 权限节点规则

权限节点规则指 Attribute3+Attribute4 构成的规则，其中 Attribute3（意见查看权限方式）分白名单(none_except)和黑名单(all_except)2 种，仅对 Attribute4（意见查看节点生效）。Attribute4 配置节点格式为“FlowName.PhaseNo”多个节点以逗号分隔。配置举例：在 phaseno=0010 的节点上配置意见查看权限：

意见查看权限方式 Attribute3	意见权限节点 Attribute4	含义
none_except	TestCommonFlow.0020,TestCommomFlow.0030	0010 节点任务被签署的意见，只能在 0020 和 0030 节点任务中被查看到。
all_except	TestCommonFlow.0020,TestCommomFlow.0030	0010 节点任务被签署的意见，在所有节点任务中都能被查看到，除了 0020 和 0030 节点任务。

1.2.7.2 特权角色规则

特权角色规则指 Attibute5 的规则，Attribute5 配置了以逗号分隔的角色 ID，凡是配置在该字段的角色都可以查看本节点任务签署的意见。该规则和“权限节点规则”是“或的关系”，满足一个即可。

1.2.7.3 意见查看逻辑

下面具体介绍意见查看权限的逻辑：

1、 取出当前流程的所有意见，循环每条意见，找到意见对应的 Flow_Task，继而找到对应 Flow_Model 的“权限节点规则”，判断该“权限节点规则”是否适应于“当前任务”，如果适应，则可展示，否则进入步骤 2。

2、 找到意见对应的 Flow_Task，继而找到对应 Flow_Model 的“特权角色规则”，判断当前登录用户的任一 RoleID 是否在 Attribute5 中，如果在，则可展示，否则不展示该条意见。

1.3 流程服务

AmarBPM 提供的流程服务 API 包括新增流程实例、取消流程、提交流程、退回流程等功能，流程服务类为 FlowService。本节介绍流程服务 API 和底层表结构。

1.3.1 流程实例表结构

流程实例存放在数据库的 Flow_Object 表中，该表记录流程实例的当前运行状态，主要字段有 SerialNo,ObjectType,ObjectNo，其中 SerialNo 为流程实例的技术主键不含业务含义，ObjectNo 为流程关联的业务对象的业务主键，而 ObjectType 标记业务对象的类型。ObjectType 是一个字符串，配置在申请类型的 ItemAttribute，该值一般维护在代码表中（CodeNo=FlowObject）。

流程实例流转信息存放在 Flow_Task 表中，该表记录流程实例的流转节点信息，每当流程被提交一次，系统就会往该表插入一条数据，来记录流程的流转情况。该表主要字段有 SerialNo、ObjectType、ObjectNo、RelativeSerialNo，其中 RelativeSerialNo 记录上一个流转节点的 SerialNo。

流程实例流转过程中的签署意见存放在 Flow_Opinion 表中，主要字段有 SerialNo、OpinionNo、PhaseOpinion。其中 SerialNo 映射 Flow_Task.SerialNo，OpinionNo 是技术主键，PhaseOpinion 是签署意见。

1.3.2 流程对象表

流程对象表（即流程实例表）Flow_Object 存放流程实例的当前状态，字段意义如下：

序号	类型	含义	描述
1.	OBJECTTYPE	对象类型	描述流程关联的业务对象类型标记
2.	OBJECTNO	对象编号	业务实体主键
3.	PHASETYPE	阶段类型	流程当前节点所属 Flow_Model 中的

			PhaseType
4.	APPLYTYPE	申请类型	
5.	FLOWNO	当前流程号	
6.	FLOWNAME	当前流程名称	
7.	PHASENO	当前阶段号	流程当前节点的 phaseno
8.	PHASENAME	当前阶段名称	
9.	OBJDESCRIBE	对象描述	
10.	OBJATTRIBUTE1	对象属性1	
11.	OBJATTRIBUTE2	对象属性2	
12.	OBJATTRIBUTE3	对象属性3	
13.	OBJATTRIBUTE4	对象属性4	
14.	OBJATTRIBUTE5	对象属性5	
15.	ORGID	创建机构	
16.	ORGNAME	创建机构名称	
17.	USERID	创建人	流程的创建人
18.	USERNAME	创建人姓名	
19.	INPUTDATE	创建日期	
20.	ARCHIVETIME	归档时间	
21.	PROCESSINSTNO	流程实例编号	外置流程引擎实例号
22.	PROCESSTASKNO	流程任务编号	外置流程引擎任务号
23.	SERIALNO	流程对象流水号	目前没有作为主键，以 objectType objectNo 作为联合主键
24.	FLOWSTATE	流程状态	未使用
25.	ARCHIVE	归档标识	
26.	VERSION	版本	
27.	BASEFLOWNO	流程号	

- PhaseType 阶段类型，即流程当前节点所属 FlowModel 的 PhaseType，描述流程当前节点的阶段，该字段值配合申请主树图节点的 where 条件（类似 PhaseType='#PhaseType'）来划分流程实例的不同阶段（在页面上体现为不同的主树图节点）。
- APPLYTYPE 申请类型，主树图节点的 where 条件会用到。

- FLOWSTATE 未使用(以前用于外置流程反向设置的状态)。

1.3.3 流程任务表

流程任务表（即流程实例流转信息表）存放在 Flow_Task 表中，该表记录流程实例的流转节点信息，每当流程被提交一次，系统就会往该表插入一条数据，具体字段含义如下：

序号	字段	含义	描述
1.	SERIALNO	流水号	流转节点技术主键
2.	OBJECTNO	对象编号	
3.	OBJECTTYPE	对象类型	
4.	RELATIVESERIALNO	上一流水号	上一个任务的 Serialno
5.	FLOWNO	流程编号	
6.	FLOWNAME	流程名称	
7.	PHASENO	阶段编号	
8.	PHASENAME	阶段名称	
9.	PHASETYPE	阶段类型	
10.	APPLYTYPE	申请类型	
11.	USERID	承办人编号	该节点工作由谁承办（审批或申请）
12.	USERNAME	承办人姓名	
13.	ORGID	承办机构编号	
14.	ORGNAME	承办机构名称	
15.	BEGINTIME	开始执行时间	任务生成时间
16.	ENDTIME	完成执行时间	为 null 或空说明节点工作没结束
17.	PHASECHOICE	阶段意见	不在使用
18.	PHASEACTION	阶段动作	提交页面上选择的动作
19.	PHASEOPINION	意见详情	不在使用
20.	PHASEOPINION1	意见详情1	提交页面上选择的意见
21.	PHASEOPINION2	意见详情2	不在使用
22.	PHASEOPINION3	意见详情3	不在使用

23.	PHASEOPINION4	意见详情4	不在使用
24.	CHECKLISTRESULT	[NEW]检查清单结果	不在使用
25.	AUTODECISION	[NEW]自动审批判断结果	不在使用
26.	RISKSCANRESULT	[NEW]风险探测结果	不在使用
27.	STANDARDTIME1	[NEW]标准审批用时	不在使用
28.	STANDARDTIME2	[NEW]最长审批用时	不在使用
29.	COSTLOB	[NEW]审批成本归属	不在使用
30.	CLIENTX	图元X坐标	不在使用
31.	CLIENTY	图元Y坐标	不在使用
32.	HEIGTH	图元高度	不在使用
33.	GROUPINFO	集团信息	不在使用
34.	WIDTH	图元宽度	不在使用
35.	PROCESSINSTNO	流程实例编号	外置流程引擎流程号
36.	PROCESSTASKNO	流程任务编号	外置流程引擎任务号
37.	RELATIVEOBJECTNO	流程关联对象流水号	以前用于外置流程引擎，不再使用
38.	FLOWSTATE	流程状态	见下面的说明
39.	VERSION	版本	
40.	BASEFLOWNO	流程号	
41.	TASKSTATE	任务状态	0 未提交，1 提交。作用不大，仍以 endTime 做为流程结束的判断
42.	FORKNO	并行分支编号	分支任务标记号
43.	ALLFORKNO	所有的并行分支编号	同批其他分支任务标记号，逗号分隔
44.	FORKSTATE	并行状态	记录分支任务状态
45.	ASSIGNEDTASKNO	指定任务编号	从哪个任务被退回，提交后直接返回
46.	RELANOTICENO	关联通知编号	暂未使用

- FlowState 代表流程状态（标记值作用会接下来章节被介绍），其值为：

  ANYBACK　退回任意阶段

  BACK　退回上一步

INPOOL 任务池中

NOTSELECT 任务池--未获取

SELECT 任务池--已获取

SUPPLY 退回补充资料

TAKEBACK 撤回

DISTRIBUTE 贷审会秘书分发

COLLECT 贷审会秘书汇总

MEETING 会议中

UNVOTED 贷审会委员未投票

VOTED 贷审会委员已投票

- ForkState 为并行分支任务特有（并行流程开发章节会详细介绍），其值为：

  INFORK 分支中

  WAIT 有其他分支未到达，等待

  JOIN 所有分支都已到达，合并

- ENDTIME 流程的结束时间，如果这个字段被设值上日期，那么表示这个节点工作已经完成。审批主树图节点就是利用这个字段区分当前任务和已完成任务的（例如当前任务的 where 条件 2 为 EndTime is null or EndTime =’’）。

1.3.4 流程意见表

流程实例流转过程中的签署意见存放在 Flow_Opinion 表中，具体字段含义如下：

序号	字段	含义	描述
1.	SERIALNO	流程任务流水号	
2.	OPINIONNO	意见序号	
3.	OBJECTTYPE	对象类型	
4.	OBJECTNO	对象编号	
5.	CUSTOMERID	客户编号	
6.	CUSTOMERNAME	客户名称	
7.	BUSINESSCURRENCY	币种	
8.	BUSINESSSUM	金额	

9.	TERMYEAR	期限(年)	
10.	TERMMONTH	期限(月)	
11.	TERMDAY	期限(日)	
12.	BASERATETYPE	基准利率类型	
13.	RATEFLOATTYPE	利率浮动方式	
14.	RATEFLOAT	利率浮动	
15.	BAILCURRENCY	保证金币种	
16.	BUSINESSRATE	执行利率	
17.	BAILRATIO	保证金比率	
18.	BAILSUM	保证金金额	
19.	PDGRATIO	手续费比率	
20.	PDGSUM	手续费金额	
21.	BASERATE	基准利率	
22.	PHASECHOICE	阶段意见	
23.	PHASEOPINION	意见详情	存放一般流程的签署意见
24.	PHASEOPINION1	意见详情1	
25.	PHASEOPINION2	意见详情2	
26.	PHASEOPINION3	意见详情3	
27.	INPUTORG	登记机构	
28.	INPUTUSER	登记人员	
29.	INPUTTIME	登记时间	
30.	UPDATEUSER	变更人员	
31.	UPDATETIME	变更时间	

1.3.5 流程服务 API

本节介绍 FlowService.java 的方法含义，但不介绍详细参数及含义，具体函数参数及含义请参加 awe 的 API。

1.3.5.1 新增流程实例

新增流程实例调用 FlowService.initFlow(applyType, objectNo, userID)方法。底层操作是往 Flow_Object 表插入一条数据， 根据 applyType 获取到 ObjectType（申请类型.ItemAttribute）和 Flowno（申请类型.Attribute2，先把 attribute2 的值当做 baseFlowNo 到 Flow_Catalog 表中找 isInUse 为 1 的流程——如果有多个，取 version 最大的那个流程，如果没有找到流程，再把 attribute2 的值当做 FlowNo 到 Flow_Catalog 表找 isInUse 为 1 的流程），生成一条 Flow_Object 表的数据并填充 ApplyType、ObjectType、ObjectNo、FlowNo 等字段。根据流程模型的 initPhase 找到对应的 Flow_Model 模型，生成一条 Flow_Task 表的数据并填充数据。

1.3.5.2 取消流程

取消流程调用 FlowService.delete(ObjectNo,ObjectType)方法。底层操作是根据 ObjectNo 和 ObjectType 删除 Flow_Object、Flow_Task、Flow_Opinion 表的对应数据。

1.3.5.3 提交流程

提交流程调用FlowService.commitAction(ftSerialNo, sPhaseAction, sPhaseOpinion1, sFlowState)底层会解析出流程任务节点 ftSerialNo 对应模型的 PostAction，传入 sPhaseAction 和 sPhaseOpinion1 和其他相关参数，执行 PostAction 脚本得到模型下一节点，然后往 Flow_Task 插入对应流转节点实例信息。

1.3.5.4 撤回流程

撤回流程调用 FlowService.takeBack(ftSerialNo),即废除之前的提交状态，回收本任务。底层会删除对应的 Task_Flow 记录，将 Flow_Object 状态复原，如果该节点是分支节点、待审会秘书分发节点或下一节点任务已提交，则不允许撤回。

1.3.5.5 退回任意阶段

可以将流程当前任务退回到之前某个流转节点，调用 FlowService.returnTask(ftSerialNo, returnPoint, assignedTaskNo)，即将当前任务 ftSerialNo 退回到 Serialno 为 returnPoint 的节点

所处的状态，底层会新增一个 Flow_Task 赋值为 returnPoint 所处的状态，并把 ASSIGNEDTASKNO 字段赋值为 assignedTaskNo，这样当任务被再次提交时会提交到 assignedTaskNo 所处的状态。

1.3.5.6 其他流程操作

其他流程操作包括从任务池抓取任务等，可参考 awe 的 API。

2. 常规流程开发

2.1 常规流程原理

ALS 中流程按其包含的节点类型可分为：常规流程、并行流程、任务池流程、待审会流程等。常规流程指流程中节点只包含“开始节点”、“一般任务节点”、“退回补充资料”、“批准”、“否决”这些常规节点的流程。

2.1.1 流程模型

2.1.2 节点类型

常规节点包括以下几种：

- 开始节点，NodeType=Start 其特性和普通节点一样，只是标记是流程实例化后第一个流转的节点，必需将 FlowCatalog 的 initPhase 设成开始节点的 PhaseNo 开始节点才有意义。
- 普通任务节点，NodeType=Task 最常用的普通流转节点。
- 结束节点，NodeType=END，包括批准节点和否决节点，其 initScript 承办人初始化脚本为：{"system"}，代表系统节点。“批准”节点和“否决”节点除了节点名称不一样其他字段配置都一样的，这导致在 ADE 中，批准和否决节点图标一样，为了解决这个问题，ADE 按照传统的节点编号来区分批准和否决，即认为 PhaseNo=1000 的节点为批准节点，PhaseNo=8000 节点为否决节点，这个小缺陷将在以后被修复。

- 退回补充资料节，NodeType=SUPPLY。在 A3Web3.1 之后不再使用该节点，补充资料被做成页面按钮来实现。

2.1.3 流程实例

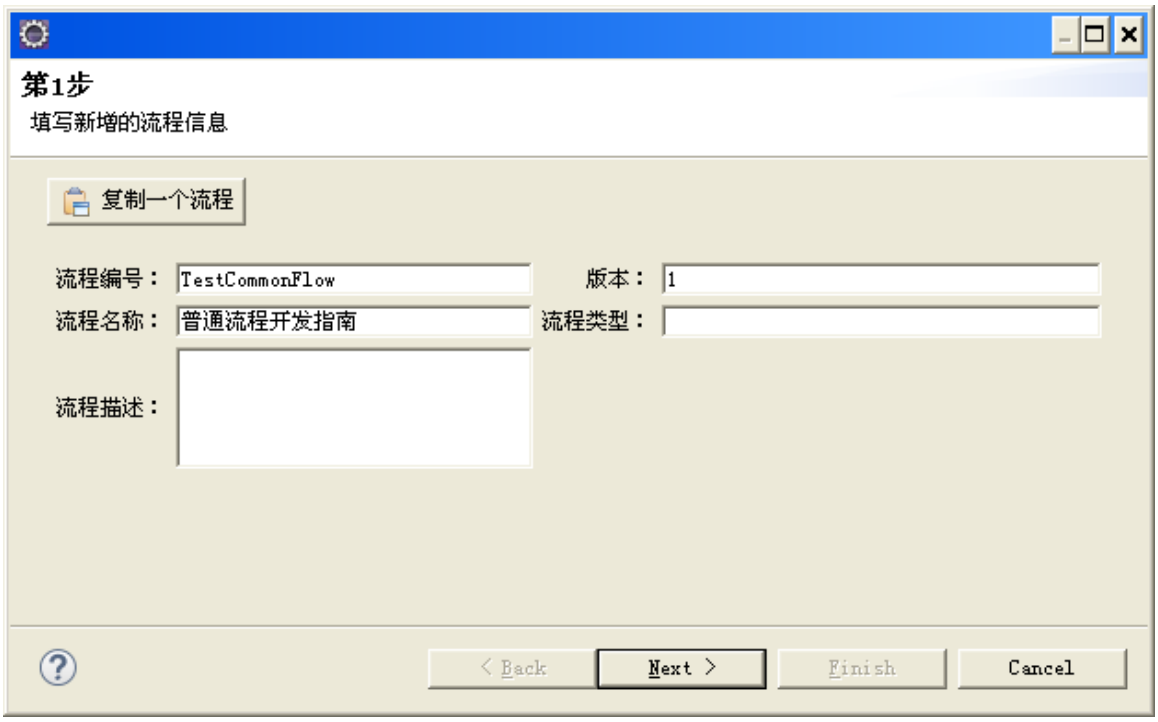
2.2 开发案例

本节将介绍借助 ADE 工具来完成流程的开发，ADE 的安装及配置请参考 ADE 发布时附带的《ADE 使用手册.pdf》 。

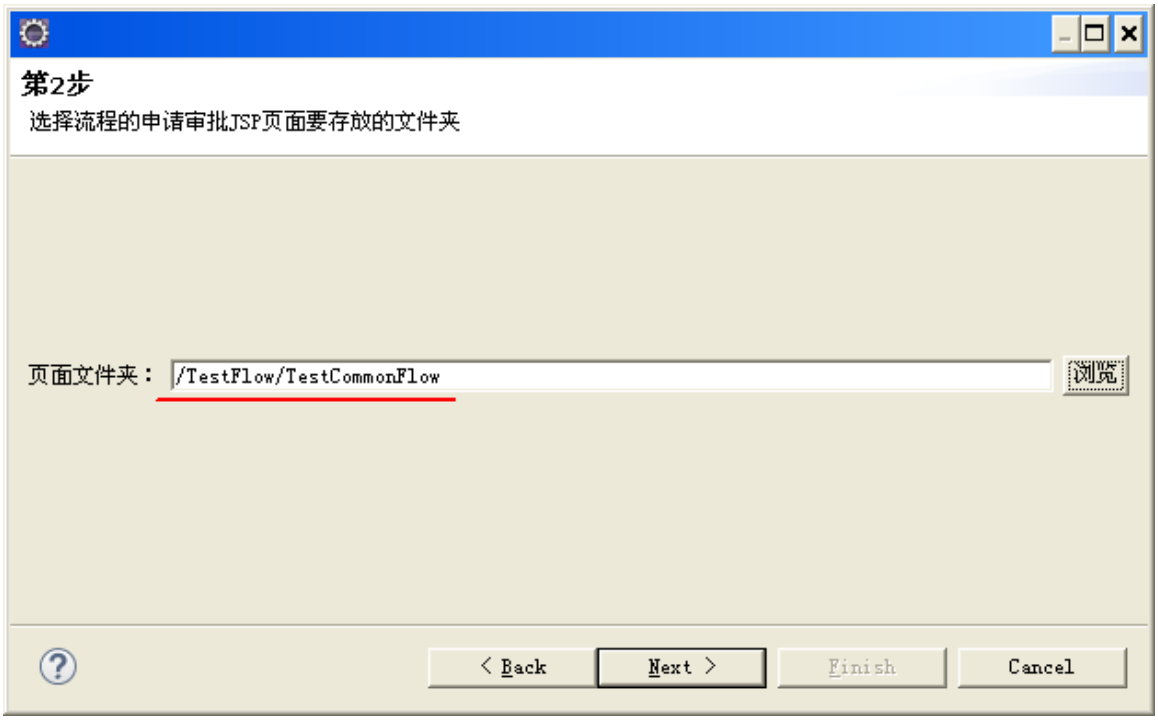
2.2.1 新建流程

新增流程时我们可以使用 ADE 的“新增流程向导”功能，通过向导来生成流程模型、申请审批主树图和相应的页面。在 Eclipse 中打开工程下的资源管理→业务流程设置→流程模型点击“从向导添加”。

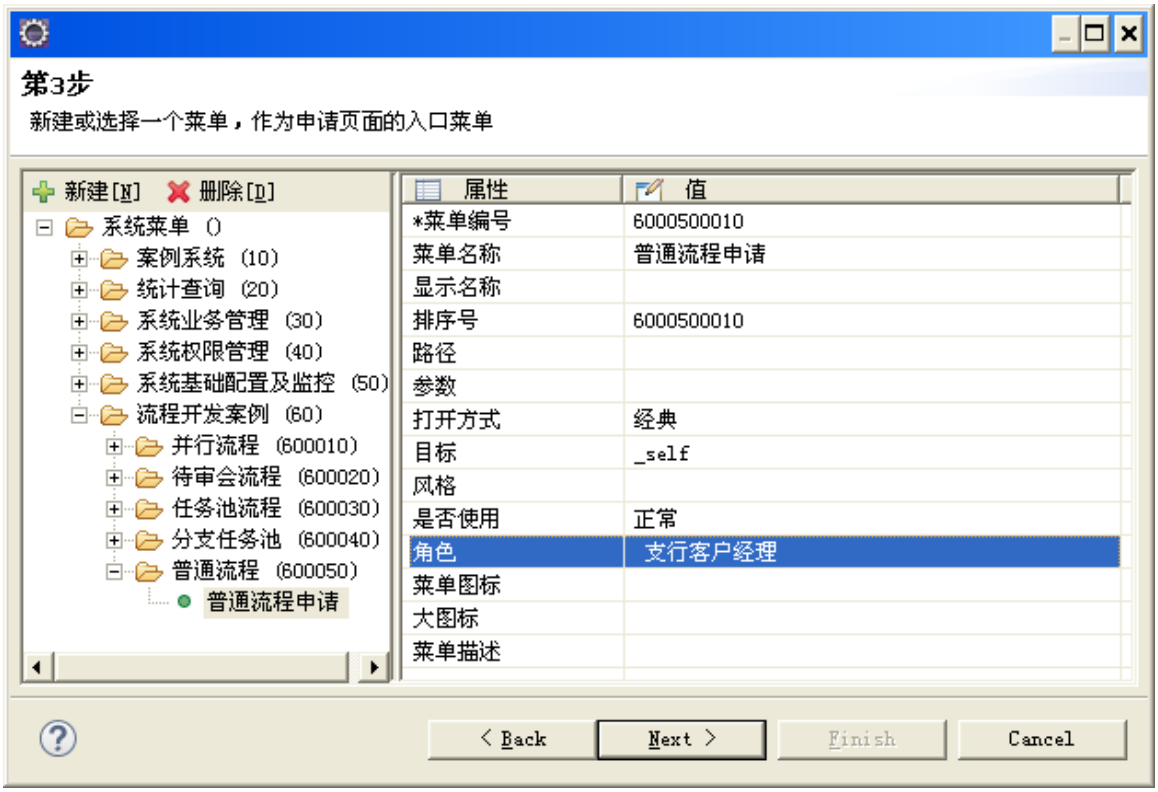
1. 填写流程基本信息：



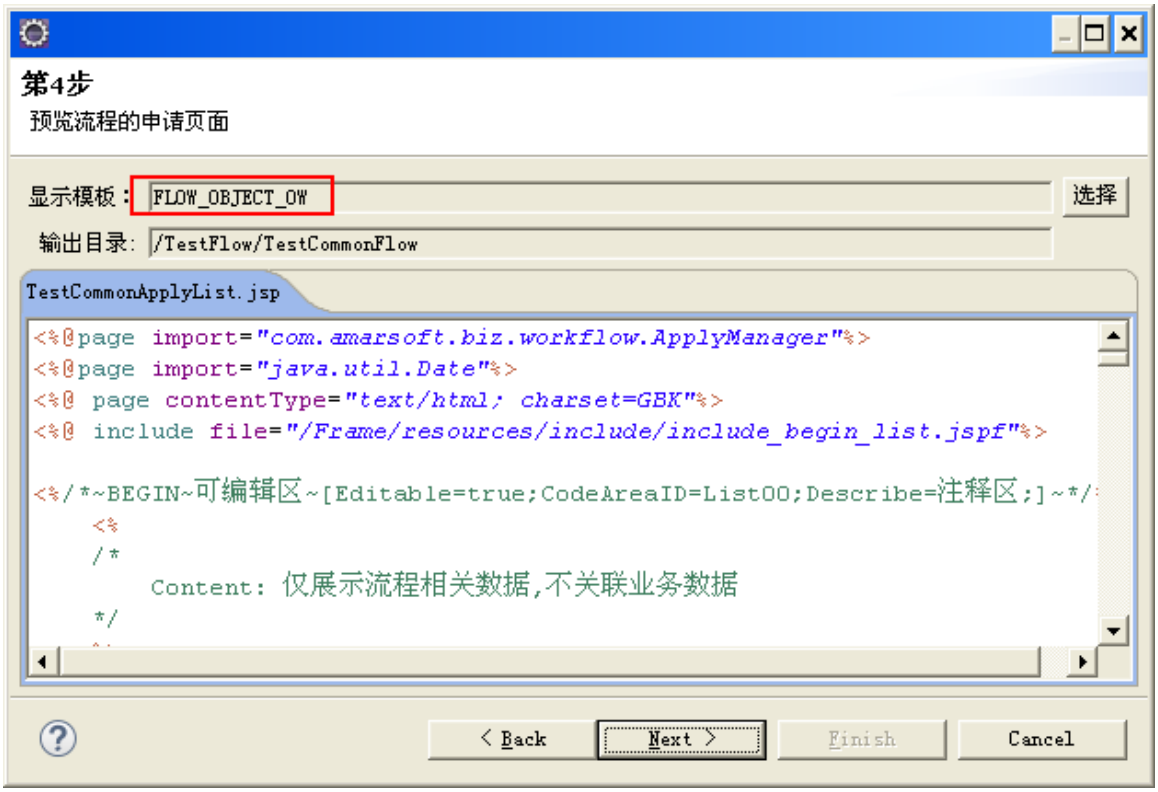
2. 选择一个文件夹，用来存放流程申请、审批的 JSP 页面，可点击“浏览”选择。



