2019年

**互联网开发培训教程**



**流程服务组件开发**

# 版权

©2019用友集团版权所有。

未经用友集团的书面许可，本文档描述任何整体或部分的内容不得被复制、复印、翻译或缩减以用于任何目的。本文档描述的内容在未经通知的情形下可能会发生改变，敬请留意。请注意：本文档描述的内容并不代表用友集团所做的承诺。

目 录

[版权 ２](#_Toc18666)

[第一章 概述 ４](#_Toc25403)

[1.1 业务场景 ４](#_Toc31690)

[1.2 应用效果 ４](#_Toc25671)

[1.3 开发方案 ４](#_Toc6726)

[1.3.1 业务场景设计 ４](#_Toc22551)

[1.3.2 库表结构设计 ４](#_Toc13479)

[1.3.3 示例功能设计 ６](#_Toc31899)

[第二章 示例配置 ７](#_Toc27844)

[2.1 应用平台配置 ７](#_Toc24937)

[2.2 开发环境配置 ７](#_Toc7307)

[2.3 流程配置 ７](#_Toc9888)

[2.4 流程对接 １１](#_Toc482)

[2.4.1 业务开发 １１](#_Toc16327)

[2.4.2 功能管理 １１](#_Toc8836)

[2.4.3 资源分配 １１](#_Toc17083)

[第三章 代码开发示例 １２](#_Toc28609)

[3.1 后端开发 １２](#_Toc19172)

[3.1.1 controller层 １２](#_Toc12620)

[3.1.2 service层 １３](#_Toc31567)

[3.1.3 实体 １５](#_Toc14211)

[第四章 API １６](#_Toc15969)

# 概述

## 业务场景

开发者经常遇到需要流程审批的需求，各种各样的单据可能都需要接入流程管理，IUAP提供的流程设计器、流程引擎、流程SDK可以让单据开发与流程平台完全解耦，开发者只需关注业务单据开发，编写少量代码即可方便快捷地实现这类需求。

* 工单开发
* 流程设计
* 工单与流程对接

## 开发方案

### 业务场景设计

本文将以“订单管理”为示例开发审批流。

工单状态（待确认、执行中、已完结）

### 库表结构设计

一主一子表结构

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| |  |  |  |  |  |  | | --- | --- | --- | --- | --- | --- | | **请购单主表： iuapd\_purchase\_order** | | | | | | | 名称 | 编码 | 类型 | 长度 | 必输项 | 备注 | | 记录Id | ID | varchar | 64 |  | PK，UUID | | 请购单编号 | ORDER\_CODE | varchar | 64 | 是 | unique index 编码规则 | | 请购单名称 | ORDER\_NAME | varchar | 64 | 是 |  | | 请购单类型 | ORDER\_TYPE | number | 1 | 是 | 1.普通采购，2委托代销，3直运采购 | | 请购单价格 | ORDER\_PRICE | number | 10 |  |  | | 请购单申请人 | ORDER\_USER | varchar | 64 |  | 参照 人员 | | 请购单申请部门 | ORDER\_DEPT | varchar | 20 |  | 参照 部门 | | 请购单申请日期 | ORDER\_DATE | varchar | 64 |  |  | | 创建时间 | CREATE\_TIME | varchar | 64 |  |  | | 创建人 | CREATE\_USER | varchar | 64 |  |  | | 修改时间 | LAST\_MODIFIED | varchar | 64 |  |  | | 修改人 | LAST\_MODIFY\_USER | varchar | 64 |  |  | | 删除标志位 | DR | decimal | 11 |  |  | | 时间戳 | TS | varchar | 64 |  |  | | 流程状态 | BPM\_STATE | decimal | 11 |  |  | | 多租户 | TENANT\_ID | varchar | 64 |  |  | |

