基于 ECA 规则和活动分解的工作流模型*

胡锦敏, 张申生, 余新颖

(上海交通大学 计算机集成技术开放实验室,上海 200030)
E-mail: jimhu@cs.utwente.nl; sszhang@sjtu.edu.cn
http://cit.sjtu.edu.cn

摘要: 企业在面临电子商务的挑战中,越来越重视业务过程重组.建立一种合理的流程模型是成功开展BPR(business process re-engineering)的关键.这样的模型应该可以集成企业许多业务相关的信息并且是可被系统解释执行的.在参考 WfMC(workflow management coalition)元模型基础上建立了一种基于ECA(event-condition-action)规则和活动分解的工作流模型.ECA 规则反映活动之间的执行依赖关系,通过重写办法把 ECA 模型变为触发器形式的 TA(trigger-action)模型,使模型的解析更高效,把事件重写为事件发生时间,使事件表达式具有更强的表达能力.活动分解的模型能很好地支持层次化项目管理.

关 键 词: 工作流模型;ECA 规则;事件驱动模型;触发器模型;重写
中图法分类号: TP311 文献标识码: A

工作流技术在实现企业过程重组、事务流水化处理以及信息流、文档流的管理等面向过程的应用以及在满足系统集成化应用需求方面显示了强大的功能和极大的应用前景.基于工作流的应用正成为现代企业软件系统的核心组成部分[1].工作流管理联盟(workflow management coalition,简称 WfMC)给工作流下的定义是:工作流是商业过程部分或全部地计算机化或自动化[2].换言之,工作流是为了达到一定的商业目的而根据一组定义的规则将文本、信息和任务在工作过程参与者之间传送的过程自动化.既然是一种流,那么工作流模型就是以一种过程的形式表现出来的.工作流模型是以过程为核心集成了以事务处理相关的其他信息的集成化过程模型.而这种模型中的集成化是体现在组成过程的单元上,也就是在活动上.因此,过程是由活动单元组成的,活动之间的关系决定了事务的处理过程.而活动在执行过程中与角色、执行者、资源、信息等其他信息联系起来.本文提出了一种基于事件-条件-动作(event-condition-action,简称 ECA)规则和活动分解的工作流模型.ECA 规则描述了触发活动的事件和内部条件,实际上也描述了执行活动的执行依赖关系.因此对工作流的解释执行主要是对与活动相关的事件的响应和对条件表达式的解析.本文提出一些理论把 ECA 模型变成了触发器-动作(trigger-action,简称 TA)模型,把活动的发生条件以触发器的形式表现出来,而触发器的最终形式就是条件表达式,使用规则重写技术可以实现它的解析.为了方便项目的管理,必须建立活动的纵向层次关系,本文提出了基于活动分解的办法来支持自顶向下(top-down)的设计.

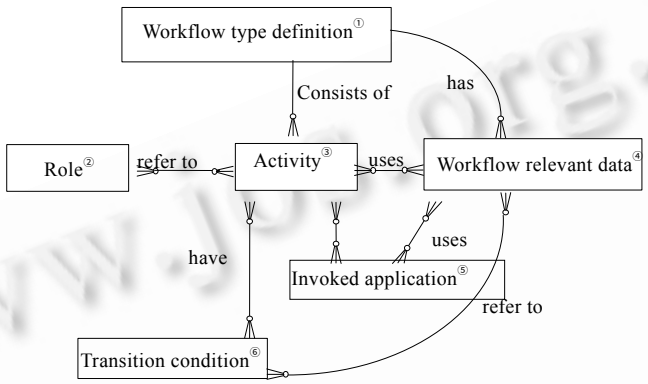
1 WfMC 定义的工作流元模型

WfMC 认识到给出一种通用的工作流定义模型能带来两方面的好处:(1) 由某种工作流建模工具产生的工作流定义可以在多种工作流运行系统中执行(enactment);(2) 通过输出一种通用的工作流定义到其他工作流系

* 收稿日期: 2000-06-26; 修改日期: 2001-01-15
基金项目: 国家自然科学基金资助项目(60073035);国家 863 高科技发展计划资助项目(2001AA415310,2001AA412010)
作者简介: 胡锦敏(1972－),男,广西平南人,博士生,主要研究领域为工作流管理系统,企业电子商务支持系统,复杂信息系统体系结构;张申生(1952－),男,上海人,博士,教授,博士生导师,主要研究领域为敏捷制造理论体系,分布异构环境集成技术;余新颖(1973－),男,硕士生,主要研究领域为工作流,并行工程.

