信产集团
IASSET GROUP

工作流技术及车贷系统实践

张宁
2018.5.9

信产集团
IASSET GROUP

PART 01

工作流引擎的基本概念

信产集团
IASSET GROUP

什么是工作流?

让在多个参与者之间按照某种预定义的规则传递文档、信息或任务的过程自动进行，从而实现某个预期的业务目标，或者促使此目标的实现

工作流的作用?

将部分或者全部的工作流程、逻辑让计算机帮你来处理，实现自动化.

工作流的优势?

1:将流程数据与业务数据解耦,各司其职.

2:灵活性高,在线上项目应用中,如果业务流程有变化,修改成本较小,可实现动态变更.

3:有一套相对完善的数据库设计，使用时无需直接操作流程数据，调用相应API即可。

信产集团
IASSET GROUP

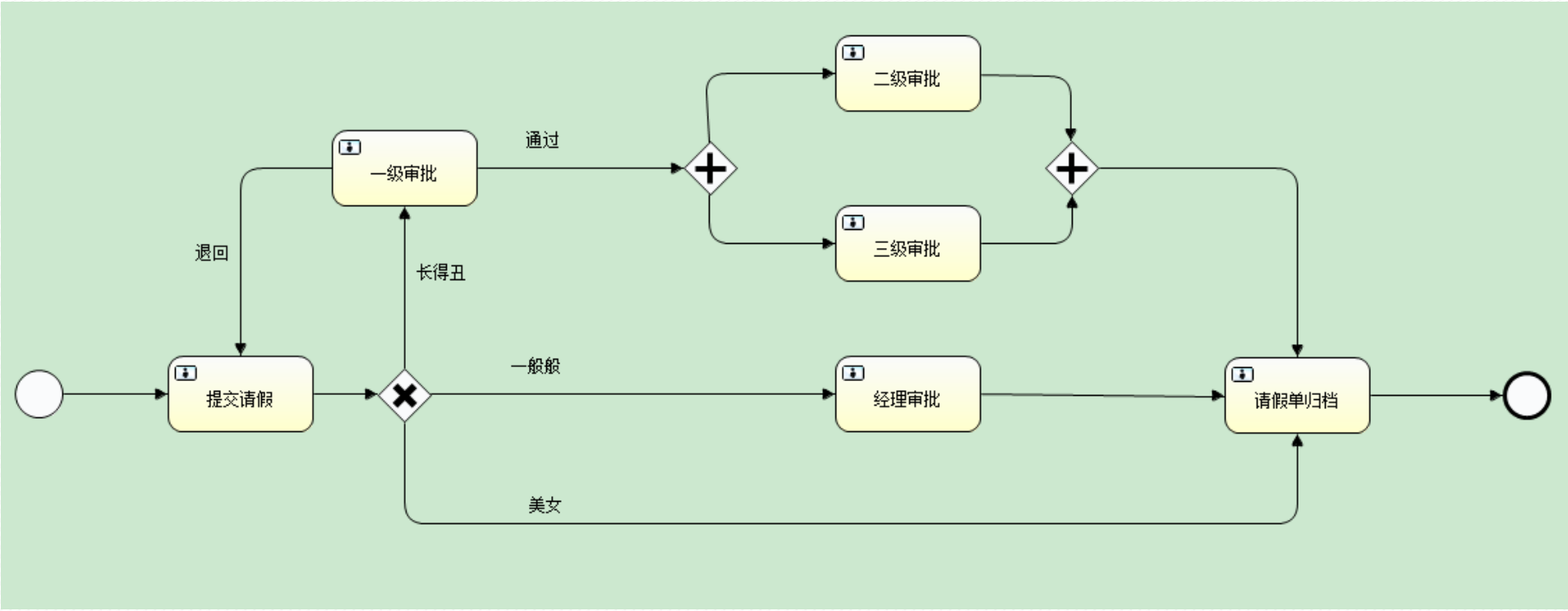
PART 02

工作流的基本用法

信产集团
IASSET GROUP

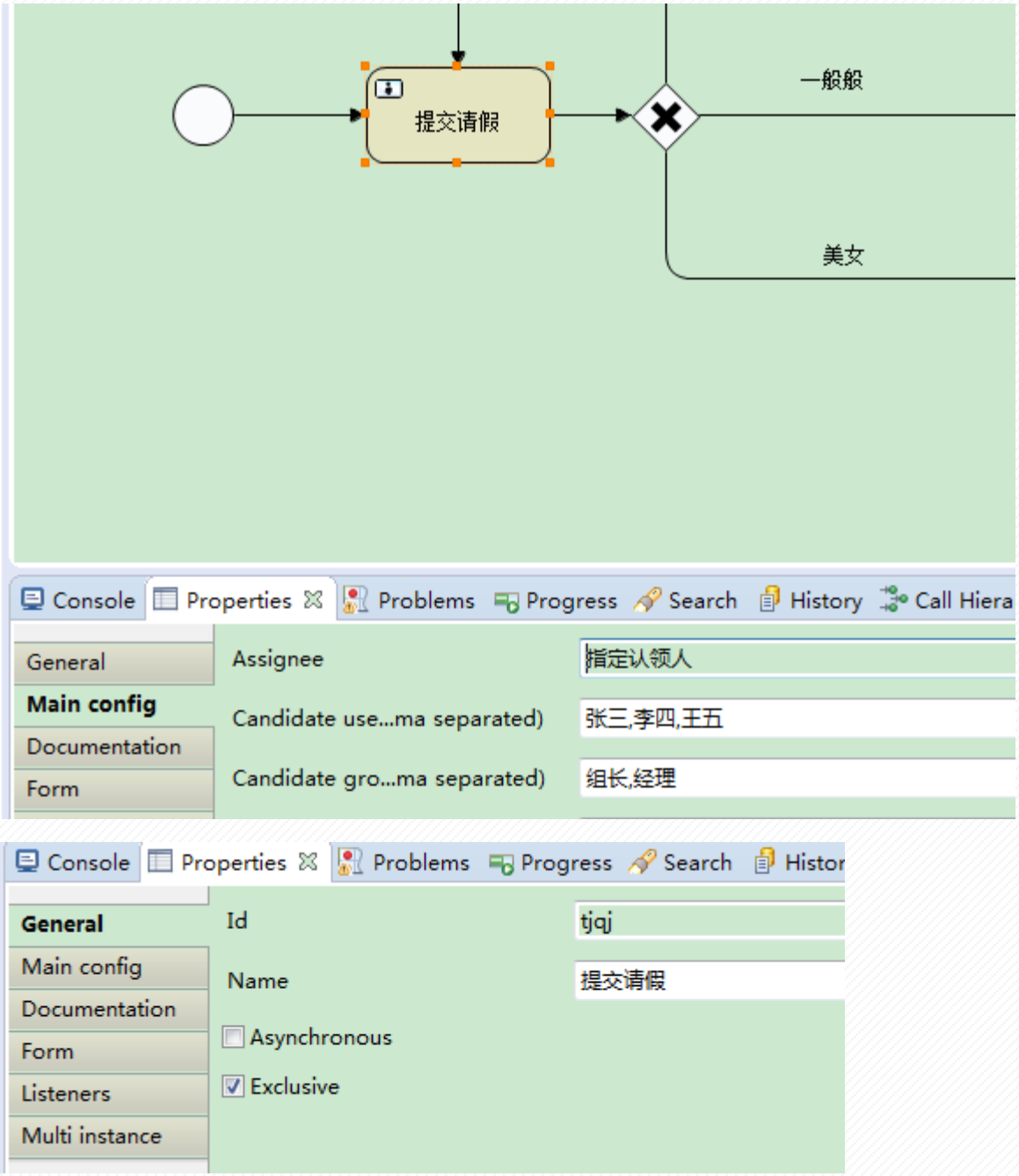
工作流的生命周期:

画流程图 → 部署 → 启动 → 流程流转 → 结束



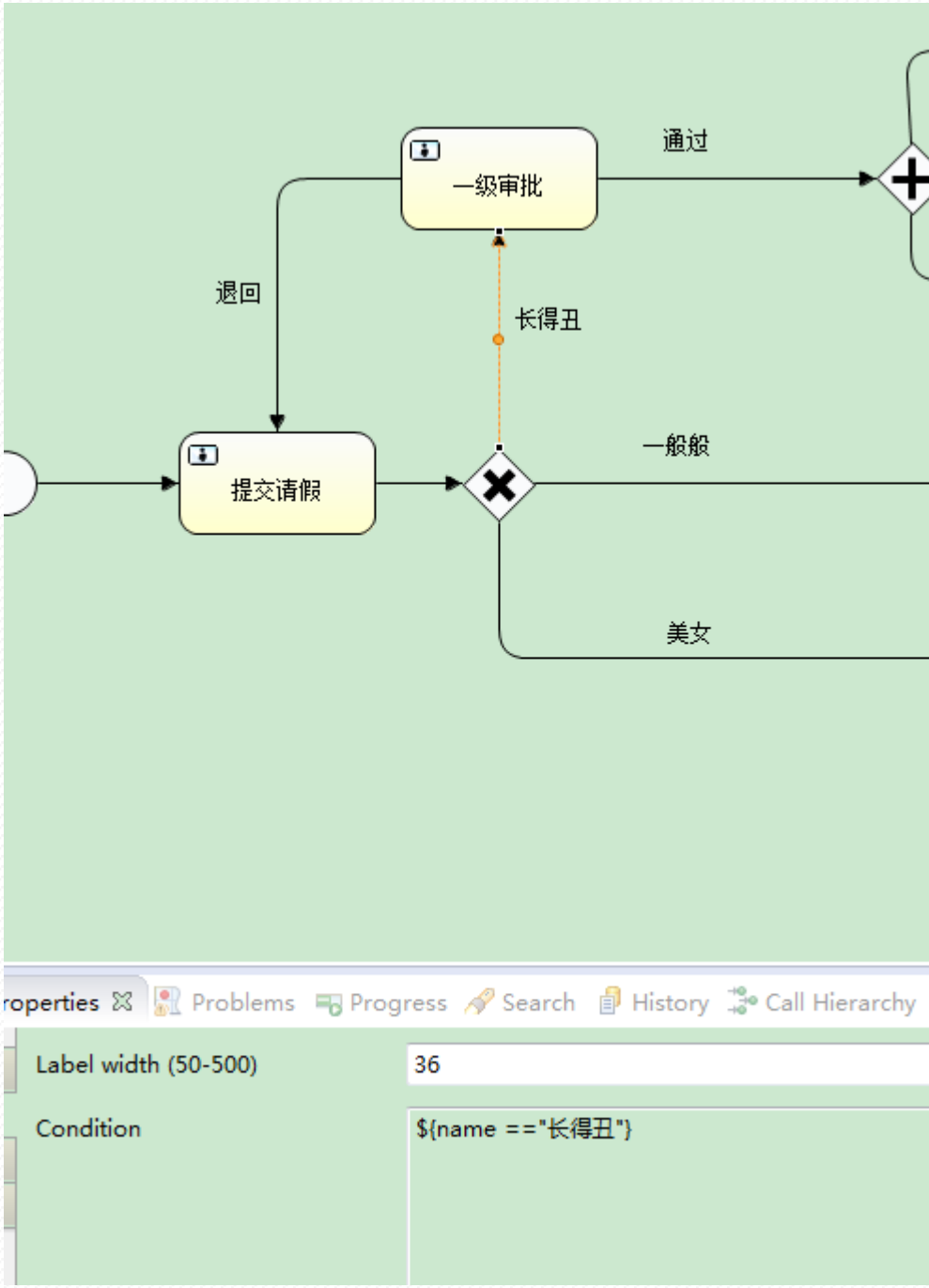
信产集团
IASSET GROUP

Task关键属性:



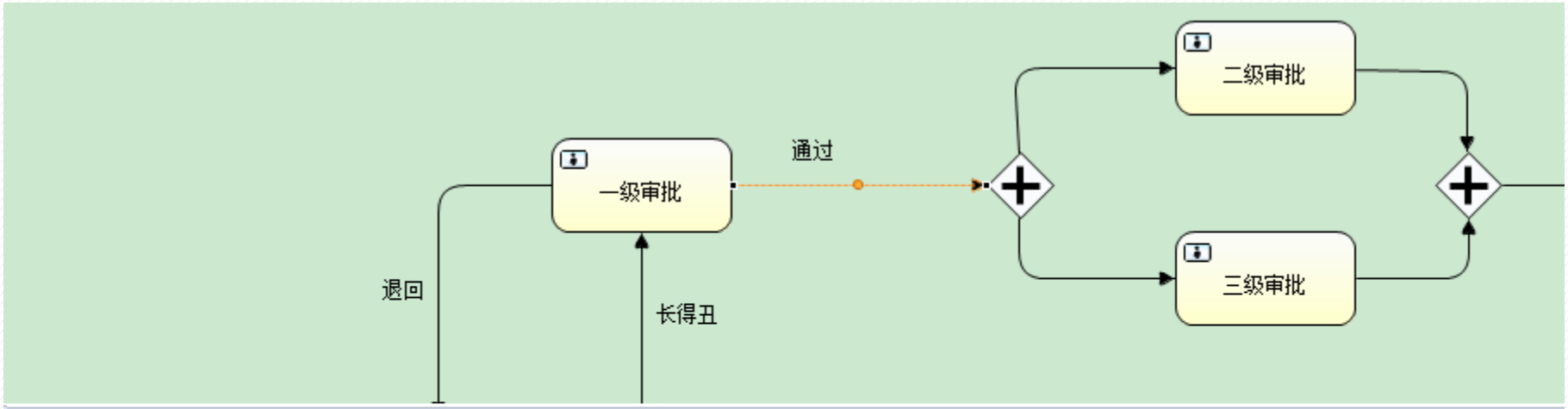
信产集团
IASSET GROUP

排他网关:(Exclusive Gateway:)



信产集团
IASSET GROUP

并行网关:Parallel Gateway



Console | Properties | Problems | Progress | Search | History | Call Hierarchy | JUnit | Debug | Servers

General
Main config
Documentation
Listeners

Label width (50-500) 24

Condition ${eventMark =="通过"}

信产集团
IASSET GROUP

核心API

属性:

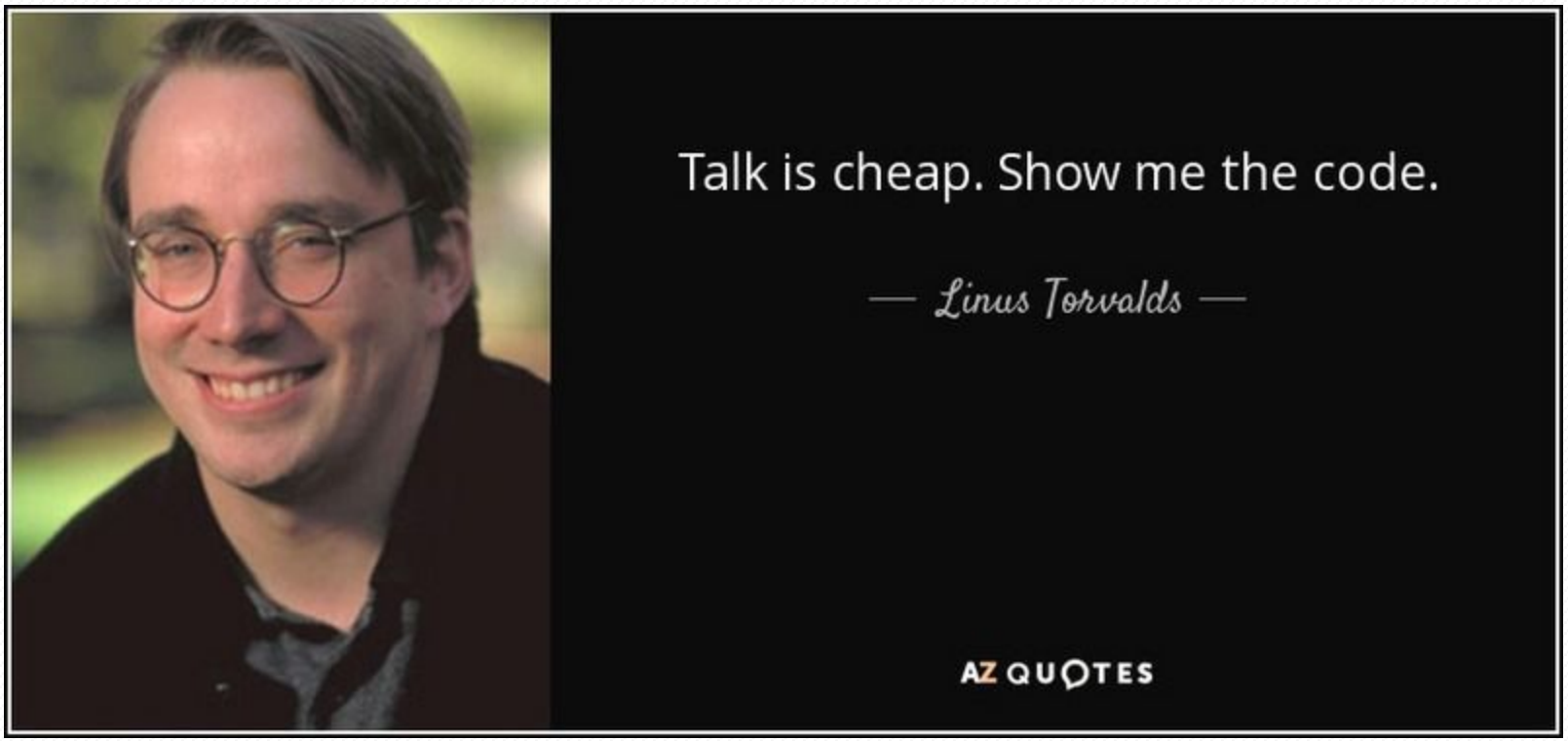
1. Deployment：流程部署对象，部署一个流程时创建。
2. ProcessDefinitions：流程定义，部署成功后自动创建。
3. ProcessInstances：流程实例，启动流程时创建。 (业务数据与流程数据的桥梁)
4. Task：任务，在Activiti中的Task仅指有角色参与的任务，即定义中的 UserTask。
5. Execution：执行计划，流程实例和流程执行中的所有节点都是Execution，如UserTask、ServiceTask等

