## 1.登陆的界面和简单说明

### 1.1.登陆界面：

如下图所示，为示例系统的登陆页面，对于新创建的项目，默认是没有开启登陆认证的逻辑，不会校验与要求；需要修改配置才能够开启登陆认证的逻辑；



图 1-1 登陆界面

### 2.2.简单说明：

登陆认证组件是基于开源框架shiro；实现的原理是客户端发起请求至服务器端，如果没有cookie（token信息等）以及有cookie并服务器端校验不通过的请求会被重定向至登陆页面；以完成登录认证过程；

过程：

1.参考图2-2、2-3中web.xml的设置，打开登陆认证组件拦截器

2.在没有登陆的情况下客户端发起请求，会被shrioFilter拦截

3.拦截之后，在匹配完shiro的规则配置之后，会使用statelessAuthcFilter进行拦截，拦截器会获取cookie中的信息；

4. statelessAuthcFilter会使用StatelessRealm的方式校验客户端传递过来的token在服务器端的redis中是否存在会话；

5.redis存在会话则验证成功，并且恢复上下文信息；不存在会话，则拦截请求并跳转至登陆的url

6.登陆页面登陆时，会调用LoginController，在进行登陆时，会通过图2-5校验登陆的用户名、密码；并通过webTokenProcessor生成cookie、token以及保存会话至redis；

7.一旦登陆成功之后，浏览器端cookie中会保有token信息，满足后续请求的认证

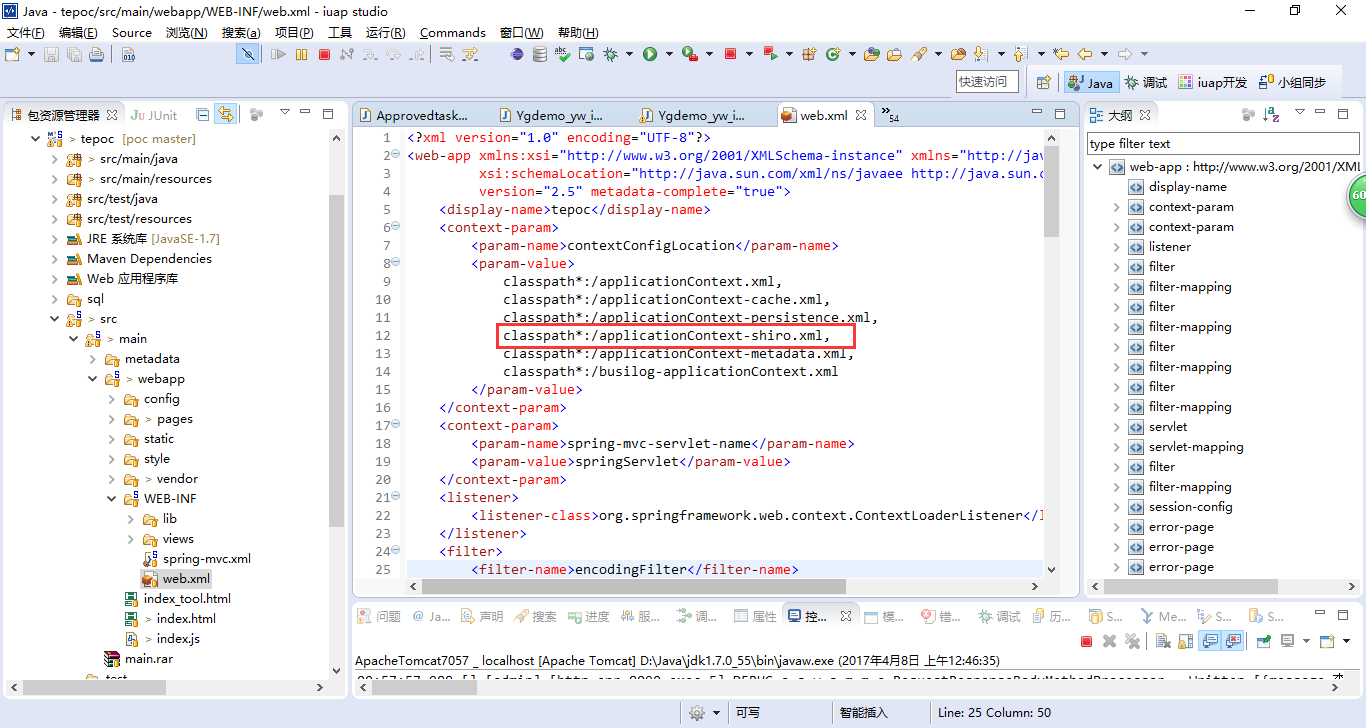


图1-2 web.xml配置applicationContext-shiro.xml



图1-3 shiroFilter开启设置

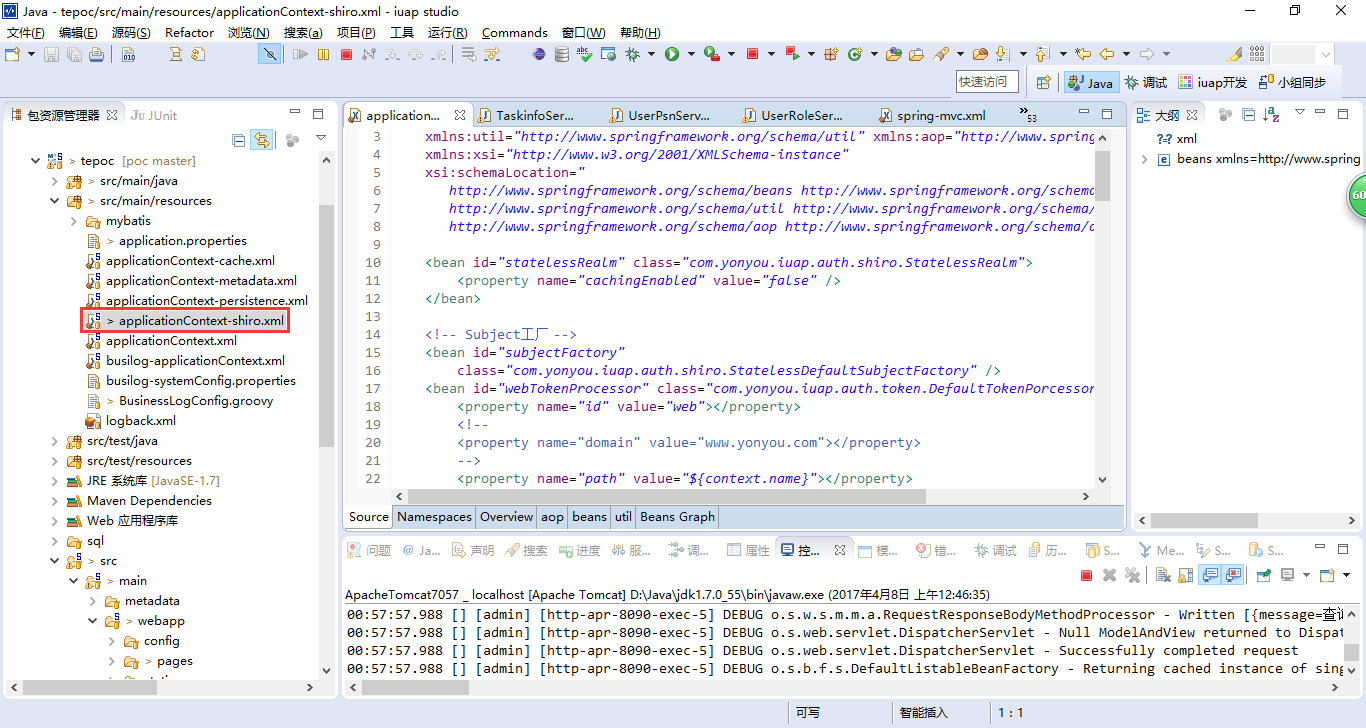


图1-4 applicationContext-shiro.xml配置



图 1-5 LoginController逻辑生成cookie、token以及redis中注册会话

## 2.单点登录

场景：tepoc系统、tepoc\_ma系统；tepoc\_ma系统通过tepoc系统的登陆机制，完成登陆并返回token信息，后续tepoc\_ma系统发起对tepoc系统的请求只需要使用token即可，不需要重复去登陆

过程：

1. tepoc\_ma 系统调用SSOLoginController发起请求至tepoc系统中进行身份认证MaLoginController.formLogin2()进行验证用户，用户验证成功之后，tepoc\_ma会接受到tepoc返回的token字符串以及cookie并存储在客户端浏览器