统可以使工作流之间实现协作.

为此,WfMC 推出一个如图 1 所示的工作流的元模型(meta model)[3]来描述工作流定义中的对象、对象关系和属性,以此来形成一个工作流信息交换的格式集合的基础.工作流元模型中定义了以下一些关系,工作流类型定义(workflow type definition)由活动(activity)来组成,而活动中会使用某些工作流相关数据(workflow relevant data),而这些相关数据会出现在活动之间的迁移条件(transition condition)中,迁移条件决定了活动之间的关系,活动的执行是由某种角色(role)来分担的,在执行过程中会调用一些可调用的应用(invoked application)来完成相应的任务.这样的元模型集中反映了工作流定义中包含的对象之间的关系,但是没有针对各对象的属性与实现进行讨论.为了让这种模型可以执行(enactment),必须对此模型进行形式化并解释它.另外,针对具体系统的应用往往还要在元模型上作必要的扩充,本文余下部分将完成这些工作.



①工作流类型定义,②角色,③活动,④工作流相关数据,⑤可调用应用,⑥转移条件.

Fig.1 Workflow meta model by WfMC

图 1 WfMC 定义的工作流元模型

2 基于 ECA 规则和活动分解的工作流模型

2.1 基于元模型和ECA的工作流定义

从 WfMC 定义的工作流元模型可以看出,活动(activity)是工作流定义(工作流模型)的基本单元,也是工作流信息集成的核心.下面以集成化的活动定义为基础,给出一种基于活动和 ECA 规则的工作流模型.活动是工作流的基本组成单元,而 ECA 规则建立了活动之间的关系.

定义 **1**. 工作流是一个二元组,WFL::=({Activities},{ECA rules}),其中{Activities}是活动的集合,{ECA rules}是一组 ECA 规则的集合,每条 ECA 规则用来刻画活动之间的迁移条件.

定义 **2**. 集成化活动是一个十元组 Activity::=(IdName,Tasks,Role,Rdata,State,f,θ,ν,σ),其中 Id 为活动的标识,在工作流模型中,每个活动的 Id 是惟一的;Name 为活动的名字;Tasks 是任务集合,Tasks::=Null∪$\{Ti|i\in N\}$,N 为自然数集,Ti 为任务,Null 表示空任务集,抽象的活动的任务集为空任务集;Role 是活动执行者对应的角色;Rdata 是与此活动相关的工作流相关数据,包括输入数据和输出数据;State 是活动所处的状态.f是一功能映射,它把执行活动中的任务与一组工具相映射;θ是角色和执行者的匹配函数,负责把此活动的角色与执行者(人/Agent)进行匹配;ν是赋值函数,在活动执行中负责给 Rdata 中的变量赋值;σ是状态转换函数,用于改变活动的状态.

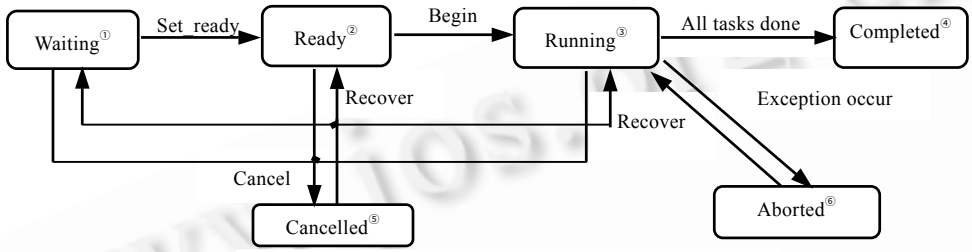
定义 **3**. 任务 T 是一个谓词表达式,T::=$\mathcal{F}(O)$,其中$\mathcal{F}$是一个逻辑谓词,表示一个动作或功能,如谓词 CreateDoc 表示创建文档;O 是动作的对象,如 CreateDoc(abc.doc)表示创建一个名为 abc.doc 的文档.这里的逻辑谓词$\mathcal{F}$对应了该任务的操作,通过功能映射函数 $\boldsymbol{f}$ 可以使它对应到供调用的应用(invoked application)的功能,如 f:CreateDoc→WinWord.exe.