3. 选择或新建一个菜单作为申请的入口菜单，需要配置菜单的角色，但不用配置“路径”和“参数”，向导完成后，菜单会被自动赋值路径和参数。

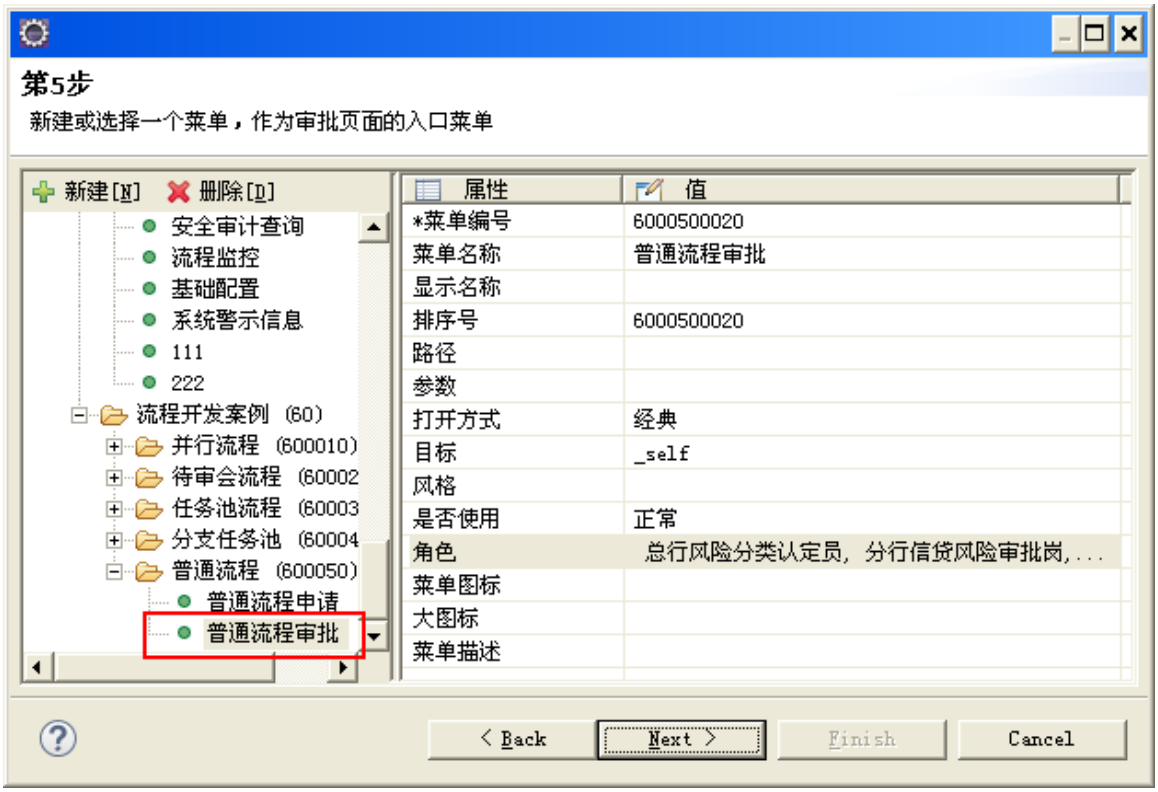


4. 选择申请页面的显示模板，这里我们使用默认的 Flow_Object_OW，这个显示模板是系统自带的不关联业务表的显示模板（只有 Flow_Object 表）。

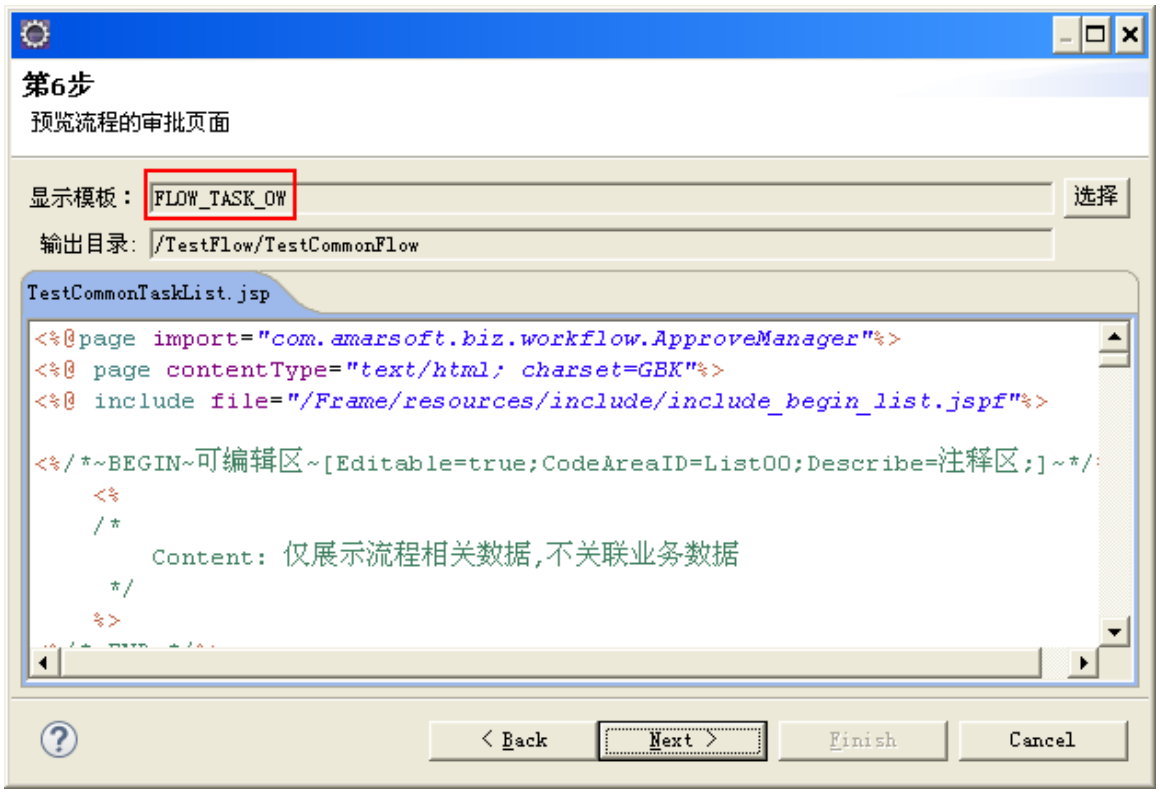


如果需要选择或新建关联业务表的显示模板，可以点击“选择”按钮，流程申请页面的显示模板一般由 Flow_Object 表和业务表关联而来的，其中 Flow_Object 必需作为主对象。同时在显示模板中 Flow_Object 的一些字段必需勾选，且不能与其他对象（表）的字段重名：OBJECTTYPE、OBJECTNO、SERIALNO、PHASETYPE、PHASENO、APPLYTYPE、FLOWNO、ORGID、USERID、INPUTDATE、FLOWSTATE。

5. 选择或新建一个菜单作为审批的入口菜单，需要配置菜单的角色，但不用配置“路径”和“参数”，向导完成后，菜单会被自动赋值路径和参数。



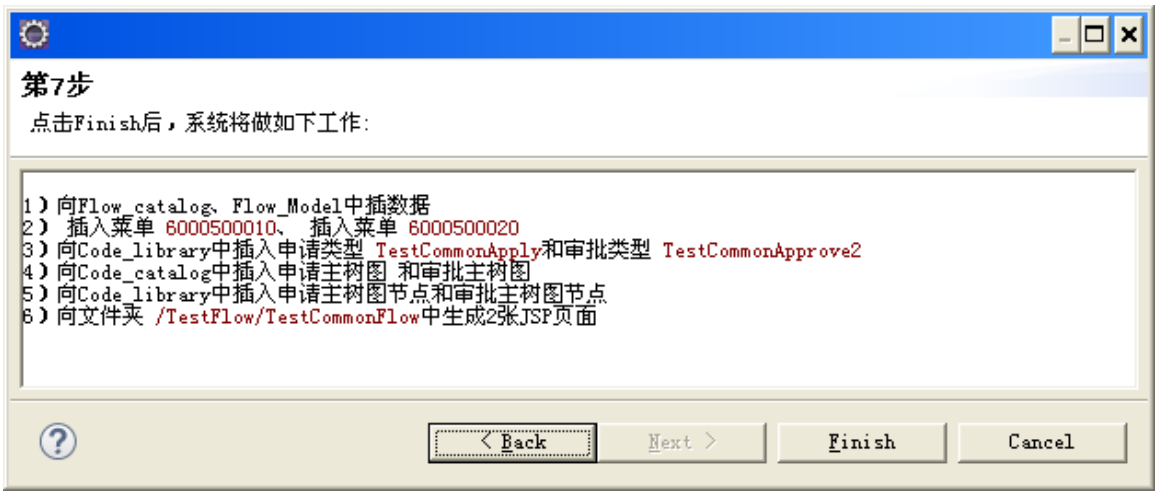
6. 选择审批页面的显示模板，这里我们使用默认的 Flow_Task_OW，这个显示模板是系统自带的不关联业务表的显示模板（只有 Flow_Task 表）。



点击“选择”按钮可以选择一个审批显示模板，同样如果没有显示模板可以新建一

个，和第 5 步不同的是审批显示模板的主表是 Flow_Task 而不是 Flow_Object。其中 Flow_Task 的一些字段必需勾选，且不能与其他对象（表）的字段重名：OBJECTTYPE、OBJECTNO、SERIALNO、PHASETYPE、PHASENO、APPLYTYPE、FLOWNO、ORGID、USERID、INPUTDATE、FLOWSTATE、BEGINTIME、ENDTIME。

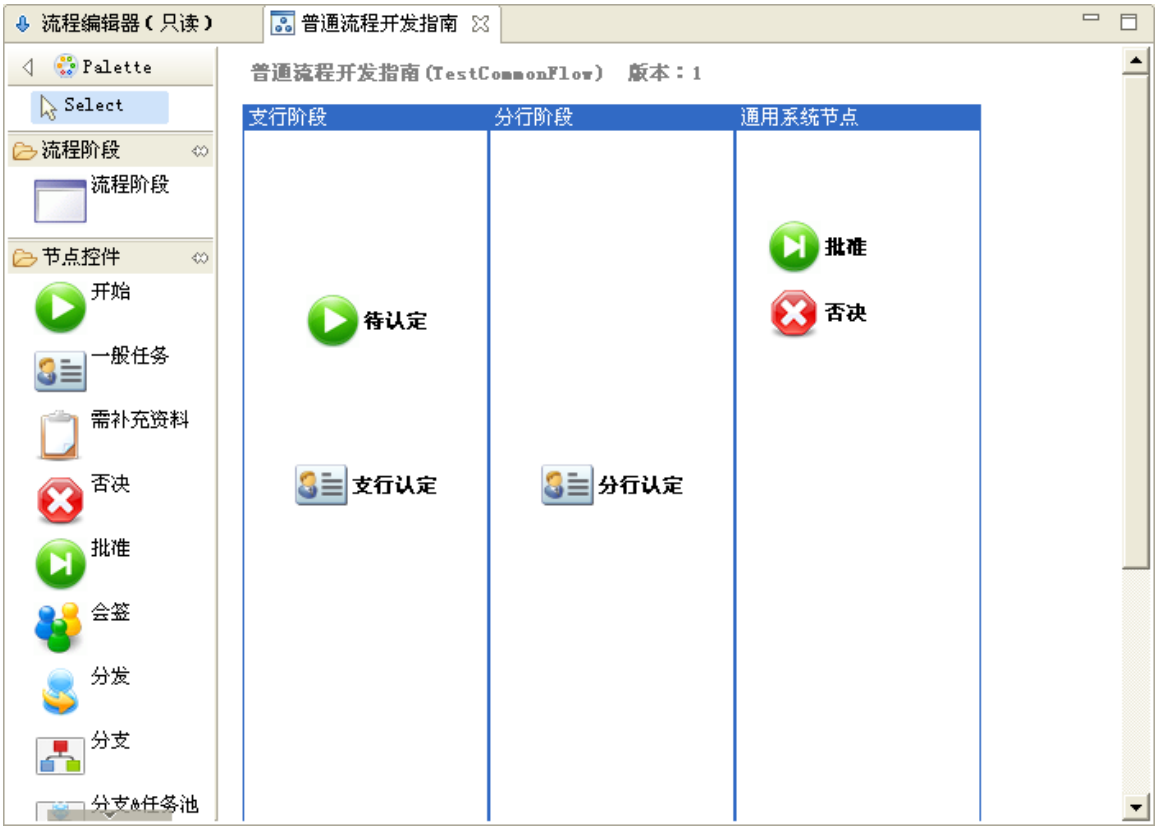
7. 预览数据生成清单：



点击 Finish 即可完成新增流程，系统会像数据库中插入流程模型、申请审批类型、主树图节点等数据。接下来需要完善这些数据。

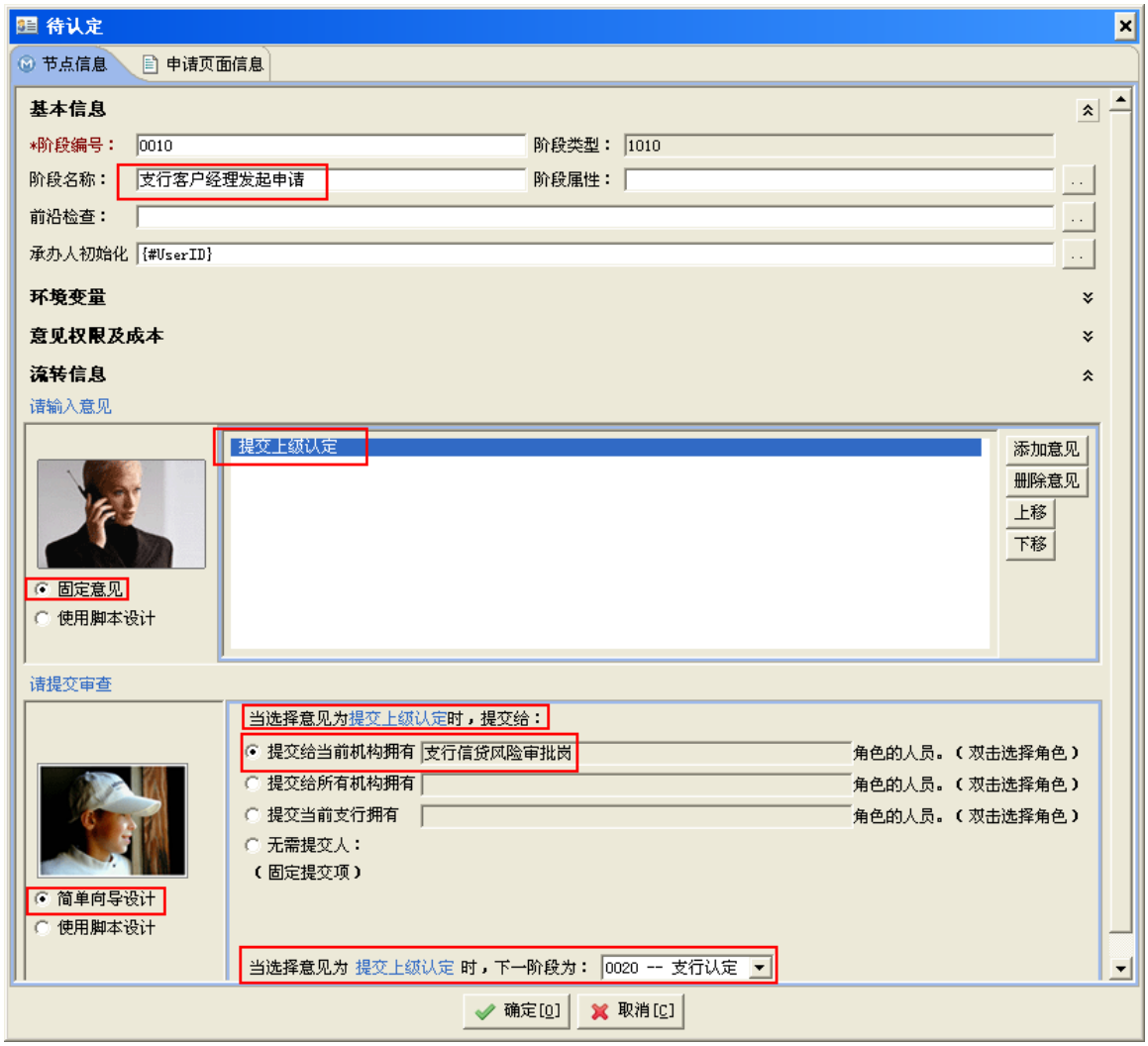
2.2.2 流程模型配置

使用 ADE 工具新建的流程默认有 5 个节点：



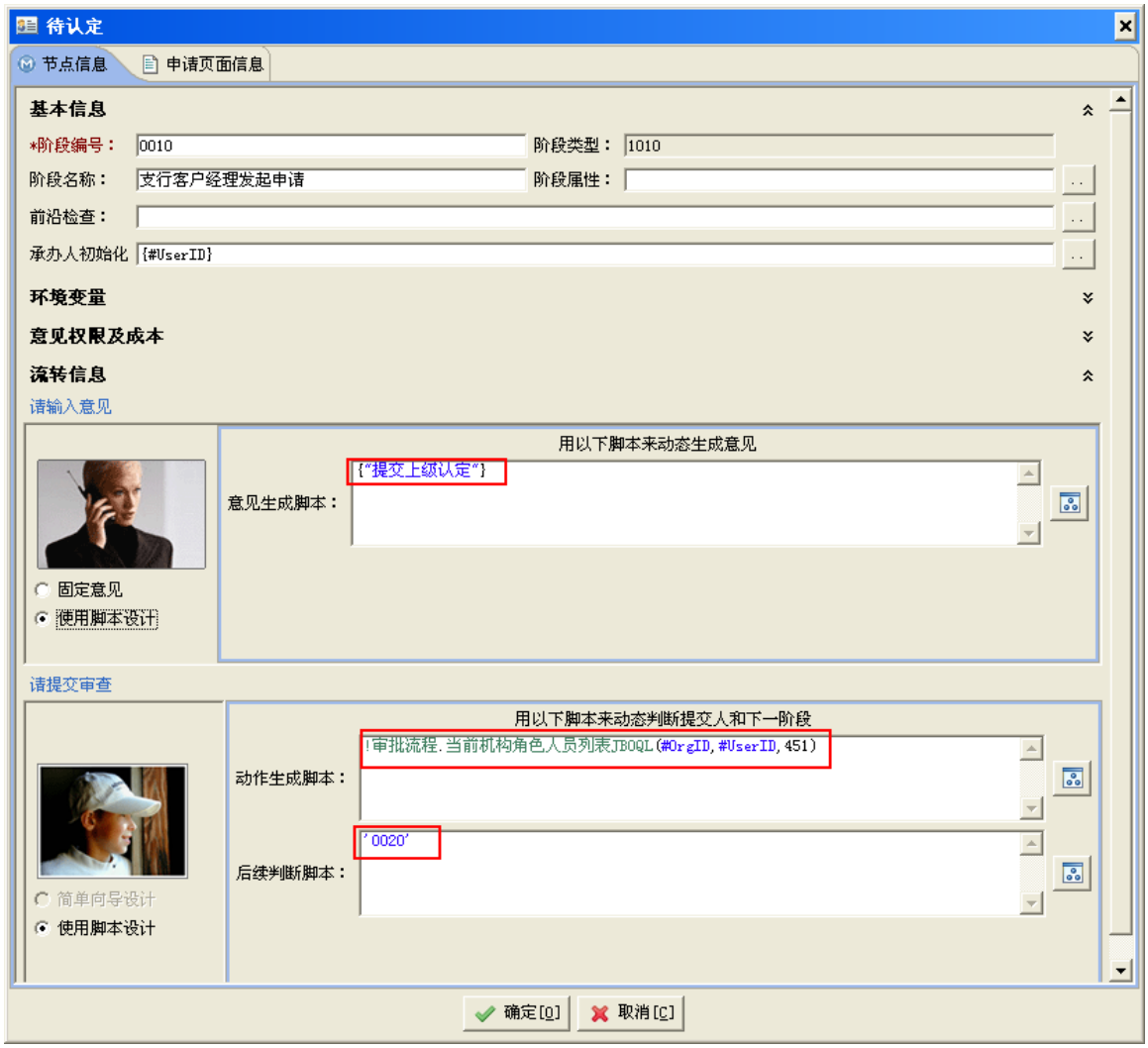
使用流程编辑器来编辑流程模型，这里我们做一个简单的流程，供即由支行客户经理发起认定→支行认定→分行认定。使用流程编辑编辑流程：

1. 单击“待认定”节点，编辑节点名称、意见脚本、动作脚本、后续脚本：



配置意见时有“固定意见”和“使用脚本设计”两种方式，固定意见使用列表界面方式新增意见，而脚本设计则是直接使用 AmarScript 脚本编写 ChoiceScript，当使用“固定意见”时 ADE 会最终将固定意见转换成 AmarScript 脚本。

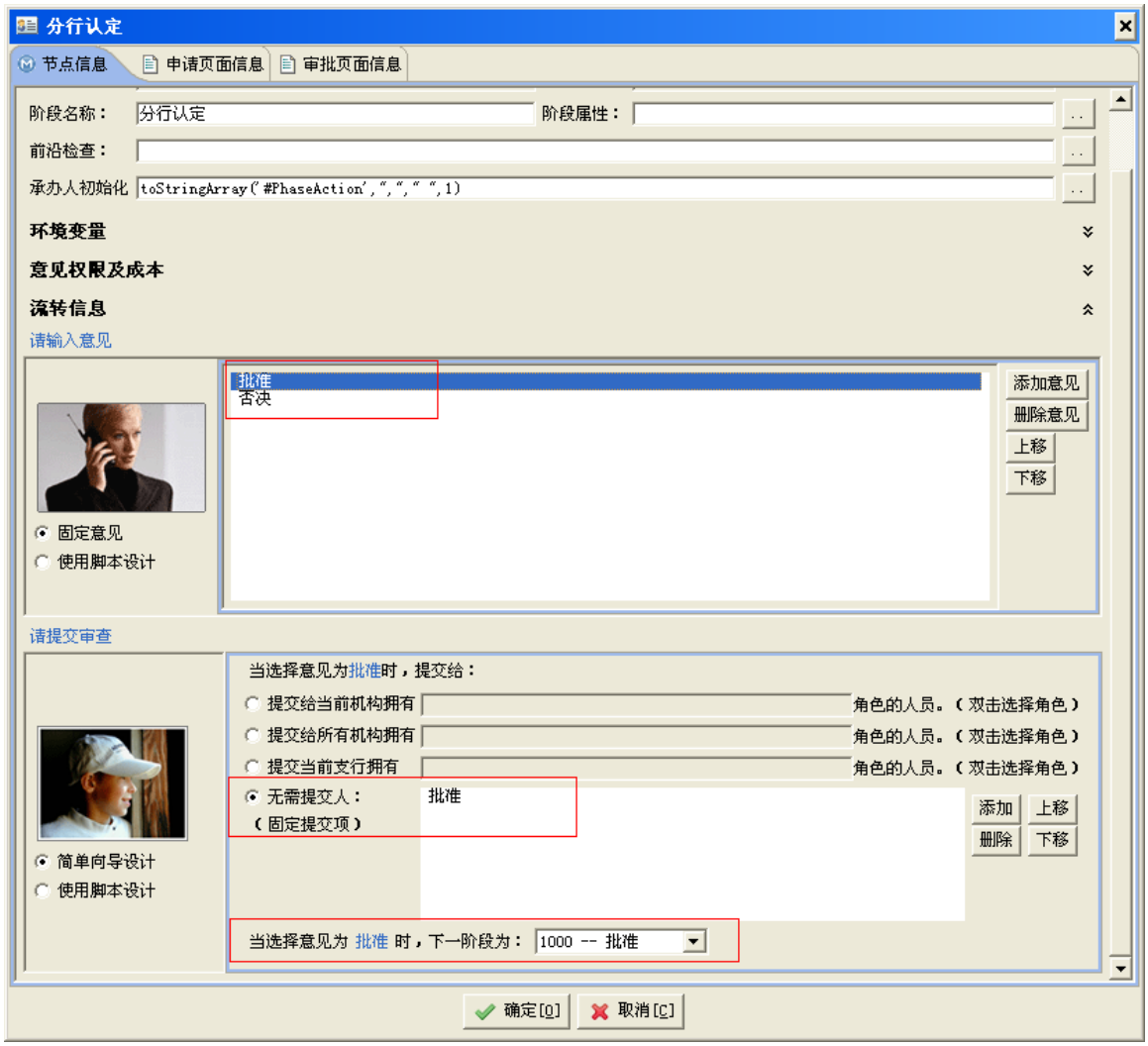
同样动作和后续配置有“简单向导设计”和“使用脚本设计”两种，简单向导设计只适用“静态路由”场景，即意见时固定的，且动作和后续脚本根据意见的不同也是设计时可知的，如上图所示。由“简单向导”配置的流转逻辑最终也将转换成 AmarScript 脚本：