2.单点登录使用了登陆认证组件的机制，每次请求拦截都会获取cookie，读取cookie中的token并进行验证；

3.所以tepoc\_ma 中通过InvocationFilter读取请求中cookie的token信息，并放置到对tepoc请求头中；

如果步骤1中用户验证不成功，对于后续向tepoc请求服务的时候，由于没有认证信息，则请求会被拦截；

在tepoc\_ma系统中进行登陆，使用与tepoc系统一致的用户，实际调用tepoc的登陆逻辑，产生token并给tepoc\_ma系统



图2-1 tepoc\_ma登陆页面

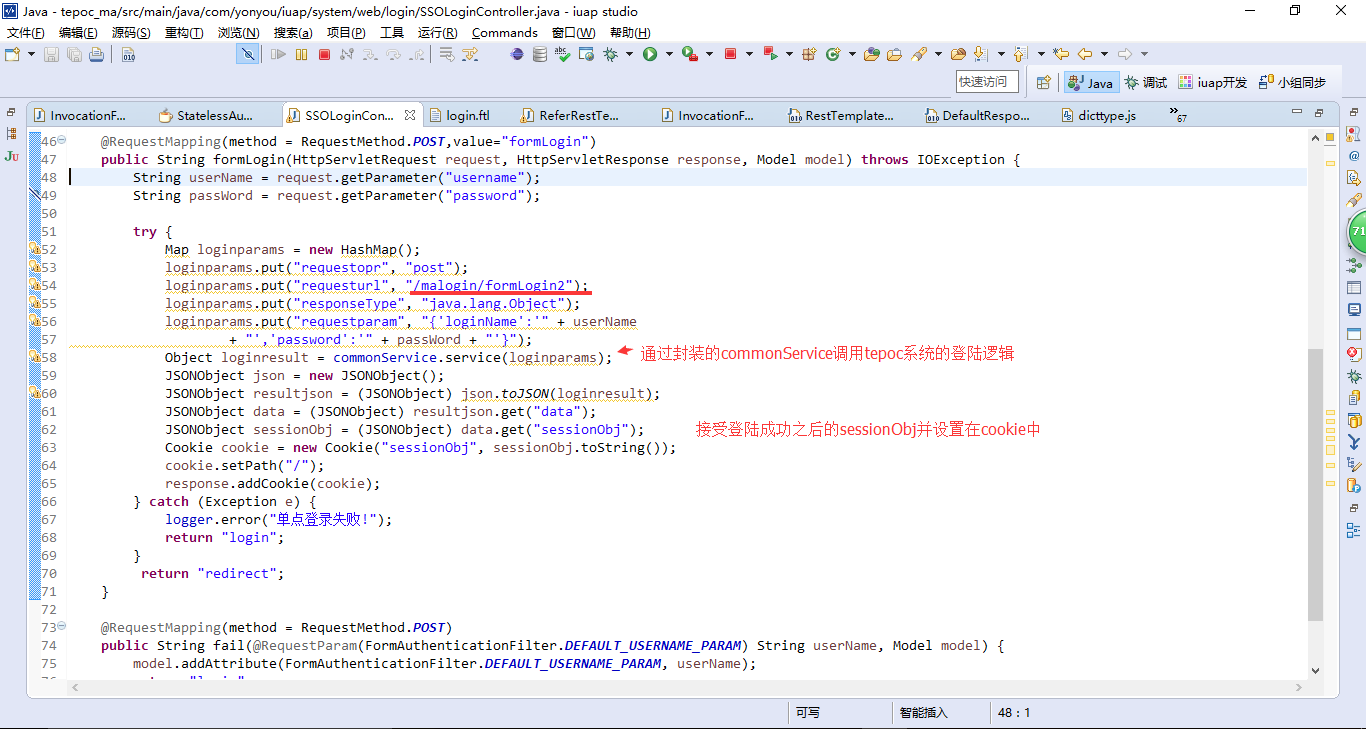


图2-2 tepoc\_ma单点登录逻辑

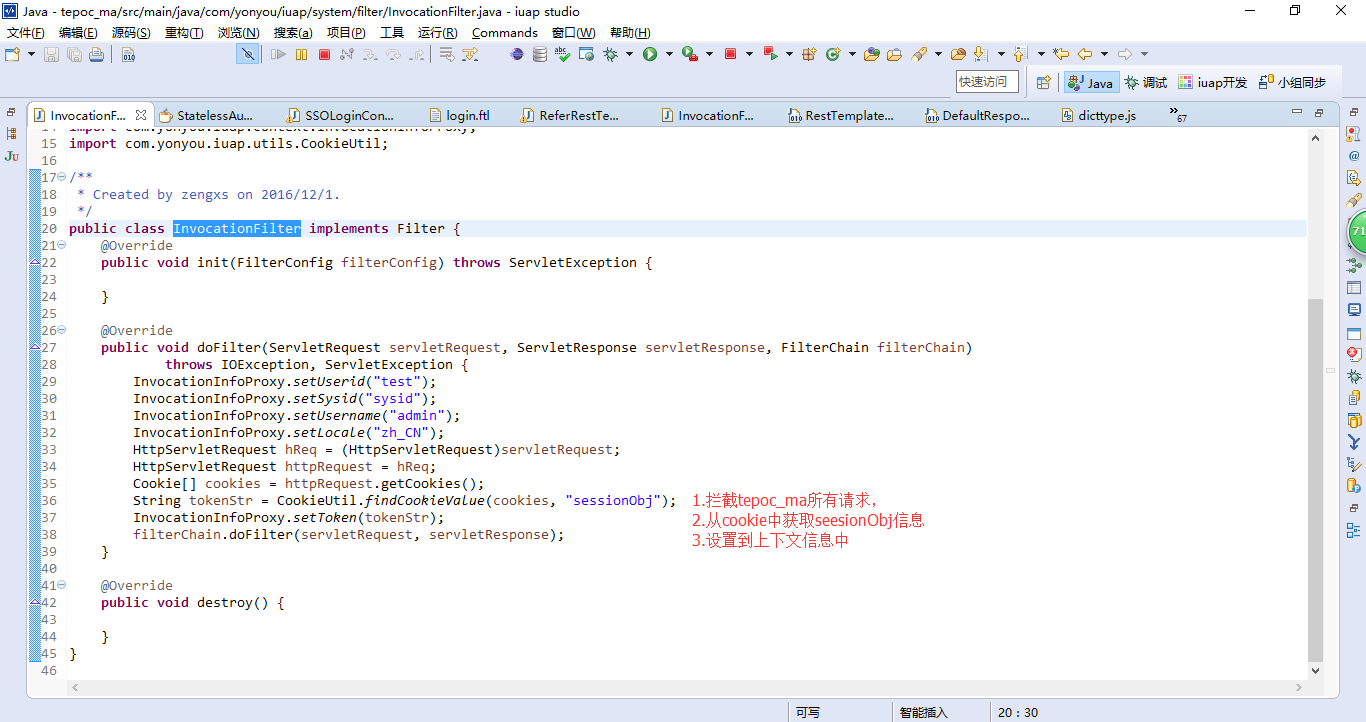


图2-3 tepoc\_ma InvocationFilter拦截器处理sessionObj

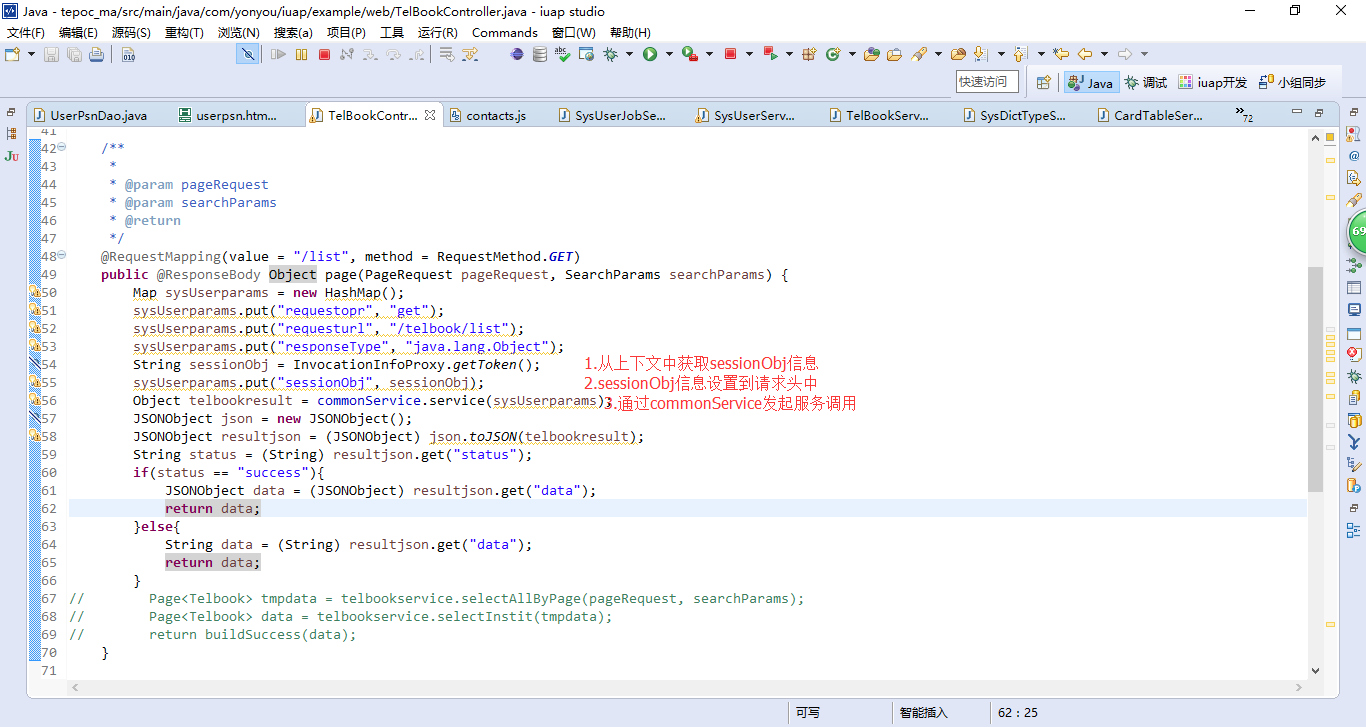


图2-4 tepoc\_ma系统业务方法请求tepoc系统中服务