定义 **4**. Role 是一个三元组,Role::=(Name,$\mathfrak{C}$,Å).Name 是角色名;$\mathfrak{C}$(capabilities)是角色的能力集合,如熟悉

VC 编程、熟悉 ProE 建立零件模型等;Â(authorizations)是一组授权集合,表明此角色对其他资源的访问或操作权限.

定义 **5**. 每个活动对应的工作流相关数据 Rdata 是与活动相关的变量及其值的列表 VarList,其中 VarList::={(VarName,VarType,VarValue)*},是变量名、变量类型及其值构成的三元组集合.

定义 **6**. 活动状态集 S={“Waiting”,“Ready”,“Running”,“Cancelled”,“Aborted”,“Completed”},其中 Waiting 表示活动触发条件未满足,活动执行条件还未就绪,是初始状态;Ready 表示活动的触发条件已经满足,可以开始执行活动中的任务;Running 表示该活动中的某些任务已经开始执行,但没有全部完成;Canceled 表示活动没有完成就不再被执行.Aborted 表示发生异常时退出执行状态;Completed 表示活动任务集中的所有任务都已执行完成.活动状态及其转换关系如图 2 所示.活动状态的变换是通过状态变换函数 σ 实现的,如 Set_Ready 就是一个状态变换函数,它把活动状态从 Waiting 变为 Ready.



①等待,②就绪,③运行,④完成,⑤取消,⑥非正常终止.

Fig.2 Activity state transition chart

图 2 活动状态转换图

定义 **7**. ECA 规则的形式是:

```
WHEN Events
    IF Conditions THEN
        Action
    ENDIF
ENDWHEN
```

例 1:一个工作流定义中的活动集合为{A1,A2,A3,A4,A5,A6},ECA 集合为如下 4 条规则:

```
WHEN Done(A1)
    IF A1.Rdata(“Money”)>1000 THEN
        Set_Ready(A2)
    ENDIF
ENDWHEN                                   (1)

WHEN Done(A1)
    IF A1.Rdata(“Money”)<=1000 THEN
        Set_Ready(A3)
    ENDIF
ENDWHEN                                   (2)

WHEN Done(A2) OR Done(A3)
    IF TRUE THEN
        Set_Ready(A4);
        Set_Ready(A5)
    ENDIF
ENDWHEN                                   (3)

WHEN Done(A4) AND Done(A5)
    IF TRUE THEN
        Set_Ready(A6)
    ENDIF
ENDWHEN                                   (4)
```

其中 ECA 规则中的 Done(Ax)均表示 Ax 完成这样的事件.此工作流定义所描述的工作流图可以如图 3 所示.

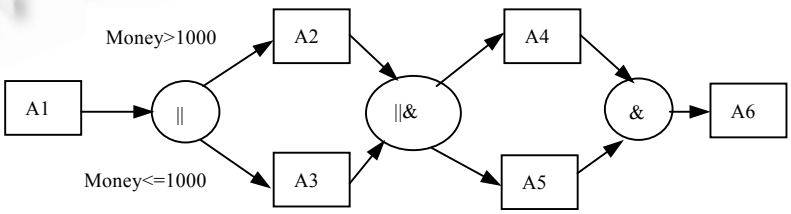


Fig.3 Workflow chart

图 3 工作流图

图 3 中的圆圈是表示活动之间的连接点(junction),||表示或(OR),&表示与(&),一个连接点中的圆圈只有一

种关系符表示它要么是分叉点(split,当其只有一个输入活动而有两个以上输出活动时),要么是汇合点(join,当其只有一个输出活动而有多于两个输入活动时).当其中有两个连接符时表示它既是汇合点又是分叉点.

显然,通过一组 ECA 规则,可以把活动的执行关系刻画出来,而且可以在 ECA 条件中加入内部限制.