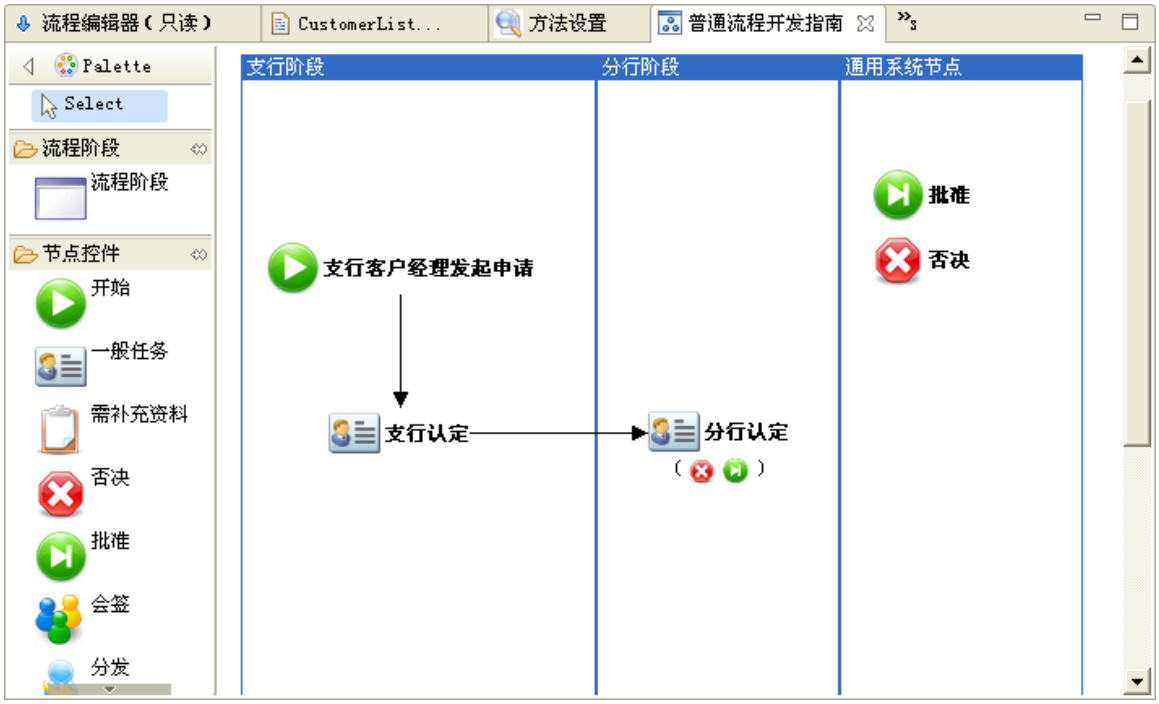
这里需注意的是，在 A3Web3.1 之前正常的提交上级的意见必须包含“提交”2 字，3.1 之后无此限定。

2. 双击支行认定节点，配置其下一阶段为“分行认定”：

3. 双击“分行认定”节点，配置其可以批准或否决流程。

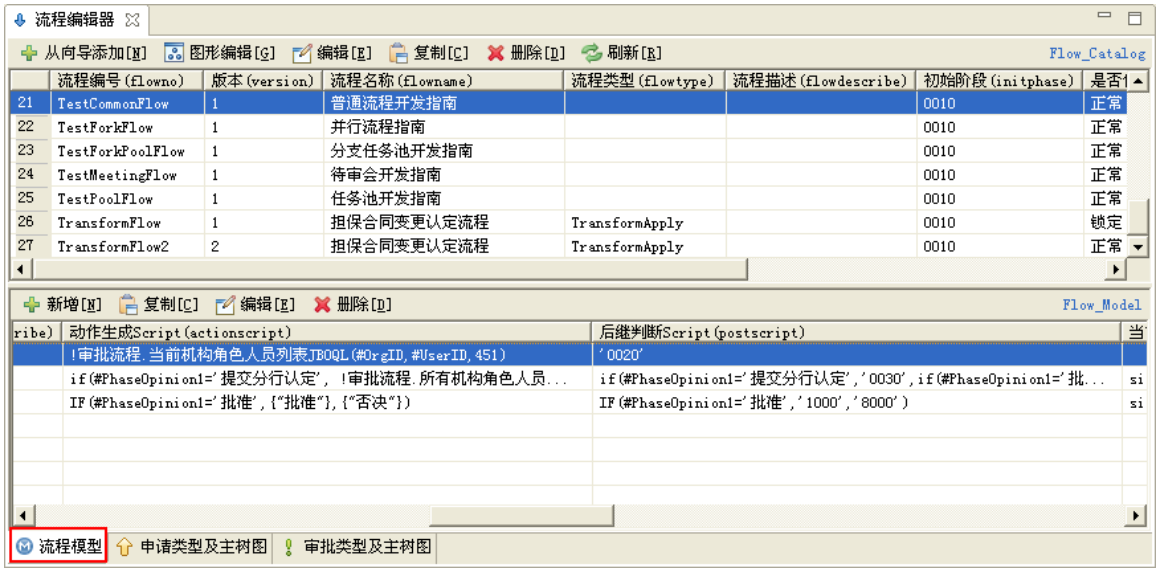

分行认定
节点信息
申请页面信息
审批页面信息
阶段名称：
分行认定
阶段属性：
前沿检查：
承办人初始化
toStringArray('#PhaseAction',",",",",1)
环境变量
意见权限及成本
流转信息
请输入意见
批准
否决
添加意见
删除意见
上移
下移
固定意见
使用脚本设计
请提交审查
当选择意见为批准时，提交给：
提交给当前机构拥有
角色的人员。（双击选择角色）
提交给所有机构拥有
角色的人员。（双击选择角色）
提交当前支行拥有
角色的人员。（双击选择角色）
无需提交人：
（固定提交项）
批准
添加
上移
删除
下移
简单向导设计
使用脚本设计
当选择意见为 批准 时，下一阶段为：
1000 -- 批准
确定[O]
取消[C]

最终的流程图如下：


流程编辑器（只读）
CustomerList...
方法设置
普通流程开发指南
Palette
Select
流程阶段
流程阶段
节点控件
开始
一般任务
需补充资料
否决
批准
会签
分发
支行阶段
分行阶段
通用系统节点
支行客户经理发起申请
支行认定
分行认定
批准
否决

图中泳道信息即“支行阶段”、“分行阶段”和“通用系统节点”只是在设计时为节点做了一个分类，该分类没有业务含义，泳道信息存放在 Flow_SwimLane 表中。

以上使用图形化编辑器来维护模型信息，可以为使用表格编辑器来修改流程模型信息，具体方法是打开资源管理→业务流程设置→流程模型，在“流程模型”页下可修改模型信息：



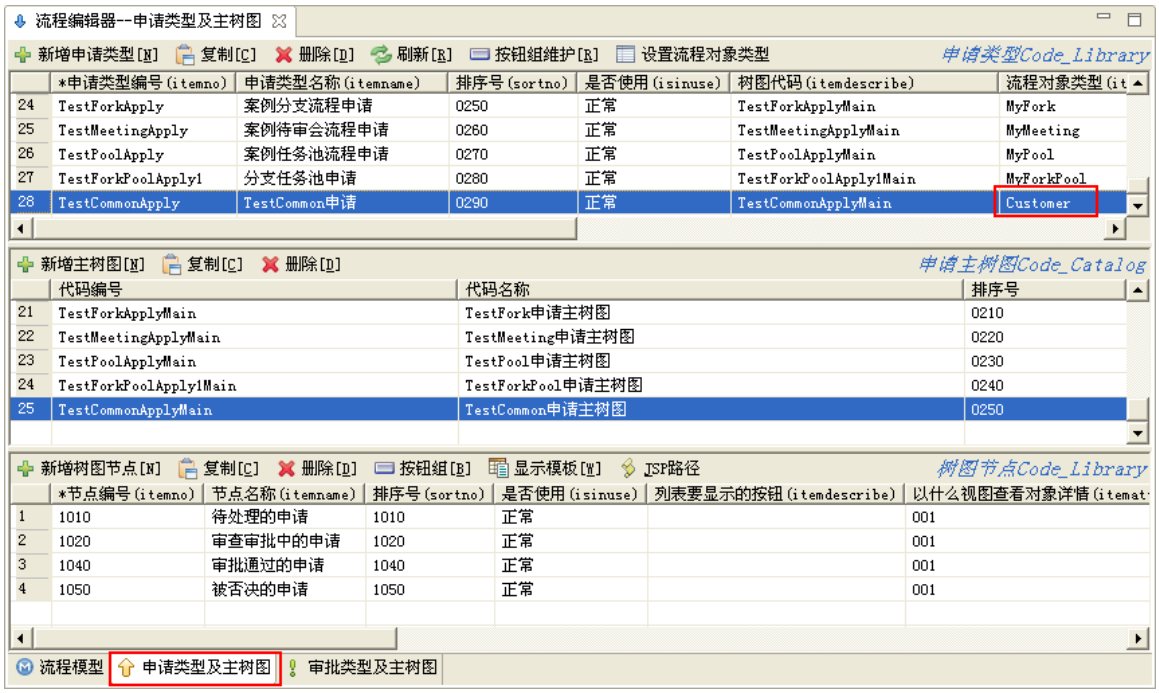
（注意：图形编辑器和流程编辑器维护的是同一份数据，所以当图形编辑器被打开时，表格编辑器就自动变成只读的了）。

2.2.3 配置申请类型及主树图

设置申请类型及主树图节点在 ADE 中有 2 种编辑方法：纯表格编辑和预览式编辑。在本节我们使用预览式编辑，表格编辑只做介绍。

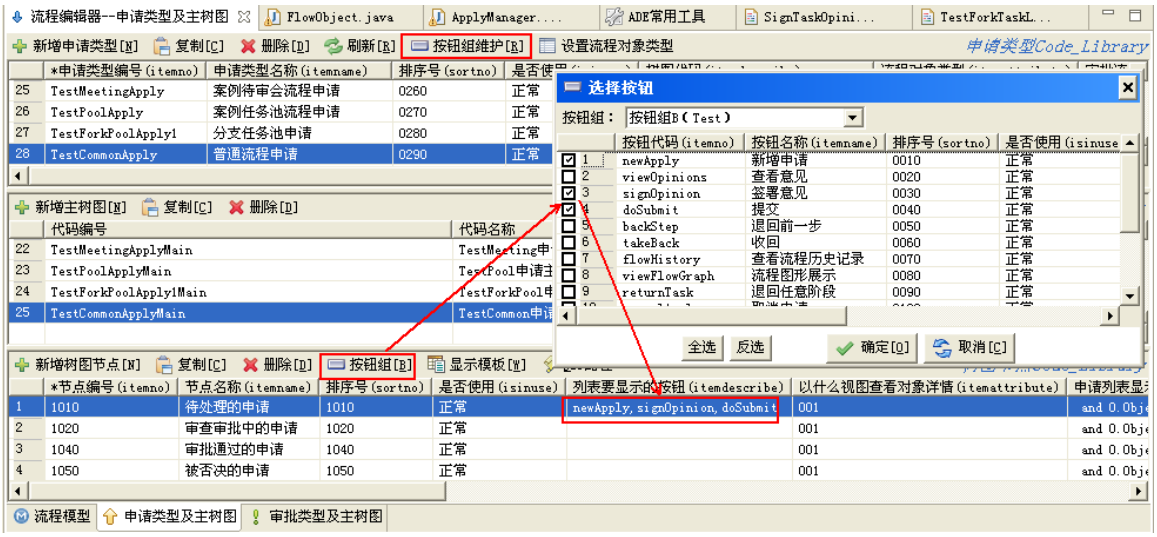
2.2.3.1 使用表格编辑器

纯表格式编辑类似直接修改数据库表，在 Eclipse 中打开资源管理→业务流程设置→流程模型，选中我们新建的流程 TestCommonFlow，点击编辑器下方标签切换到“申请类型及主树图”编辑页，即可编辑对应的申请类型即主树图：



在本界面可以编辑申请类型及主树图节点的信息，上图中高亮的申请类型及主树图是我们使用流程向导自动新建出来的，有些默认值需要我们修改：

1. 申请类型中默认生成的按钮组是 ButtonSetI，可点击“按钮组维护”来更换或新增，本案例将按钮组改为 ButtonSetB（Test）。
2. 申请类型默认的流程对象类型是 Customer，本案例需要将其修改为 MyFlow。
3. 主树图节点的按钮组配置，我们这里给“待处理的申请”节点加上“新增申请”、“签署意见”和“提交”三个按钮。配置方法：单击树图节点记录，点击“按钮组”按钮或双击 ItemDescribe 字段即可：

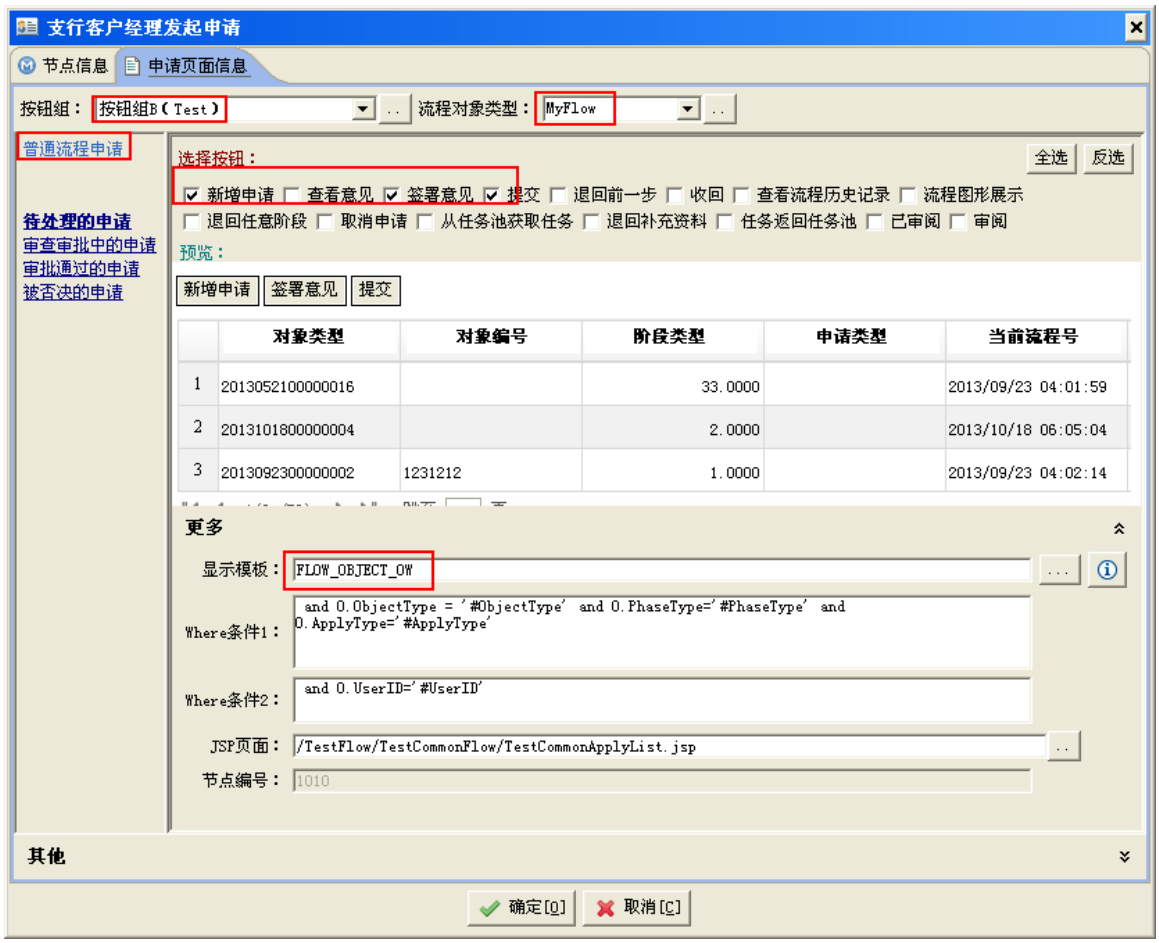


4. 主树图节点的 attribute1 和 attribute2 是显示模板拼接过滤条件，这里默认配成 and O.ObjectType = '#ObjectType' and O.PhaseType='#PhaseType' and

O.ApplyType='#ApplyType' 和 and O.UserID='#UserID'，即过滤出 Flow_Object 中本申请类型、本流程对象类、处于本节点下且承办人为当前登录用户的数据。

2.2.3.2 使用预览编辑器

在流程编辑器中双击任意流程点，在弹出的对话框中选择“申请页面信息”标签，即可进入预览式编辑：



在预览式编辑界面可以维护按钮组（本案例改成 ButtonSetB(Test)）、编辑流程对象类型（本案例改成 MyFlow）。点击左侧树图节点名称右侧出现页面配置界面，本案例需要给待处理的申请勾选“新增申请”、“签署意见”和“提交”3 个按钮。

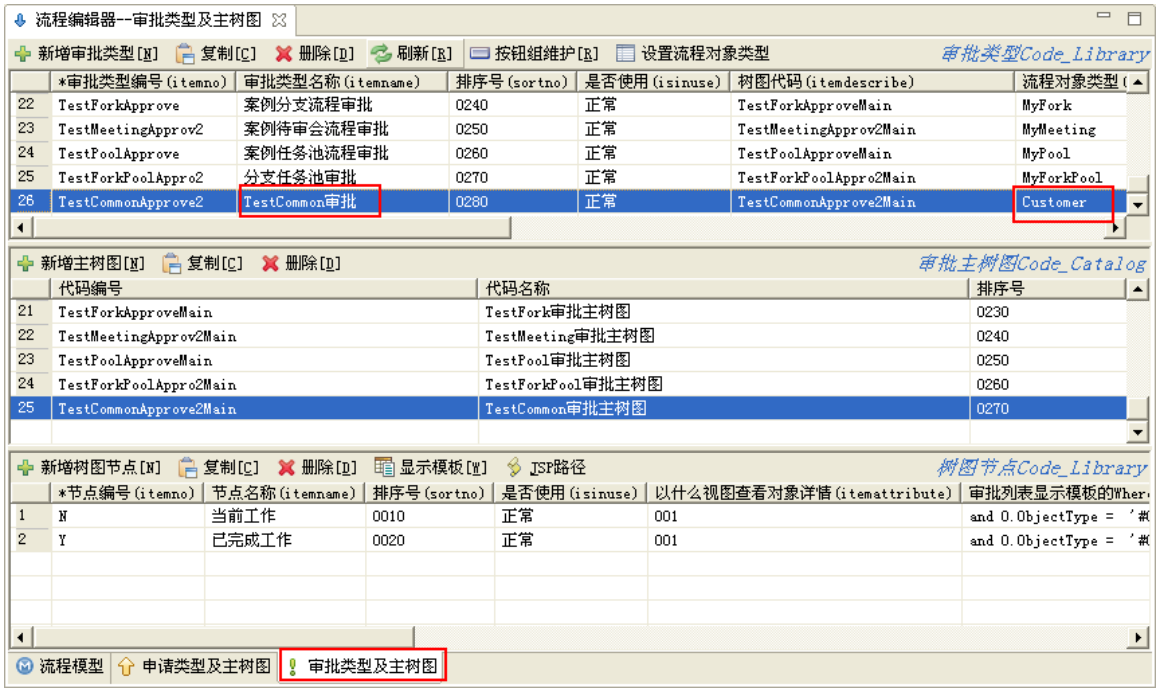
注意：在本界面修改的流程对象类型会同时更新到申请和审批类型的 attribute2。

2.2.4 配置审批类型及主树图

设置审批类型及主树图节点在 ADE 中有 2 种编辑方法：纯表格编辑和预览式编辑。在本节我们使用预览式编辑，表格编辑只做介绍。

2.2.4.1 使用表格编辑器

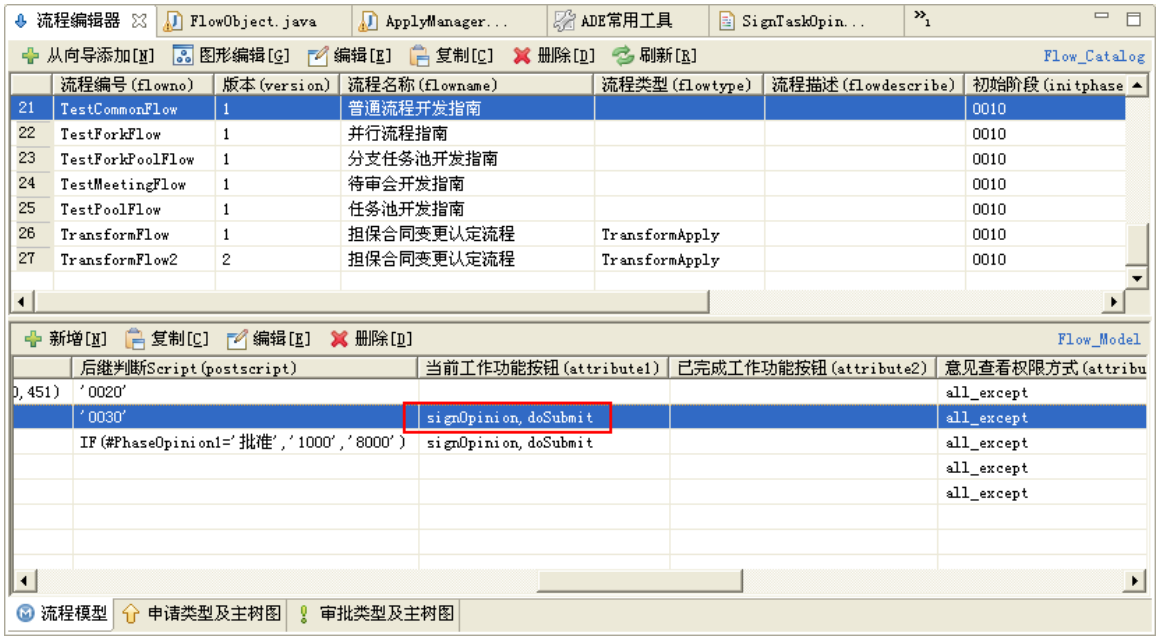
在 Eclipse 中打开资源管理→业务流程设置→流程模型，选中我们新建的流程 TestCommonFlow，点击编辑器下方标签切换到“审批类型及主树图”编辑页，即可编辑对应的审批类型即主树图：



在本界面可以维护按钮组、流程对象类型和审批树图节点信息。

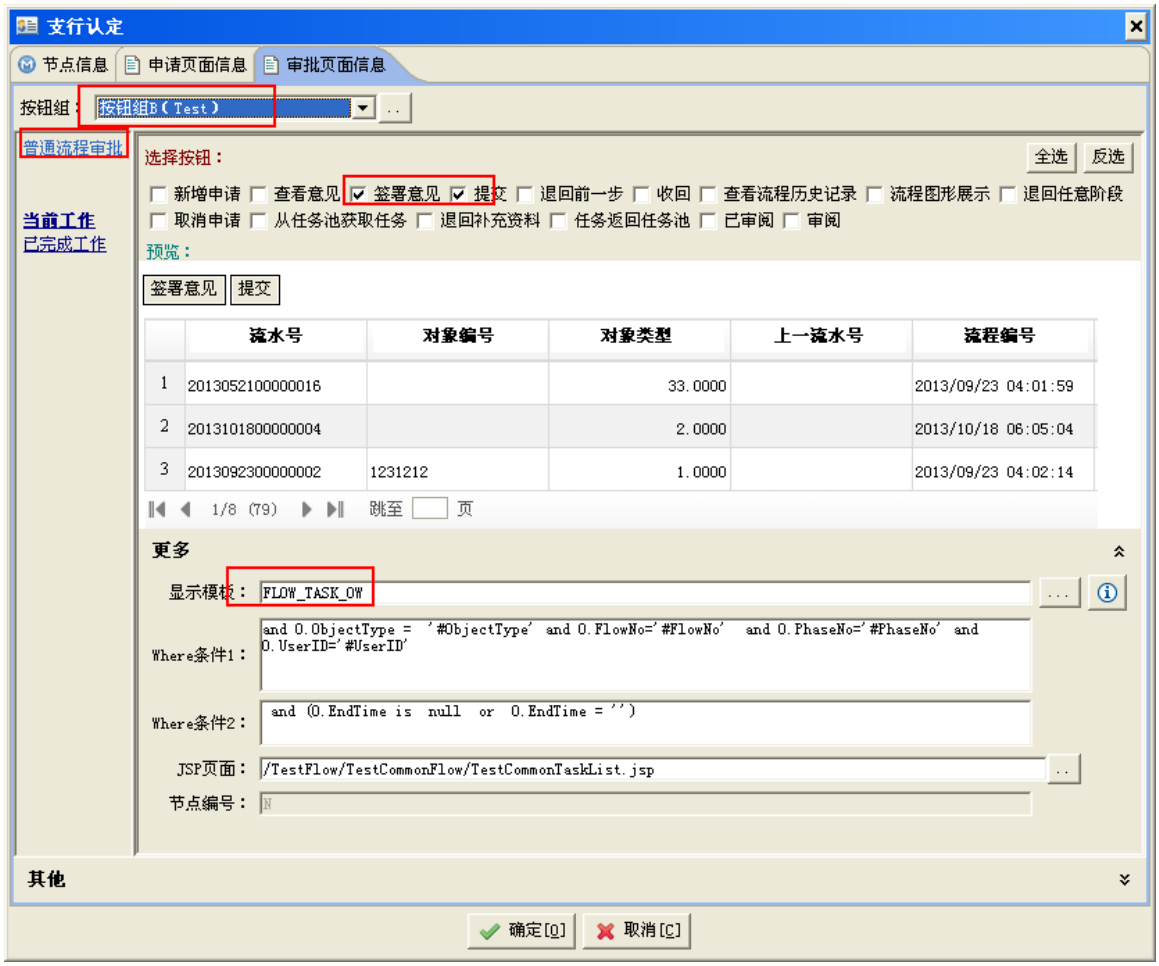
1. 审批树图节点只有固定两个，N 和 Y 分别代表当前工作和已完成工作。其中已完成工作的 attribute1 和 attribute2 为：and O.ObjectType = '#ObjectType' and O.FlowNo='#FlowNo' and O.PhaseNo='#PhaseNo' and O.UserID='#UserID' 和 and (O.EndTime is null or O.EndTime = '')。即过滤 Flow_Task 中本流程的、本流程对象类型的、没有被提交的、且承办人为当前登录用户的数据。未完成的工作类似，只是其 attribute2 为 and O.EndTime is not null and O.EndTime <> ''，代表以提交（即已完成）的工作。

2. 审批页面按钮，审批页面的按钮不是在审批主树图节点中配置的，而是在流程模型 Flow_Model 中来配置的，因为审批页面中的可见按钮根据流程所处流转节点的不同而不同。Flow_Model.Attribute1 代表当前工作按钮，Flow_Model.Attribute2 代表未完成工作按钮。配置方法：切换到“流程模型”标签，双击 Attribute1 进入单元格，再双击一次弹出按钮组选择对话框：