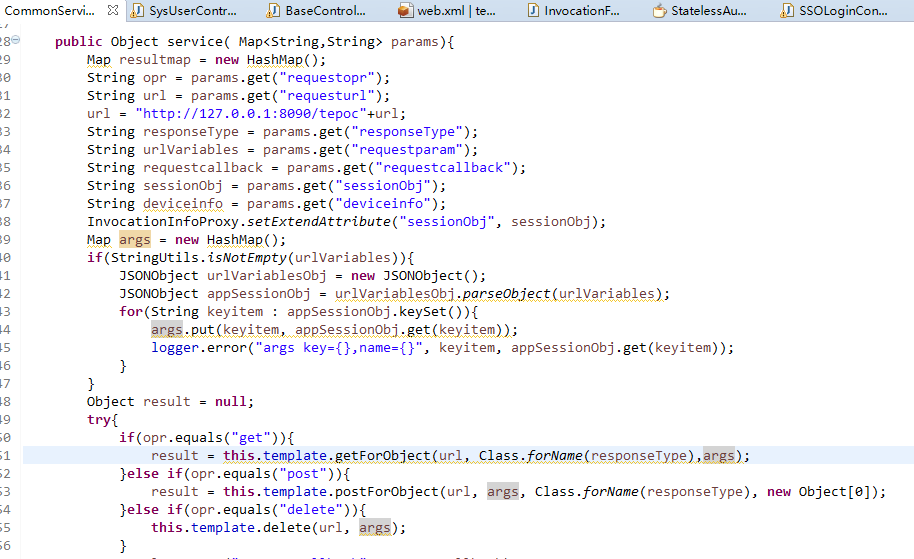


图 2-5 CommonService工具

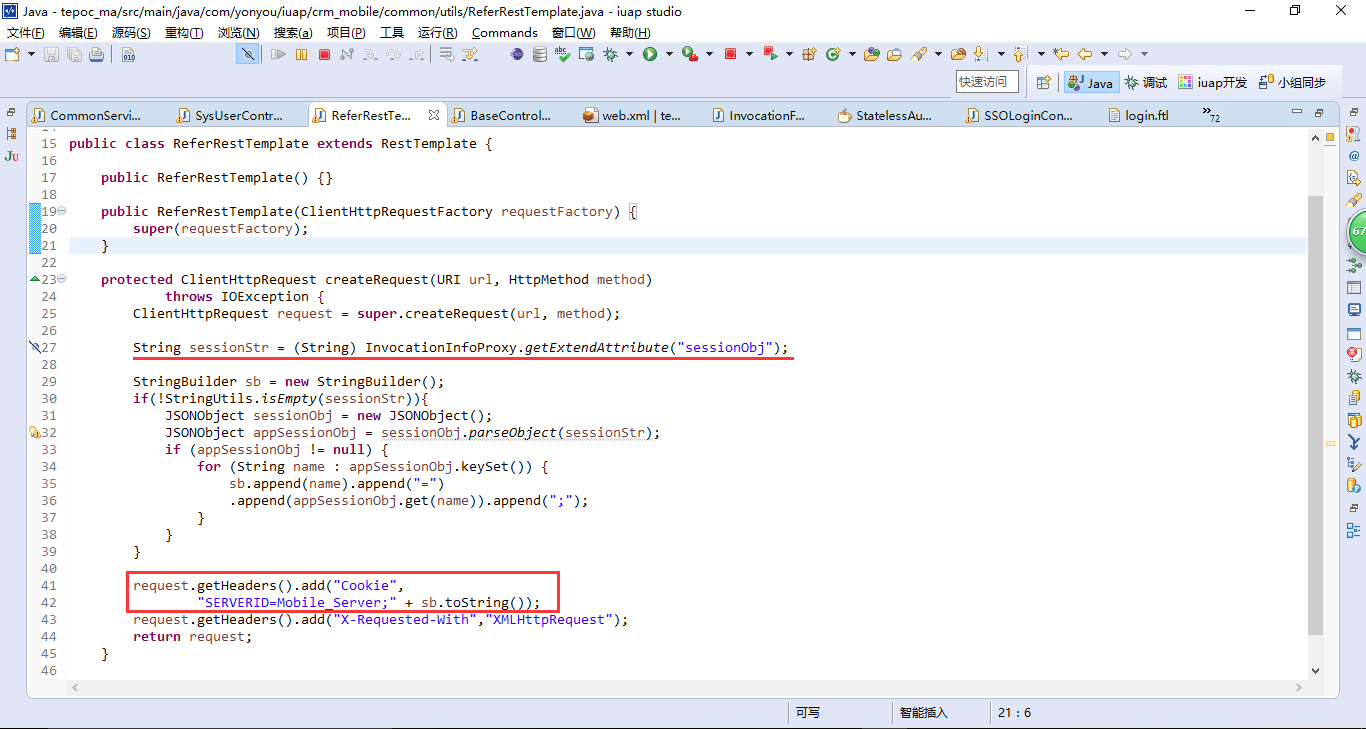


图2-6 ReferRestTemplate处理sessionObj并设置到请求头，给tepoc项目

经过单点登录之后，服务可用，演示效果如下图：



图2-7 经过单点登录之后，服务可用

未经过单点登录，服务不可用，演示效果如下图：

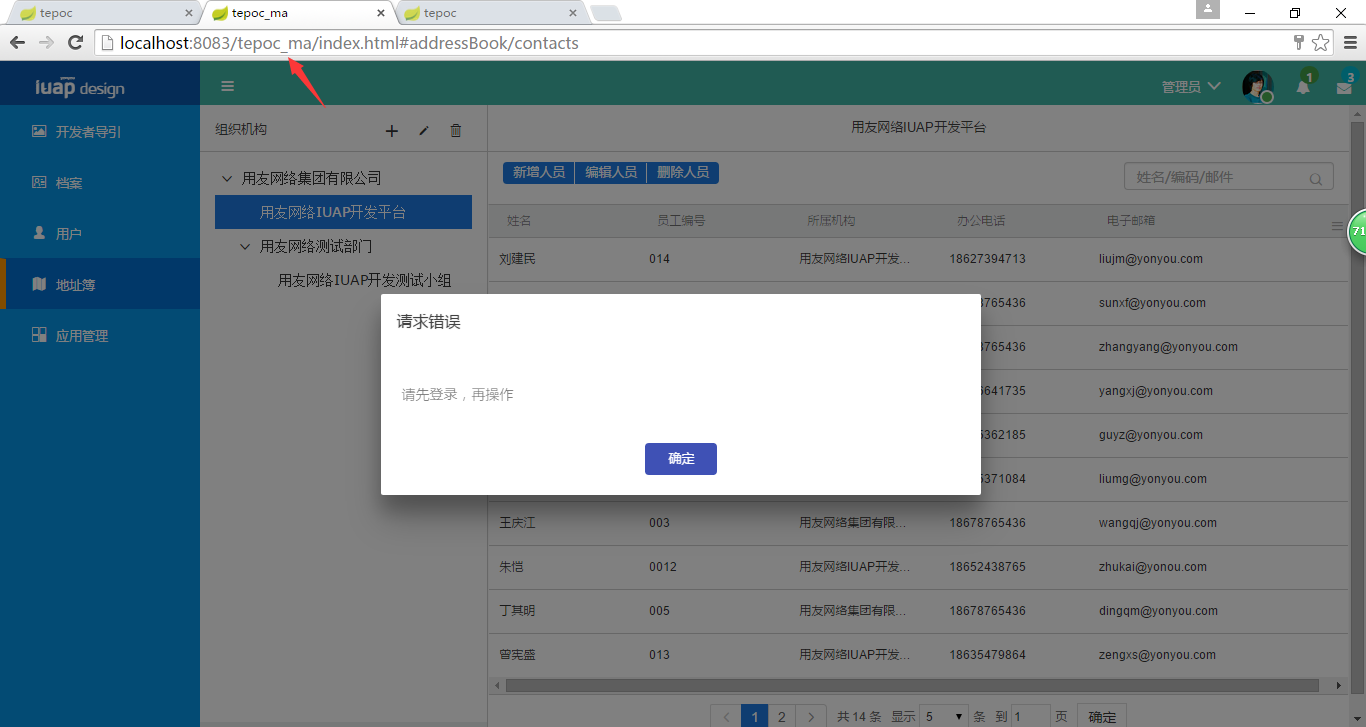


图2-8 未经过单点登录，服务不可用

## 3.用户管理示例节点

按照poc需求提供示例规范，设计用户管理页面，支持界面不同的状态:编辑态、浏览态；如图3-1为浏览态；图3-2为编辑态；

如：新建、编辑按钮动作触发之后，跳转到卡片页面的编辑态

查阅跳转到卡片页面的浏览态

在图 3-1中提供grid的头列锁定功能

调整了图3-2卡片页面的css样式，满足了label与输入控件同行的情况；tab页面支持了背景色的设置，并且子表设置了固定的留白区域，在多个也签的情况下，能够保证一致，不会在不同的也签切换时候，页面会抖动；