2.2 ECA模型的TA形式

从上节中知道,利用 ECA 规则和一定的事件定义(如 Done)以及状态变换函数(如 Set_Ready)就能很好地表示活动之间的关系.但是这样的模型是把活动实体(activities)及其实体关系(ECA rules)分开描述,给模型存储和工作流引擎对其解释都带来了一定的困难,在实现时,如把它们分开存放在数据库中,则给解析带来不便.为此,这里把 ECA 规则变为一个触发器以作为一种活动单元的一种属性,表示活动是否可以触发.

定义 **8**. 触发器 Trigger 是一个二元组(E,ℓ),E 是事件逻辑表达式,ℓ是条件表达式.

E 表达式中的项是事件(后面扩充为包括事件的运算).事件的表现形式是$\Re(O)$,$\Re$是一元关系符,可以理解为事件的类型或是事件所发出的动作,O 是与此事件相关的对象.在本系统中,规定了两种与活动相关的事件,Begin(*A*)和 Done(*A*),分别表示活动开始或活动完成这两种事件.

条件表达式ℓ的形式如一般程序语言的逻辑表达式,在实现时用类 VBScript 的逻辑表达式实现.其表达式中的项的可能形式为:VBScript 定义的关系表达式,而表达式出现的变量必须是工作流中所定义的活动相关数据 Rdata,其表示形式为 A.RData(VarName).*A* 为活动 Id,VarName 是活动 *A* 中 RData 中所定义了的变量名.例 2 是一个合法的条件表达式的例子.

例 2:RData(“Count”)>5 And B.RData(“Money”)<10000+A.RData(“Money”)为一条件表达式,当活动 *A* 中的 Count 的值大于 5 并且活动 *B* 中的 Money 的值小于活动 *A* 中的 Money 的值时,其值为真.

当触发器中的事件表达式和条件表达式都得到满足时,触发器的值为真,否则为假.

定义 **9**. 关系 Occ(*A*)表示活动 *A* 的一个发生(执行).它表示活动 *A* 的状态变为 Running 这一时刻到活动 *A* 的状态变为 Completed 这一时刻之间的过程事实.

定义 **10**. BeginOf(Occ(*A*))表示活动状态变为 Running 这一时间点(timepoint,也叫时刻),EndOf(Occ(*A*))表示活动状态变为 Completed 这一时间点.因为没有理解上的差异,可以简单地把它们记为 BeginOf(*A*)和 EndOf(Occ(*A*)).

公理 **1**. 任一活动发生都开始于某一时刻且结束(包括中断、取消和退出)于某一时刻.

规则 1. 表示与活动 *A* 发生相关事件的 Begin(*A*)和 Done(*A*)分别重写为“BeginOf(*A*)”和“EndOf(*A*)”表示.

规则 1 把事件表达式中与活动相关的事件重写为事件发生的时间,这样不但不会影响事件表达式的一致性,反而可以增强它的表达能力,也给对它的解析带来了方便.

考察事件表达式 Done(*B*) OR Begin(*C*),当活动 *B* 完成或活动 *C* 开始时,其中的一项被替换成 TRUE,其值得到满足.如果把它重写为 EndOf(*B*) OR BeginOf(*C*),显然它表示当活动 *B* 完成这一时刻到达或活动 *C* 开始这一时刻到达时,其值为真.但是,这时的表达式中的项是一个时间点(timepoint)而不是关系式,无所谓 TRUE 或 FALSE,因此引入一时间点 CURRENTTIME,它表示计算这一表达式时的时刻.这样,上述表达式可以被重写为 (EndOf(*B*) <=CURRENTTIME) OR (BeginOf(*C*)<=CURRENTTIME),显然,此表达式可以拿去解析.这样,表示活动开始和完成这两个事件实际上是被重写为两事件发生的时刻到达这样的事件.