2.2.4.2 使用预览编辑器

在流程编辑器中双击某个流程点击，在弹出的对话框中选择“审批页面信息”标签，即可进入审批页面预览式编辑（注意：有些节点没有对应的审批页面，如开始节点）：



在此界面配置的树图节点信息（显示模板、where 条件等）会更新到审批树图节点中，但是勾选的按钮会更新到当前流程节点的 Attribute1、Attribute2 中。本案例中需将按钮组改成 ButtonSetB（Test），给当前工作勾选“签署意见”和“提交”2 个按钮即可。

2.2.5 编写申请、审批 JSP 页面

我们使用流程向导新建流程时，默认已经新增了申请页面和审批页面。

其中 TestCommonApplyList.jsp 为申请页面，里面已经默认实现了“新增申请”、“签署意见”、“提交”、“退回”、“取消申请”、“查看意见”等 JS 方法，一般开发流程时需要重写“新增申请”向 Flow_Object 插入数据并关联业务对象。本案例只突出流程特性暂不关联业务对象，所以 ObjectNo 为随机生成。即默认的 jsp 页面不做改动。

流程服务类是 FlowService，但页面通过 JS 方法调的服务类是 FlowAction，FlowAction 对 FlowService 封装，并返回直接结果的字符串。

下面来解释一下申请页面的默认 js 实现：

1. newApply()新增申请

```
function newApply(){
    //根据applyType,objectNo,userID 初始化流程
    var d = new Date();
    var sObjectNo = ""+d.getFullYear()+getNum(d.getMonth()+1, 2)+getNum(d.getDate(), 2)+getNum(d.getHours(), 2)
            +getNum(d.getMinutes(), 2)+getNum(d.getSeconds(), 2)+getNum(d.getMilliseconds(), 3);
    var sRet = AsControl.RunJavaMethodTrans("com.amarsoft.app.flow.FlowAction","initFlow",
            "ApplyType=<%=sApplyType%>,ObjectNo="+sObjectNo+",UserID=<%=CurUser.getUserID()%>");
    if(sRet=="success"){
        alert("操作成功. ");
    }else{
        alert("操作失败! ");
    }
    reloadSelf();
}
```

上图看出 sObjectNo 为随即生成，然后调用 FlowAction.initFlow，生成一条流程实例。

2. cancelApply()取消申请

```
function cancelApply(){
    //获得申请类型、申请流水号
    var sObjectType = getItemValue(0,getRow(),"OBJECTTYPE");
    var sObjectNo = getItemValue(0,getRow(),"OBJECTNO");
    if (typeof(sObjectNo)=="undefined" || sObjectNo.length==0){
        alert(getHtmlMessage('1'));//请选择一条信息!
        return;
    }
    // 删除flow_object,flow_task,flow_opinion三张表的数据
    var sRet = AsControl.RunJavaMethodTrans("com.amarsoft.app.flow.FlowAction","delete"
            ,"objectType="+ sObjectType +",objectNo="+sObjectNo);
    if(sRet=="success"){
        alert("操作成功. ");
    }else{
        alert("操作失败! ");
    }
    reloadSelf();
}
```

取消申请即调用 FlowAction.delete 方法，删除流程对象、任务和意见。

3. takeBack()撤回流程

```
function takeBack(){
    var sObjectType = getItemValue(0,getRow(),"OBJECTTYPE");
    var sObjectNo = getItemValue(0,getRow(),"OBJECTNO");
    var sFlowNo = getItemValue(0,getRow(),"FLOWNO");
    var sUserID = getItemValue(0,getRow(),"USERID");
    if (typeof(sObjectNo)=="undefined" || sObjectNo.length==0){
        alert(getHtmlMessage('1'));//请选择一条信息!
        return;
    }
    if(!confirm("确认撤回?")) return;
    var sRet = AsControl.RunJavaMethodTrans("com.amarsoft.app.flow.FlowAction","takeBack",
            "ObjectType="+ sObjectType+",ObjectNo="+sObjectNo+",FlowNo="+sFlowNo
            +",UserID=<%=CurUser.getUserID()%>" );
    if(sRet=="success"){
        alert("操作成功. ");
    }else{
        alert("操作失败! ");
    }
    reloadSelf();
}
```

撤回即调用 FlowAction.takeBack,在下个阶段提交前回收流程。

4. doSubmit()提交流程，在该 JS 方法中调用 FlowSubmitDialog.jsp 打开流程对话框，

FlowSubmitDialog 中会调用 FlowAction.getActionList 等方法获取动作列表等信息，生成提交页面。最终调用 FlowAction.submit 方法提交流程。该方法默认不需要改动。

5. signOpinion()签署意见，在该 JS 方法中打开 SignTaskOpinionInfo.jsp 页面，在该页面可进行签署意见操作。
6. viewFlowGraph()以图形的形式展现当前流程的运行情况。

审批的页面的方法 TestCommonTaskList 在本案例里不需要修改，其中的默认实现除了提交、签署意见、撤回之外还有如下 JS 方法有：

1. returnTask()退回任意阶段，即退后到之前流程流转的任意阶段，调用 TaskReturnList.jsp 页面选择历史流转阶段，然后调用 FlowAction,returnTask 退回选择的阶段。
2. backStep()退回上一步。

```
function backStep() {
    //获取任务流水号
    var sTaskNo = getItemValue(0,getRow(),"SERIALNO");
    if(typeof(sTaskNo) == "undefined" || sTaskNo.length == 0){
        alert(getHtmlMessage('1'));//请选择一条信息!
        return;
    }
    if(!confirm(getMessageText("ALS70509"))) return; //您确认要将该申请退回上一环节吗?
    var sRet = AsControl.RunJavaMethodTrans("com.amarsoft.app.flow.FlowAction","backStep","TaskNo="+ sTaskNo );
    if(sRet=="success"){
        alert("操作成功. ");
    }else{
        alert("操作失败! ");
    }
    parent.reloadSelf();
}
```

即调用 FlowAction.backSetp 退回上一个流转阶段。

3. flowHistory()查看流程历史流转。查看流程的历史流转信息，显示 Flow_Task 的数据。
4. fetchTask()从任务池获取任务。

```
function fetchTask(){
    var sRet = AsControl.RunJavaMethodTrans("com.amarsoft.app.flow.FlowAction","fetchTask",
            "FlowNo=<%=sFlowNo%>,PhaseNo=<%=sPhaseNo%>,UserID=<%=CurUser.getUserID()%>" );
    if(sRet.length>0){
        alert("获取任务成功. ");
    }else{
        alert("没有待获取的任务池任务! ");
    }
    reloadSelf();
}
```

调用 FlowAction.fetchTask 来获取任务池的任务。只有当前任务是任务池任务，且自己有权限才可以获取到任务池中的任务。

5. returnTaskToPool()把任务退回到任务池。

```
function returnTaskToPool(){
    var sObjectType = getItemValue(0,getRow(),"OBJECTTYPE");
    var sObjectNo = getItemValue(0,getRow(),"OBJECTNO");
    var sTaskNo = getItemValue(0,getRow(),"SERIALNO");
    var sFlowState = getItemValue(0,getRow(),"FLOWSTATE");
    if (typeof(sTaskNo)=="undefined" || sTaskNo.length==0){
        alert(getHtmlMessage('1'));//请选择一条信息!
        return;
    }
    if (typeof(sFlowState)=="undefined" || sFlowState.length==0 || sFlowState!="SELECT"){
        alert("不是从任务池获取的任务!");
        return;
    }
    var sRet = AsControl.RunJavaMethodTrans("com.amarsoft.app.flow.FlowAction","returnTaskToPool",
        "ObjectType="+sObjectType+",ObjectNo="+sObjectNo+",FlowNo=<%=sFlowNo%>,PhaseNo=<%=sPhaseNo%>" );
    if(sRet=="success"){
        alert("操作成功. ");
    }
    parent.reloadSelf();
}
```

这里调用 FlowAction.returnTaskPool 来把已经获取的任务重新放回任务池。

6. returnSupply()退回补充资料。

```
function returnSupply(){
    var sTaskNo = getItemValue(0,getRow(),"SERIALNO");
    var sObjectType = getItemValue(0,getRow(),"OBJECTTYPE");
    var sObjectNo = getItemValue(0,getRow(),"OBJECTNO");
    if (typeof(sTaskNo)=="undefined" || sTaskNo.length==0){
        alert(getHtmlMessage('1'));//请选择一条信息!
        return;
    }
    var sRet = AsControl.RunJavaMethodTrans("com.amarsoft.app.flow.FlowAction","returnSupply"
            ,"ObjectType="+sObjectType+",ObjectNo="+sObjectNo+",TaskNo="+ sTaskNo );
    if(sRet=="success"){
        alert("操作成功. ");
    }else{
        alert("操作失败! ");
    }
    parent.reloadSelf();
}
```

在 A3Web3.0 之后，退回补充资料不在以流程节点的形式实现，而是在页面中放一个退回按钮来实现，这里调用 FlowAction.returnSupply 来实现退回补充资料。

2.2.6 运行流程

启动 Tomcat，用 test11 登录系统点击菜单“流程开发案例”→普通流程→普通流程申请，即可打开申请页面。

1. 点击“新增申请”可新增一笔无业务的流程。此时后台向 Flow_Object 插入一条数据，其 ObjectType=MyFlow，ObjectNo 为随机生成。同时找到 TestCommonFlow 在 Flow_Catalog 中 initPhase 字段为 0010，所以想 Flow_Task 中插入初始任务，其 PhaseNo=0010。

2. 点击新增的流程，点击签署意见按钮，签署意见并保存，后台向 Flow_Opinion 插入一条数据且 Flow_Opinion.Serialno=Flow_Task.Serialno。点击提交按钮，打开

FlowSubmitDialog.jsp 页面，页面加载流程任务 0010 的系统变量和环境对象，运行 0010 节点的 ChoiceScript 脚本生成意见：“提交上级认定”。当点击“提交上级认定”意见时，后台加载 ActionScript 脚本（*!审批流程.当前机构角色人员列表JBOQL(#OrgID, #UserID, 451)*）并运行生成提交人列表。当点击一个提交人时，后台加载 PostScript 脚本（’0020’）并运行生成下一节点的节点编号。提交前页面如下图：

点击提交后，被选择的提交人“*test12 test12 支行客户经理 451 支行信贷风险审批岗*”会被记录到当前任务(Flow_Task)的 PhaseAction 字段。

同时后台取出“支行认定”模型（PhaseNo=0020），先运行其 PreScript 脚本，再运行 InitScript 脚本（toStringArray('#PhaseAction',",," ",1)，把上个节点的 PhaseAction 先按逗号分隔，再按空格分隔成二维矩阵，然后取第 1 列）其执行结果只有一行数据为“test12”，所以只生成一条任务并把其 UserID 赋值为 test12。

3. 使用 test12 登录，审批该流程提交给 test22，后台生成分行认定任务（Flow_Task），其 UserId=test22。

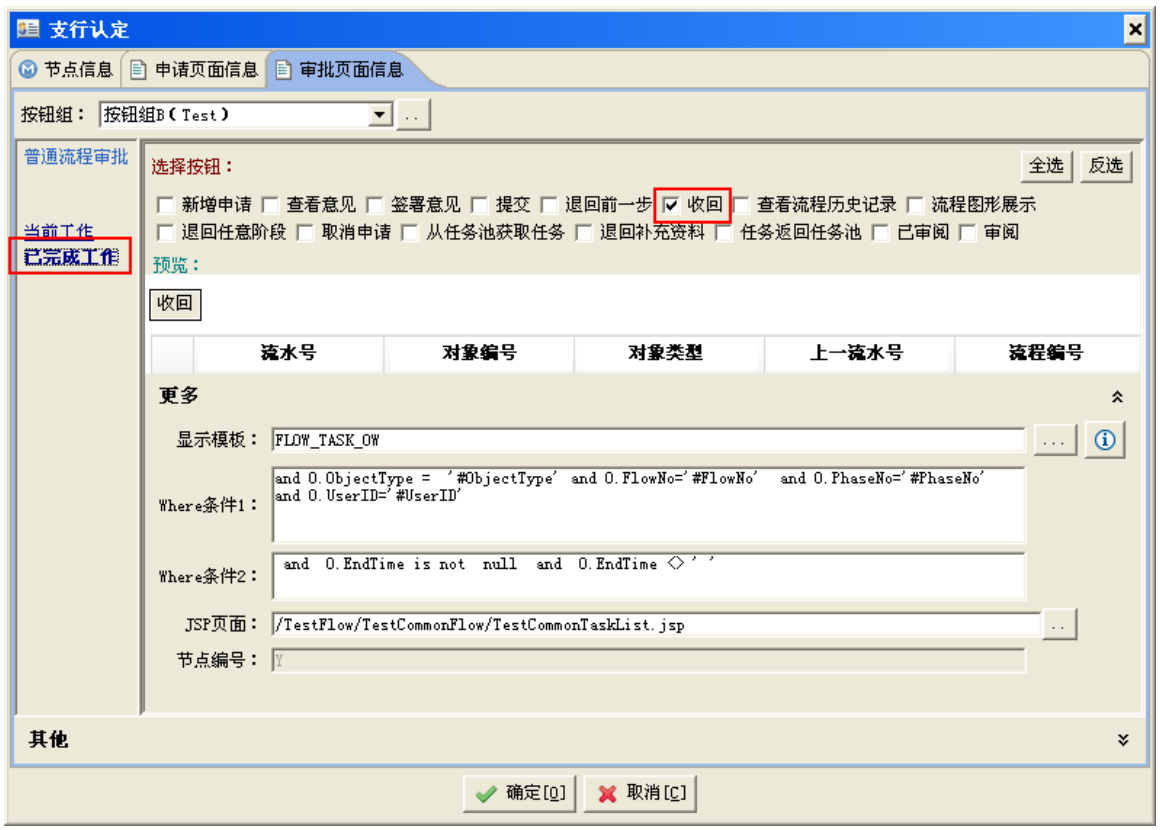
4. 使用 test22 登录，审批该流程批准该流程，后台生成一个批准任务，其 EndTime 自动被赋值当前时间，其 PhaseAction 为 AutoFinish，代表该节点不需要审批，自动完成。至此流程流转结束。

2.2.7 高级特性开发

2.2.7.1 撤回流程

也叫收回流程，流程当前任务被提交后同时在产生的下一个任务没有被提交前，可以撤回流程的提交。撤退流程在页面中调用 takeBack()函数，后台调用 FlowAction.takeBack 函数撤回流程。配置方法：

在 ADE 中用图形编辑器打开流程 TestCommonFlow，双击“支行认定”节点，点击“审批页面信息”标签，点击“已完成的工作”树图，配置其按钮为：回收。如下图：

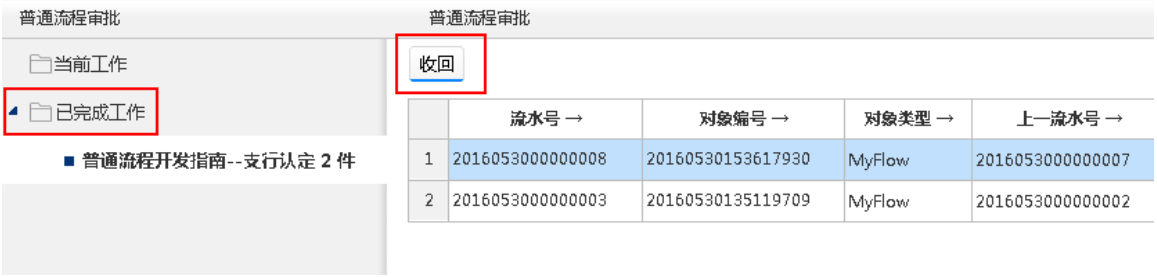


运行：

1. 用 test11 登录系统点击菜单“流程开发案例”→普通流程→普通流程申请，可打开申请页面，点击“新增申请”，签署意见并提交给 test12。生成 test12 任务。

2. 用 test12 登录，审批该流程，提交给 test22，生成 test22 任务。

3. 用 test12 点击左侧“已完成工作”，选择被提交的流程，点击“回收”按钮。



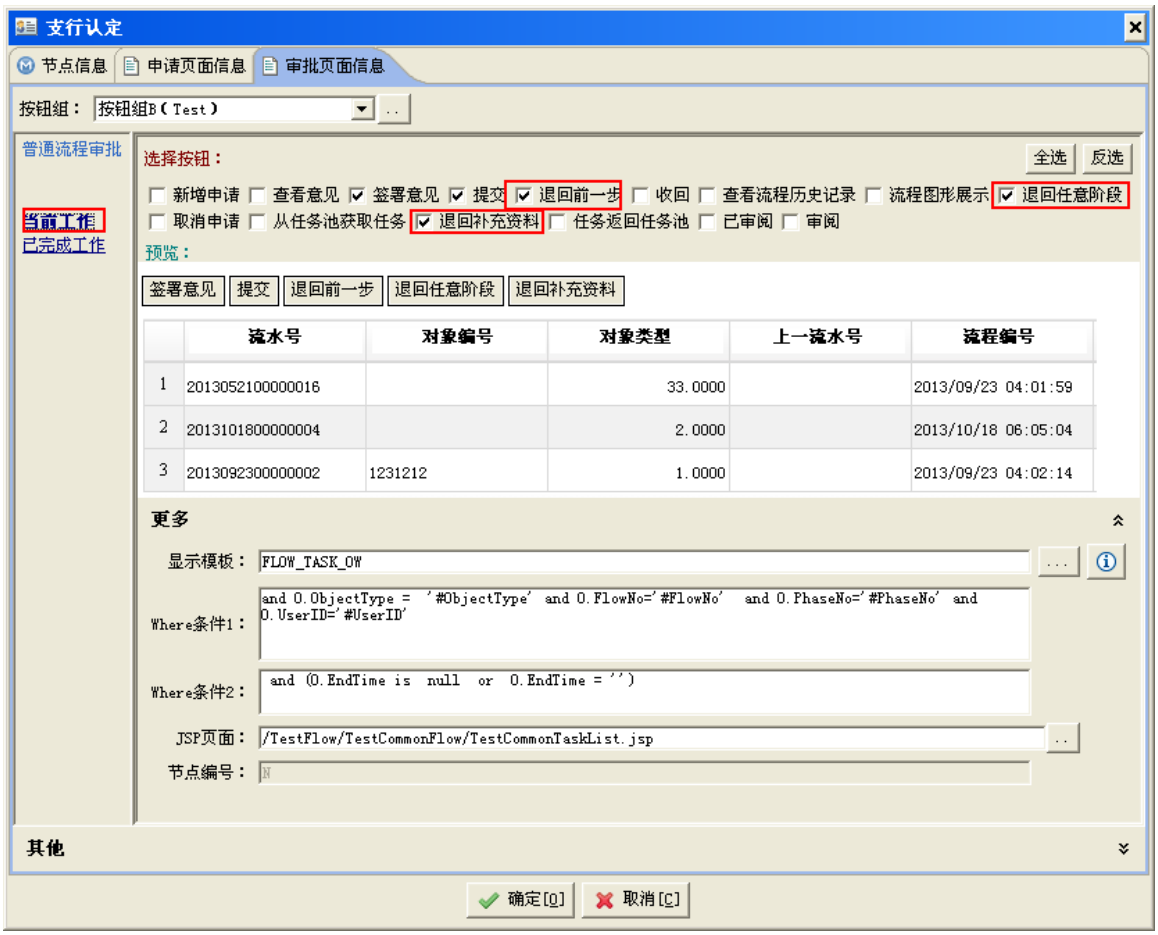
回收后，后台会删除 test22 的任务，把 test12 的任务 EndTime 置空，并更新 Flow_Object 状态。

4. 在“当前工作”中可继续提交被收回的流程。

2.2.7.2 退回任意阶段

退回任意阶段是将流程退回之前任一流转状态，前台调用 returnTask，后台调用 FlowAction,returnTask 函数。

在 ADE 中用图形编辑器打开流程 TestCommonFlow，双击“支行认定”节点，点击“审批页面信息”标签，点击“当前工作”树图，添加按钮“退回任一阶段”、“退回前一步”、“退回补充资料”。如下图：



1、用 test11 登录系统点击菜单“流程开发案例”→普通流程→普通流程申请，可打开申请页面，点击“新增申请”，签署意见并提交给 test12。生成 test12 任务。

2、用 test12 登录，点击菜单“普通流程审批”，在当前工作中选中流程，点击“退回任务阶段”。弹出选择要退回的阶段：

点击“退回并重新提交我”，后台调用 FlowAction.returnTask(其中 Recommit=Y)，在 Flow_Task 中生成一条返回任务，其 ASSIGNEDTASKNO=test12 任务的 Serialno，其 FowState=ANYBACK。

3、 使用 test11 登录，重新提交该流程，此时不弹出提交框，而是确认框询问是否将流程重新提交到退回人：

点击确定后，系统会生成新的任务其中 PhaseNo=0020，UserID=test12。

4、使用 test12 登录，选中该流程，点击“退回任意阶段”按钮，选中要退回的阶段，点击“退回”按钮。后台调用 FlowAction.returnTask(其中 Recommit=N)，在 Flow_Task 中生成一条返回任务，其 ASSIGNEDTASKNO 为空，其 FowState=ANYBACK。

5、使用 test11 登录，重新提交该流程，此时不会直接提交给退回人，而是继续解析 ActionScript、PostScript 重新流转流程。

2.2.7.3 退回上一步

退回前一步是退回任意阶段的特例，即流程退回到上一任务的状态，操作步骤：

1、用 test11 登录系统点击菜单“流程开发案例”→普通流程→普通流程申请，可打开申请页面，点击“新增申请”，签署意见并提交给 test12。生成 test12 任务。

2、用 test12 登录，点击菜单“普通流程审批”，在当前工作中选中流程，点击“退回上一步”。后台最终调用 FlowAction. backStep ()，在 Flow_Task 中生成一条返回任务，其 ASSIGNEDTASKNO 空，其 FowState=BACK。

3、使用 test11 登录，重新提交该流程，此时不会直接提交给退回人，而是继续解析 ActionScript、PostScript 重新流转流程。

2.2.7.4 退回补充资料

退回补充资料是退回任意阶段的特例，即退回到流程的“初始任务”所处状态，操作步骤：

1、用 test11 登录系统点击菜单“流程开发案例”→普通流程→普通流程申请，可打开申请页面，点击“新增申请”，签署意见并提交给 test12。生成 test12 任务。

2、用 test12 登录，点击菜单“普通流程审批”，在当前工作中选中流程，点击“退回补充资料”。后台最终调用 FlowAction.`returnSupply`()，在 Flow_Task 中生成一条退回任务，其 ASSIGNEDTASKNO=test12 任务.Serialno，其 FowState=SUPPLY。

3、使用 test11 登录，重新提交该流程，此时不会直接提交给退回人，而是继续解析 ActionScript、PostScript 重新流转流程。

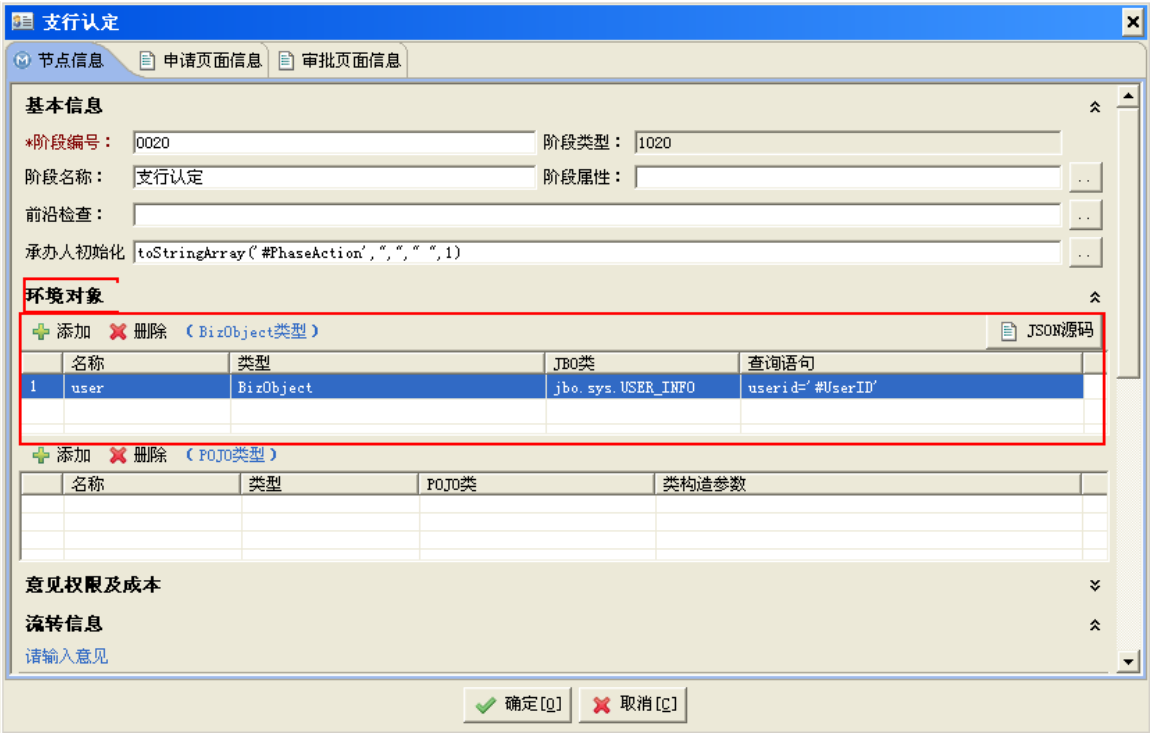
2.2.7.5 使用环境对象

本节介绍如何使用 ADE 配置流程节点的环境对象，并在流程脚本中使用环境对象，环境对象配置 JSON 串存放在 Flow_Model 表的 FlowPhaseContext 字段。本节案例假设流转逻辑做了如下变更（该变更纯粹是为了演示环境对象的使用）：即提交“支行认定”任务时，如果当前登录用户的备注（User_Info.Remark）为“特权”二字，则由其自行决定通过还是否决（无需提交上级）。