以及通过参照的方式，提供了公司、部门以及子页签上所属公司、所属部门的数据设置方式；

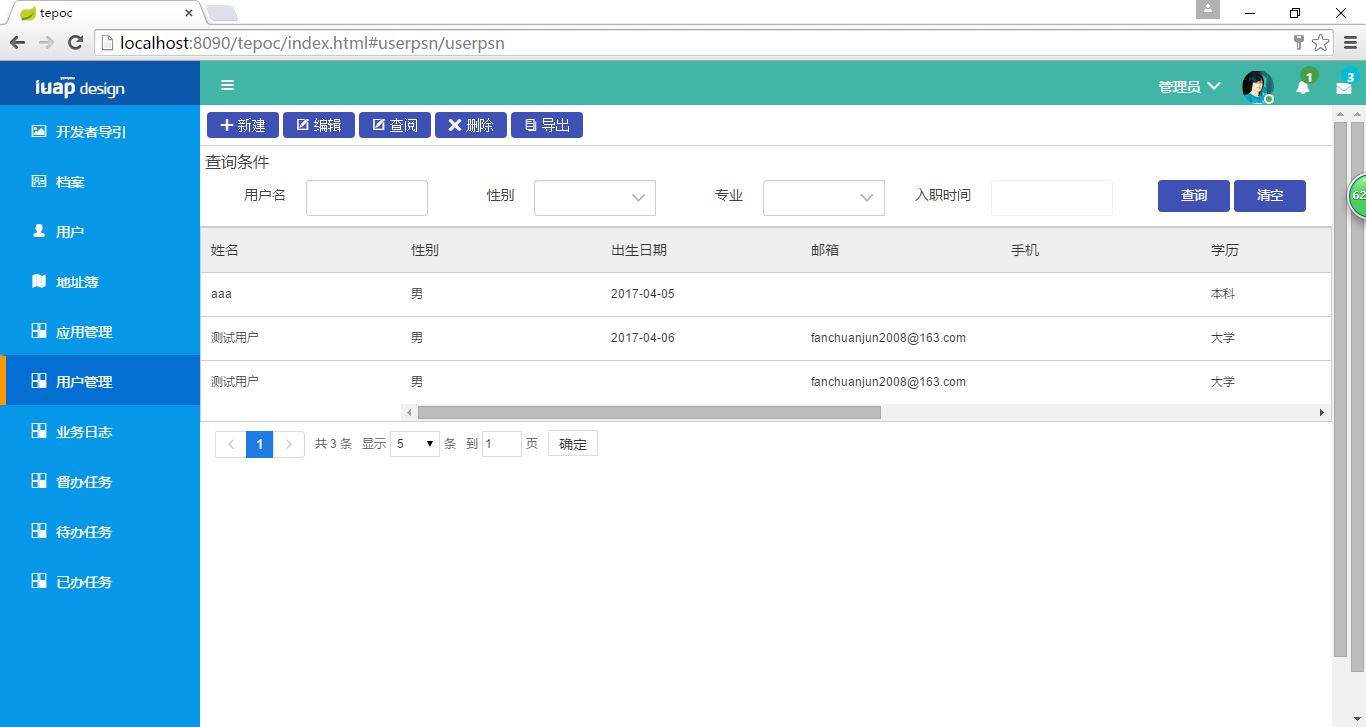


图 3-1用户管理节点列表页面

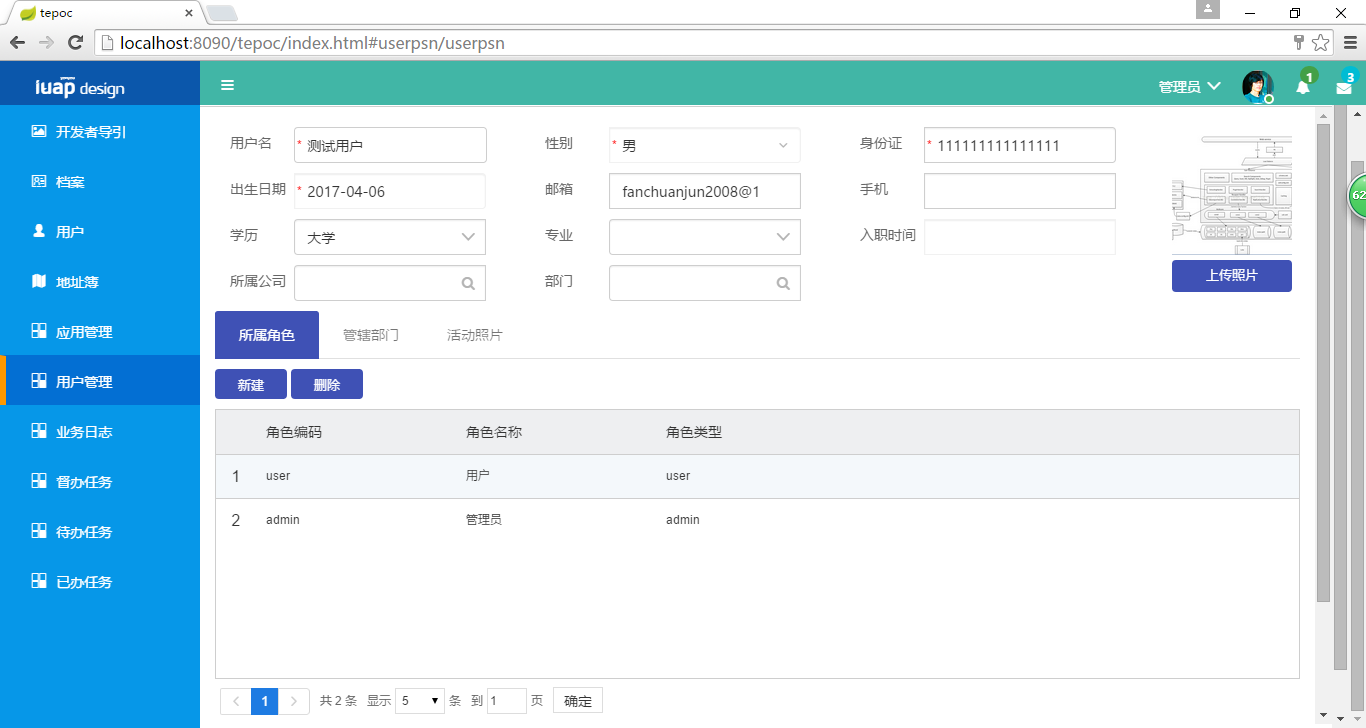
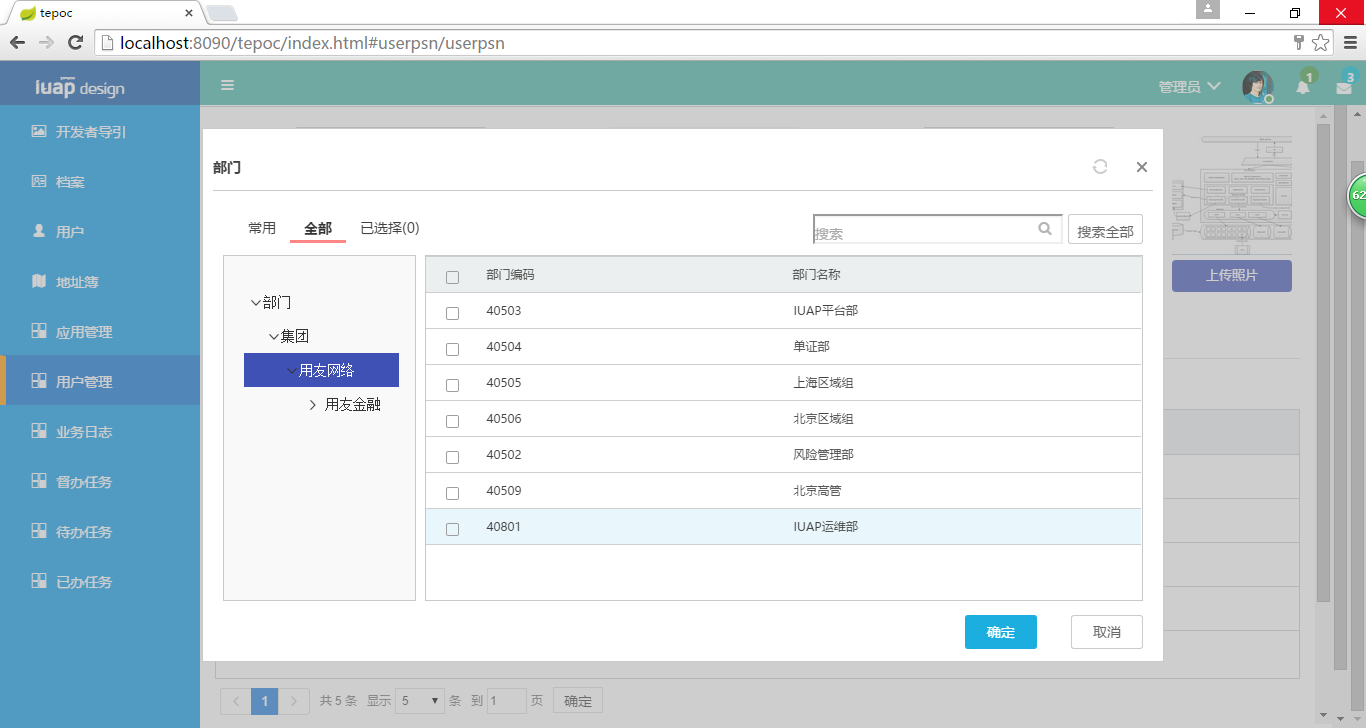