重写后可以进一步加强事件表达式的表达能力,在重写以前,事件表达式只表达事件之间的逻辑关系,而不能进行算术运算.在重写以后,事件表达式中的每一项都是关系表达式,关系表达式中的项是时间或时间的运算.如表达式(EndOf(*B*)+20<=CURRENTTIME) OR (BeginOf(*C*)<=EndOf(*B*)),当活动 *B* 完成后 20(小时)或活动 *C* 在活动 *B* 完成前开始,该表达式得到满足.另外,为了表示某活动在某时刻被触发这样的事件,我们引入事件 AtTime(*tp*),其中 *tp* 是一时刻,如“5/10/00 13:20:00”,AtTime(“5/10/00 13:20:00”)就表示 5/10/2000 14:28:01 这一时刻的到达.对于 AtTime(*tp*)这样的事件,可以把它重写为 *tp*<=CURRENTTIME.这些在第 3 节中将进一步讨论.

由于事件表达式可以重写为条件表达式的形式,因此可以把 Trigger 中的事件表达式和条件表达式合并起来,这样有

性质 1. Trigger::=E∧ℓ.可以表示为(E AND ℓ).

这样对 Trigger 的解析变成了对类似 VBScript 的条件表达式的解析,给分析 Trigger 的值带来了方便.

定义 **11**. 活动单元 AU::=(Trigger,Activity).活动单元的结构图如图 4 所示,Trigger 决定活动状态变为 Ready 的时刻,反映了活动之间以及活动与时间之间的关系.Activity 表达了活动的属性,包括 Tasks,Role,Rdata,State 等以及方法f,θ,ν,σ等.

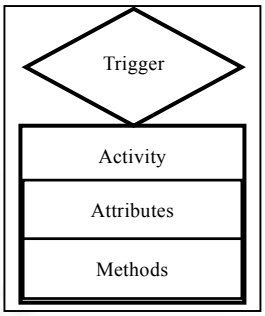


Fig.4 Activity unit
图 4 活动单元

由于活动单元中包含了活动和活动的触发器(反映了该活动与其他活动之间的关系及与时间的关系),所以有以下推论.

推论 **1**. 工作流是活动单元的集合,WFL::={AUs}.

在实现时,往往把 Trigger 也作为活动的一个属性,其值为布尔型(Boolean).这样,活动单元就变成了活动,工作流就变成了活动的集合,而活动之间的横向依赖关系通过其属性 Trigger 来反映.活动的纵向关系也可以通过扩展一个称为 ParentId(父亲,记录其父活动的 Id)来表达活动之间的父子关系,也就是其纵向关系,当然还要扩展一种方法来维护这种关系,它是 Sup(x)的实现,记为λ.因此扩展后的活动定义是:Activity^N::=(Id,Name,Tasks,Role,Rdata,State,Trigger,ParendId,f,θ,ν,σ,λ).

推论 **2**. 工作流就是一组建立了纵向关系和横向关系活动的集合,WFL::={Activity^N s}.

在实现时,当用关系数据库来存放工作流定义时,可以把扩展活动的信息作为一条记录存入到关系数据库中,而工作流则是一组活动记录的集合.工作流引擎就是要对这些记录及这些记录中的字段进行操作.至此,一个基于 ECA 规则和活动分解的工作流模型被实现并可存入到关系数据库中供工作流引擎执行.工作流引擎必须实现活动中f,θ,ν,σ,λ这些操纵活动属性和活动关系的方法.这些实现都比较简单,此处不再作进一步的描述.而对工作流引擎来说,决定哪些活动该发生是最核心的,它其实是对该模型中的 ECA 规则进行解析,也就是对 Trigger 的解析,我们把它称为工作流模型解释,将在第 3 节中进行讨论.

2.3 活动分解模型

为了有效地管理整个工作流项目,在工作流定义和任务分解的过程中,往往要建立活动之间的纵向关系以支持自顶向下(top-down)的项目流程设计,这样能够更好地反映出活动和任务管理的层次,也能和企业的组织结构的层次对应起来[4].

定义 **12**. 关系 Sub($A1$,A)::=(<,$A1$,A)是一个严格的偏序关系,表示活动的父子关系,$A1$ 是 A 的子活动.

定义 **13**. Sup(x)为一元运算符,并且 Sub(x,y)→Sup(x)=y 为真.

规则 2. 每个活动最多只有一个父活动.即$\forall y_1y_2$(Sup(x)=y_1∧Sup(x)=y_2))→(y_1=y_2∨Sup(x)=∅).