本节案例只介绍 BizObject 类型的对象配置及使用，POJO 类型的环境对象用户可以自己尝试配置。

2.2.7.5.1 配置环境对象

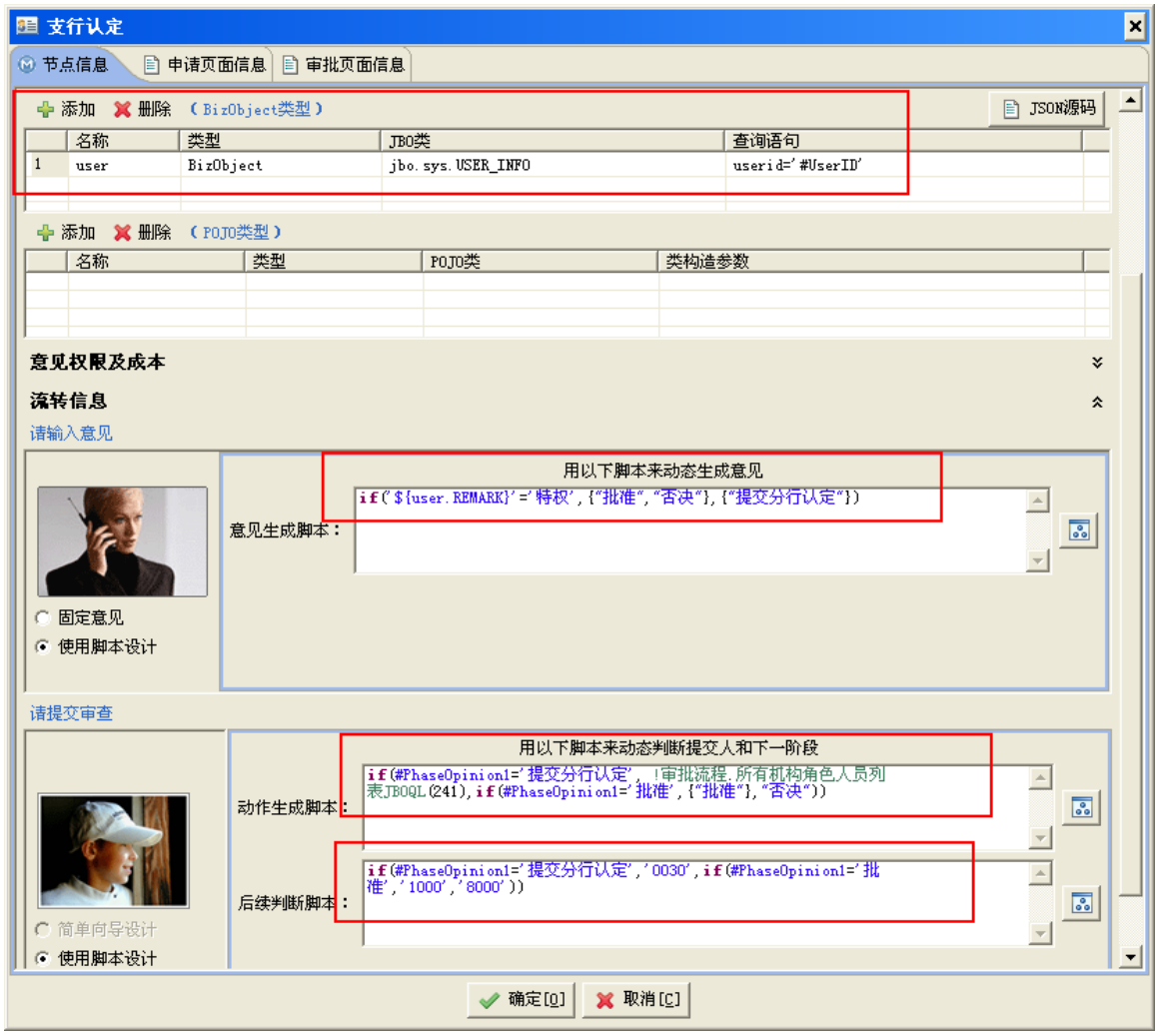
在 ADE 的流程编辑器中打开 TestCommonFlow 流程，双击“支行认定节点”，点开“环境对象”条目配置环境变量：



新建一个 BizObject 类型的环境对象，名称为 user，JBO 类为 jbo.sys.USER_INFO（可双击选择），查询语句为 userid='#UserID'（可双击编写）。这里我们定义了一个环境对象 user，其是用 JBOQL 查询得到的 JBO 对象，其查询语句为 userid='#UserID'，其中 #UserID 是流程环境变量。

2.2.7.5.2 使用环境对象

在 ChoiceScript 中使用环境对象，改写其 ChoiceScript、ActionScript、PostScript 为：



其中 ChoiceScript 为：*if('${user.REMARK}'='特权',{"批准","否决"},{"提交分行认定"})* 即根据当前登录用户的备注 Remark 来决定生成的意见。

同时 ActionScript 根据选中的意见生成提交人，脚本为：*if(#PhaseOpinion1='提交分行认定', !审批流程.所有机构角色人员列表 JBOQL(241),if(#PhaseOpinion1='批准',{"批准"},"否决"))* 。

同时 PostScript 中根据选中的意见决定下一流转节点，脚本为：*if(#PhaseOpinion1='提交分行认定','0030',if(#PhaseOpinion1='批准','1000','8000'))*。

2.2.7.5.3 运行结果

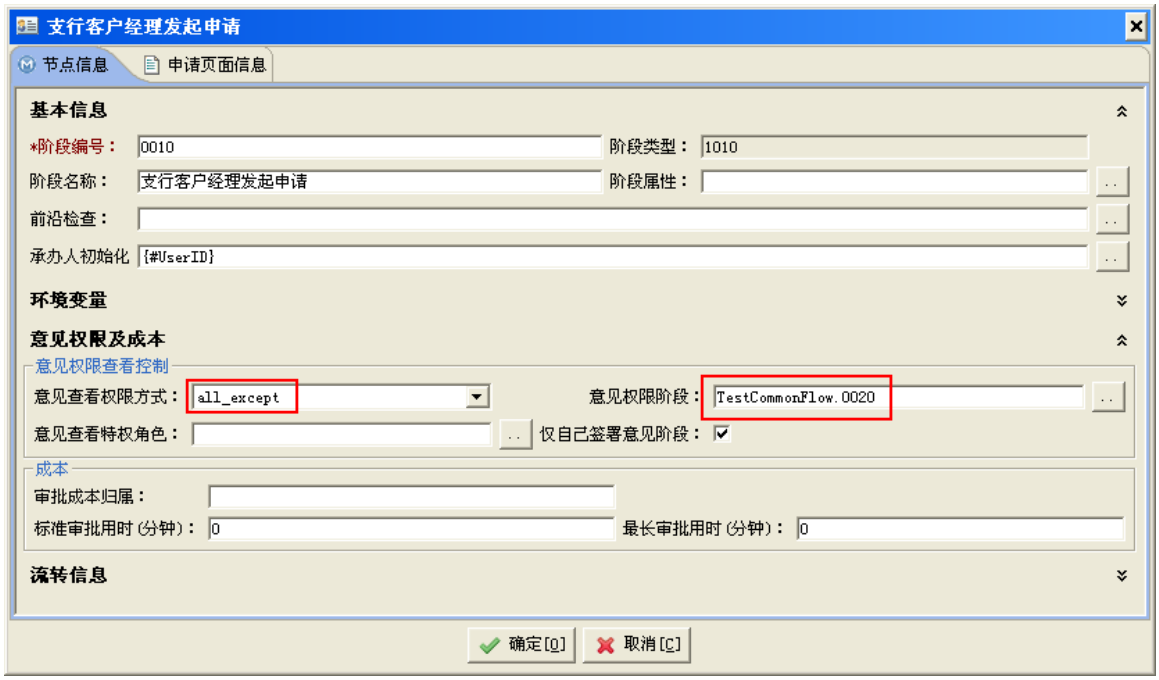
在 User_Info 表中修改 test12 的 Remark=“特权”，当流程走到支行认定并使用 test12 审批时，可以看到提交页面的意见为：批准和否决，如下图：

2.2.7.6 意见查看权限

意见查看规则逻辑在 AmarBPM 原理章节介绍过，这里就不介绍了。本节只简单配置“支行客户经理发起申请”的意见对“支行认定”不可见，使用“权限节点规则”，对于“特殊角色规则”用户可以自行尝试配置。

2.2.7.6.1 配置意见查看规则

1、使用流程编辑器打开 TestCommonFlow 流程，双击“支行客户经理发起申请”节点，点开“意见权限及成本”条目，配置其意见查看权限方式为 all_except，意见权限阶段为 TestCommonFlow.0020，如下图



这里配置的意思是：“支行客户经理发起申请”所签署的意见，其他节点任务都可以看见，

除了 0020 支行认定节点。

2、双击“支行认定”节点，点击“审批页面信息”页，给“已完成的工作”勾选“查看意见”按钮。

3、双击“分行认定”节点，点击“审批页面信息”页，给“已完成的工作”勾选“查看意见”按钮。

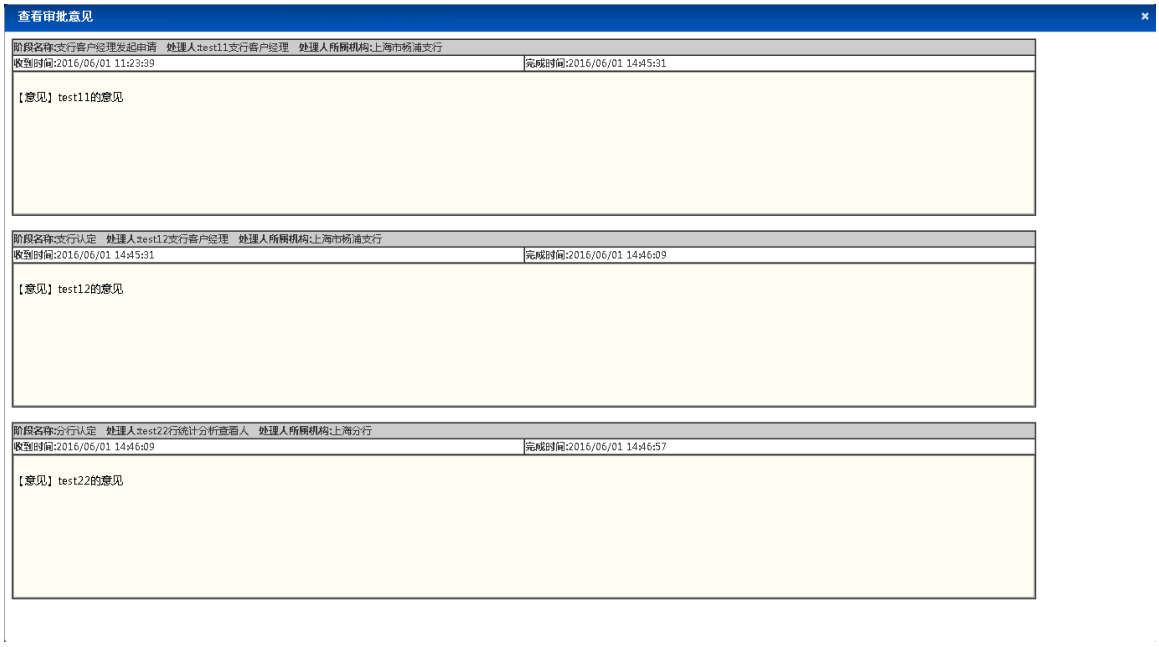
2.2.7.6.2 运行结果

1、新增一笔流程，使用 test11 签署意见为“test11 的意见”，提交 test12。

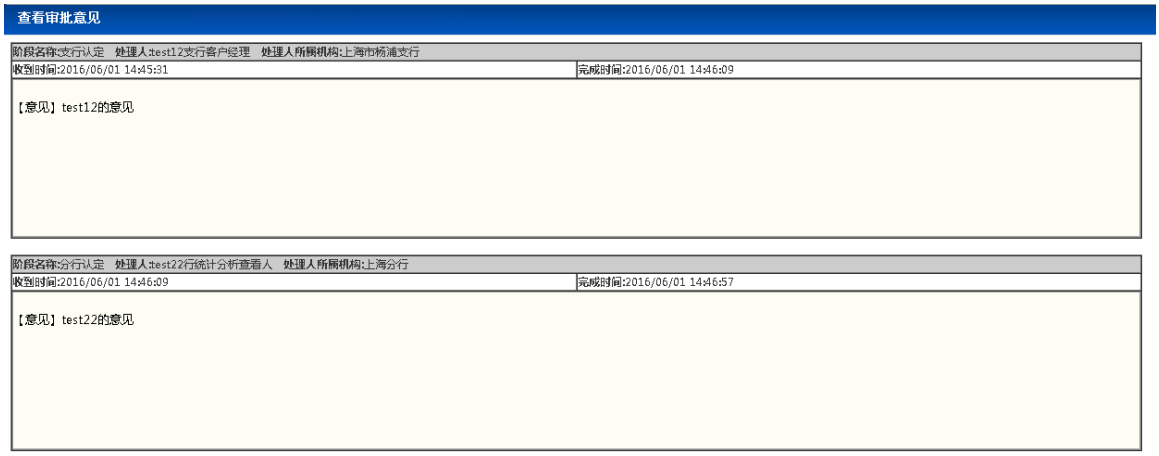
2、使用 test12 登录，签署意见为“test12 的意见”，提交给 test22。

3、使用 test22 登录，签署意见为“test22 的意见”，批准该流程。

4、使用 test22 登录，在“已完成的工作”中选中该流程，点击“查看意见”，可以看到，流程审批过程中的三条意见都能看到：



5、使用 test12 登录，在“已完成的工作”中选中该流程，点击“查看意见”，可以看到只有 2 条意见，其中 test11 的意见被屏蔽了：



3. 并行流程开发

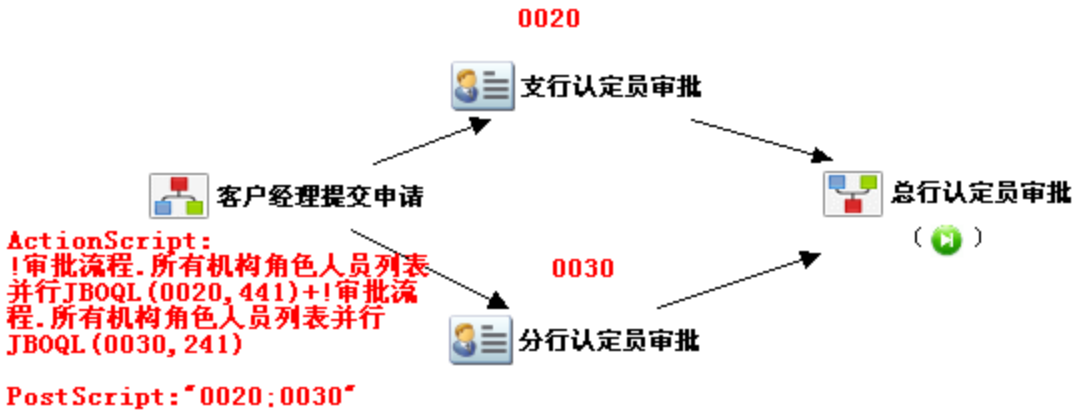
这里所谓的并行流程是指流程模型中含有“并行分支”和“并行聚合”节点的流程，当流程实例的主流转线走到“并行分支”节点后会分成有独立转换逻辑的若干分支，这些分支独立、并行流转，最终在“并行聚合”节点汇聚在一起继续走主流转逻辑。

3.1 并行流程原理

3.1.1 并行流程模型

“并行分支”节点在 Flow_Model 的 NodeType 字段中用 FORK 标记，“并行汇聚”节点的 Nodetype=JOIN，分支节点与汇聚节点必需成对出现，且分支出现在汇聚之前。

“分支”节点的分支逻辑体现在 ActionScript 和 PostScript 中。在 ActionScript 中用加号（+）来合并每个“分支动作脚本”，在 PostScript 中用分号（;）来分隔每个分支后续脚本。下图是一个简单的并行节点示意图：



其中“客户经理提交申请”是分支节点，该节点之后要产生两个流程分支实例：一个走分行审批、一个走支行审批；然后在“总行认定员审批”节点汇聚，回归到主流转线。其中“支行客户经理提交申请”的 ActionScript 配置为：!审批流程.所有机构角色人员列表并行 JBOQL(0020,441)+!审批流程.所有机构角色人员列表并行 JBOQL(0030,241) 其 PostScript 的配置为：’0020;0030’。

3.1.2 ActionScript 配置

并行节点的 ActionScript 返回一个字符数组，字符数组内字符串格式：*PhaseNo|UserID 空格 UserName 空格 RoleID 空格 RoleName* 其中 PhaseNo 用来标记一个动作（提交人）信息属于哪个分支流向。可用的 ClassMethod 方法包括：!审批流程.当前机构角色人员列表并行 JBOQL(PhaseNo,OrgID,UserID,RoleID) 和!审批流程.所有机构角色人员列表并行 JBOQL(PhaseNo,RoleID)。

这两个方法返回的是字符串数组，可以使用加上（+）将几个该函数的返回结构合并到一起。在这两个方法中传入的参数 PhaseNo 将会拼在返回数组内字符串的前面，类似 *0020|pt13 pt13 客户经理 441 支行风险分类认定员* 代表 0020 分支流向的其中一个提交人为 pt13。其中 0020 必需出现在 PostScript 最终执行结果中。

3.1.3 PostScript 配置

并行节点的 PostScript 用半角分号(;)来分隔多个分支流向节点编号 PhaseNo。例如”0020;0030”，代表分支节点有两个流向，分别是到节点 0020 和到节点 0030。

3.1.4 并行流程实例

当流程实例主流转线走到“并行分支”节点时，会生成多个分支任务，每个分支任务有唯一的 ForkNo，分支起始任务的 AllForkNo 记录着所有分支的 ForkNo 的值，以逗号分割。分支任务的 ForkState 状态为 INFORK （分支中）.。

每条分支的流程任务按照模型配置提交下去，单个分支走到汇聚节点时停止流转并将其 ForkState 为 WAIT，所有分支全部走到汇聚节点时，所有分支任务的 ForkState 为 JOIN，这时流程会沿汇聚后的主线继续流转 。

3.2 开发案例

在本节中我们开发一个简单的并行流程，该流程有 5 个节点，其中“客户经理提交申请“是并行节点，客户经理提交申请后由支行和分行独立并行审批，支行分行都完成审批后最终提交到总行审批。

3.2.1 新建流程

在 Eclipse 中打开资源管理→业务流程设置→流程模型，点击从向导添加按钮，新建流程模型、菜单、申请审批类型及主树图。新增流程向导在第四章已经介绍过，这里不在详细说明步骤，只列出最终新增的内容：

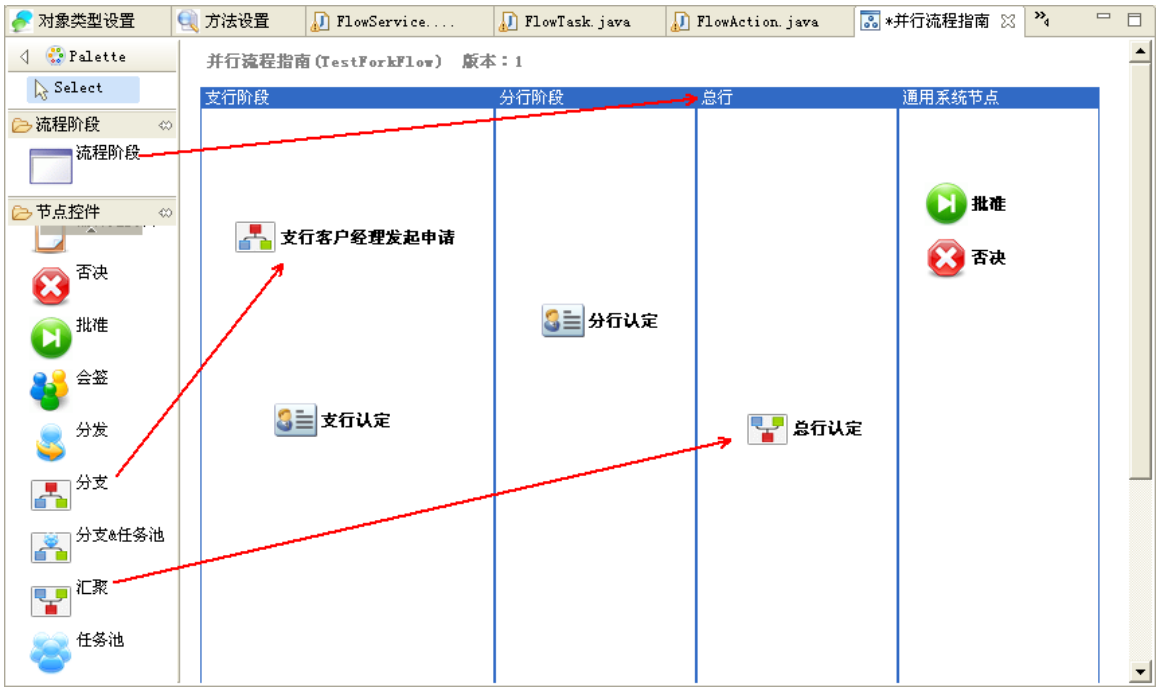
1. 新建流程 TestForkFlow。
2. 新建菜单：流程开发案例→并行流程→并行流程申请、

   流程开发案例→并行流程→并行流程审批。
3. 在/FlowTest/TestForkFlow 下新增两个 JSP 页面，即申请页面和审批页面，显示模板使用系统内置无业务模 FFLOW_OBJECT_OW 和 FLOW_TASK_OW。
4. 新建申请审批类型、主树图及主树图节点。

3.2.2 流程模型配置

使用图形编辑器编辑我们刚才新增的流程 TestForkFlow，删掉开始节点节点，从左侧

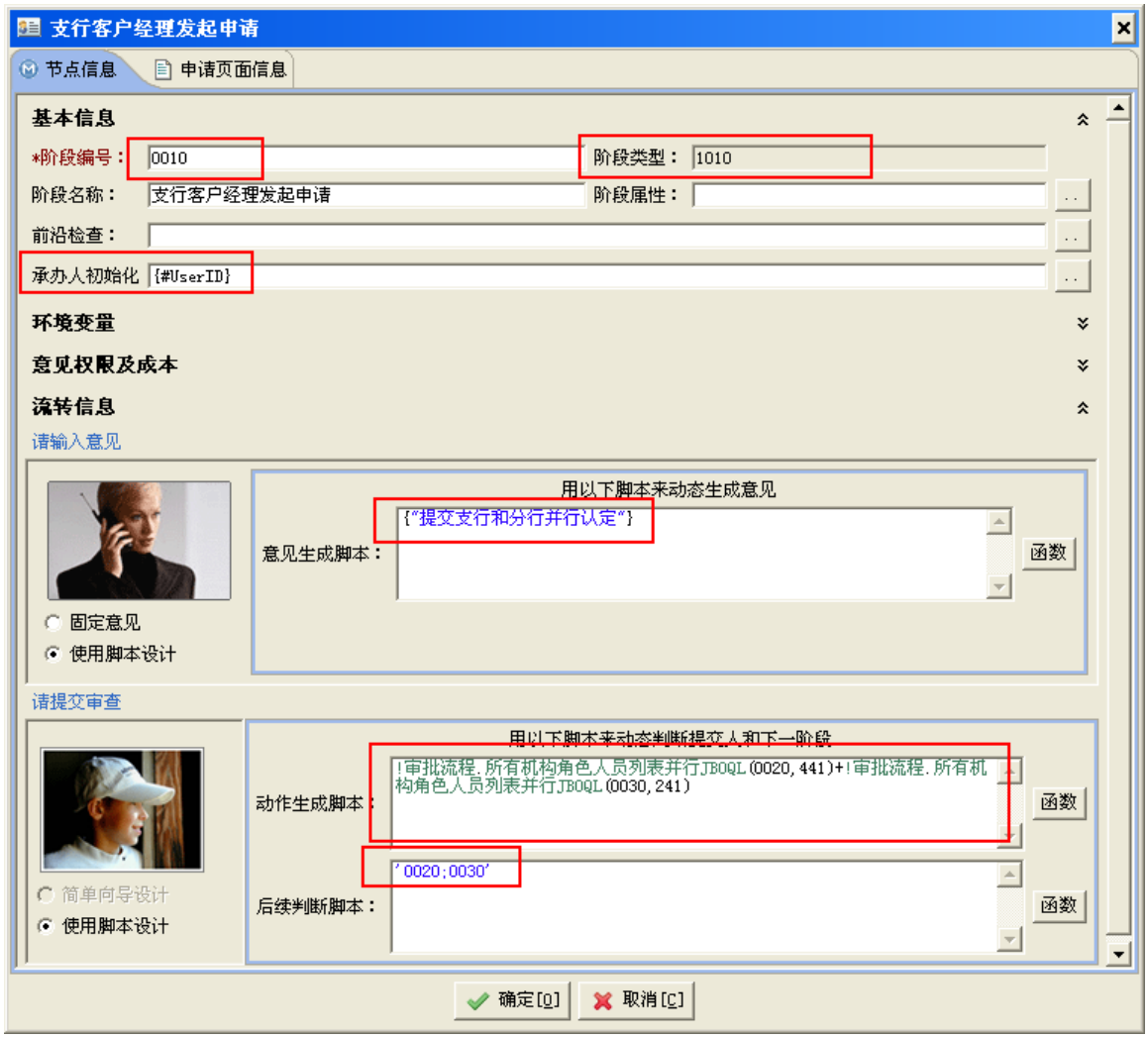
画板中拖入一个流程阶段（不必须）、“分支“节点、”汇聚“节点，单击节点可修改节点名称，如下图：



1. 单击“支行客户经理发起申请”节点，在悬浮操作栏点击最后一个按钮

来配置其 PhaseType=1010。

2. 双击“支行客户经理发起申请”节点，修改其阶段编号=0010，承办人初始化因为是第一个节点所以配成{#UserID}代表当前登录用户，配置其意见脚本、动作脚本和后续脚本：



