智能工作流系统 黑板模型 人工智能 (12)

51-53

计算机科学1998 Vol. 25 №. 4

基于黑板模型的智能工作流系统

A Blackboard-Based Intelligent Workflow System

TP18

胡 华[1,2] 何志均 高 济

(杭州商学院计算机与信息工程系 杭州 310035)[1] (浙江大学计算机科学与工程学系 杭州 310027)[2]

摘 要 Workflow and BPR are the most prominent technologies in 1990s. But the current workflow systems still have many problems when they are applied in the practical environment. Based on the setting of State-transforming rules and active spans of task, this paper presents a scheme for designing and realizing of a Blackboard-based intelligent workflow system by technology of Knowledge base and studying agent. Ths system can be widely applied in fields of workflow control and BPR.

关键词 Workflow, BPR, Task decompose, Knowledge-base, Blackboard model

工作流(Workflow)和企业流程重组(BPR: Business Process Reengineering)是90年代达到全盛的企业操作管理技术。通过从传统的关注产品转向到关注流程(Process),工作流和企业流程重组从本质上反思企业工作流程,对之进行彻底的重新设计,从而达到在成本、质量和服务等方面的重大改善。

早在七十和八十年代的办公室信息系统(OIS)中,人们就提出了采用计算机的流程管理观念[2,4]。但是由于技术和理论上的欠缺,OIS 对分布环境下的企业流程协同工作缺乏有效的支持,未能取得预期的成功。八十和九十年代信息和网络技术的迅猛发展,为工作流和企业流程重组的兴起与发展打下了坚实的基础。有很多企业已开发出了分布式环境下的流程控制软件系统(如:IBM 公司的 Lotus Notes 和 Microsoft 公司的 Exchange 等)。这些系统不仅继承了早期 OIS 的流程控制任务特点,而且还利用现有技术成果在用户界面和分布式协作等方面比以往的流程控制系统有了很大的改进。尽管如此,现有的工作流系统在实际使用过程中仍然存在着适用范围、灵活性和意外情况处理等方面的许多问题[4]。匹茨堡大学的 Mahling 等人提出采用 Agent 和基于目标的智能工作流方法来解决适用范围和灵活性的问题[4],但是 Mahling 的方案对于流程系统中经常出现的处理无限期延长和异常处理等问题仍未给出有效的方法。本文通过给任务对象设置生存周期和状态转换规则的任务分解方法,结合基于目标的智能工作流方案设计并实现了一个基于黑板模型的智能工作流系统(BBIWFS),该系统在保持现有工作流系统的分布协同工作特点的基础上,较好地解决了适用范围、控制灵活性和意外情况处理等问题,从而可广泛用于企业加强内部管理,为企业实现降低生产成本,提高工作效率,赢得市场竞争提供有效的支持。

一、任务分解及其语义模型

企业管理工作中的每次活动都可以与一个抽象的任务对应起来,这样企业组织的一次具体管理活动的执行和完成过程,本质上可以用一次任务的执行完成过程来描述,而管理过程中的每一具体步骤细节过程又对应于任务分解成子任务后的子任务完成过程。根据这个原理,我们可以采用如图1的任务层次分解和控制的方式来控制、描述和分析企业的生产活动情况。图中,最高层的任务为总任务,各任务下层中用直线相连的任务为其子任务,子任务的执行次序为从左向右,但用弧线关联的子任务表示为可并发执行的子任务(如 subtask22和 subtask23等),每个任务完成的条件为其各子任务都已完成。

定义1 一个任务实例是一个六元组 T(N,A,C,M,T,P)。其中:①N 为系统全局唯一定义任务实例标识;②A 为任务所关联的信息属性集合;③C 为任务实例的当前状态:执行(E)、就绪(R)、挂起(H)、完成(C)、超时执行(T)和非正常结束(A)等;④M 为对任务可执行的操作;⑤T 为任务实例已经经过的状态轨迹;⑥P 为任务实例可活动存在的时间跨度。

任务实例的状态转换图见图2。其中,若任务实例有下级子任务则:1)任务处于"完成"态的必要条件为:所有关联子任务的状态为"完成";2)任务处于"执行"态的必要条件为:有一个或多个关联子任务的状态为"执行"。

在实际实现时,为防止对任务的恶意挂起,与"挂起"状态相关的操作应有权限限制 (如:在 BBI-

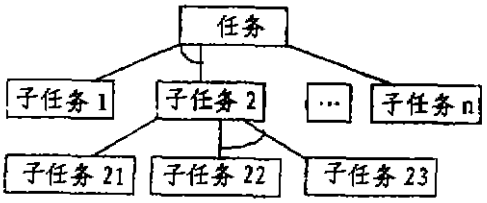

图1 任务的层次分解

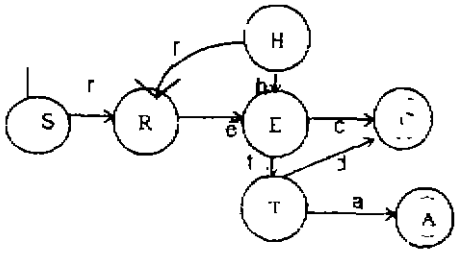

图2 任务状态转换图

WFS中只有系统管理员才能执行)。

这样我们可得到如下七条任务状态转换规则(其中,"∅"为空字符):