图 3-2 用户管理节点卡片页面



## 4.集成演示部分

### 4.1 IUAP与MDM集成

IUAP与MDM主数据集成:读取主数据提供的webservice接口，并将数据展现在用户管理节点的前端页面上；

主数据的接口提供方式为webservice；如下图在IUAP中的该接口的client端：

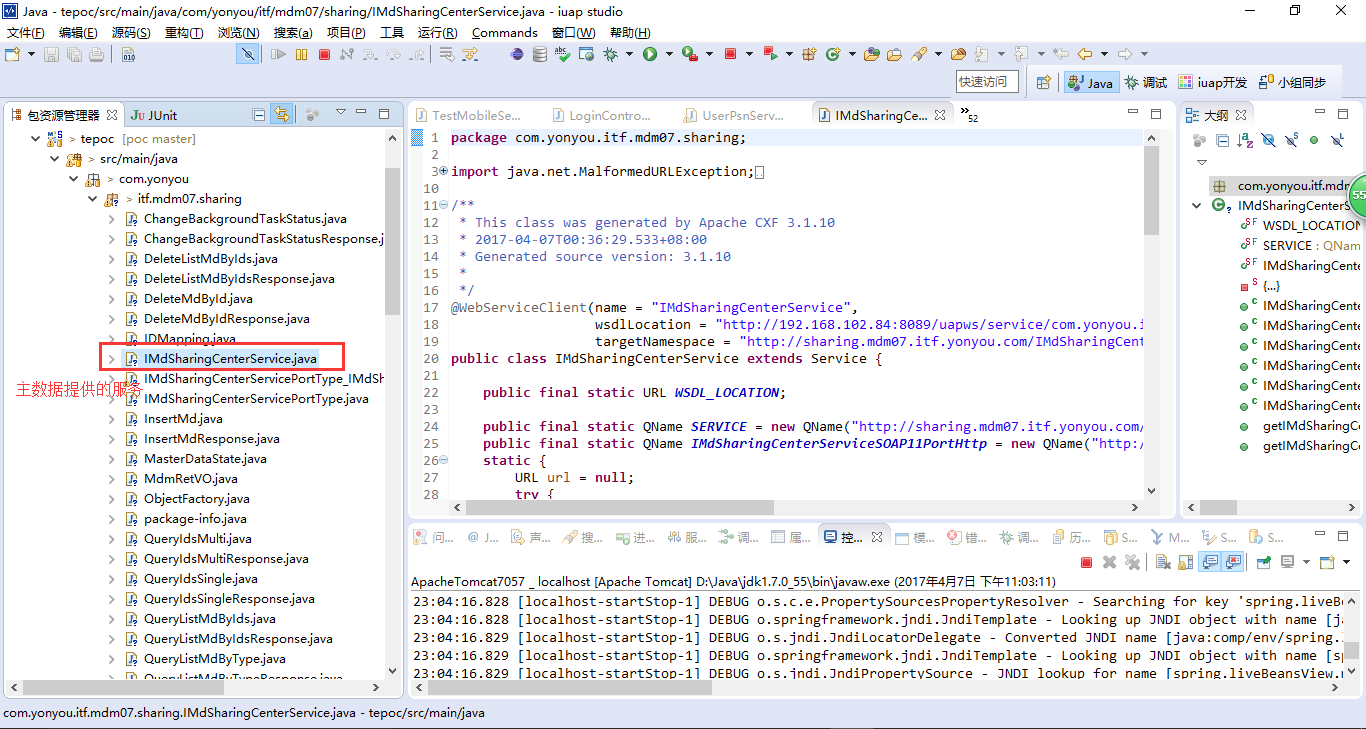


图4-1 主数据服务接口

通过图4-3、4-4中的服务提供主数据服务数据展现在图4-2中用户管理界面上；



图4-2主数据获取数据在用户管理界面展现

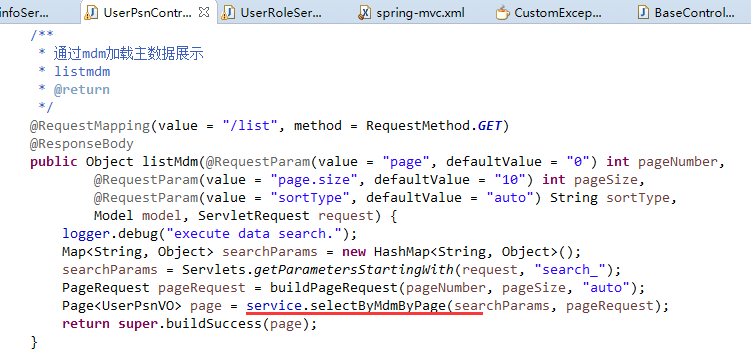


图4-3提供给用户管理界面的后台服务

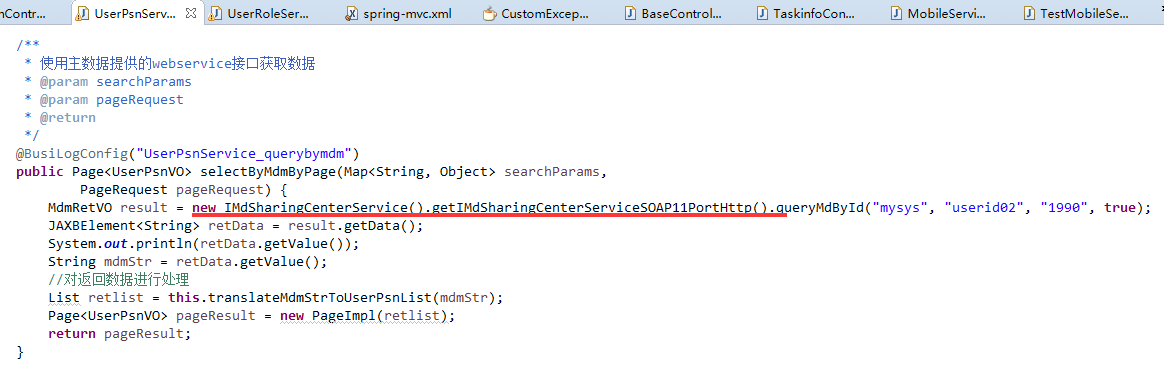


图4-4集成主数据服务

### 4.2 IUAP与Mobile与MDM集成

如图4-5中，在IUAP中提供了restful服务给移动端，在MA中通过网关配置的方式调用IUAP中提供的restful服务，实现移动通过IUAP与MDM集成；

