数据驱动

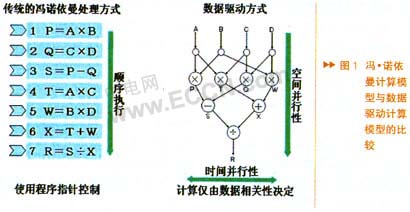
# 概述

数据驱动运算执行的顺序由输入数据的有效性决定。数据驱动的执行顺序只由数据的依赖关系决定，只要数据到达就可以进行运算，有效地挖掘了程序的时间并行性和空间并行性，非常适合于并行处理。

数据流计算的运算顺序不是预先确定的，而是在程序运行时动态确定，采用运行时的数据依赖性作为运算顺序的判定机制。

运行时数据流图，每个操作的结果有一个或多个目的地；只要操作数有效，操作将会被安排尽快进行。

数据驱动简单方便地挖掘出运算的时间并行性和空间并行性，使用这种结构将有助于提高系统的安全性和处理能力。

******

# 设计思想

1. 数据驱动的执行顺序只由数据的依赖关系决定。
2. 每个操作的结果有一个或多个目的地。
3. 数据流计算的运算顺序不是预先确定的，而是在程序运行时动态确定。
4. ***沿用XPDL文件作为模型的内容载体。其中控制流（Transitions）用于表示数据流，其对应的是任务节点间的输入、输出关系。***
5. ***沿用现有的ER数据库关系，丰富数据驱动的相关内容。（弱化数据库中的外键关系，以逻辑关系替代）***
6. ***对于数据池（数据中心）中的数据源采用驱动注入的方式，为任务的数据注入相应的驱动事件，在数据状态相应变更后，可以自动反馈驱动引擎，以控制任务的流转。***

# 设计方案

***建模：（XPDL）***

*<?xml version="1.0" encoding="UTF-8"?><Package Id="wfpkg604098" Name="数据驱动" xmlns="http://www.w3.org/2000/svg" xmlns:xpdl="http://www.w3.org/2000/svg" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xsi:schemaLocation="http://www.w3.org/2000/svg http://wfmc.org/standards/docs/TC-1025\_schema\_10\_xpdl.xsd">*