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| |  |  |  |  |  |  | | --- | --- | --- | --- | --- | --- | | **物料表（请购单详情表）： iuapd\_purchase\_order\_detail** | | | | | | | 名称 | 编码 | 类型 | 长度 | 必输项 | 备注 | | 记录Id | ID | varchar | 64 |  | PK，UUID | | 请购单ID | ORDER\_ID | varchar | 64 | 是 | 主表id | | 物料名称 | DETAIL\_NAME | varchar | 64 | 是 |  | | 物料型号 | DETAIL\_MODEL | varchar | 64 | 是 |  | | 物料数量 | DETAIL\_COUNT | NUMBER | 64 |  |  | | 需求日期 | DETAIL\_DATE | varchar |  |  |  | | 创建时间 | CREATE\_TIME | varchar | 64 |  |  | | 创建人 | CREATE\_USER | varchar | 64 |  |  | | 修改时间 | LAST\_MODIFIED | varchar | 64 |  |  | | 修改人 | LAST\_MODIFY\_USER | varchar | 64 |  |  | | 删除标志位 | DR | decimal | 11 |  |  | | 时间戳 | TS | varchar | 64 |  |  | | 流程状态 | BPM\_STATE | decimal | 11 |  |  | | 多租户 | TENANT\_ID | varchar | 64 |  |  | |

### 示例功能设计

* 提交订单
* 撤回订单

# 示例配置

## 应用平台配置

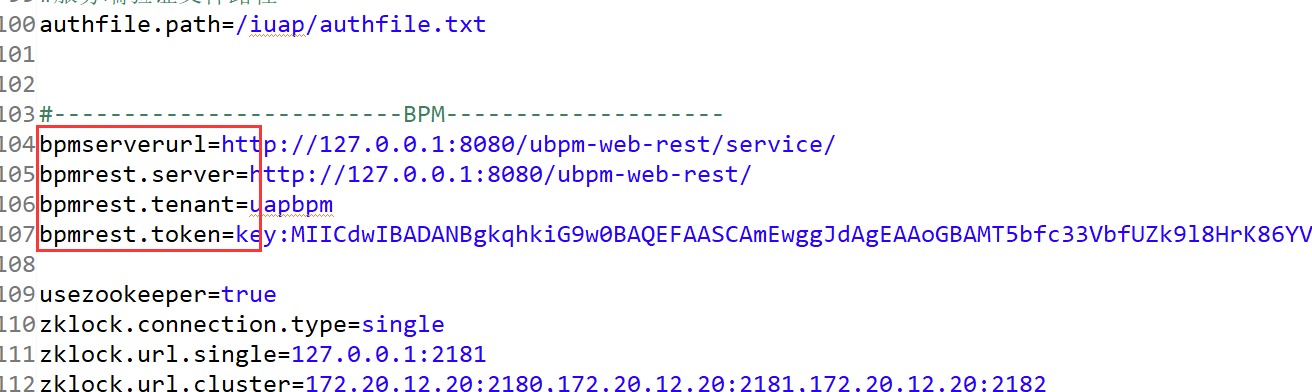
1. 在sysconfig.bat/sysconfig.sh配置数据库的时候,如果数据库类型为Oracle,用户名应该要大写,否则在新建流程的时候会创建失败.
2. 修改eiap-plus下的

/iuap-server/workdir/eiap-plus/pages/flow/bpmapproveref/bpmhandler.html文件

|  |
| --- |
|  |

## 开发环境配置

1. 配置流程平台服务地址，配置文件为：appliaction.properties



说明：

bpmserver.server为ubpm-web-rest应用工程部署的地址

bpmserverurl为流程rest服务地址

## 流程配置

1. 新建流程模型

* 使用系统管理员账号，登录应用中心，打开“流程管理——>流程模型”