接口:

1. ProcessEngine：一切服务的入口
2. RepositoryService：流程部署,文件操作相关
3. RuntimeService：运行时操作及查询相关流程数据.
4. TaskService：当前节点的操作接口
5. IdentityService：组合用户管理.
6. ManagementService：流程引擎维护接口,定制功能的时候用到
7. HistoryService：历史信息查询

信产集团
IASSET GROUP

放码过来



信产集团
IASSET GROUP

工作流数据探究

信产集团
IASSET GROUP

小结:

1:冷热数据分离,增加了活动节点的查询效率.

2:关键操作校验,避免重复修改.

3:并发修改的情况以数据库乐观锁的方式规避,保证数据安全.

4:数据操作留痕(认领,通过,回退,部署),方便查询跟踪.

信产集团
IASSET GROUP

PART 03

工作流在车贷系统的实践

信产集团
IASSET GROUP

使用工作流所遇到并解决的问题:

1:权限无法交给act管理:

重写源码中GroupEntityManager类的findGroupsByUser方法,将获取用户角色的地方改为从SSO获取.

2:需求中大量退回的操作,流程图过于复杂:

操作底层API,实现任意节点跳转.

3:检索的时候,流程状态和业务状态关联一起查询,得查两次,而且分页情况下筛选无法查询

关联数据表

工作流的权限托管

1:使用IdentityService接口同步系统数据到ACT_ID_*的表中;

优点:不破坏、不修改源码，面向接口编程

```
/**
 * @Comment:保存关系
 * @Author: ZhangNing
 * @Date:   2018/5/8 18:27
 */
@Test
public void UserShip(){
    User user = identityService.newUser("角色");
    identityService.saveUser(user);
    Group group = identityService.newGroup("关系");
    identityService.saveGroup(group);
    //绑定关系
    identityService.createMembership(user.getId(),group.getId());
}
```

工作流的权限托管

2:使用同名视图替换相关表结构,删除ACT_ID_*相关表

优点:不需要编写Java代码，只需要创建同名视图即可，对于现有系统的集成

```
<!-- Activiti配置初始化-->
<bean id="processEngineConfiguration"
      class="org.activiti.spring.SpringProcessEngineConfiguration">
    <property name="dbIdentityUsed" value="false"/>
    <property name="databaseType" value="mysql"/>
    <property name="databaseSchemaUpdate" value="true" />
```

```
SELECT DISTINCT
  RES.*
FROM
  ACT_RU_TASK RES
INNER JOIN ACT_RU_IDENTITYLINK I ON I.TASK_ID_ = RES.ID_
WHERE
  RES.ASSIGNEE_ IS NULL
AND I.TYPE_ = 'candidate'
AND (I.GROUP_ID_ IN('配置的角色'))
ORDER BY
  RES.ID_ ASC
```

信产集团
IASSET GROUP

工作流的权限托管

3:重写IdentifyService接口的默认实现,从SSO动态查询并组装成接口中需要的数据结构.

```
<!-- Activiti配置初始化-->
<bean id="processEngineConfiguration"
      class="org.activiti.spring.SpringProcessEngineConfiguration">
    <property name="dbIdentityUsed" value="false"/>
    <property name="dataSource" ref="dataSource" />
    <property name="databaseType" value="mysql"/>
    <property name="databaseSchemaUpdate" value="true" />
    <property name="jobExecutorActivate" value="false" />
    <property name="activityFontName" value="宋体"/>
    <property name="labelFontName" value="宋体"/>
    <property name="transactionManager" ref="transactionManager" />
    <!--自定义用户权限-->
    <property name="customSessionFactories">
        <list>
            <bean class="common.ActUserEntityServiceFactory"/>
            <bean class="common.ActGroupEntityServiceFactory"/>
        </list>
    </property>
</bean>
```