*<PackageHeader>*

*<XPDLVersion>1.0</XPDLVersion>*

*<Vendor>Together</Vendor>*

*<Created>2012-5-2 14:55:20</Created></PackageHeader>*

*<RedefinableHeader PublicationStatus="UNDER\_TEST"/>*

*<ConformanceClass GraphConformance="NON\_BLOCKED"/>*

*<WorkflowProcesses>*

*<WorkflowProcess AccessLevel="PUBLIC" Id="wfprcs604098" Name="数据驱动">*

*<ProcessHeader DurationUnit="D">*

*<Created>2012-5-2 14:55:20</Created>*

*<Description></Description>*

*</ProcessHeader>*

*<RedefinableHeader PublicationStatus="UNDER\_TEST"/>*

***<DataFields>***

***<DataField Id="f667855" IsArray="FALSE" Name="结构书"><DataType><BasicType Type="REFERENCE"/></DataType>***

***<ExtendedAttributes><ExtendedAttribute Name="multifiles" Value="false"/>***

***</ExtendedAttributes></DataField>***

***<DataField Id="f672847" IsArray="FALSE" Name="标准规范"><DataType><BasicType Type="REFERENCE"/></DataType>***

***<ExtendedAttributes><ExtendedAttribute Name="multifiles" Value="false"/>***

***</ExtendedAttributes></DataField>***

***</DataFields>***

*<Activities>*

*<Activity Id="F606201" Name="结构设计">*

*<Implementation><No/></Implementation>*

*<Performer>default\_participant\_id</Performer>*

*<Description></Description>*

*<StartMode><Automatic/></StartMode>*

*<FinishMode><Automatic/></FinishMode>*

*<ExtendedAttributes>*

*<ExtendedAttribute Name="ParticipantID" Value="default\_participant\_id"/>*

*<ExtendedAttribute Name="XOffset" Value="117"/>*

*<ExtendedAttribute Name="YOffset" Value="138"/>*

***<ExtendedAttribute Name="VariableToProcess\_OUT" Value="f667855"/>***

***<ExtendedAttribute Name="VariableToProcess\_OUT" Value="f672847"/>***

*<ExtendedAttribute Name="FormulaId" Value="1001"/>*

*<ExtendedAttribute Name="FormulaName" Value="数学公式"/>*

*<ExtendedAttribute Name="Source" Value="2"/>*

*<ExtendedAttribute Name="DataPackage" Value="1"/>*

*<ExtendedAttribute Name="DataCommitTo" Value="2"/>*

*<ExtendedAttribute Name="autoRun" Value="no"/>*

*<ExtendedAttribute Name="Ifaudit" Value="false"/>*

*<ExtendedAttribute Name="UrlContent" Value="0"/>*

***<ExtendedAttribute Name="TransferredType" Value="wait"/>***

***<ExtendedAttribute Name="TransferredExpression" Value="/>***

*</ExtendedAttributes>*

*</Activity>*

*<Activity Id="F628698" Name="容积率计算">*

*<Implementation><No/></Implementation>*

*<Performer>default\_participant\_id</Performer>*

*<Description></Description>*

*<StartMode><Automatic/></StartMode>*

*<FinishMode><Automatic/></FinishMode>*

*<ExtendedAttributes>*

*<ExtendedAttribute Name="ParticipantID" Value="default\_participant\_id"/>*

*<ExtendedAttribute Name="XOffset" Value="329"/>*

*<ExtendedAttribute Name="YOffset" Value="259"/>*

***<ExtendedAttribute Name="VariableToProcess\_IN" Value="f667855"/>***

***<ExtendedAttribute Name="VariableToProcess\_IN" Value="f672847"/>***

*<ExtendedAttribute Name="FormulaId" Value="1001"/>*

*<ExtendedAttribute Name="FormulaName" Value="数学公式"/>*

*<ExtendedAttribute Name="Source" Value="2"/>*

*<ExtendedAttribute Name="DataPackage" Value="1"/>*

*<ExtendedAttribute Name="DataCommitTo" Value="2"/>*

*<ExtendedAttribute Name="autoRun" Value="no"/>*

*<ExtendedAttribute Name="Ifaudit" Value="false"/>*

*<ExtendedAttribute Name="UrlContent" Value="0"/>*

***<ExtendedAttribute Name="TransferredType" Value="expression"/>***

***<ExtendedAttribute Name="TransferredExpression" Value="( f667855 gt 100) or (f672847 le 200)"/>***

*</ExtendedAttributes>*

*</Activity>*

*</Activities>*

***<Transitions>***

***<Transition From="F606201" Id="TRF606201F628698" Name="结构设计\_容积率计算" To="F628698">***

***<ExtendedAttributes><ExtendedAttribute Name="RoutingType" Value="NOROUTING"/>***

***<ExtendedAttribute Name="BreakPoint" Value="394;161;F653405;F606201;F628698"/>***

***</ExtendedAttributes></Transition>***

***</Transitions>***

*<ExtendedAttributes>*

*<ExtendedAttribute Name="StartOfWorkflow" Value="default\_participant\_id;F606201;174;78;NOROUTING;STARTF606201"/>*

*<ExtendedAttribute Name="EndOfWorkflow" Value="default\_participant\_id;F628698;530;274;NOROUTING;ENDNDF628698"/>*