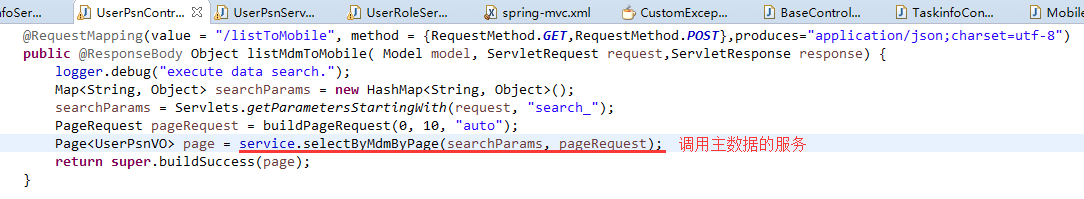


图4-5 IUAP中提供给Mobile的用户管理节点查询服务（对接MDM）

### 4.3 IUAP与BPM及Mobile集成

如下图所示，提供了一个具有审批流程动作的业务单据；先进行制单，再通过“提交审批”按钮操作，驱动单据审批流程的启动



图4-6 IUAP中含有审批流程的业务单据

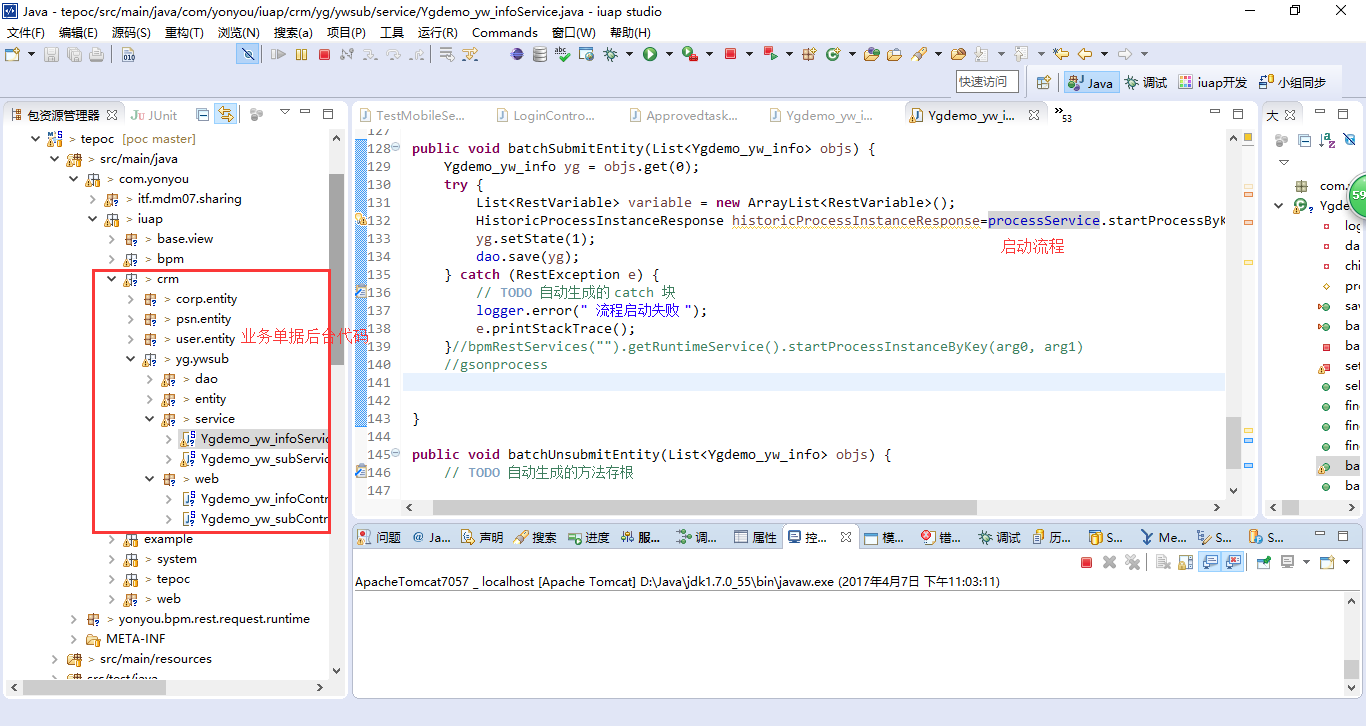


图4-7 业务单据后台代码及启动流程实例

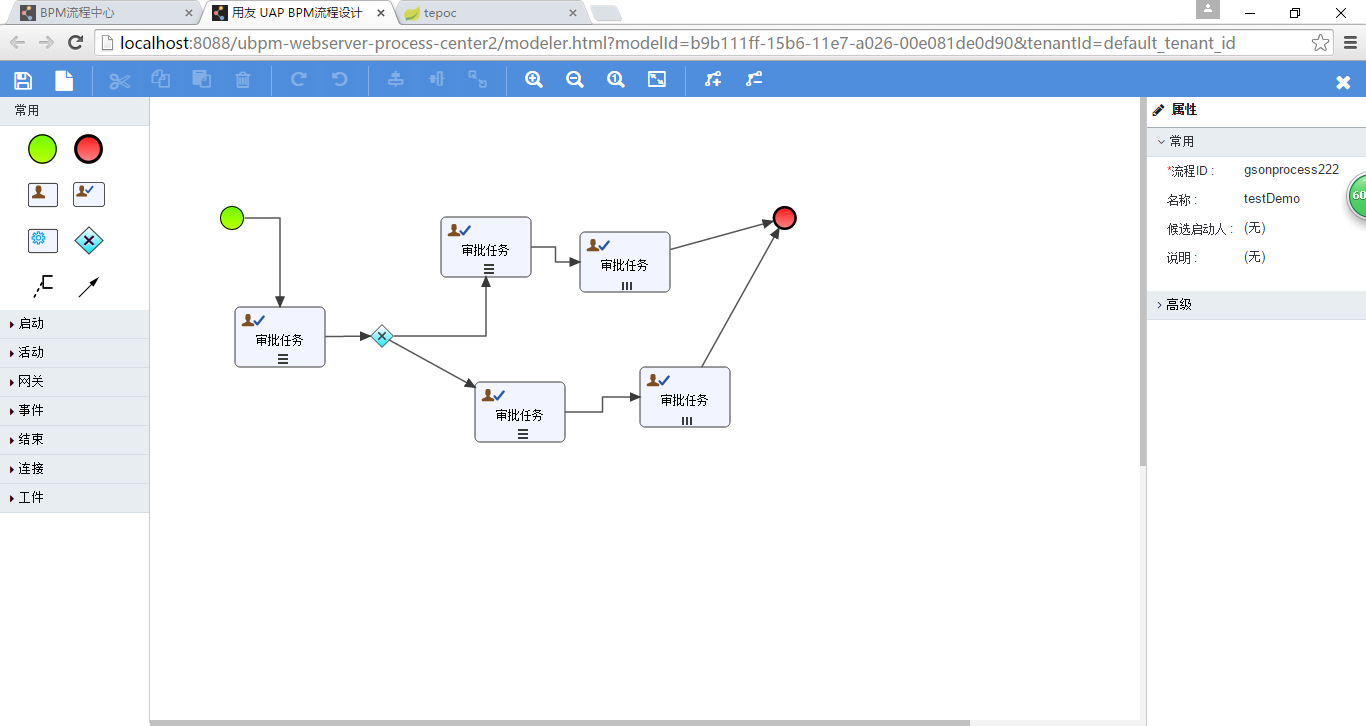


图4-8 bpm审批流程定义

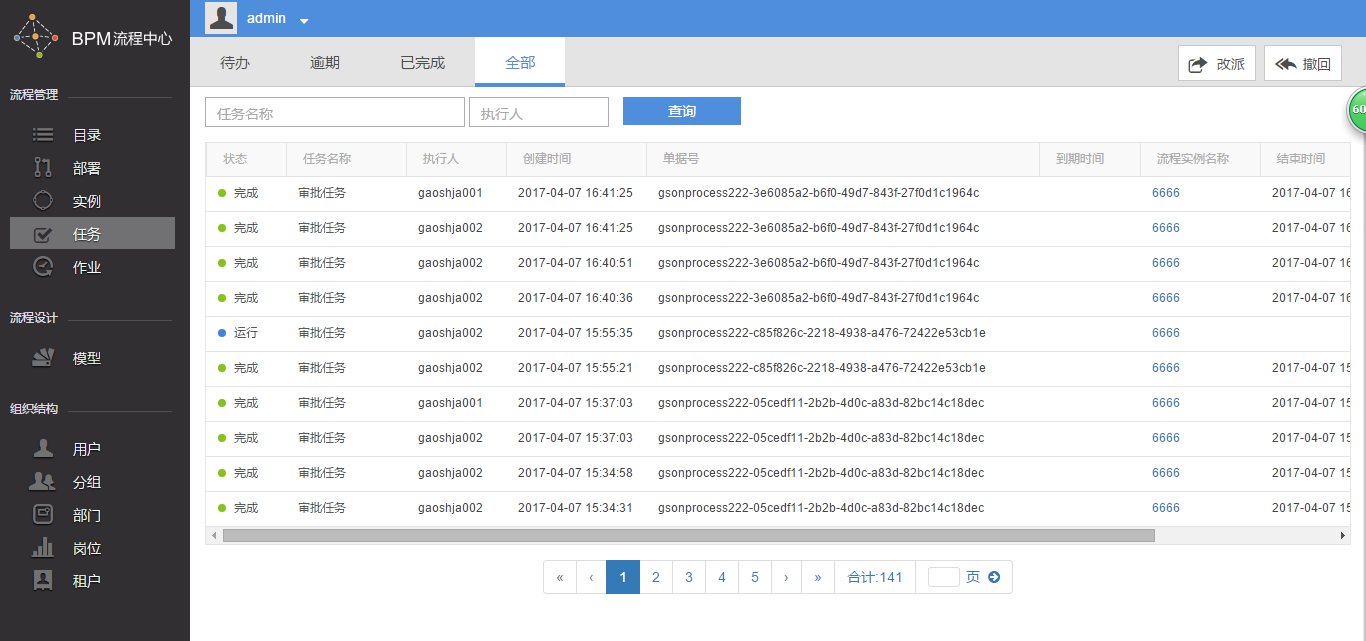


图 4-9 测试产生的所有审批流程任务

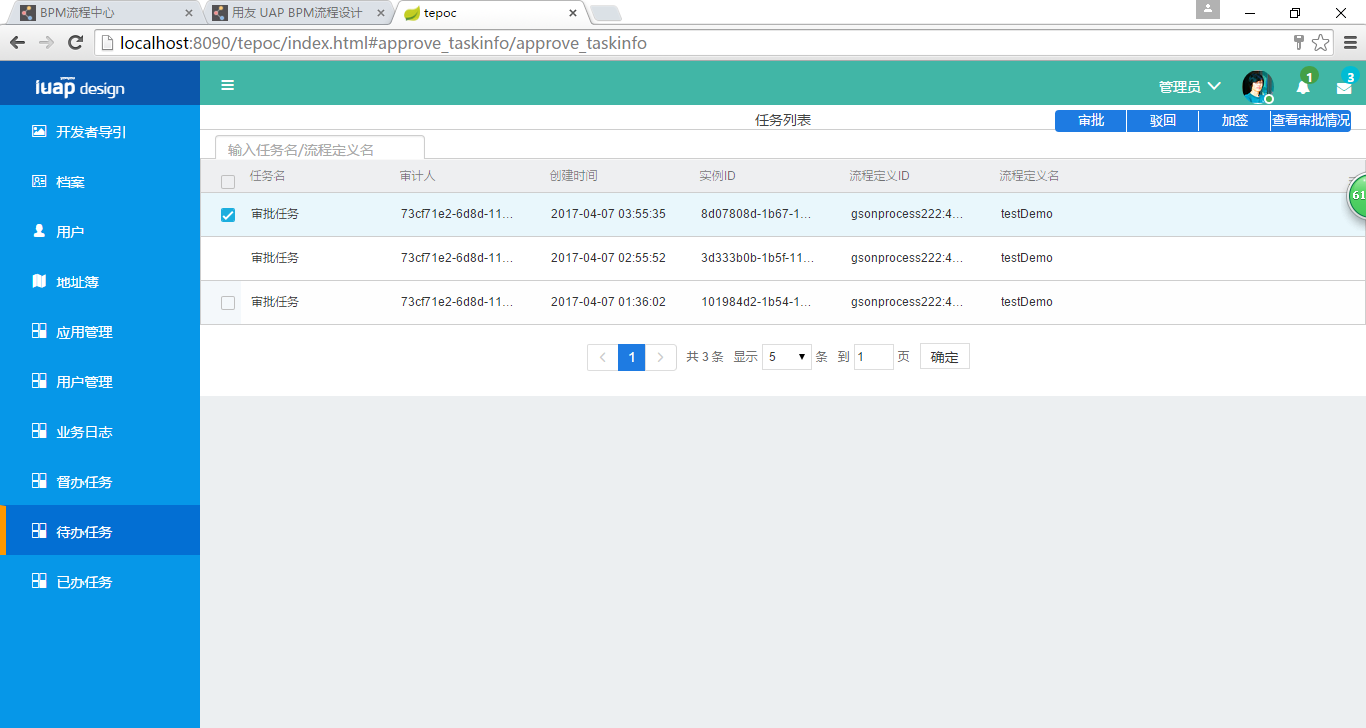


图4-10 待办任务列表界面