定义 **14**. 原子活动是没有子活动的活动,若 A 是原子活动,则$\forall x$ Sub(x,A)→x=∅为真.

定义 13 定义了与 Sub 具有对称关系的操作运算 Sup(x),就是取父活动的运算,规则 2 保证了 Sup(x),只有单值.可以把一个工作流对应的整个任务看成是一个大活动,而通过自顶向下进行活动的划分,划分出许多层次化的活动,一直分到原子活动(原子活动是直接分配给执行者完成的活动).而父子活动之间通过 Sub 关系来刻画,这样的划分保证了规则 1 的规定,即每个活动要么自己是根活动(最顶层的活动),否则它有且仅有一个父活动,这样就把工作流用活动树的形式表现出来了.活动树把工作流任务通过纵向层次的抽象,使得对项目的组织管理更适合实际的应用.若某一层次的某项活动交给某个组织部门或角色负责,则它有权利和义务组织好其下层活动的开展.活动的最终执行是反映在活动树的叶子上,也就是原子活动上.

至此,建立了基于活动层次化的工作流模型,也就是自定向下地建立了工作流模型中活动的纵向关系.这种 Top-Down 式的活动分解对项目管理非常有用.活动的层次关系的建立同样会影响活动的横向关系的表示.如活动 A 有 3 个子活动 $A1$,$A2$,$A3$,活动 B 的触发事件是 Done(A),显然,因为 A 是抽象活动,它不会被具体执行,被执行的是原子活动 $A1$,$A2$,$A3$,因此收不到 Done(A)这样的消息.同样,可以通过重写把该条件重写为 Done($A1$) AND Done($A2$) AND Done($A3$).当触发事件是 Begin(A)时,可重写为 Begin($A1$) OR Begin($A2$) OR Begin($A3$).

3 工作流模型的解释

工作流模型的解释实际上是指解释执行该工作流模型,这里只讨论模型解释的核心部分,即对 ECA 规则的解释,也就是对 Trigger 的解析.在第 2.2 节,已把 ECA 中的事件表达变成了时间表达.为此,先引入一个操作函数来给时间取值.

定义 **15**. TimePoint(*tp*)为一时间操作函数,其参数 *tp* 为一时间点,而函数的值是该时间点与相对时间点的差值,差值的单位反映了时间的粒度,可以根据系统的要求来选取.

例 3:设 tp 为"5/10/2000 13:20:00",而相对时间点是以 1/1/2000 00:00:00 为基准时间,时间粒度为秒,TimePoint(*tp*)的值是 24067210.

规则 3. 对事件表达式中的绝对时间值可以重写成相对时间值,如 AtTime(*tp*)重写成 TimePoint(*tp*),EndOf(*A*)重写成 TimePoint(EndOf(*A*)),CURRENTTIME 的取值直接取相对时间值.为了简单起见,对于 EndOf(*A*)和 BeginOf(*A*)在取值时也直接取相对时间值,而在表达时可以仍然写为 EndOf(*A*)和 BeginOf(*A*).

下面考察对以下 ECA 规则的解析:

```
WHEN Done(A) OR Done(B) OR AtTime("5/10/2000 14:28:01")
    IF  A.RData("Money")<100.0*1000 Or B.RData("Count")<10 THEN
        Set_Ready(C)
    ENDIF
ENDWHEN
```

在模型变换后,该 ECA 变为 Trigger-A 时,Trigger 表达式变为[(EndOf(*A*)<=CURRENTTIME OR BeginOf(*B*)+30*24*3600<=CURRENTTIME OR TimePoint("5/10/2000 14:28:01")<=CURRENTTIME)] And [(A.RData("Money")<100.0*1000 Or B.RData("Count")<10)],为了说明变换后的运算能力有所加强,表达式在 BeginOf(*B*)中故意加了 30*24*3600(秒)以说明问题,这是传统 ECA 规则中事件表达式难以表达的.