*<ExtendedAttribute Name="processType" Value="static"/>*

*</ExtendedAttributes>*

*</WorkflowProcess>*

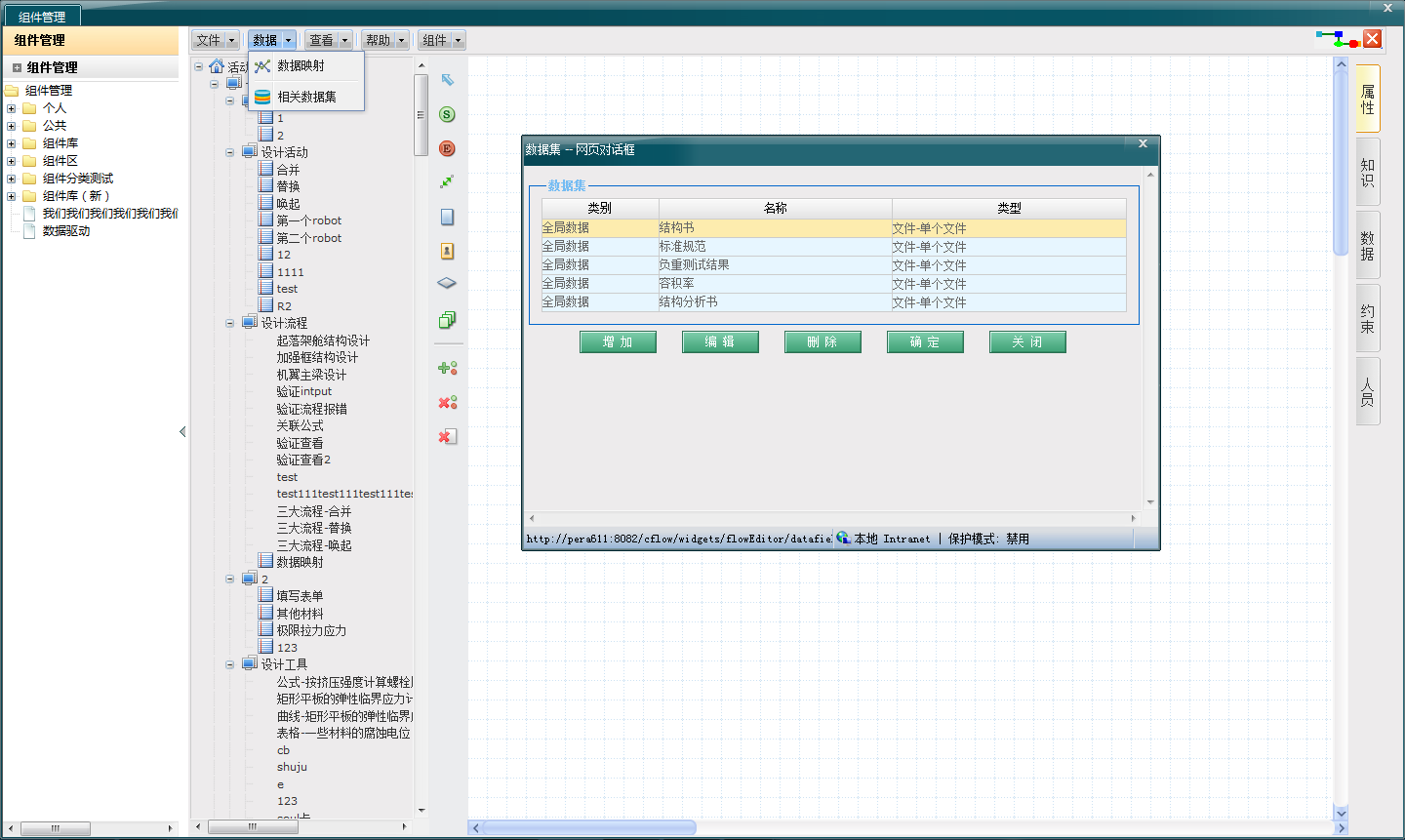