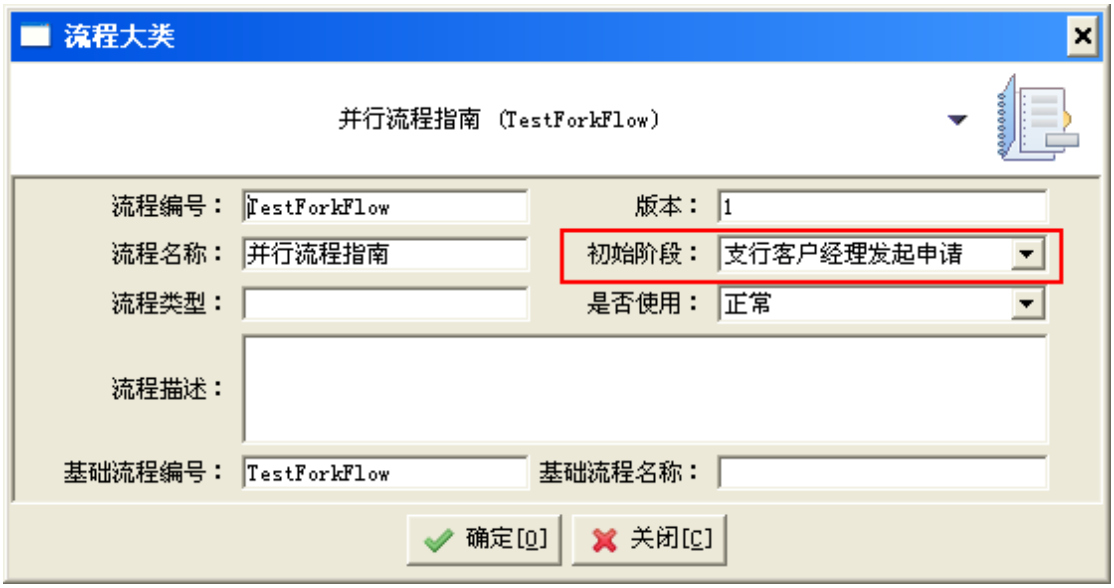
ActionScript 为：!审批流程.所有机构角色人员列表并行 JBOQL(0020,441)+!审批流程.所有机构角色人员列表并行 JBOQL(0030,241)其 PostScript 的配置为：'0020;0030'。

3. 双击支行认定，配置其流转到“总行认定“节点，其 ActionScript 为!审批流程.所有机构角色人员列表 JBOQL(041)。

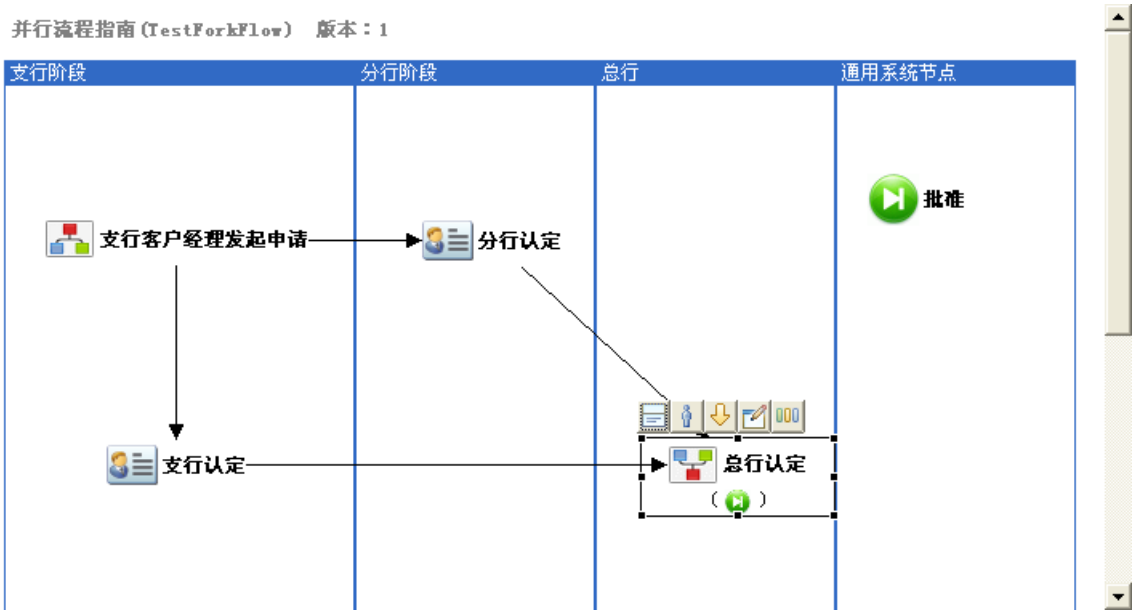
4. 双击分行认定，配置其流转到“总行认定“节点，其 ActionScript 为!审批流程.所有机构角色人员列表 JBOQL(041)。

5. 双击总行认定，配置其流转到“批准“节点。

6. 左键单击流程图图形之外的空白区域，右键→配置流程图属性，将流程的初始阶段配置成“支行客户经理发起申请”节点。



最终流程图为：

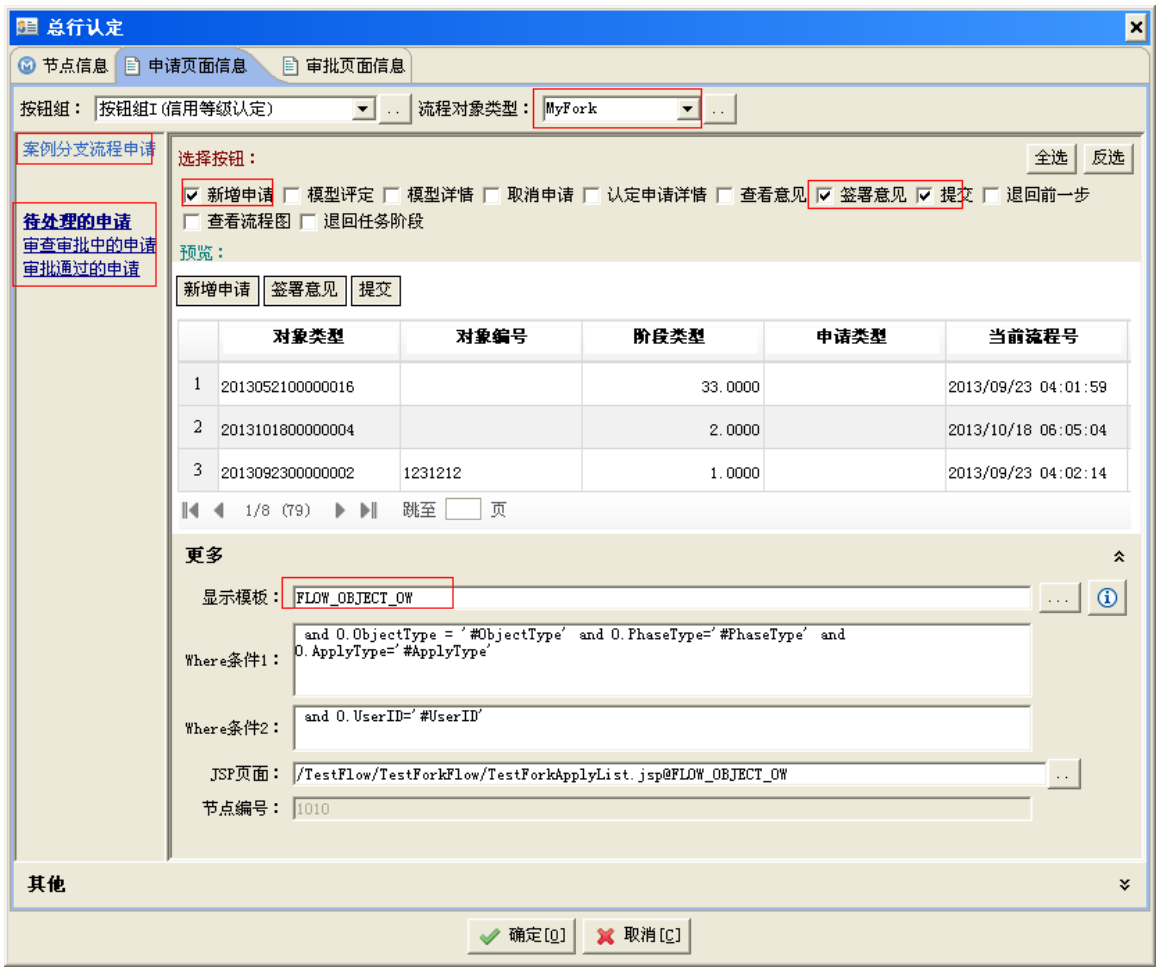


3.2.3 配置申请类型及主树图

双击支行客户经理发起的申请节点，切换到“申请页面“标签：

- 删除“被否决的申请”树图。
- 给“待处理的申请”树图勾选“新增申请”、“签署意见”和“提交”3 个按钮。
- 修改流程对象类型为“MyFork”。
- 修改树图标题为“案例分支流程申请”。

最终配置结果界面如下：

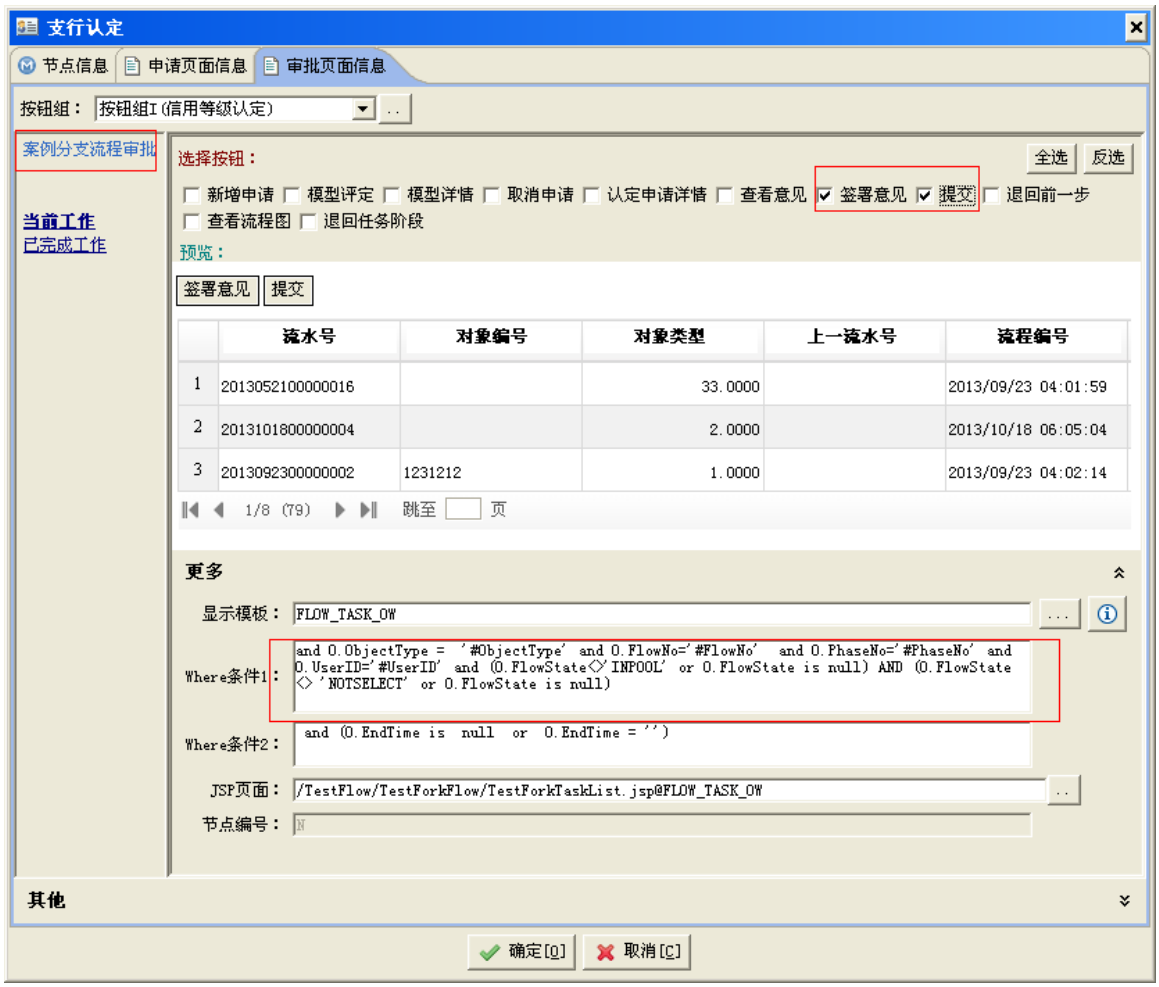


3.2.4 配置审批类型及主树图

双击支行认定节点，切换到“审批页面“标签：

- 给“当前工作”树图勾选 “签署意见”和“提交”2 个按钮。
- 修改树图标题为“案例分支流程审批”。

最终配置结果界面如下：



其中 where 条件中：and (O.FlowState<>'INPOOL' or O.FlowState is null) AND (O.FlowState <> 'NOTSELECT' or O.FlowState is null)　这句 Where 是 ADE 自动生成的，用来过滤没被获取的任务池任务，本案例没有任务池节点，可以不加这串条件。

分行认定和总行认定的按钮组做类似配置，即给当前工作勾上“签署意见”和“提交”按钮。

3.2.5 运行流程

启动 Tomcat，用 test11 登录系统点击菜单“流程开发案例”→并行流程→并行流程申请，即可打开申请页面，点击“新增申请”可新增一笔无业务的流程。

1. 签署意见后提交，其提交页面如下：

提交之后将产生 2 条分支实例（Flow_Task 中）两条数据，我们称之为“支行 Task”和“分行 Task”其 ForkState 均为 INFORK。

2. 使用 test50 登录，将该“支行 Task”提交给 test53，将在 Flow_Task 产生一条“总行 Task”数据，其 ForkState 为 Wait，其 UserID 为*test53(这样该条数据不会被申请审批页面查到)。在之后的任务流转中“支行 Task”和“分行 Task”的 ForkState 会依然保持为 INFORK。

3. 使用 test51 登录，将“分行 Task”提交给 test53（注意：因为支行 Task 已经提交给 test53 了，所以分行 Task 也必需提交给 test53），系统会再生成一条数据“总行 Task2”（而不是再原数据上修改）其 ForkState 为 JOIN，USERID 为 test53。

4. 使用 test53 登录，批准该申请，流程至此结束。

4. 待审会流程开发

待审会描述这样的场景：审批业务进行到特定阶段时需要走一个待审会讨论流程，这个

讨论由待审会秘书发起，提交给待审会委员，待审会委员各自发表意见汇总到秘书处，然后秘书根据讨论结果决定审批业务接下来如何流转。

在 ALS 中待审会流程即包含“分发”和“会签”节点的流程。“分发”节点代表贷审会秘书分发阶段，“会签”节点代表贷审会委员审批阶段。当流程走到“分发”节点时，待审会秘书在提交页面选择若干委员并将流程提交给他们。每个委员签署自己的意见，并将意见提交给秘书，秘书根据委员意见签署最终意见并提交，流程继续往下流转。

4.1 待审会流程模型

待审会流程即包含“分发”和“会签”节点的流程。“分发”节点即 NodeType=Distribute 的节点，会签节点即 NodeType=Meeting 的节点。“分发”和“会签”必需成对出现，且“分发”节点之后必需是“会签”节点。

“分发”节点的 ActionScript 和 PostScript 和普通流程节点类似。“会签”节点一般会根据秘书提交的意见来生成 ChoiceScript，继而在 PostScript 中决定流程继续流转还是否决。一般 ChoiceScript 类似：IF(!审批流程.秘书汇总意见(#SerialNo)='2',{'否决'},{'提交总行领导'})即调用审批流程.秘书汇总意见该方法返回秘书最终汇总的意见代码，1 代表同意，2 代表否决。

4.2 待审会流程实例

当流程走到“分发”节点秘书提交后，产生多条贷审会委员任务（Flow_Task）其 FlowState 任务状态为 MEETING，同时会生成一条秘书的汇总任务 FlowState 状态为 COLLECT。即秘书将待审会任务提交给委员后，自己也产生了一条汇总任务。

待审会委员任务被签署意见并提交后其 FlowState 任务状态变为 VOTED,秘书汇总意见并提交后，所有未提交的委员的 FlowState 任务状态变为 UNVOTED。

4.3 开发案例

4.3.1 新建流程

在 Eclipse 中打开资源管理→业务流程设置→流程模型，点击从向导添加按钮，新建流程模型、菜单、申请审批类型及主树图。新增流程向导在第四章已经介绍过，这里不在

详细说明步骤，只列出最终新增的内容：

5. 新建流程 TestMeetingFlow。

6. 新建菜单：流程开发案例→待审会流程→申请、

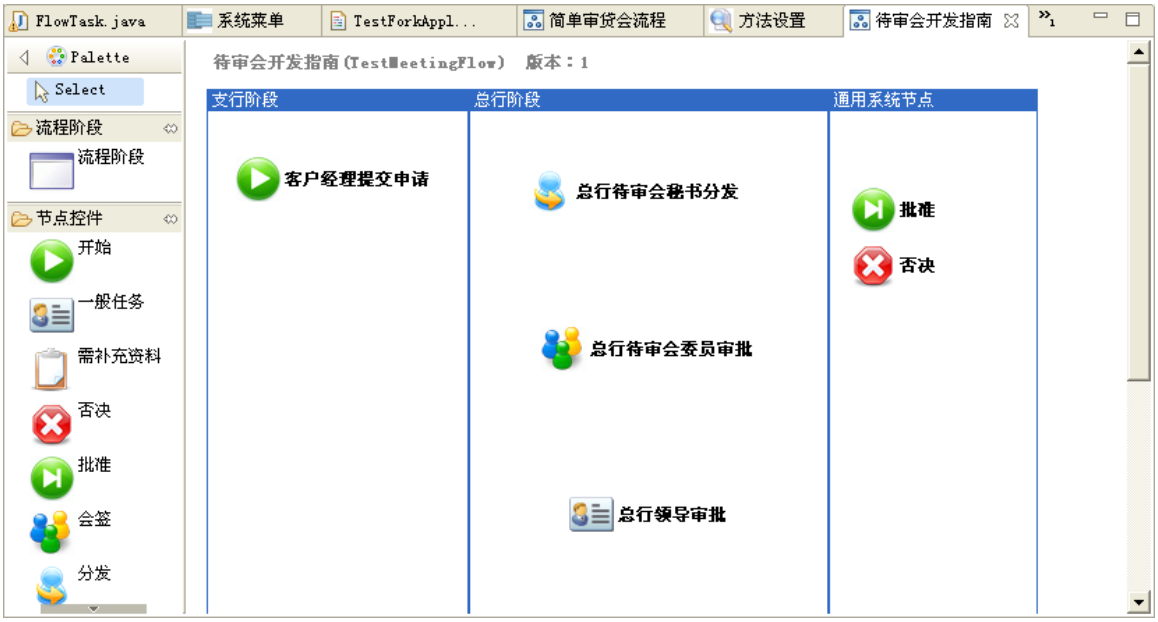
   流程开发案例→待审会流程→审批。

7. 在/FlowTest/TestMeetingFlow 下新增两个 JSP 页面，即申请页面和审批页面，显示模板使用系统内置无业务模 FFLOW_OBJECT_OW 和 FLOW_TASK_OW。

8. 新建申请审批类型、主树图及主树图节点。

4.3.2 配置流程模型

使用图形编辑器编辑我们刚才新增的流程 TestMeetingFlow，删除支行认定节点，单击“分行阶段”流程阶段大框将其改名为“总行阶段”（不必须）、从左侧画板拖入“会签”、“分发”节点，单击节点可修改节点名称，如下图：



1. 双击客户经理提交申请节点，配置 ActionScript 为!审批流程.所有机构角色人员列表JBOQL(015)即提交到总行委员会秘书，配置 PostScript 为提交到“总行待审会秘书分发”节点。

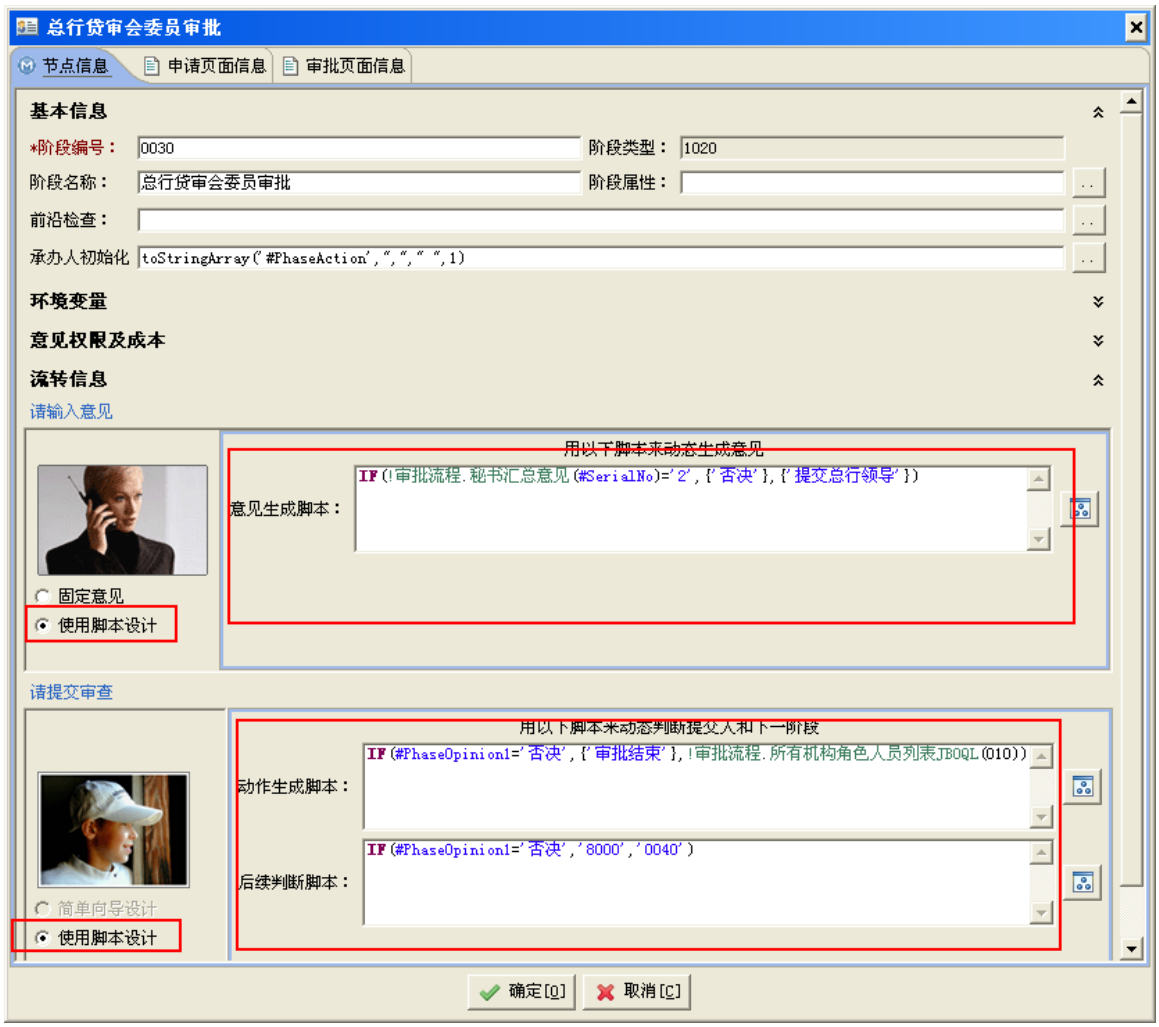
2. 双击总行待审会秘书分发节点，配置 ActionScript 为：!审批流程.所有机构角色人员列表

JBOQL(014)即提交到总行审批委员会委员，配置 PostScript 为提交到“总行待审会委员审批”节点。

3. 双击总行待审会委员审批节点，配置 ChoiceScript 为：*IF(!审批流程.秘书汇总意见(#SerialNo)='2',{'否决'},{'提交总行领导'})*

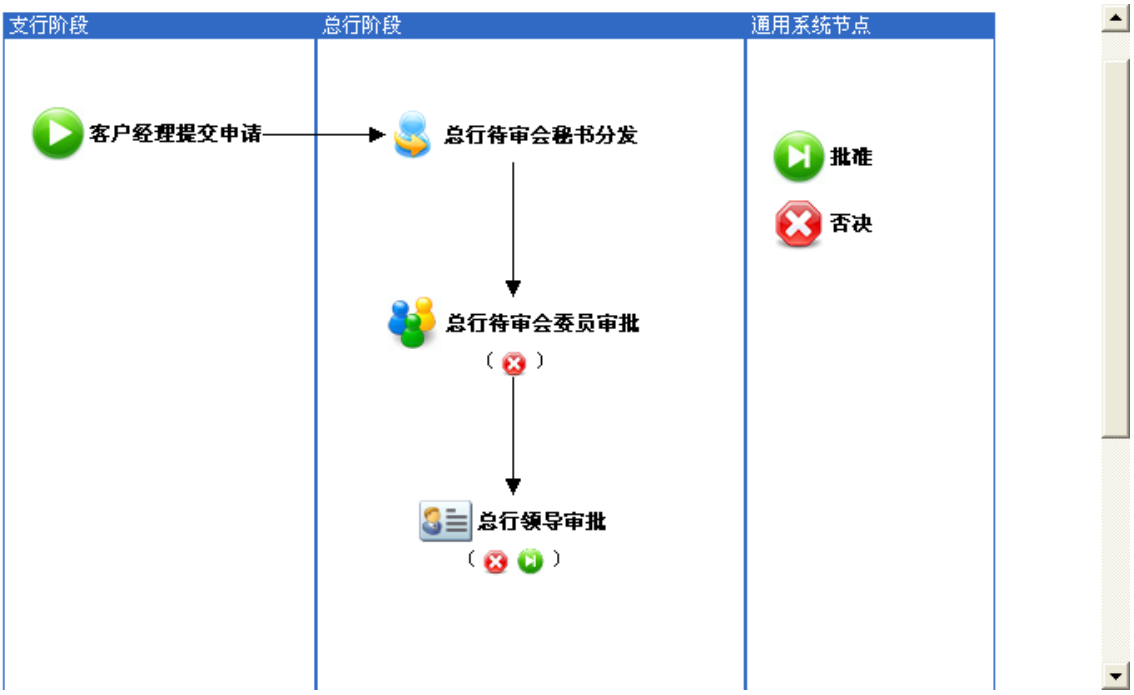
配置 ActionScript 为：IF(#PhaseOpinion1='否决',{'审批结束'},!审批流程.所有机构角色人员列表 JBOQL(010))