信产集团
IASSET GROUP

工作流的权限托管

```java
/**
 * Created by ZhangNing on 2016/5/5.
 */
public class ActGroupEntityServiceFactory implements SessionFactory{
    @Autowired
    private ActGroupEntityService actGroupEntityService;

    @Override
    public Class<?> getSessionType() {
        // 返回原始的GroupIdentityManager类型
        return GroupEntityManager.class;
    }

    @Override
    public Session openSession() {
        System.out.println("用的是我的方法");
        // 返回自定义的GroupEntityManager实例
        return actGroupEntityService;
    }
}
```

```java
/**
 * @Comment:自定义用户操作类，重写相关组权限查询方法
 * @Author: ZhangNing
 * @Date:   2016/5/9 14:41
 */
@Service
public class ActGroupEntityService extends GroupEntityManager {
    @Autowired
    private SSOClientService ssoClientService;

    public Group createNewGroup(String groupId) { return new GroupEntity(groupId); }
```

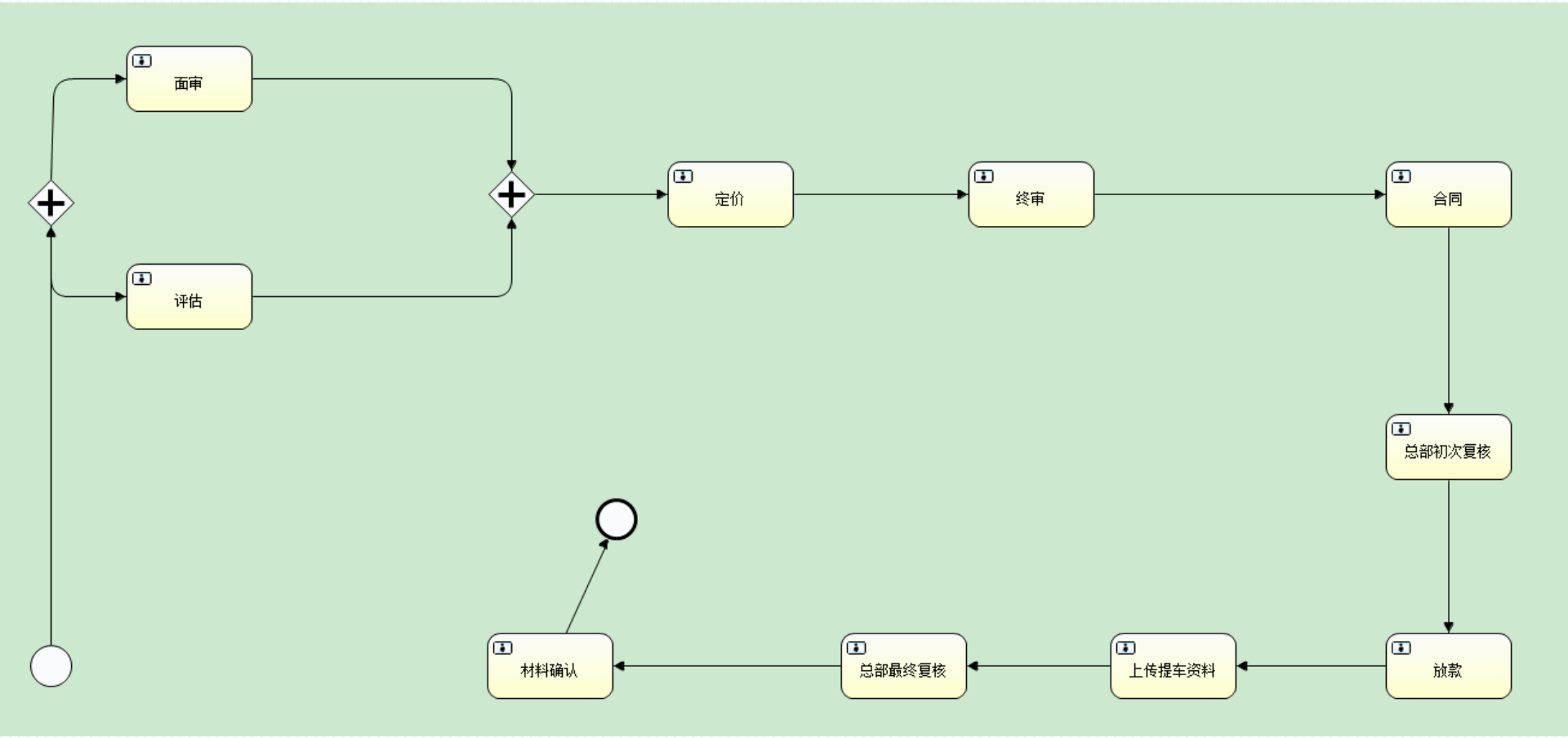
信产集团
IASSET GROUP

工作流的权限托管

```
public List<Group> findGroupsByUser(String userId) {
    /**
     * 修改返回参数，从sso中根据角色id获取相关组信息，以角色名称作为组ID（业务上强制规范角色名称不会相同）
     */
    List<Group> list =new ArrayList<Group>();
    List<SSORole> rolesByUserId = ssoClientService.getRolesByUserId(userId);
    for (SSORole role : rolesByUserId) {
        Group grop=new GroupEntity();
        grop.setId(role.getRolename());
        grop.setName(role.getRolename());
        list.add(grop);
    }
    return list;
}
```