*</WorkflowProcesses>*

*<ExtendedAttributes>*

*</ExtendedAttributes>*

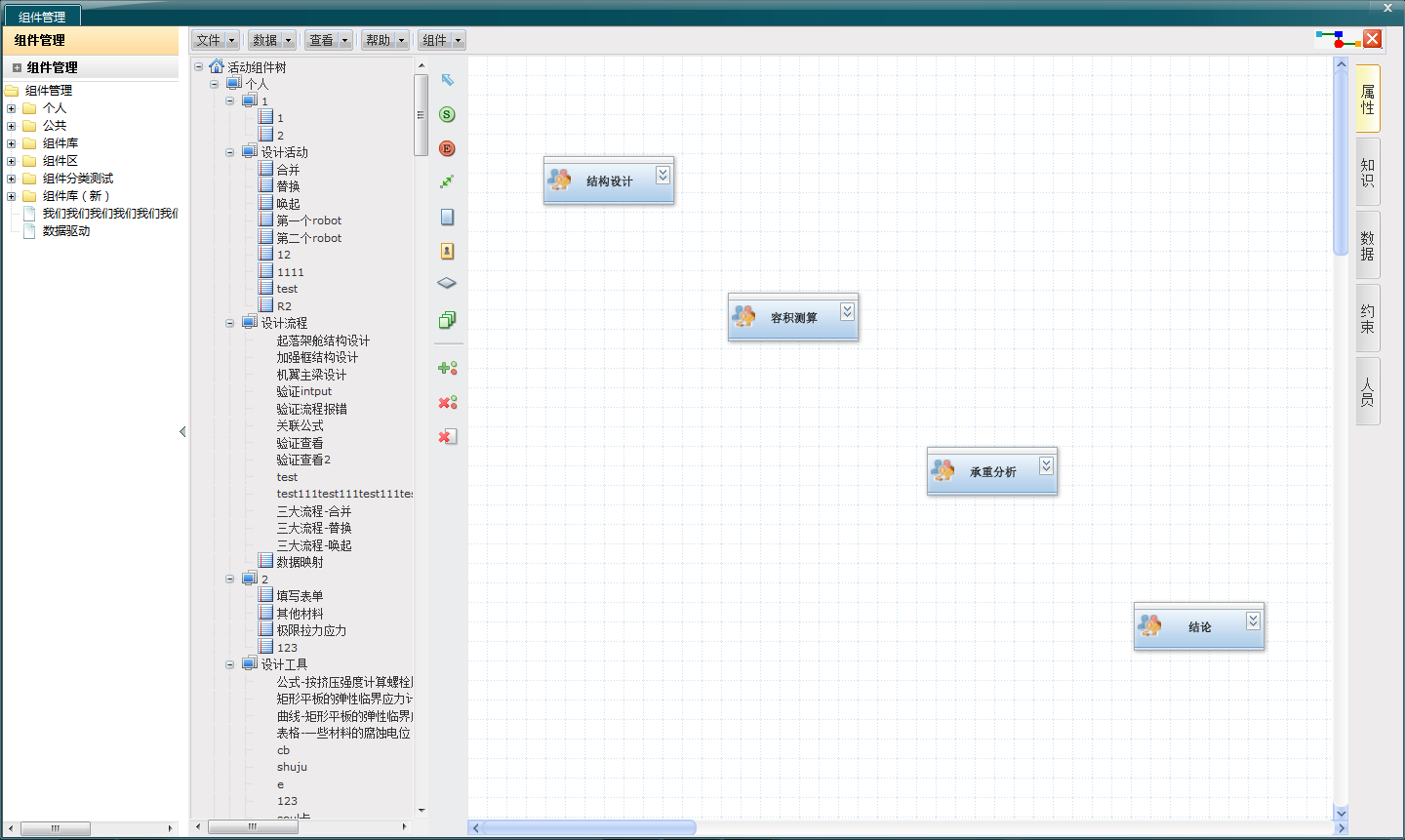
*</Package>*