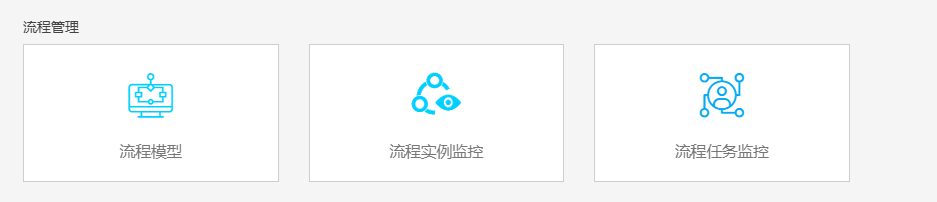


图1

* 新建流程分组



图2

* 新增流程——订单管理示例流程

在新建的分组上新增流程

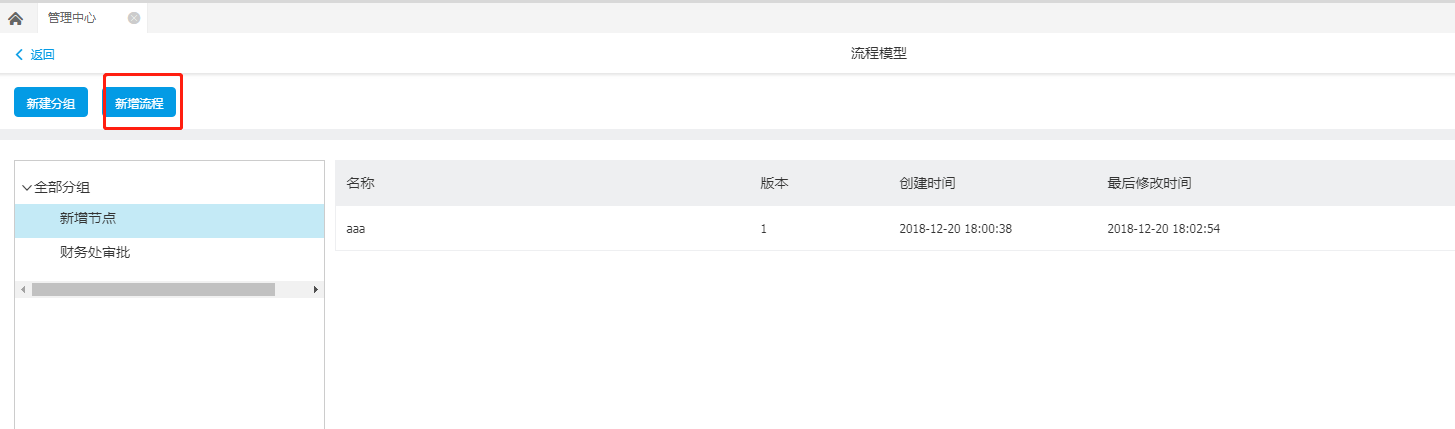


图3



图4

配置保存后：



图5

1. 流程设计
2. 打开流程设计页面



图6

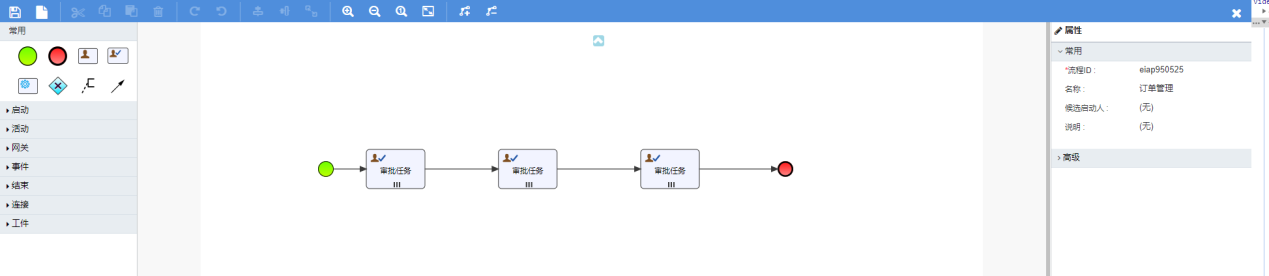


图7

1. 设计路程环节，并为流程环节配置相关信息

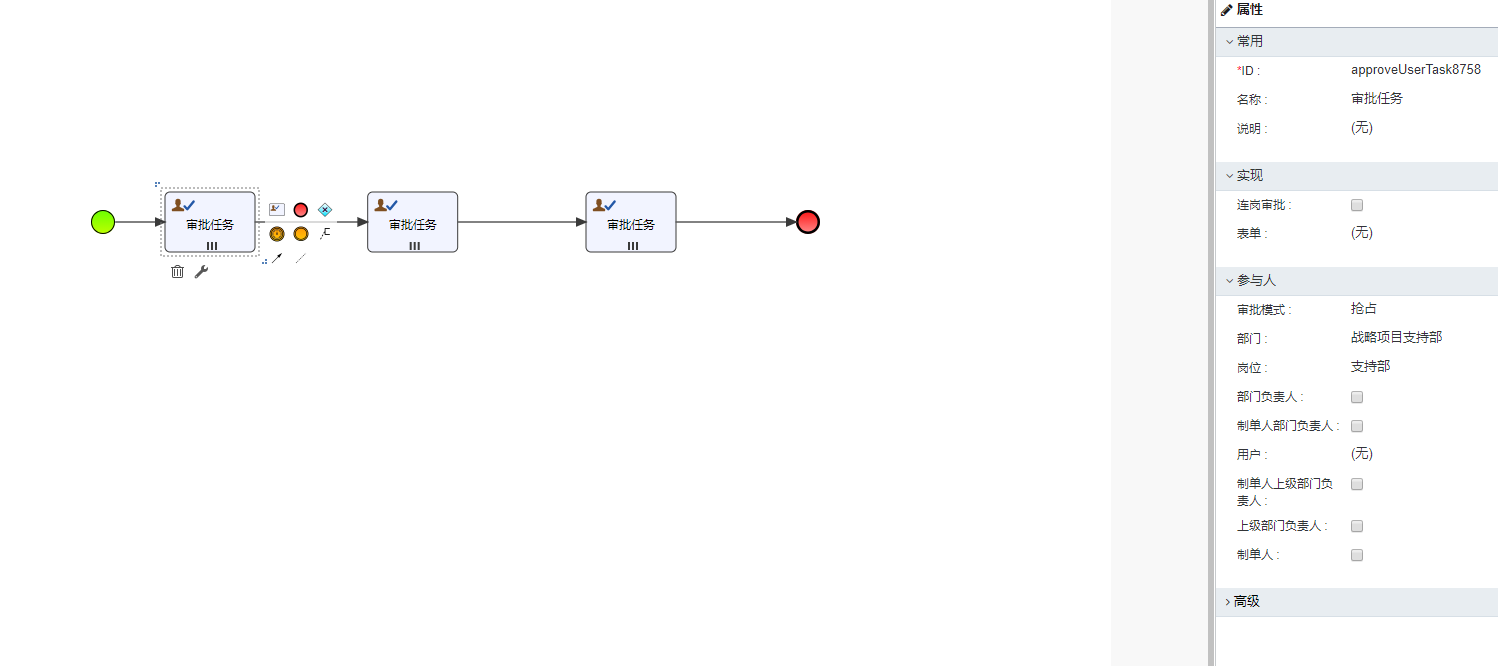


图8

这里流程配置参与人以用户为例：

流程配置需要的组织机构、部门、岗位和人员信息，并新增用户，进行人员与用户的关联。

**注：人员与用户关联通过手机号码进行绑定。**



图9



图10



图11



图12

1. 发布流程

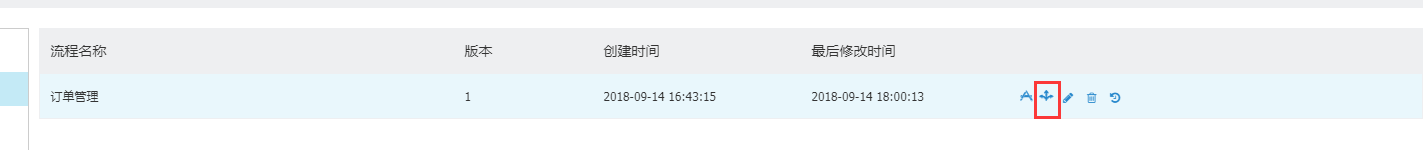


图13

## 流程对接



图14

### 业务开发

注：订单管理流程示例应用具体业务开发过程在第三章中详述

### 功能管理

订单管理开发完成后，在管理中心注册“订单管理”功能、菜单，并进行授权。

### 资源分配

资源分配主要完成业务功能和流程模型的关系映射，实现流程的动态配置，并对BPM进行启用、停用控制。

选择需要加流程的功能节点，这里选择的是“订单管理”功能节点，配置资源信息，并启用BPM。

注：nodekey在业务开发时使用。



图15

# 代码开发示例

## 后端开发

### controller层

|  |
| --- |
| package com.yonyou.iuap.purchaseorder.controller;  import com.yonyou.iuap.baseservice.bpm.controller.GenericBpmController;  import com.yonyou.iuap.purchaseorder.entity.PurchaseOrder;  import com.yonyou.iuap.purchaseorder.service.PurchaseOrderBpmService;  import java.util.Map;  import javax.servlet.http.HttpServletRequest;  import org.slf4j.Logger;  import org.slf4j.LoggerFactory;  import com.yonyou.iuap.mvc.annotation.FrontModelExchange;  import com.yonyou.iuap.mvc.type.SearchParams;  import org.springframework.beans.factory.annotation.Autowired;  import org.springframework.data.domain.PageRequest;  import org.springframework.stereotype.Controller;  import org.springframework.web.bind.annotation.RequestMapping;  /\*\*  \* 