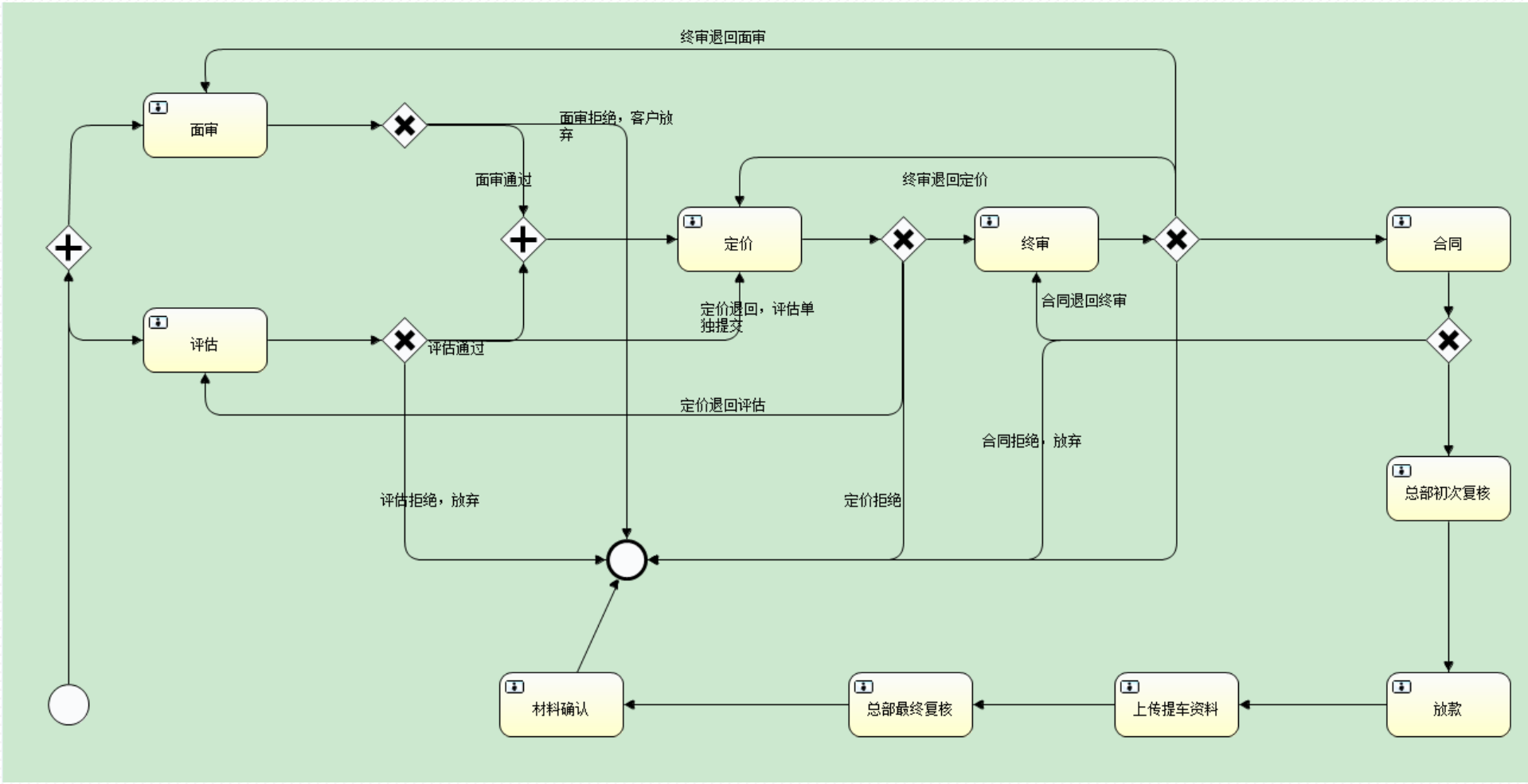
信产集团
IASSET GROUP

解决”中国式”回退问题



信产集团
IASSET GROUP

解决”中国式”回退问题



信产集团
IASSET GROUP

Act的流程图解析与运行过程是怎样的?

1:流程部署以后,bpmn文件会以二进制的形式存在数据库中

2:节点流转的时候,会动态的解析存在库中的bpmn,根据流程变量及状态找寻下一个跳转节点

3:操作底层api,移动并保存相应数据,完成节点流转.

源码中，节点流转的时候都干了什么？

```
public void claim(String taskId, String userId) {
  commandExecutor.execute(new ClaimTaskCmd(taskId, userId));
}

public void unclaim(String taskId) {
  commandExecutor.execute(new ClaimTaskCmd(taskId, userId: null));
}

public void complete(String taskId) {
  commandExecutor.execute(new CompleteTaskCmd(taskId, variables: null));
}

public void complete(String taskId, Map<String, Object> variables) {
  commandExecutor.execute(new CompleteTaskCmd(taskId, variables));
}

public void complete(String taskId, Map<String, Object> variables,boolean localScope) {
  commandExecutor.execute(new CompleteTaskCmd(taskId, variables, localScope));
}
```

```
/**
 * @author Joram Barrez
 */
public class CompleteTaskCmd extends NeedsActiveTaskCmd<Void> {
```

模拟流转，跳过解析流程图的步骤，调用底层相关方法，完成节点跳转

```java
/**
 * @Comment:自定义节点跳转
 * @Author: ZhangNing
 * @Date:   2018/5/7 20:20
 */
@Test
public void jumpTask(){
    String taskId="30003";
    String taskKey="tjqj";
    //根据传入的taskId查询流程实例id
    Map map=new HashMap();
    TaskServiceImpl taskServiceImpl=(TaskServiceImpl)taskService;
    taskServiceImpl.getCommandExecutor().execute(new TaskCommitCmd(taskId,taskKey, _type: "jump",map));
}
/**
```

信产集团
IASSET GROUP

```java
/**
 * @Comment:跳转到任意节点
 * @Author: ZhangNing