***定义数据池（全局数据集）***



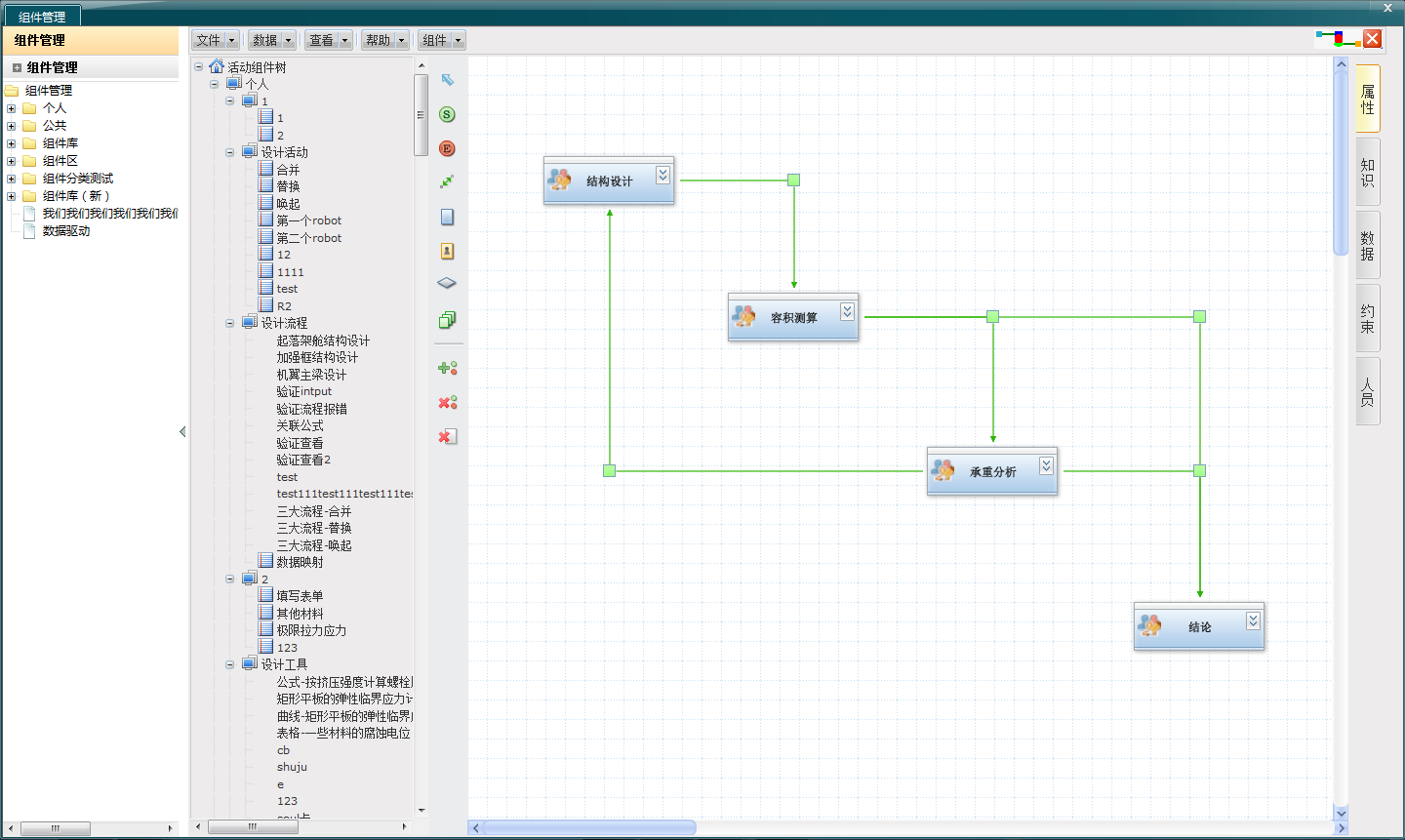
（全局数据集）

***定义任务***

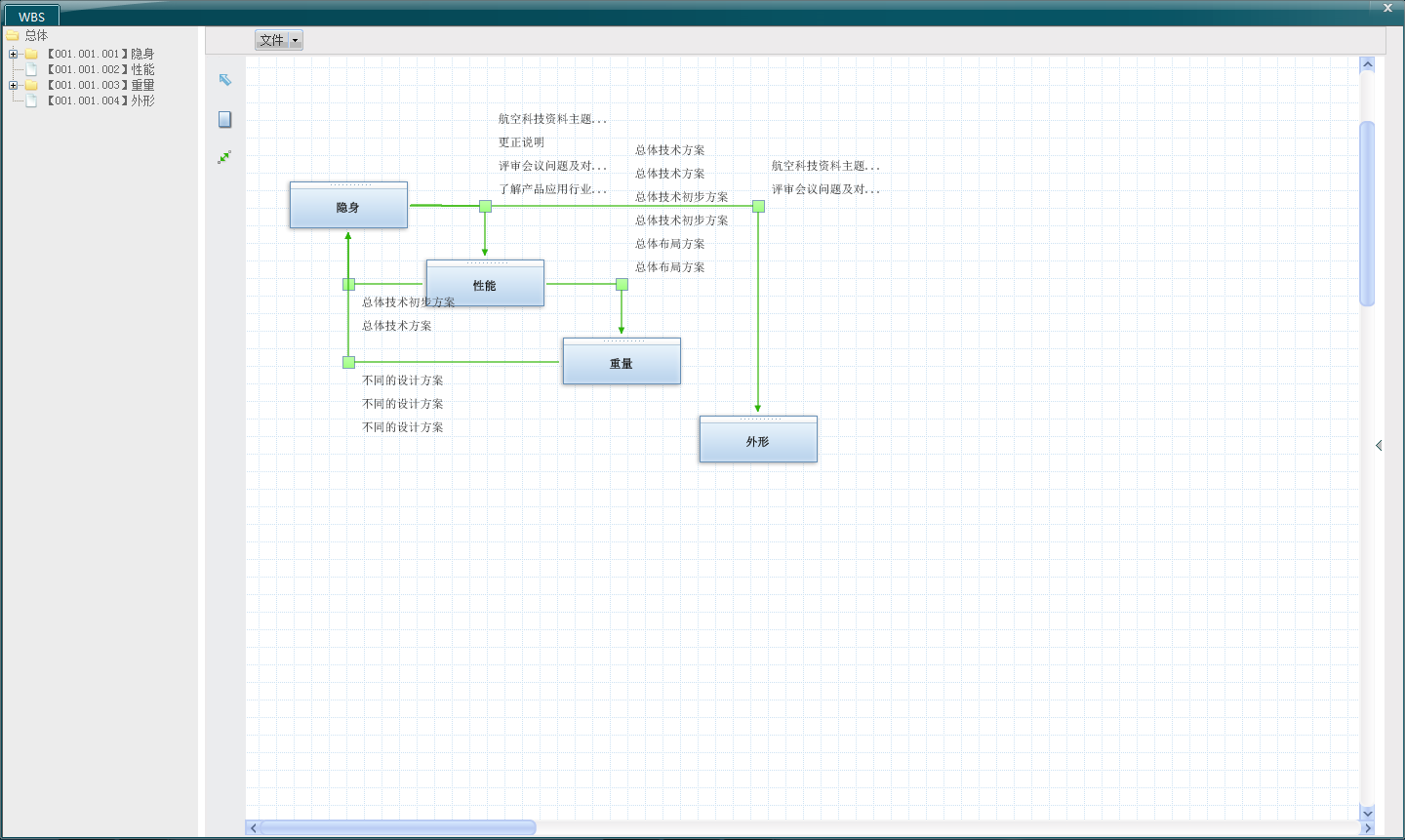


（搭建模型）