1. S→rR;　　5. T→cC|aA;

2. R→eE;　　6. C→∅;

3. E→hH|tT|cC;　　7. A→∅。

4. h→rR;

对于没有子任务的任务实例,以上规则执行将得到该任务的执行状态轨迹;而对于还有后继子任务的任务实例,其执行状态轨迹则由其本身执行以上规则的执行状态轨迹和其各子任务执行以上规则的执行状态轨迹共同组成的轨迹集合。

由于每个任务实例具有一个活动生存的跨度,从而使得控制系统能够迅速发现该任务是否存在延时问题,并作出相应的处理,如结束当前任务的执行以避免无限延期等以保证任务的最终结束。

最后,一个任务通常是作为一类任务的实例而存在的,这类任务形成一个任务类,任务类表示这样一组任务对象;它们不仅有相同的操作和属性,而且任务状态经过的轨迹亦满足相同的约束。由任务类产生任务实例需要进行实例命名和状态初始化。

定义2 任务对象类是一个六元组 T(I,A,M,S,ST,F),其中:①I是一个命名系统,为每一个任务实例分配一个全局唯一的标志名字;②A和M同定义1;③S为该类任务状态序列在时间上的有限半序集合(任务状态轨迹集合);④ST为S的终止集;⑤F为S→S的映射集合(此处为转换规则集合),且有:1) ∀ x∈S,∀ f∈F,f(x)≥x;2)∀ x∈ST,∀ f∈F,f(x)=x。

二、任务分解知识库和学习代理

在用户给定了需求目标后,流程处理的进一步关键处理就是根据目标设定任务实例并进行任务分解。

BBIWFS支持两种方式的任务分解:人工分解和基于任务分解知识库的分解。前者采用交互式模块支持人类管理专家对所求目标任务的分解,在生成任务实例的同时将分解所得到的流程知识分别组织成任务类描述模板和分布逻辑模板存入任务分解知识库以备将来使用;基于任务分解知识库的分解则由系统根据需求目标寻找存于任务分解知识库中且完成类似的目标的任务类描述模板以进行相应的任务流程分解并生成有关的任务流程实例集。

任务分解知识库主要由任务类描述模板表和任务分布逻辑模板表两个模板表组成。

1. 任务描述模板表

```
CREATE TABLE dba. Taskdescrible(
    tasktype char(10)NOT NULL,//任务类别
    infotype char(5),//信息相关标识
    taskname char(10), //任务名
    subtasktype char(2),//子任务类别
    subtaskno int, //子任务执行序号
    span int, //任务存在周期
    taskinfo char(10)//关联信息表名
```

此表主要存放任务的组成与属性描述信息。其中,子任务执行序号相同的子任务为可并行执行的子任务,关联信息表为本任务要处理的实际信息,而信息相关标识主要用于无子任务的任务中,相同标识的任务的关联信息在其中之一进行处理时,将同时显示以视为同一处理的不同阶段。

2. 任务分布逻辑模板表结构

```
CREATE TABLE dba. Taskmodel(
    modeltype char(5)NOT NUULL, //模板类别
    tasktype char(10)NOT NULL, //任务类别号
    op.dept char(12)NOT NULL, //操作部门
    operator char(8) //操作人员
```

此表主要为无子任务的任务而设。这样在流程执行时,除非系统管理员干涉或本表中未设置,子任务的分发目的地都由调度程序参考本表内容决定。

此外,在任务执行过程中,BBIWFS还有一个任务分解知识学习代理运行于服务器的后台。学习代理采用结构类比匹配学习方法监控任务流程的执行。

一个任务对象A可由若干个子任务对象 $a_1,a_2,\cdots,a_n$ 及其子对象关系R组成,不妨记为:$A=(\{a_1,\cdots,a_n\},R)$;则若已知任务分解知识库中的对象B结构 $(\{b_1,\cdots,b_n\},R_b)$ 和靶对象T的结构 $(\{t_1,\cdots,t_n\},R_t)$,将T的关系及子对象结构与B中的关系及子对象结构按下面公式计算结构类比拟合度:

$$S(a,b)=\alpha|P(a)\cup P(b)|-\beta|P(a)-P(b)|-\gamma|P(b)-P(a)|$$

(其中,P(a)为属性集,"−"表示差集,α、β和γ为大

于0的常数)。若计算出的类比拟合度小于规定的阈值,则表明B和T很相似,属同一类别,否则T是与B不同的类别对象。

在实际执行中,学习代理在每步流程执行完成后,对当前任务对象和有关任务模板对象进行类比匹配计算,若计算出对所有有关任务模板对象匹配的拟合度都大于规定的阈值,则确定有新模板生成,此时提醒系统管理控制员确认后将新模板加入任务分解知识库。