 * @Date:   2016/6/15 17:52
 */
public class TaskCommitCmd  extends NeedsActiveTaskCmd<Void>{
    private static final long serialVersionUID = 1L;
```

```java
    /**
     * @Comment:执行任意跳转
     * @Author: ZhangNing
     * @Date:   2016/6/16 10:23
     */
    @Override
    protected Void execute(CommandContext commandContext, TaskEntity task) {
        //缓存id
        if (variables != null) task.setExecutionVariables(variables);
        ExecutionEntity execution = task.getExecution();
        EngineServices engineServices = execution.getEngineServices();
        //流程定义id
        String procDefId = execution.getProcessDefinitionId();
        //获取服务
        RepositoryServiceImpl repositoryService = (RepositoryServiceImpl) engineServices.getRepositoryService();
        //获取流程定义的所有节点
        ProcessDefinitionImpl processDefinitionImpl = (ProcessDefinitionImpl) repositoryService.getDeployedProcessDefinition(procDefId);
        //获取需要提交的节点
        ActivityImpl toActivityImpl = processDefinitionImpl.findActivity(this.toTaskKey);
        if (toActivityImpl == null) {
            throw new ActivitiException("找不到Key为" + this.toTaskKey + "的流程节点，请根据流程图检查节点信息配置！");
        } else {
            //删除相关exetion数据
            task.fireEvent("complete");
            Context.getCommandContext().getTaskEntityManager().deleteTask(task, this.type, false);
            execution.removeTask(task);
            //执行规划的线
            execution.executeActivity(toActivityImpl);
        }
        return null;
    }
```

信产集团
IASSET GROUP

谢谢！