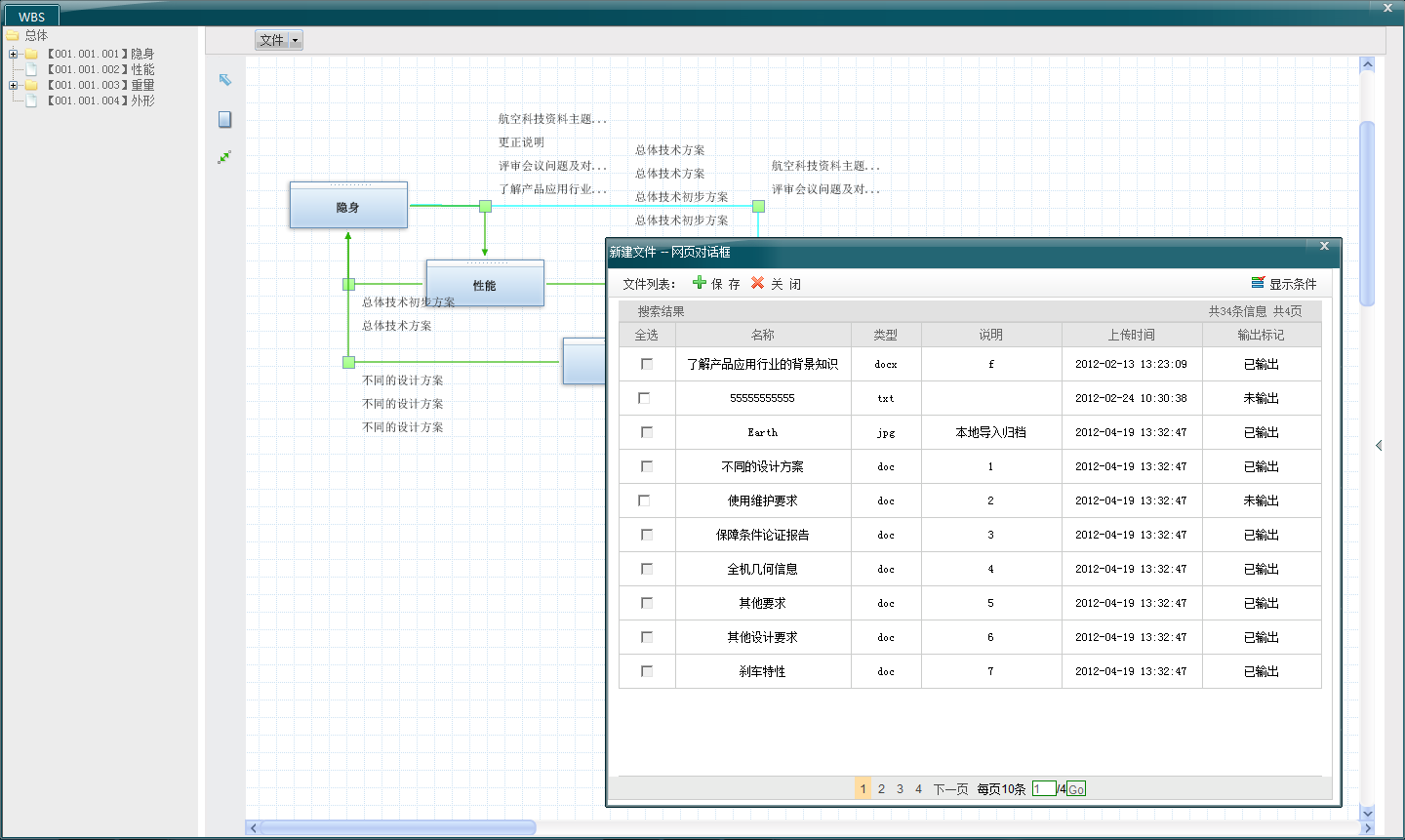
***定义数据流（IDEF0图）***



（IDEF0图—数据流）

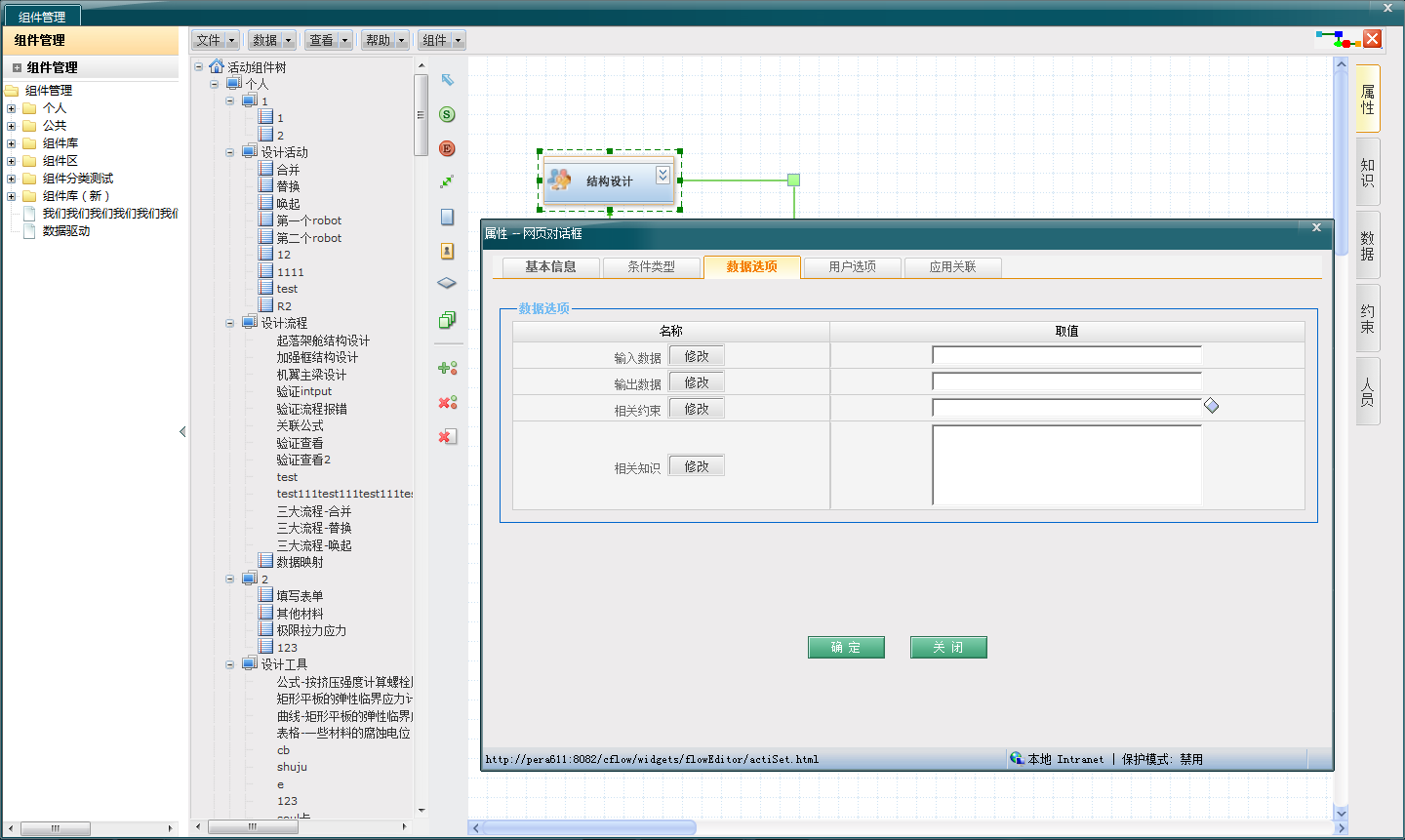


（IDEF0图—数据展现）