说明：请购单主表PurchaseOrder流程控制Controller——提供流程提交、收回、审批回调等rest接口  \* @date 2018-11-5 14:01:30  \*/  @Controller  @RequestMapping(value="/purchase\_order")  PublicclassPurchaseOrderBpmController extends GenericBpmController<PurchaseOrder>{  private Logger logger = LoggerFactory.getLogger(PurchaseOrderController.class);  private PurchaseOrderBpmService service;  @Autowired  public void setService(PurchaseOrderBpmService service) {  this.service = service;  super.setService(service);  }  } |

这里流程类继承了GenericBpmController，GenericBpmController这个类实现了流程前端调用的所有接口

### service层

|  |
| --- |
| package com.yonyou.iuap.purchaseorder.service;  import java.io.UnsupportedEncodingException;  import java.net.URLDecoder;  import org.springframework.beans.factory.annotation.Autowired;  import org.springframework.stereotype.Service;  import com.yonyou.iuap.baseservice.bpm.service.GenericBpmService;  import com.yonyou.iuap.bpm.pojo.BPMFormJSON;  import com.yonyou.iuap.context.InvocationInfoProxy;  import com.yonyou.iuap.purchaseorder.dao.PurchaseOrderMapper;  import com.yonyou.iuap.purchaseorder.entity.PurchaseOrder;  @Service  public class PurchaseOrderBpmService extends GenericBpmService<PurchaseOrder> {  private PurchaseOrderMapperpurchaseOrderMapper;  @Autowired  public void setPurchaseOrderMapper(PurchaseOrderMapperpurchaseOrderMapper) {  this.purchaseOrderMapper = purchaseOrderMapper;  super.setIbatisMapperEx(purchaseOrderMapper);  }  @Override  public BPMFormJSONbuildBPMFormJSON(String processDefineCode, PurchaseOrder entity) {  