将bpm的获取代办任务的接口通过iuap提供给移动MA，进行集成

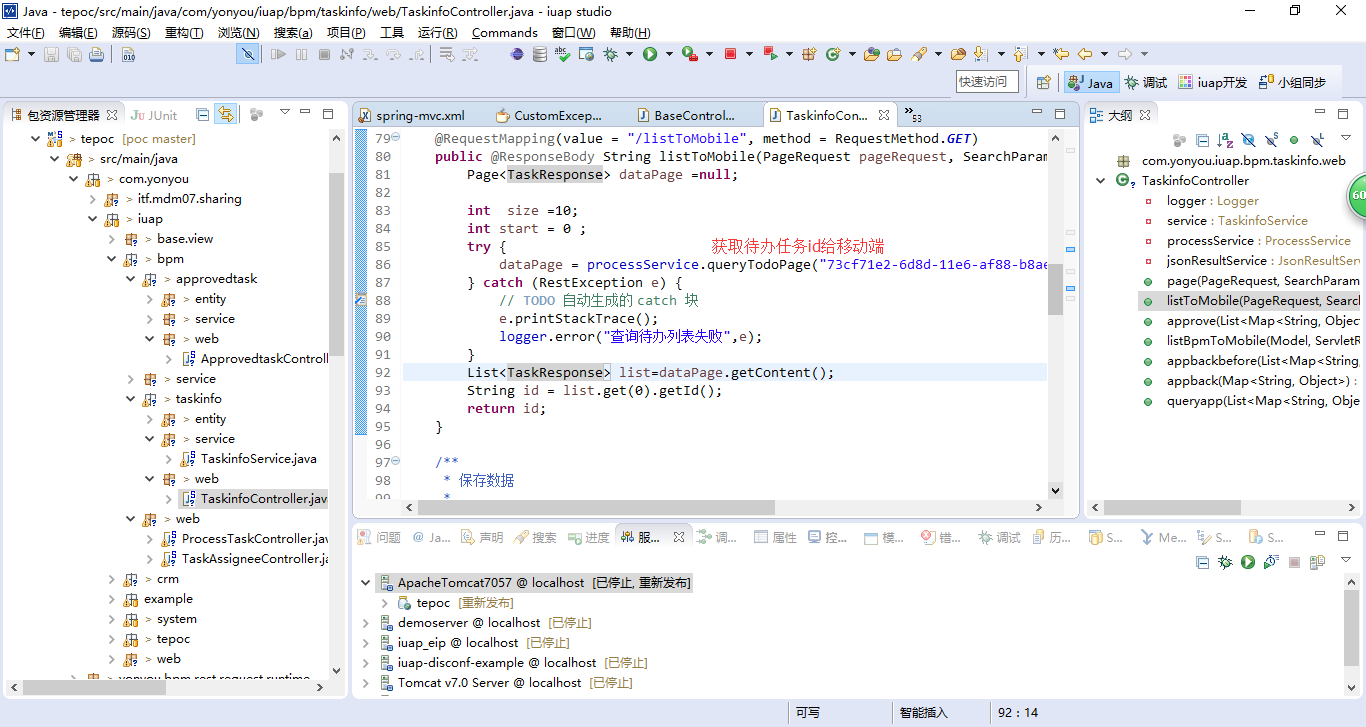


图 4-11 iuap将bpm服务提供给mobile服务

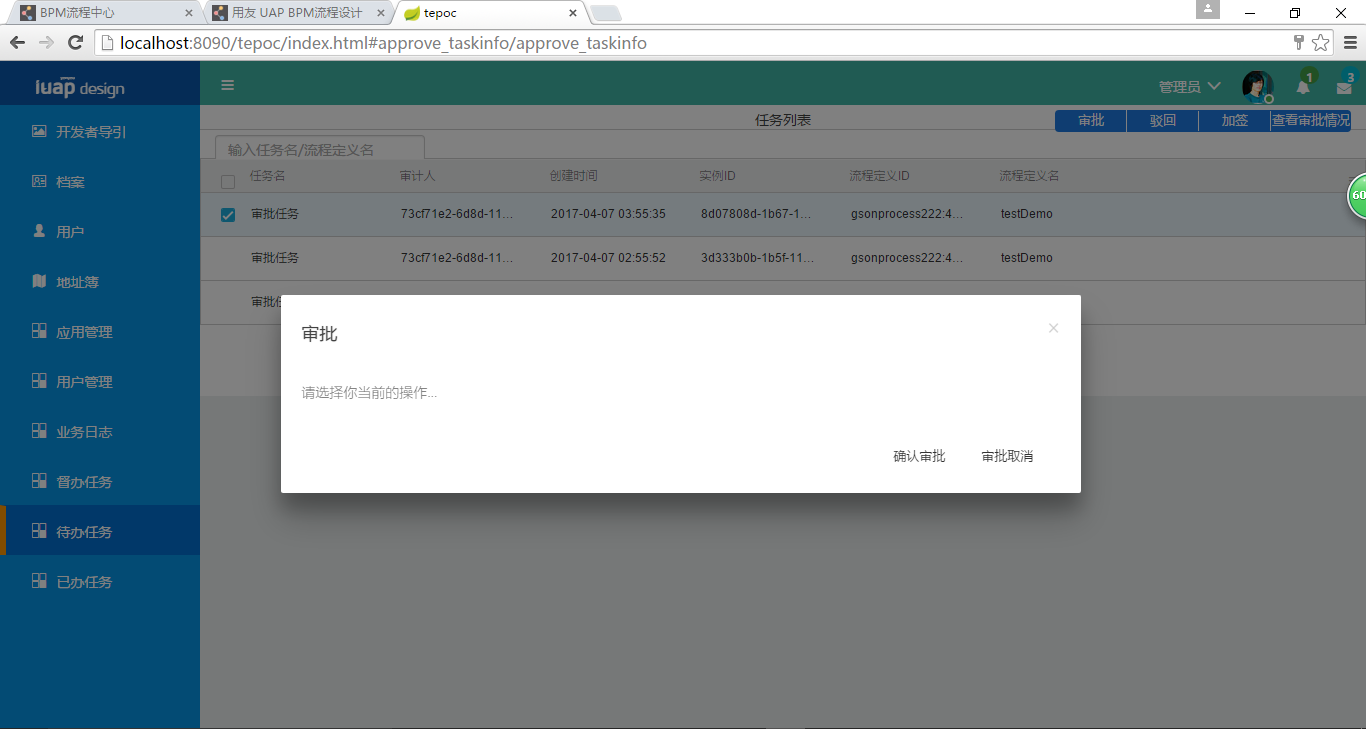


图 4-12 审批界面

将bpm审批动作同样通过iuap提供的restful与mobile集成

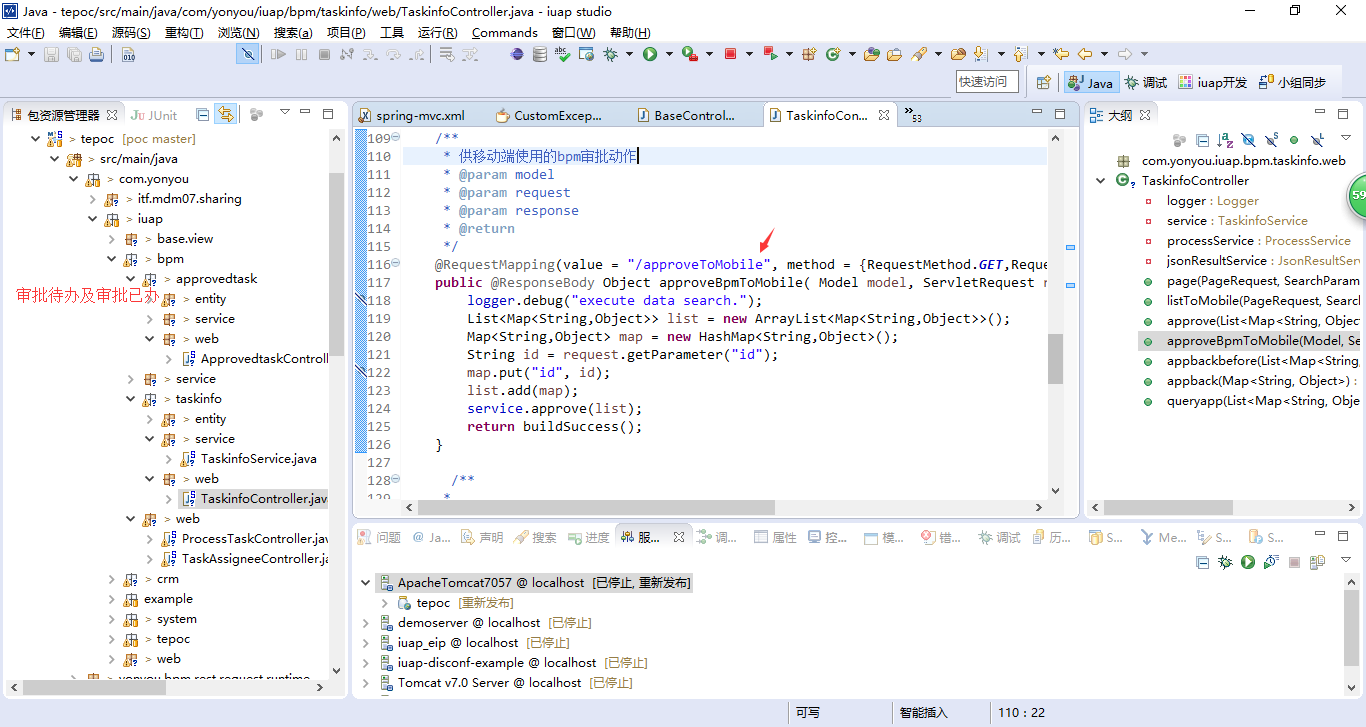


图 4-13提供bpm审批动作通过iuap提供的restful与mobile集成

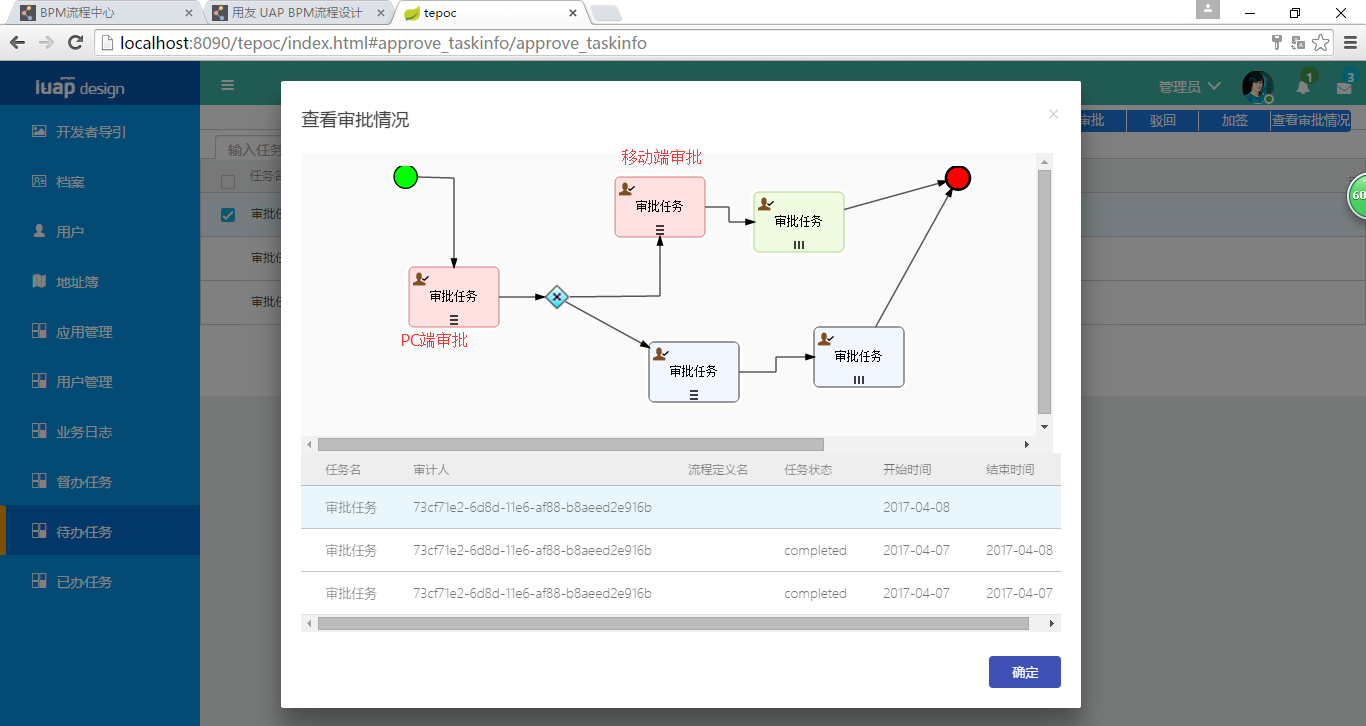


图 4-14 待办-查看审批情况

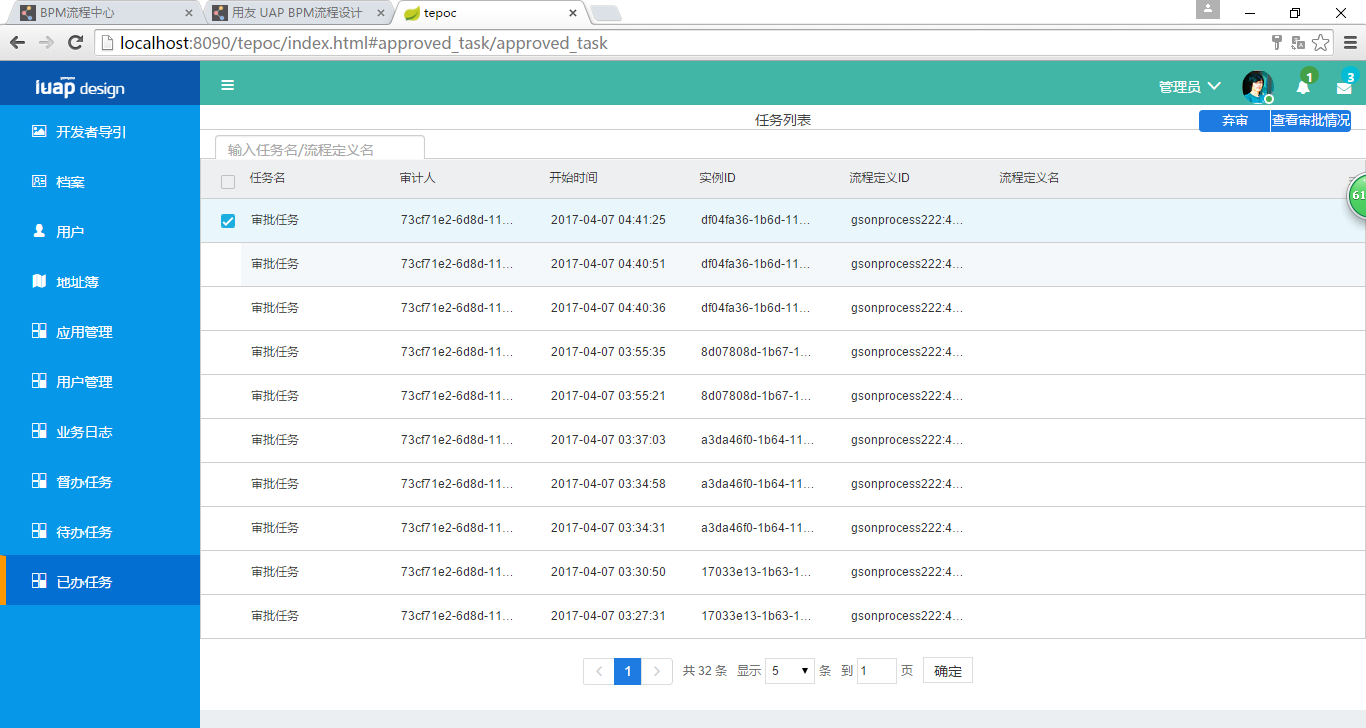


图 4-13 已办任务界面

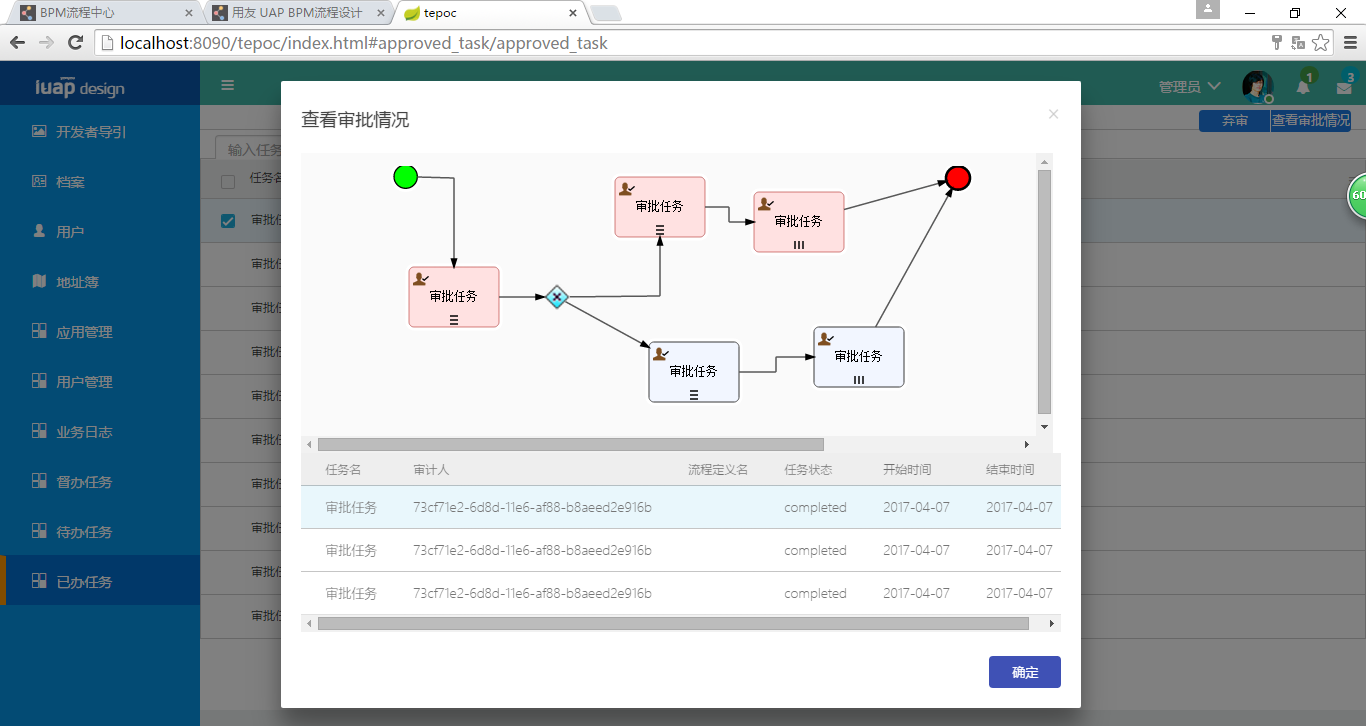


图 4-14 已办-查看审批情况