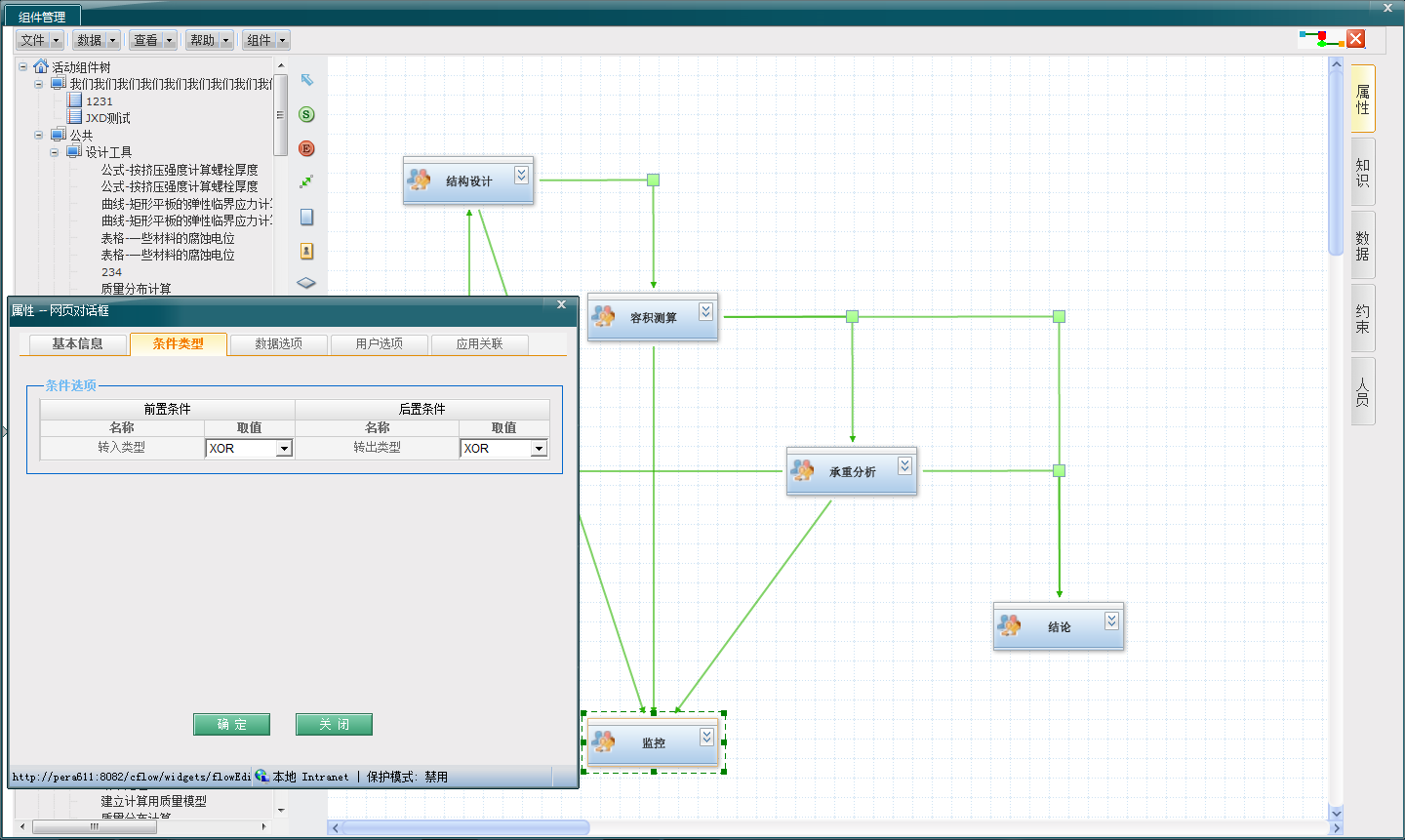
（IDEF0图-选择数据）

***定义任务输入、输出***



（输入、输出定义）

***触发条件（自动发起、等待输入、条件表达式）***

******

(转入表达式)