配置 PostScript 为 IF(#PhaseOpinion1='否决','8000','0040')。0040 为“总行领导审批”节点的 PhaseNo。如下图：



其中*!审批流程.秘书汇总意见(#SerialNo)='2'* 查询秘书汇总意见是不是为“否决”，如果为否决，则 ActionScript 为“审批结束”同时 PostScript 流转到 8000 即否决节点。如果秘书汇总意见不是“否决”，则流转到“总行领导审批”节点。

4. 双击总行领导审批节点，配置其可以“通过”或“否决”。最终配置流转结果如下：

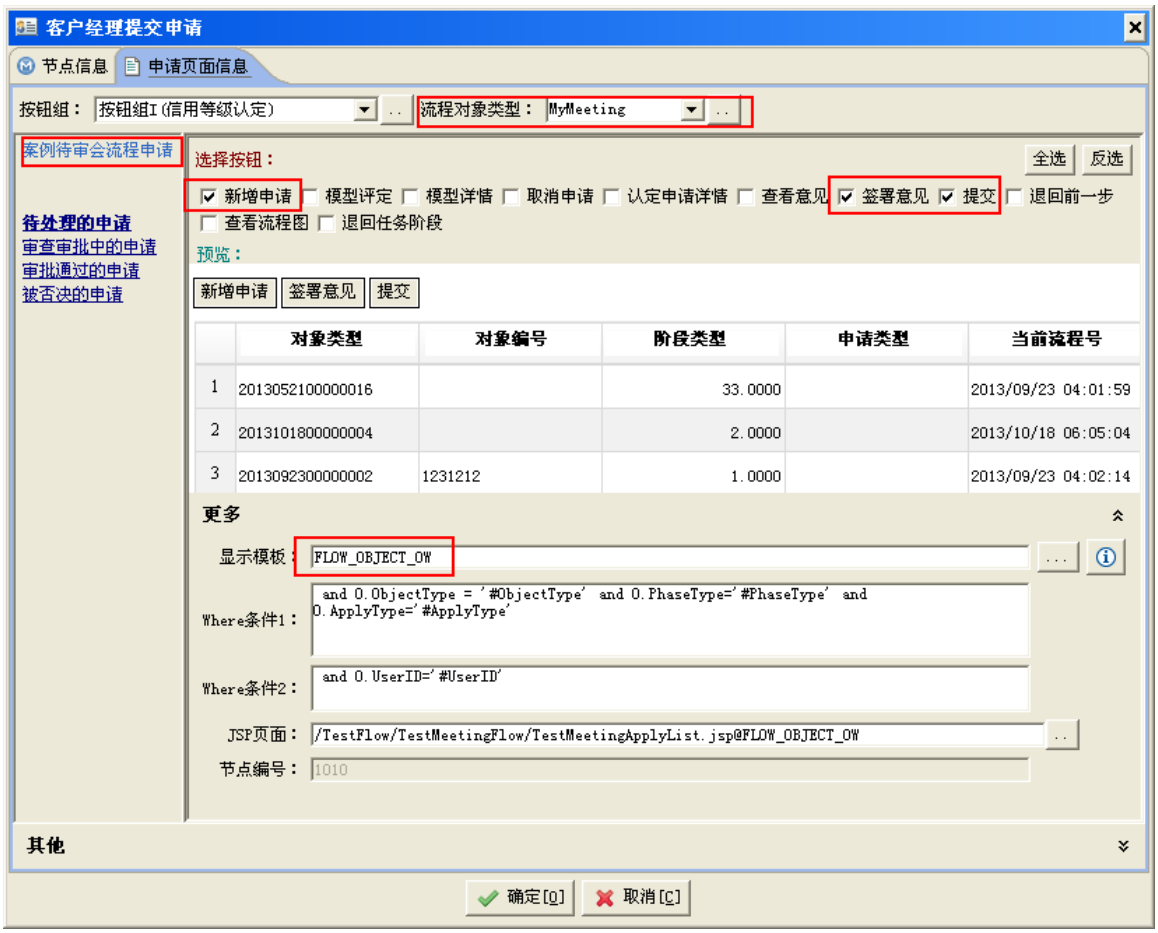


4.3.3 配置申请类型及主树图

双击支行客户经理提交申请节点，切换到“申请页面“标签：

- 给“待处理的申请”树图勾选“新增申请”、“签署意见”和“提交”3 个按钮。
- 修改流程对象类型为“MyMeeting”。
- 修改树图标题为“案例待审会流程申请”。

最终配置结果界面如下：

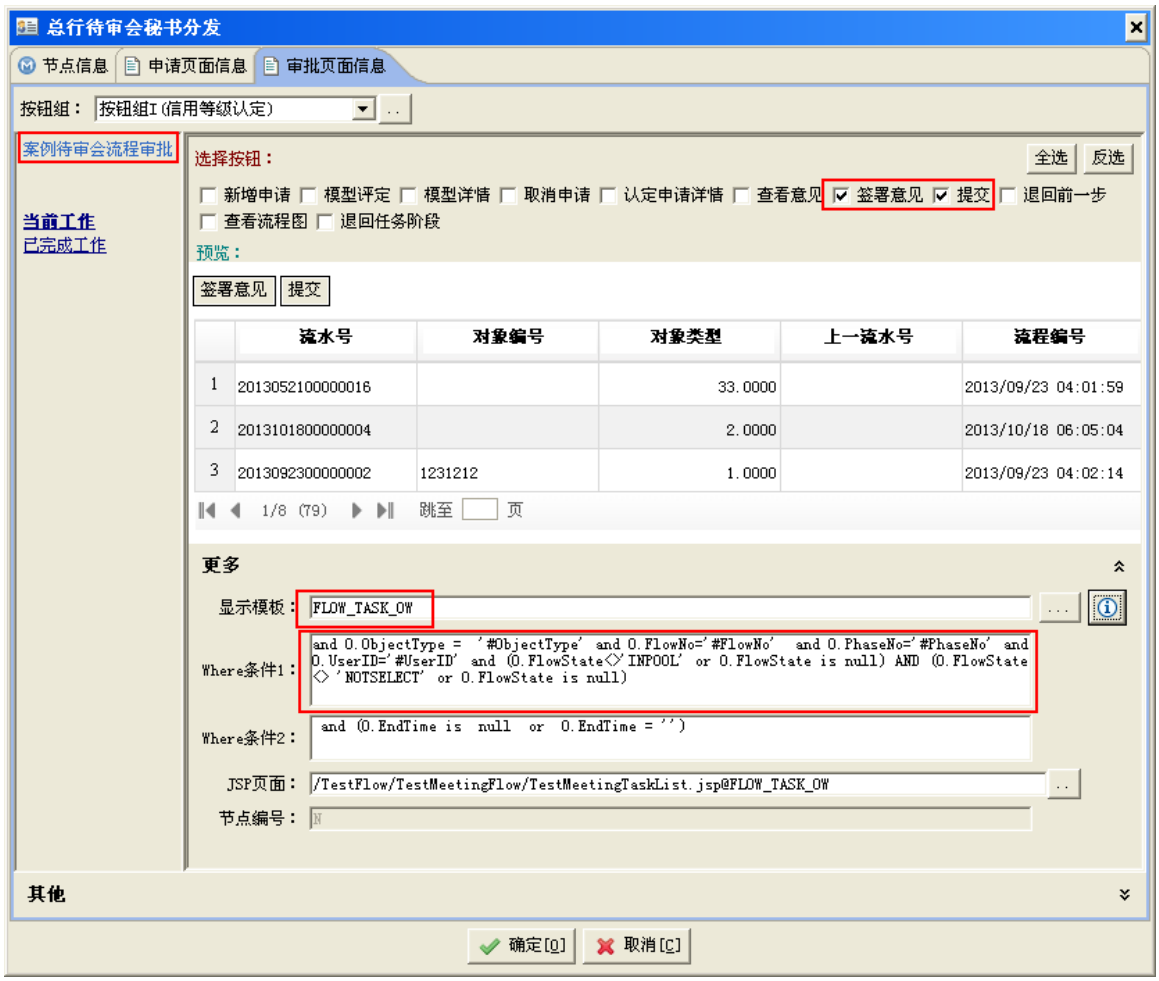


4.3.4 配置审批类型及主树图

双击总行待审会秘书分发节点，切换到“审批页面“标签：

- 给“当前工作”树图勾选 “签署意见”和“提交”2 个按钮。
- 修改树图标题为“案例待审会流程审批”。

最终配置结果界面如下：



其中 where 条件中的：and (O.FlowState<>'INPOOL' or O.FlowState is null) AND (O.FlowState <> 'NOTSELECT' or O.FlowState is null) 这句 Where 是 ADE 自动生成的，用来过滤没被获取的任务池任务，本案例没有任务池节点，可以不加这串条件。

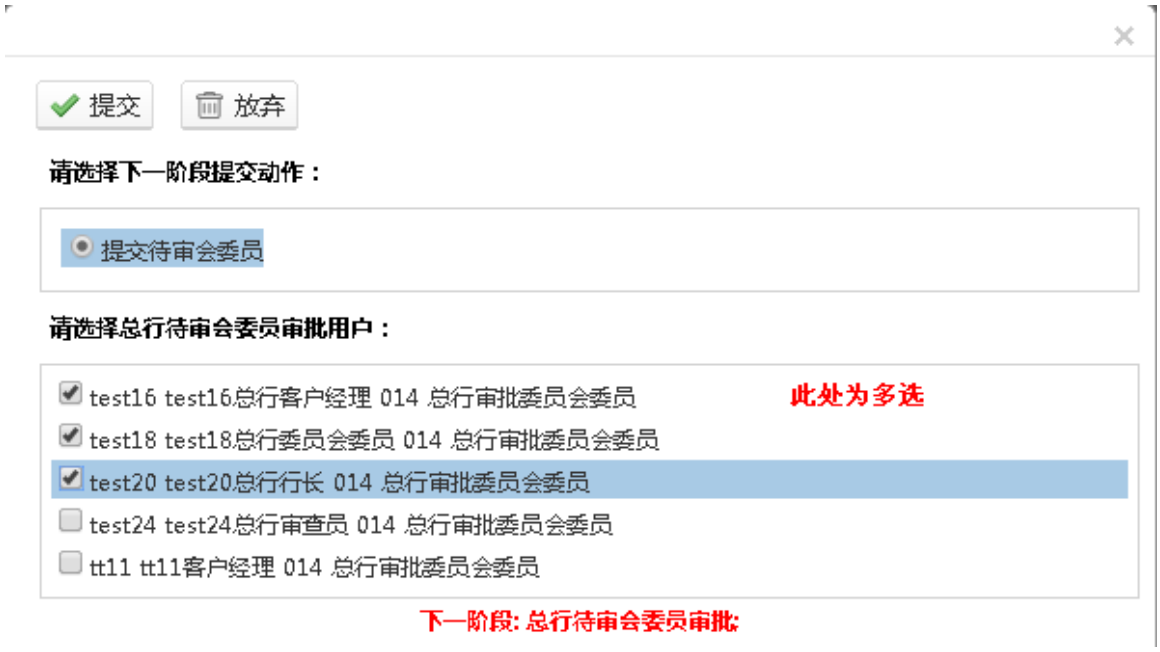
总行待审会委员审批和总行领导审批的按钮组做类似配置，即给当前工作勾上“签署意见”和“提交”按钮。

4.3.5 运行流程

启动 Tomcat，用 test11 登录系统点击菜单“流程开发案例”→待审会流程→申请，即可打开申请页面，点击“新增申请”可新增一笔无业务的流程。

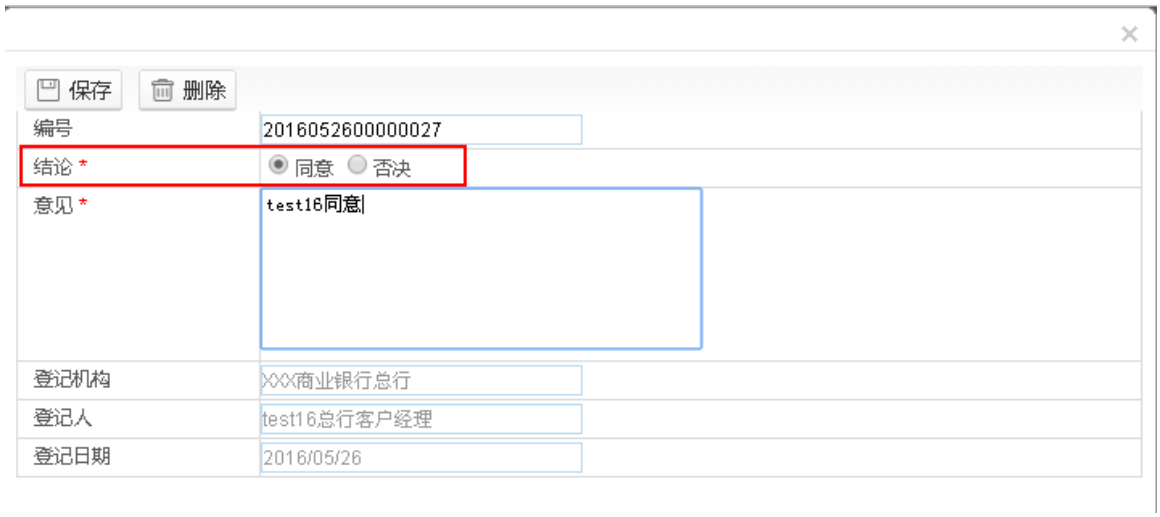
1. 签署意见后提交，提交给 test19。此时 Flow_Task 会生成一条秘书分发 Task，其 FlowState 为 DISTRIBUTE。

2. 用 test19 登录，继续提交该流程，提交页面如下：



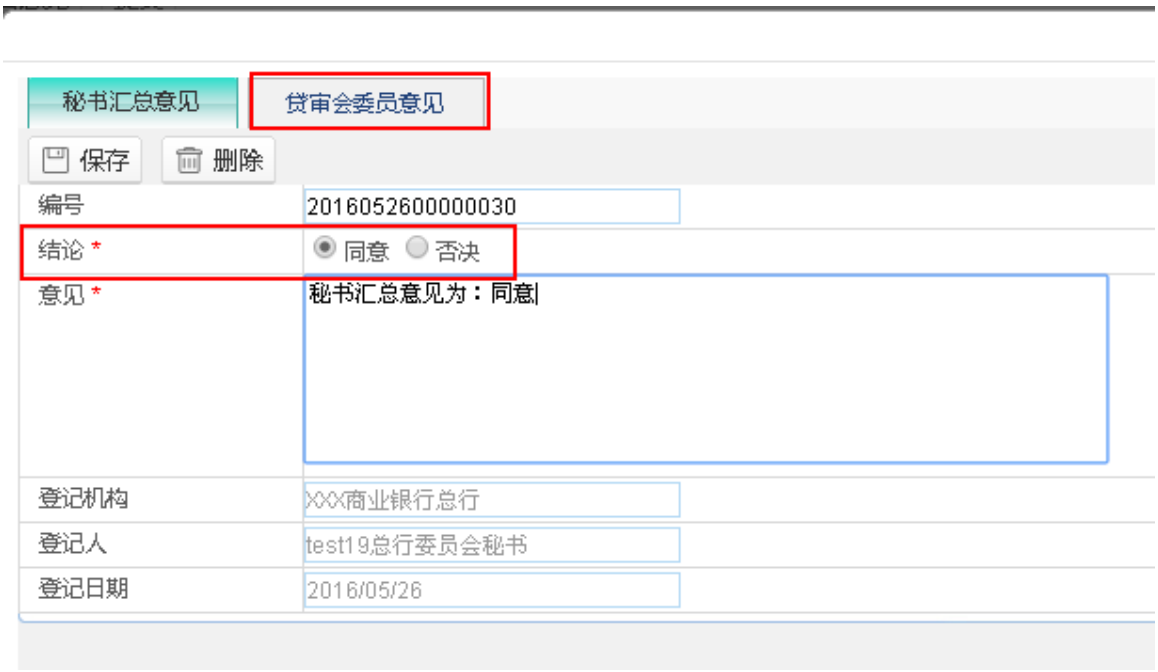
选择 3 个委员 test16,test18,test20，提交掉。后台会向 Flow_Task 中生成 4 条数据，其中 3 条是委员会任务其 FlowState 均为 MEETING（代表委员正在投票）。另外还会生成一条秘书汇总任务其 FlowState 为 COLLECT，且 UserID=test19。

3. 使用 test16 登录系统进行委员投票，即选中流程任务签署“同意”意见，并提交流程。这个签署意见的界面和以前签署意见界面有点不同，即有一个“结论”项，点击同意即可。



本步骤执行完后，会将 test16 对应的委员会任务的 FlowState 置成 VOTED。

4. 使用 test19 登录，进行秘书汇总操作。即选择流程任务，填写最终意见（同意或否决），并提交流程。秘书汇总阶段填写的意见页面可以查看到其他委员的意见：



本步骤（秘书汇总任务）提交给 test35 之后，test18 和 test20 两个委员会任务（Flow_Task）的 FlowState 被置成 UNVOTED。

5. 使用 test35 登录，批准该流程，至此流程结束。

5. 任务池流程开发

任务池描述这样的场景：当审批任务走特定阶段时，不指定具体的审批人，而是由拥有权限的审批人主动获取任务并处理。

5.1 任务池流程原理

5.1.1 任务池流程模型

在 ALS 中任务池流程指模型中包含“任务池”节点流程，“任务池”节点的 NodeType 用 POOL 来标记。其他属性如 ActionScript 和 PostScript 和其他普通节点一致。

5.1.2 任务池流程实例

当流程走到“任务池”节点的前一个节点时，在前一个节点的提交页面上不用选择提交人，系统会执行 ActionScript 生成所有的提交人，并为所有提交人生成“待获取任务”

（Flow_Task 中），其 FlowState 为 INPOOL，表示在任务池中。

当任何一个有权限的用户 “从任务池获取任务”后，其对应的任务池阶段任务的 FlowState 会从 INPOOL 变为 SELECT, 其他的任务池阶段的任务状态变为 NOTSELECT，其他用户将不能再次获取。

被获取的任务可以被“返回任务池”，此时该池任务的 FlowState 会由 SELECT 变成 INPOOL，同时其他同批其他任务池任务（即 RelativeSerialno 相同的其他池任务）的 FlowState 将由 NOTSELECT 变回 INPOOL。代表该任务重新回到任务池中，可以被再次获取。

任务池任务节点被提交后，同批其他任务池任务（即 RelativeSerialno 相同的且 FlowState 为 NOTSELECT）将被删除。

5.2 开发案例

5.2.1 新建流程

在 Eclipse 中打开资源管理→业务流程设置→流程模型，点击从向导添加按钮，新建流程模型、菜单、申请审批类型及主树图。新增流程向导在第四章已经介绍过，这里不在详细说明步骤，只列出最终新增的内容：

1. 新建流程 TestPoolFlow。
2. 新建菜单：流程开发案例→任务池流程→申请、

   流程开发案例→任务池流程→审批。
3. 在/FlowTest/TestPoolFlow 下新增两个 JSP 页面，即申请页面和审批页面，显示模板使用系统内置无业务模 FFLOW_OBJECT_OW 和 FLOW_TASK_OW。
4. 新建申请审批类型、主树图及主树图节点。

5.2.2 配置流程模型

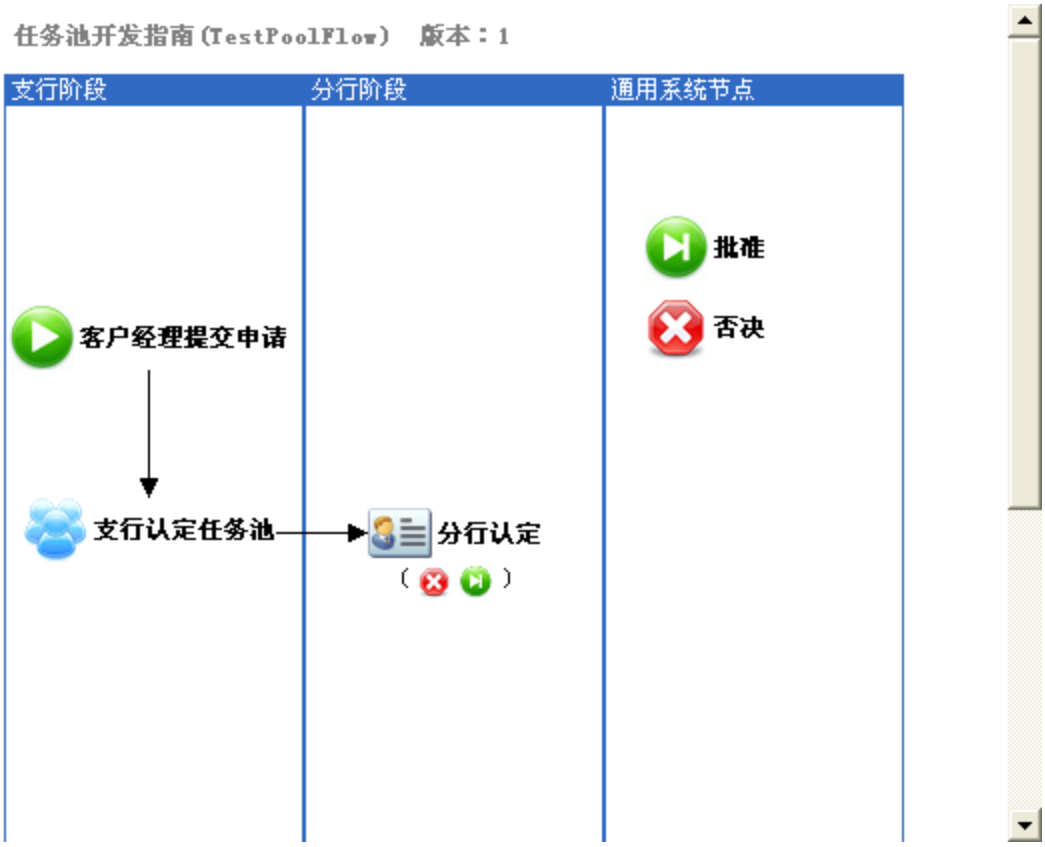
使用图形编辑器编辑我们刚才新增的流程 TestMeetingFlow，删除支行认定节点，从

左侧画板拖入“任务池”节点，单击节点可修改节点名称，修改后的节点如下图：

1. 双击支行客户经理申请节点，配置其 ActionScript 为：!审批流程.所有机构角色人员列表 JBOQL(441)即支行认定员，配置 PostScript 为提交到“支行认定任务池”节点。这里演示任务池需要提交人员有多个，所以配成“所有机构人员列表”。

2. 双击支行认定任务池节点，配置其 ActionScript 为：!审批流程.所有机构角色人员列表 JBOQL(241)即分行认定员，配置其 PostScript 为提交到“分行认定”节点。

3. 双击分行认定节点，配置其可以批准或否决。最终流程图如下：

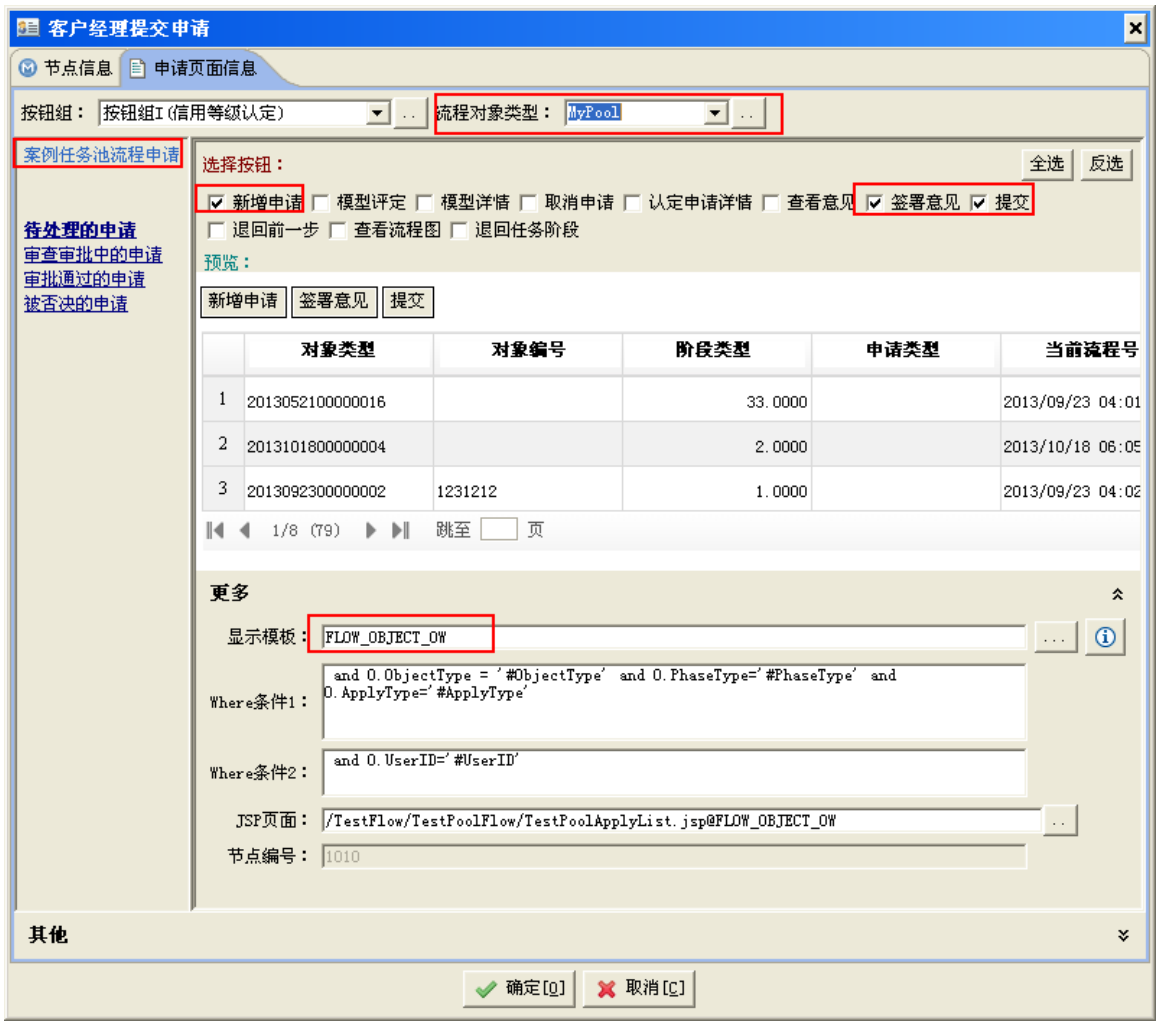


5.2.3 配置申请类型及主树图

双击客户经理提交申请节点，切换到“申请页面“标签：

- 给“待处理的申请”树图勾选“新增申请”、“签署意见”和“提交”3 个按钮。
- 修改流程对象类型为“MyPool”。
- 修改树图标题为“案例任务池流程申请”。