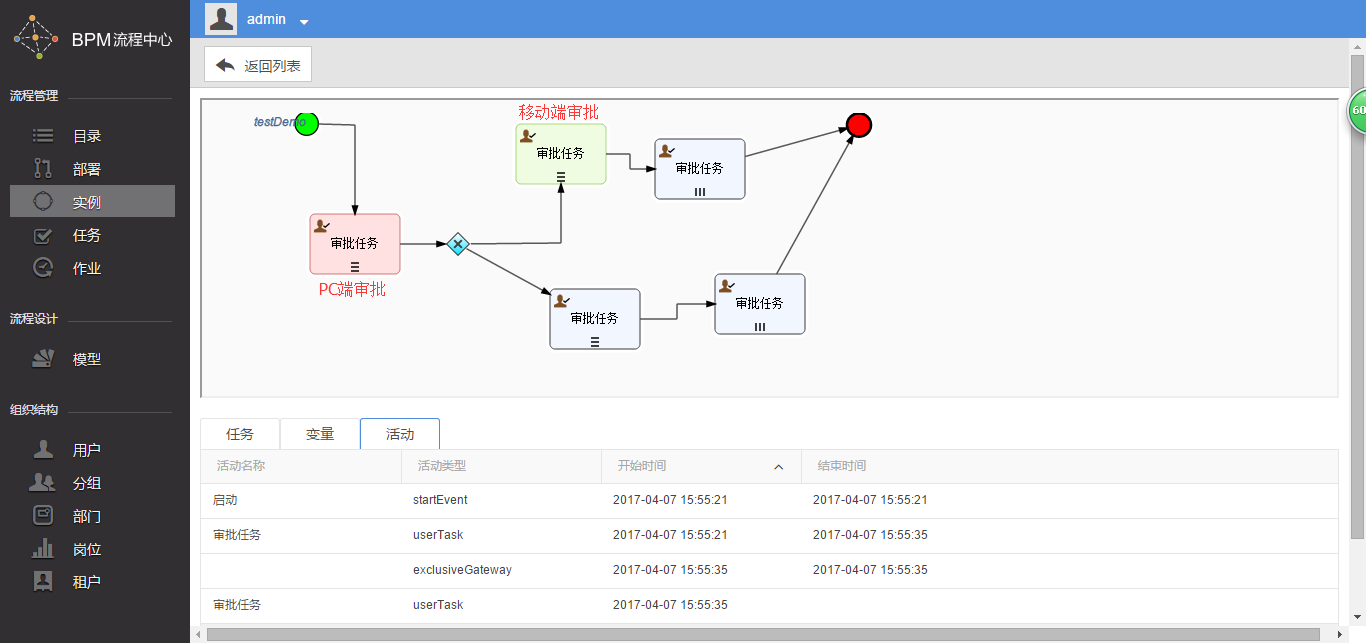


图4-15 在bpm追溯执行的示例及活动

补充：(该部分已经规避了)

1. 在对接过程中，出现mobile通过MA的网关直接调用MDM的webservice接口失败的问题
2. 出现mobile通过MA网关直接调用IUAP发布的restful服务不成功，提示406的问题:org.springframework.web.HttpMediaTypeNotSupportedException: Content type 'application/x-www-form-urlencoded;charset=UTF-8' not supported；

解决方式：通过在MA上直接调用IUAP发布的restful服务，不是通过网关的方式规避问题；

1. 移动调用bpm服务的审批动作通过传递id不成功，发现对于不同的任务，及代办任务id是不同的；故移动端需要加载代办事项，通过代办任务获取id，以便进行审批动作
2. 移动审批之后，在查看审批情况，并不能够区分是移动端发起的审批还是pc端发起的；