***运行：***

***解析模型（沿用XPDL规范）***

*import org.enhydra.shark.xpdl.XMLCollectionElement;*

*import org.enhydra.shark.xpdl.XMLElement;*

*import org.enhydra.shark.xpdl.XMLInterface;*

*import org.enhydra.shark.xpdl.XMLInterfaceForJDK13;*

*import org.enhydra.shark.xpdl.XMLUtil;*

*import org.enhydra.shark.xpdl.elements.Deadline;*

*import org.enhydra.shark.xpdl.elements.Deadlines;*

*import org.enhydra.shark.xpdl.elements.Package;*

*import org.enhydra.shark.xpdl.elements.Activity;*

*import org.enhydra.shark.xpdl.elements.BasicType;*

*import org.enhydra.shark.xpdl.elements.DataField;*

*import org.enhydra.shark.xpdl.elements.DataType;*

*import org.enhydra.shark.xpdl.elements.EnumerationType;*

*import org.enhydra.shark.xpdl.elements.EnumerationValue;*

*import org.enhydra.shark.xpdl.elements.ExtendedAttribute;*

*import org.enhydra.shark.xpdl.elements.PackageHeader;*

*import org.enhydra.shark.xpdl.elements.ProcessHeader;*

*import org.enhydra.shark.xpdl.elements.Transition;*

*import org.enhydra.shark.xpdl.elements.WorkflowProcess;*

*import org.enhydra.shark.xpdl.elements.WorkflowProcesses;*

*public boolean parseXpdl(Package pkg, PeraXpdl xpdl) {*

*try {*

*Map processMap = new HashMap();*

*Map nodeSubFlowMap = new HashMap();*

*if (log.isDebugEnabled()) {*

*log.debug("Loading package[" + pkg.getName() + "]...");*

*}*

*this.createXpdlExtendAttribute(xpdl, pkg);*

*WorkflowProcesses wfps = (WorkflowProcesses) pkg*

*.get("WorkflowProcesses");*

*for (Iterator it\_wfp = wfps.toElements().iterator(); it\_wfp*

*.hasNext();) {*

*WorkflowProcess wfp = (WorkflowProcess) it\_wfp.next();*

*if (log.isDebugEnabled()) {*

*log.debug("workflowProcess is " + wfp.getName());*

*}*

*ProcessHeader proch = wfp.getProcessHeader();*

*// create PeraXpdlProcess*

*PeraXpdlProcess xpdlProcess = new PeraXpdlProcess();*

*xpdlProcess.setCreationDate(proch.getCreated());*

*xpdlProcess.setDescription(proch.getDescription());*

*xpdlProcess.setName(wfp.getName());*

*xpdlProcess.setProcessId(wfp.getId());*

*xpdlProcess.setXpdl(xpdl);*

*// extended attribute*

*Map positionNodeMap = this.createProcessStartAndEndNodeMap(wfp);*

*// add by zhangtao on 2009-06-15*

*// set process's type to dynamic or static*

*this.setProcessType(wfp, xpdlProcess);*

*xpdl.getXpdlProcessMap().put(xpdlProcess.getProcessId(),*

*xpdlProcess);*

*processMap.put(wfp.getId(), xpdlProcess);*

*LinkedHashMap relevantDataMap = new LinkedHashMap();*

*// save peraXpdlRelevantData*

*for (Iterator it\_dataField = wfp.getDataFields().toElements()*

*.iterator(); it\_dataField.hasNext();) {*

*DataField df = (DataField) it\_dataField.next();*

*this.createPeraXpdlRelevantData(wfp, xpdlProcess, df,*

*relevantDataMap);*

*}*

*Map activityMap = new HashMap();*

*for (Iterator it\_act = wfp.getActivities().toElements()*

*.iterator(); it\_act.hasNext();) {*

*Activity act = (Activity) it\_act.next();*

*PeraXpdlNode pxn = this.createPeraXpdlNode(xpdlProcess,*

*act, positionNodeMap, relevantDataMap, activityMap,*

*nodeSubFlowMap);*

*if (act.getActivityType() == 3) {*

*pxn.setSynchronous(act.isSubflowSynchronous());*

*if (XMLUtil.getSubflowProcess(xml, act) == null) {*

*log.warn("Loading package[" + pkg.getName()*

*+ "] error!");*

*log.warn("Load workflowProcess[" + wfp.getName()*

*+ "] error!");*

*return false;*

*}*

*if (log.isDebugEnabled()) {*

*log.debug("Node[" + act.getId() + "] is "*

*+ act.isSubflowSynchronous());*

*log.debug(XMLUtil.getSubflowProcess(xml, act)*

*.getId());*

*}*

*nodeSubFlowMap.put(XMLUtil.getSubflowProcess(xml, act)*

*.getId(), pxn);*

*}*

*}*

*positionNodeMap.clear(); // clear positionNode*

*relevantDataMap.clear();*

*for (Iterator it\_tra = wfp.getTransitions().toElements()*

*.iterator(); it\_tra.hasNext();) {*

*Transition tra = (Transition) it\_tra.next();*

*this.createPeraXpdlEdge(xpdlProcess, tra, activityMap);*

*}*

*activityMap.clear();*

*}*

*// create PeraXpdlNodeSubFlow*

*this.createPeraXpdlNodeSubFlow(nodeSubFlowMap, processMap);*

*// clear all map*

*nodeSubFlowMap.clear();*

*processMap.clear();*

*} catch (Exception e) {*

*e.printStackTrace();*

*return false;*

*}*

*return true;*

*}*

***运行时的数据依赖性作为任务流转的判定机制***

1. *严格驱动：*

*根据输入数据的有效性判断该任务是否发起，在同版本下的输入符合规范时，自动发起该任务。*

1. *补充输入：*

*可以为未开始的任务补充本地的输入数据作为临时输入，发起该任务。*

*在任务提交时，需要判断输入数据的有效性。*

1. *同步执行*

*模型执行后，所有的任务自动发起，在任务提交时判断输入数据的有效性。*

1. *条件触发*

*数据流中可以设置针对数据的触发条件,当该数据满足条件时,触发制定任务的响应事件,自动发起任务。包含：自动发起、等待输入、条件表达式。*

***数据推送（引擎注入）***

1. *模型的全局数据集扩展：区分数据来源（内部、外部【工作包、知识、质量】）*
2. *模型在发布之后，将全局数据及其规则注入到数据管理中。*
3. *模型启动时，首先需要取得所有全局数据集的状态，根据规则发起相关任务。*
4. *模型启动后，在数据管理中的数据状态在变更时，需要根据注入的规则通知综合设计环境，进行相应的任务操作。*
5. *任务执行过程中允许用户增加自己的输入数据，其中输入数据可以在型号工作区中任务查找。（是否考虑权限？）加入的数据应同步加入到数据管理的规则注入中。*
6. *任务执行过程中允许用户增加自己的参考资料，考虑是在知识和质量中选择。*

*场景：*

*建模：*

1. *创建模型*
2. *增加全局数据集（数据池）*
3. *定义任务*
4. *数据映射*
5. *触发条件*

*运行：*

1. *解析模型（发布）*
2. *模型启动*
3. *任务发起*
4. *任务领取*
5. *输入展现（内部、平台、外部）*
6. *输入引入（本地、平台）*
7. *数据提交*
8. *任务提交*
9. *条件判定*