最终配置结果界面如下：

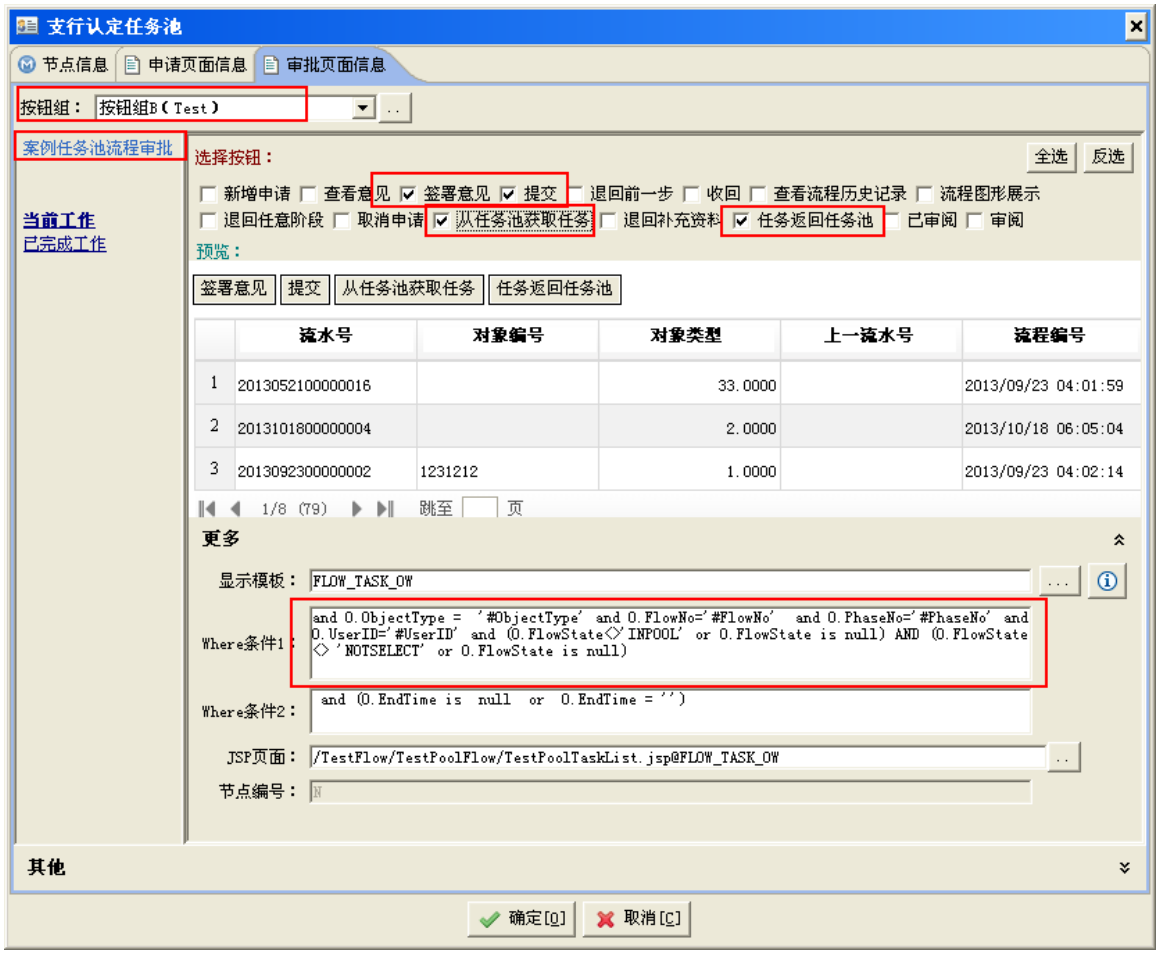


5.2.4 配置审批类型及主树图

双击支行认定任务池节点，切换到“审批页面“标签：

- 选择按钮为：按钮组 B（Test），因为这里我们要用到任务池相关按钮。
- 给“当前工作”树图勾选 “签署意见”和、提交”、“从任务池获取任务”和“任务返回任务池”4 个按钮。
- 修改树图标题为“案例任务池流程审批”。

最终配置结果界面如下：



其中 where 条件为：and (O.FlowState<>'INPOOL' or O.FlowState is null) AND (O.FlowState <> 'NOTSELECT' or O.FlowState is null)　这句 Where 是 ADE 自动生成的，用来过滤没被获取的任务池任务。

双击支行认定节点给按钮组做类似配置，即给当前工作勾上“签署意见”和“提交”按钮。

5.2.5 运行流程

启动 Tomcat，用 test11 登录系统点击菜单“流程开发案例”→任务池流程→申请，即可打开申请页面，点击“新增申请”可新增一笔无业务的流程。

1. 签署意见后提交，因为下一节点时任务池节点，所以提交页面上无需选择提交人：

提交之后系统会向 Flow_Task 插入 3 条池任务，且其 FlowState 为 INPOOL。

2. 使用 test50 登录，在审批页面点击“从任务池获取任务”，即可获取任务。此时 test50 对应的池任务的 FlowState 被设置为 SELECT，而 pt13,、tt12 对应的池任务的 FlowState 被设置成 NOTSELECT。

3. 还在原来的界面 test50 选择刚获取的任务，然后点击“任务返回任务池”。此时 test50、pt13、tt12 对应的池任务的 FlowState 都被设成 INPOOL。

4. 再次点击“从任务池获取任务”，签署意见并提交给 test51。提交后 pt13、tt12 对应的池任务被删除。

5. 使用 test51 登录，批准该流程，至此流程运行结束。

6. 分支任务池开发

6.1 分支任务池原理

分支任务池是流程中有“分支&任务池”节点，该节点同时拥有“分支”节点和任务池节点特性。当流程流转到该节点时，用户需要通过任务池的方式获取任务，当用户提交任务时需要向提交并行流程那样为每个分支分别选择提交人。

6.1.1 分支任务池模型

在 ALS 中分支任务池指模型中包含“分支&任务池”节点流程，节点的 NodeType 用 FORK&POOL 来标记。其 ActionScript 和 PostScript 的配置同“分支”节点的配置一致。

6.1.2 分支任务池实例

当流程走到“任务池”节点的前一个节点时，在前一个节点的提交页面上不用选择提交人，系统会执行 ActionScript 生成所有的提交人，并为所有提交人生成“待获取任务”（Flow_Task 中），其 FlowState 为 INPOOL，表示在任务池中。

当任何一个有权限的用户 “从任务池获取任务”后，其对应的任务池阶段任务的 FlowState 会从 INPOOL 变为 SELECT, 其他的任务池阶段的任务状态变为 NOTSELECT，其他用户将不能再次获取。

被获取的任务可以被“返回任务池”，此时该池任务的 FlowState 会由 SELECT 变成 INPOOL，同时其他同批其他任务池任务（即 RelativeSerialno 相同的其他池任务）的 FlowState 将由 NOTSELECT 变回 INPOOL。代表该任务重新回到任务池中，可以被再次获取。这些特性同任务池是一样的。

当用户提交该流程时，会生成多个分支任务，每个分支任务有唯一的 ForkNo, 分支起始任务的 AllForkNo 记录着所有分支的 ForkNo 的值，以逗号分割。分支任务的 ForkState 状态为 INFORK （分支中）.。

每条分支的流程任务按照模型配置提交下去，单个分支走到汇聚节点时停止流转并将其 ForkState 为 WAIT, 所有分支全部走到汇聚节点时，所有分支任务的 ForkState 为 JOIN，这时流程会沿汇聚后的主线继续流转 。这些特性和并行流程是一样的。

6.2 开发案例

6.2.1 新建流程

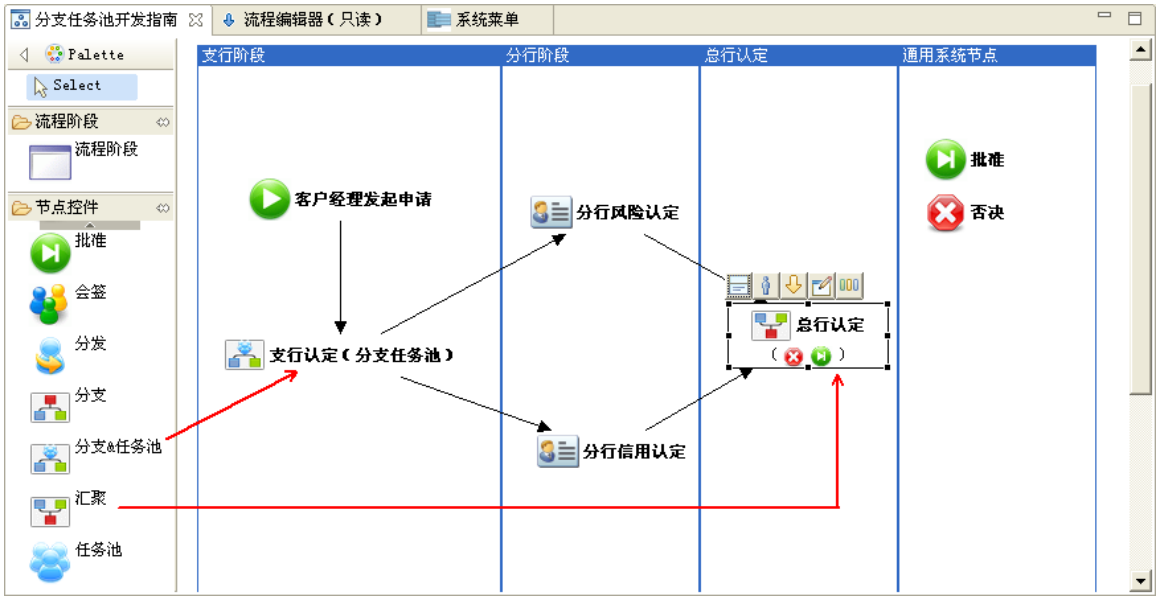
在 Eclipse 中打开资源管理→业务流程设置→流程模型，点击从向导添加按钮，新建流程模型、菜单、申请审批类型及主树图。新增流程向导在第四章已经介绍过，这里不在详细说明步骤，只列出最终新增的内容：

5. 新建流程 TestForkPoolFlow。
6. 新建菜单：流程开发案例→分支任务池流程→申请、

   流程开发案例→分支任务池流程→审批。
7. 在/FlowTest/TestForkPoolFlow 下新增两个 JSP 页面，即申请页面和审批页面，显示模板使用系统内置无业务模 FFLOW_OBJECT_OW 和 FLOW_TASK_OW。
8. 新建申请审批类型、主树图及主树图节点。

6.2.2 配置流程模型

使用图形编辑器编辑我们刚才新增的流程 TestForkPoolFlow，从左侧画板拖入“分支&任务池”节点，单击节点可修改节点名称，按照步骤配置模型，最终流程图如下：



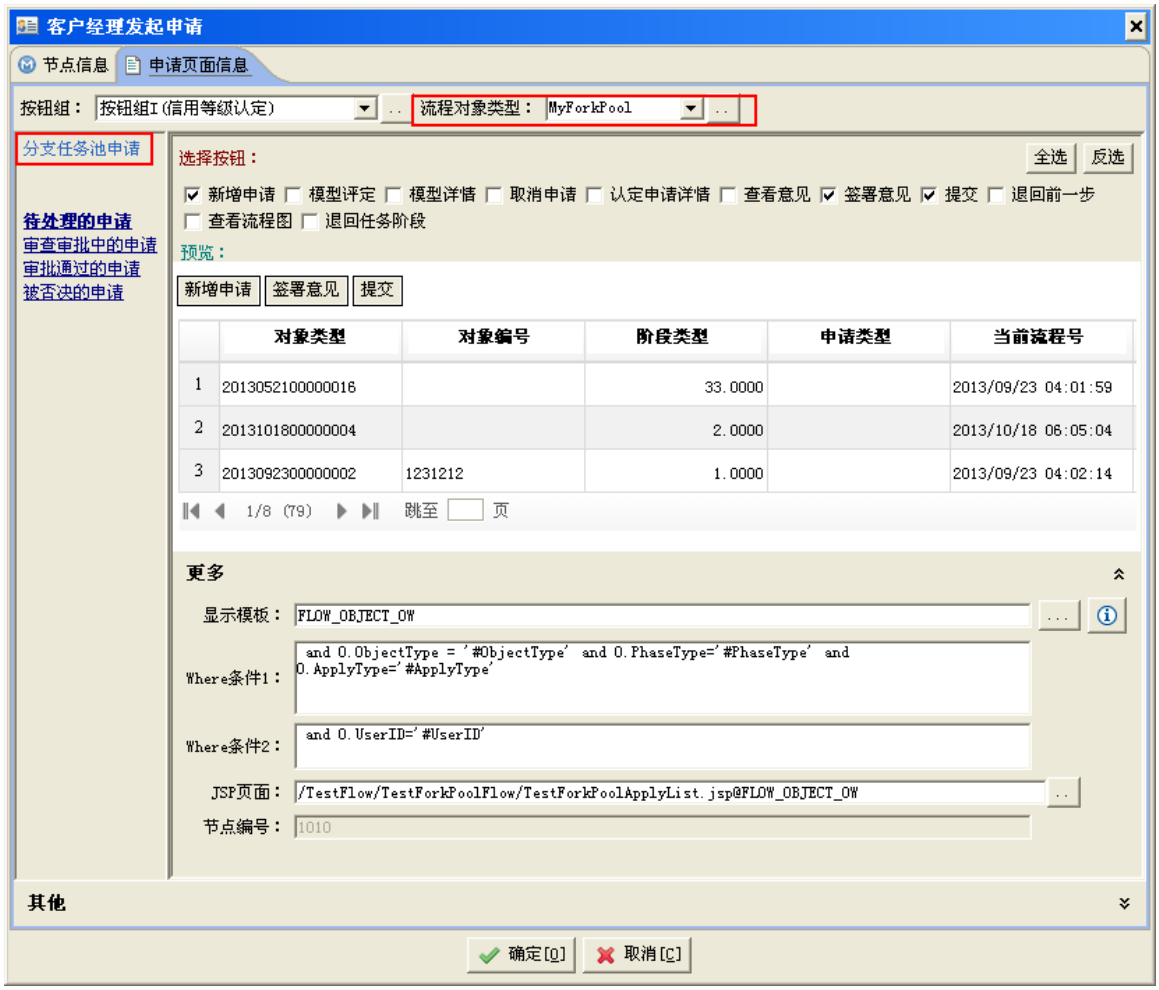
1. 双击支行客户经理申请节点，配置其 ActionScript 为：!审批流程.所有机构角色人员列表 JBOQL(441)即支行认定员，配置 PostScript 为提交到“支行认定任务池”节点。这里演示任务池需要提交人员有多个，所以配成“所有机构人员列表”。
2. 双击“支行认定（分支任务池）”节点，配置其 ActionScript 为：!审批流程.所有机构角色人员列表并行(0030,241)+!审批流程.所有机构角色人员列表并行(0040,242)，配置其 PostScript 为’0030;0040’。
3. 双击分行风险认定节点，配置其提交到总行。
4. 双击分行信用认定，配置其提交到总行。
5. 双击总行认定，配置其可以批准和否决。

6.2.3 配置申请类型及主树图

双击客户经理提交申请节点，切换到“申请页面“标签：

- 给“待处理的申请”树图勾选“新增申请”、“签署意见”和“提交”3 个按钮。
- 修改流程对象类型为“MyForkPool”。
- 修改树图标题为“分支任务池流程申请”。

最终配置结果界面如下：

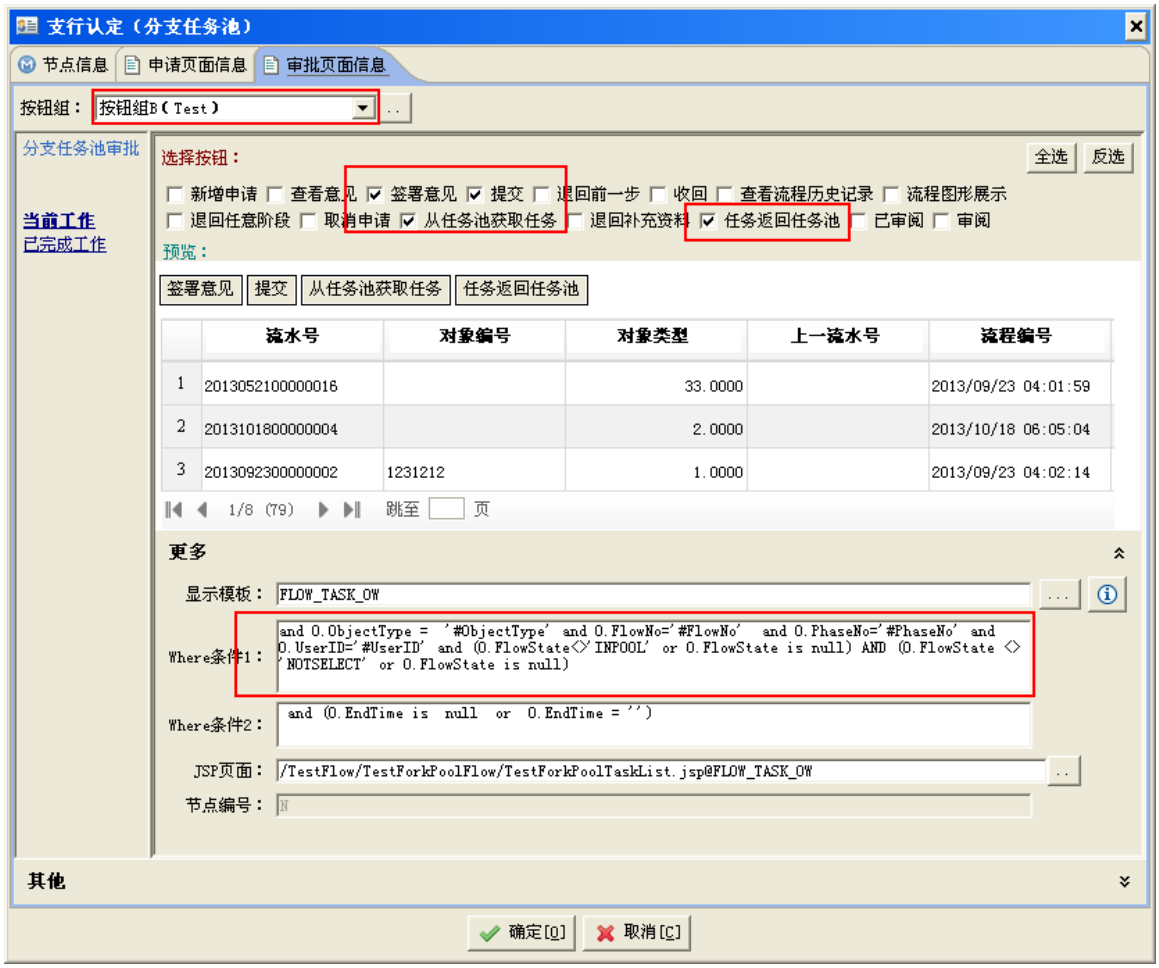


6.2.4 配置审批类型及主树图

双击支行认定任务池节点，切换到“审批页面“标签：

- 选择按钮为：按钮组 B（Test），因为这里我们要用到任务池相关按钮。
- 给“当前工作”树图勾选 “签署意见”和、提交”、“从任务池获取任务”和“任务返回任务池”4 个按钮。
- 修改树图标题为“案例任务池流程审批”。

最终配置结果界面如下：



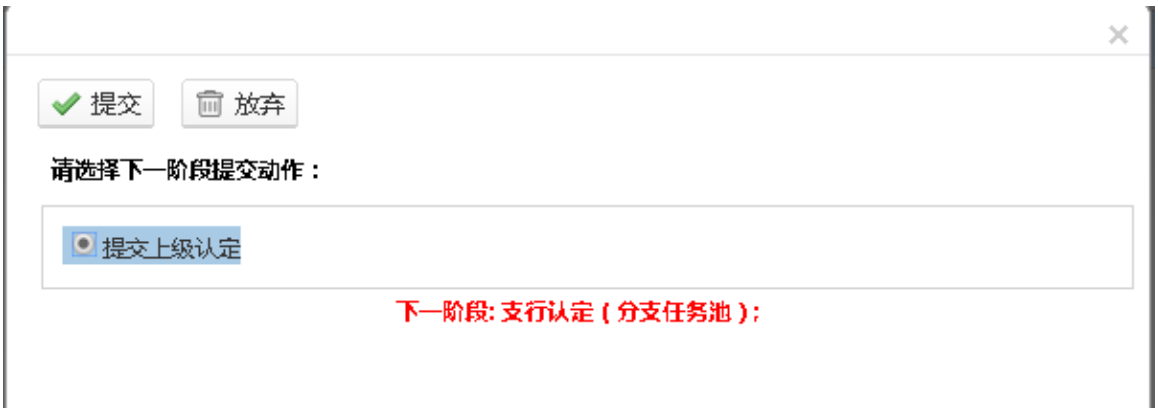
其中 where 条件为：and (O.FlowState<>'INPOOL' or O.FlowState is null) AND (O.FlowState <> 'NOTSELECT' or O.FlowState is null)　这句 Where 是 ADE 自动生成的，用来过滤没被获取的任务池任务。

双击其他节点给按钮组做类似配置，即给当前工作勾上“签署意见”和“提交”按钮。

6.2.5 运行流程

启动 Tomcat，用 test11 登录系统点击菜单“流程开发案例”→分支任务池流程→申请，即可打开申请页面，点击“新增申请”可新增一笔无业务的流程。

1. 签署意见后提交。因为下一节点时任务池节点，所以提交页面上无需选择提交人：



点击提交后系统会向 Flow_Task 插入 3 条池任务，且其 FlowState 为 INPOOL。（这里有一点和“提交给任务池”不同，这里并没有提示任务池任务接受的三个人 pt13、tt12、test50）

2. 使用 test50 登录，在审批页面点击“从任务池获取任务”，即可获取任务。此时 test50 对应的池任务的 FlowState 被设置为 SELECT，而 pt13,、tt12 对应的池任务的 FlowState 被设置成 NOTSELECT。

3. 还在原来的界面 test50 选择刚获取的任务，然后点击“任务返回任务池”。此时 test50、pt13、tt12 对应的池任务的 FlowState 都被设成 INPOOL。

4. 再次点击“从任务池获取任务”，签署意见并提交给 test22 和 test31，因为任务节点为“分支&任务池”节点，所以提交时显示出并行流程的特性：

上图中，风险认定和信用认定是同一批人，所以提交人都是一样的。提交后 pt13、tt12 对应的池任务被删除（任务池特性），并生成 test22 的任务和 test31 的任务（并行特性）。

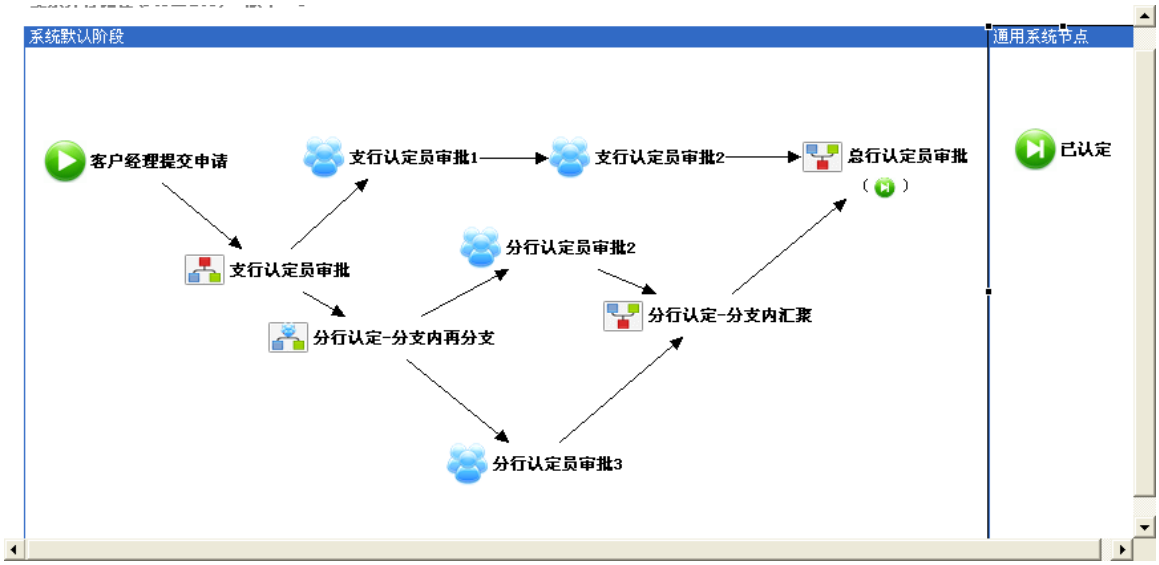
5. 使用 test22 登录，审批该流程提交给 test35，生成总行认定任务其 FlowState 为 WAIT，其 USERID 为*test35。

6. 使用 test31 登录，审批该流程提交给 test35，重新生成总行认定任务其 FlowState 为 JOIN，其 USERID 为 test35。

7. 使用 test35 登录，批准该流程。至此流程运行结束。

7. 复杂模式流程参考

以上描述的并行、待审会、任务池均是单一场景，复杂场景可以将这些场景组合到一起，可以参见 A3Web 的系统案例→审批流程实例→复杂模式流程申请。其流程图如下：


系统默认阶段
通用系统节点
客户经理提交申请
支行认定员审批1
支行认定员审批2
总行认定员审批
已认定
支行认定员审批
分行认定员审批2
分行认定-分支内再分支
分行认定-分支内汇聚
分行认定员审批3