try {  BPMFormJSONbpmform = new BPMFormJSON();  bpmform.setProcessDefinitionKey(processDefineCode);  String userName = InvocationInfoProxy.getUsername();  try {  userName = URLDecoder.decode(userName, "utf-8");  userName = URLDecoder.decode(userName, "utf-8");  } catch (UnsupportedEncodingException e) {  userName = InvocationInfoProxy.getUsername();  }  // title  String title = userName + "提交的单据,单号是" + entity.getOrderCode() + ", 请审批";  bpmform.setTitle(title);  // 实例名  bpmform.setFunCode("masterdetail-one");  // 单据id  bpmform.setFormId(entity.getId().toString());  // 单据号  bpmform.setBillNo(entity.getOrderCode());  // 制单人  bpmform.setBillMarker(InvocationInfoProxy.getUserid());  // 其他变量  bpmform.setOtherVariables(buildEntityVars(entity));  // 单据url  bpmform.setFormUrl("/iuap\_walsin\_fe/masterdetail-one#/order?btnFlag=2&search\_id=" + entity.getId()); // 单据url  // 流程实例名称  bpmform.setProcessInstanceName(title); // 流程实例名称  // 流程审批后，执行的业务处理类(controller对应URI前缀)  bpmform.setServiceClass("/iuap\_walsin\_demo/purchase\_order");// 流程审批后，执行的业务处理类(controller对应URI前缀)  // 设置单据打开类型uui/react  bpmform.setFormType(BPMFormJSON.FORMTYPE\_REACT);  return bpmform;  } catch (Exception ex) {  throw new RuntimeException(ex.getMessage());  }  }  } |

Service继承了GenericBpmService这里重写了buildBPMFormJSON这个方法，这个方法是用来数据组装，提交之前业务需进行参数数据组装，把需要传入流程的数据组装为com.yonyou.iuap.bpm.pojo.BPMFormJSON对象数据

### 实体

实体继承了AbsBpmModel，添加流程需要的属性

# API

流程服务调用需要使用iuap流程组件中的BPMSubmitBasicService的相关API

buildBPMFormJSON方法用于组装提交流程服务的参数