现在考察此 Trigger 的解析,当 *A* 完成这一事件发生时(假设 *B* 未开始),工作流引擎会收到 Done(*A*)这一消息,设此时的时间为 5/10/2000 13:20:00,若此时解析 Trigger,则 Trigger 变为[TimePoint("5/10/00 13:20:00")<= TimePoint("5/10/00 13:20:00") OR FALSE OR TimePoint("5/10/00 14:28:01")<=TimePoint("5/10/00 13:20:00")] And [97888.0<100.0*1000 OR FALSE].在此解析过程中,主要是重写表达式,重写中把含有未知其值的变量的逻辑项置为 False(假),这样不会影响 Trigger 真正的意义,又能使其处于可归约状态(可计算状态).如 BeginOf(*B*)+30*24*3600<=CURRENTTIME 这一逻辑项因为 BeginOf(*B*)没有发生,则其值未知,所以置为 FALSE.再通过用相对时间替换,该 Trigger 的值就被计算出来为 TRUE.

在实现时,把事件的值、相关数据 Rdata 中变量的值以(变量名,类型,值)的形式存放在数据库中.当计算 Trigger 的值时,对表达式中出现的变量,从数据库中读取其值,并替换式中的变量以使该表达式可解析.

4 结 论

参考 WfMC 提出的工作流元模型,本文提出了一种基于 ECA 规则和活动分解的工作流模型,对模型进行了形式化.同时,针对 ECA 模型中对事件运算表达能力不强以及规则解析较难等缺点,本文把其变为了 TA 形式,并用重写的办法给出其解析办法.我们已经开发了建立该工作流模型的图视化建模工具软件.在国家 863/CIMS 重大攻关项目"海南新大洲并行工程实施"中,我们设计并实现了该工作流模型的解释和运行,以支持摩托车的并行设计过程管理.事实证明,单纯的过程模型无法反映活动管理的层次化关系,单纯的层次化关系无法控制流程的执行.结合这两方面的工作流模型能克服单纯过程模型的不足.基于 ECA 的过程模型给过程的调整和动态变化带来了方便,增强了其适应变化的能力,变形为触发器形式给解释执行该模型带来了极大的方便.该模型有很大的实际应用意义,可以在实现 BPR、并行工程、商业流程管理、供应链管理等各种面向过程和工作流的应用系统中得到很好的应用.

Reference

[1] Leymann, F., Roller, D. Workflow-Based application. IBM System Journal, 1997,36(1):102~123.

[2] Workflow Management Coalition. The workflow reference model. Document Number TC00-10003. 1994.

[3] Workflow Management Coalition. Interface 1: process definition interchange process model. Document Number WfMC TC-1016-P. 1998.

[4] Hu, Jin-min, Zhang, Sheng-shen. An agile workflow system supports virtual enterprise. Computer Research and Development, 1999,36(12):1517~1523 (in Chinese).

附中文参考文献:

[4] 胡锦敏,张申生.支持企业动态联盟的敏捷工作流系统.计算机研究与发展,1999,36(12):1517~1523.

A Workflow Model Based on ECA Rules and Activity Decomposition*

HU Jin-min, ZHANG Shen-sheng, YU Xin-ying

(*Laboratory of Computer Integrated Technology*, *Shanghai Jiaotong University*, *Shanghai* 200030, *China*)

E-mail: jimhu@cs.utwente.nl; sszhang@sjtu.edu.cn

http://cit.sjtu.edu.cn

Abstract: While facing the challenge of e-commerce, the enterprises pay more attention to the business process re-engineering (BPR). A reasonable model to describe the business process is the key point to BPR sucess. Such a model should integrate all related business information and should be enacted by a system. Referencing on the WfMC (workflow management coalition) meta-model, a workflow model based on ECA rules and business activity decomposition is presented. The ECA rules imply the execution dependencies among the activities. Using rewriting techniques, the ECA (event-condition-action) model is transformed to Trigger-Action model to make the interpretation more efficient. Rewriting the event to the occurrence time of the event makes the event expression have more expressive semantics. Activity decomposition model can support the hierarchical project management.

Key words: workflow model; ECA rules; event-driven model; trigger model; rewriting

* Received June 26, 2000; accepted January 15, 2001

Supported by the National Natural Science Foundation of China under Grant No.60073035; the National High Technology Development 863 Program of China under Grant Nos.2001AA415310, 2001AA412010