三、系统实现

我们在Windows NT网络环境中综合使用Sybase,Visual C++和PowerBuilder等工具对BBIWFS进行了实现,如图3所示,其中除了将任务分解知识库作为共享信息存放于服务器上外,而将已经创建且未完成的有关每一类任务实体组成的树表放入一个存放于服务器上任务实体表的共享信息表中,以利流程的控制执行。已创建的任务实体表结构如下:

```
CREATE TABLE dba.Taskobject(
    tasktype char(10)NOT NULL,//任务类别
    infotype char(5),//信息相关标识
    pretasktag char(10), //父任务实体号
    tasktag char(10),//任务实体号
    op.dept char(12),//操作部门
    operator char(8),//操作人员
    subasktag char(10),//子任务实体号
    subtaskno int, //子任务执行序号
    span int ,//任务存在周期
    taskinfo char(10),//关联信息表名
    taskstate char(2),//任务执行状态
```

其中,任务执行状态的各位取值含义如下:

低位:0-未执行,1-执行,2-超时执行,3-完成。

高位:0-可执行,1-挂起。

此外,图3中,运行于服务器上的工作流执行引擎和学习代理等进程利用在对任务分解知识库中任务描述模板表和任务分布逻辑模板表中的内容分析权衡的基础上,负责调度与监控系统中各任务的执行与分发;公共信息板则充当黑板的角色:实时反映任务执行的情况和状态,以供调度程序分发任务激活运行于客户端的知识源;运行于客户端的用户进程为黑板结构的知识源,其在提供用户界面接口和全程数据映射接口的同时,还使用运行于客户端的后台监控程序响应系统任务状态变化、显示私有信息并向公共信息板发送信息;当所有代表知识源的用户进程都完成了其承担的某一任务的子任务时,该任务便被成功地完成。

最后,位于服务器上的工作流引擎在初次启动时,同时还激活一个名为活动任务监控程序的后台进程,该进程的作用为每隔一定时间扫描状态为"E"的各任务,并对其执行的时间进行累加,若该时间大于规定的生存时间跨度,则向系统管理控制员报警请求处理;而运行于每个客户端的后台监控程序亦每隔一定时间对自己负责的任务执行的时间进行检测,若该时间大于规定的生存时间跨度,则向用户私有信息板显示报警。

目前,BBIWFS已成功地运用于浙江"好来西"集团公司的生产流程控制中,并获得了有关使用人员的好评。

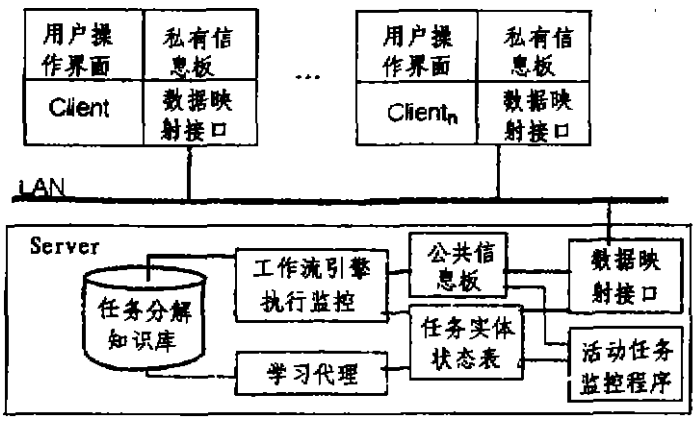

图3 BBIWFS的实现结构

参考文献

[1]M. Gaspari et al.,An open Framework for Cooperative Problem Sloving,IEEE Expert,June,1995

[2]W. B. Croft and L. Lefkowitz,Task Support in an Office System,Trans. Office Infor. Systems,2(3),1984

[3]K. Swanson,Visual Support for Reengineering Work Processes,Conf. Organizational Computing Systems,ACM Press,1993

[4]D. E. Mahling et al.,From Office Automation to Intelligent Workflow Systems,Same to [1]

[5]D. N. Chorafas and H. Steinmann, Object-Oriented Databases,Prentice-Hall,1994

[6]G. Pomberger and G. Blaschek,Object-Orientation and Prototyping in Software Engineering, Prenticd-Hall, 1995

[7]E. A. Edmonds et al., Support for Collaborative Design: Agents and Emergence,CACM,37(7),1994

[8]M. R. Genesereth and S. P. Ketchpel, Software Agents,Same to [7]

[9]R. V. Guha and D. B. Lenat ,Enabling Agents to Work Together,Samt to [7]

[10]Lotus Notes Realease 4.5:A Developer's HandBooks,Lotus Corp.

[11]冯玉琳、黄涛、李京,面向对象的构造,软件学报,7(3) 1996

[12]冯玉琳、李京、黄涛,对象语义理论和行为约束推理,计算机学报,16(11) 1993

[13]田盛丰,人工智能原理与应用:专家系统,机器学习,面向对象的方法,北京理工大学出版社 1993