

产品开发过程中的数据流建模与分析

李伟刚¹⁾ 王文斌²⁾ 沈钧毅²⁾

¹⁾(西北工业大学软件与微电子学院 西安 710072)

²⁾(西安交通大学电子与信息工程学院 西安 710049)

摘 要 通过分析产品开发过程管理中各种数据的特征,指出数据流建模和分析在面向产品开发过程的工作流管理系统中格外重要.基于提出的工作流模型,开发了一种面向版本控制的数据流模型,它支持包含任意环的过程结构.引入变量影响列表描述这种数据流模型并给出该模型正确性分析的方法.能够适应面向产品开发过程的工作流管理系统中数据管理的需求.

关键词 产品开发过程;工作流;数据流模型;分析;检验

中图法分类号 TP391

Data Flow Modeling and Verification in Product Development Process

LI Wei-Gang¹⁾ WANG Wen-Bin²⁾ SHEN Jun-Yi²⁾

¹⁾(College of Software and Microelectronics, Northwestern Polytechnical University, Xi'an 710072)

²⁾(Department of Computer Science and Technology, Xi'an Jiaotong University, Xi'an 710049)

Abstract This paper discusses perspectives of data in product development process management, such as data type, data storage mechanism, and data structure. It is shown that data flow modeling and verification is especially important in workflow management system for product development process. Based on the proposed workflow model with arbitrary cycles, this paper proposes a version control based data flow model which meets the need of product development. To describe the data flow intuitively and efficiently, the authors introduce variable influence table, and develop an algorism for verifying this data flow model. Earlier practices indicate that the approach meets the data management requirements in workflow management system for product development process.

Keywords product development process; workflow; data flow model; analysis; verification

1 引 言

航空发动机等复杂机电产品的开发需要多学科领域的人员参与,开发过程跨越时间和空间的限制,应引入过程管理信息系统对开发人员、活动、信息等进行有效的协调管理.这主要在两个级别上进行:活动级和对象级.活动级协调旨在对产品开发过程中

各个任务步骤进行调度和安排,使得群组工作有序地进行;而对象级协调主要关心数据对象的生成、共享和并发控制.当前,工作流技术被广泛地用于产品开发过程管理^[1-2],适宜的工作流建模方法和正确的工作流模型是实现产品开发过程协调和管理的基础和关键.

工作流模型应能从不同的角度对产品开发过程给予描述^[3-4],包括功能方面、行为方面、信息方面、

操作方面和组织方面。一般认为,定义过程中活动的执行控制依赖关系的行为方面是 workflow 模型中最基本和最重要的方面,是描述 workflow 其它方面的基础^[4-5],因此得到了广泛的研究^[6]。然而,定义过程中的数据和数据流的信息方面尚未得到充分的研究。产品数据管理中的 workflow 虽然研究了与产品相关的文档、数据的发布及其状态控制过程^[2],但是,一般并不涉及 workflow 引擎为执行过程本身所需的数据结构及其建模方法。

另一方面,workflow 模型在被执行以前必须保证其正确性,因为纠正由于不合理的工作流模型而引起的运行时错误会付出昂贵的代价^[7]。与 workflow 模型的研究状况相似,对 workflow 模型正确性验证方面的研究,也主要集中在控制流(或称过程结构)方面^[4-5,8],而对数据流正确性验证的研究十分有限^[9-10]。而且,这些研究成果大多面向通用 workflow 管理系统,没有考虑产品开发过程的特殊需求。本文从产品开发过程管理的实际需求出发,分析这种特殊的过程中数据和数据流的特征,在提出的面向产品开发过程的 workflow 模型的基础上,研究了一种面向版本控制的数据流模型,并给出其形式化定义和实现方法。最后,讨论了数据流模型正确性验证的基本思路和算法。

2 产品开发过程管理中的数据

2.1 数据类型

对产品开发过程进行 workflow 管理时,需处理各种类型的数据。伴随着 workflow 的执行,这些数据在任务间传递,workflow 管理系统(Workflow Management System, WfMS)的重要功能之一是协调这种数据流动的过程。一般来说,workflow 中的数据无外乎属于如下 3 种类型^[11]:

(1) workflow 控制数据。它是 WfMS 和/或 workflow 引擎管理的内部数据。

(2) workflow 相关数据。它是被 WfMS 使用以决定过程实例状态变更的数据。

(3) workflow 应用数据。它是 workflow 应用所特有的数据,不能被 WfMS 操纵。

workflow 控制数据一般用于描述过程实例和活动实例的状态,以及 WfMS 中其它的内部属性信息。这种数据不能被外部应用系统直接使用,但可以通过 WfMS 提供的功能函数访问被授权的有限信息内容。workflow 控制数据可分成静态数据和动态数据,

静态数据构成 workflow 实例化所必需的上下文,一般在 workflow 创建时已经确定,在运行时不会变化,如具有初始值的 workflow 变量、workflow 模型的作者等。动态数据在 workflow 实例的运行过程中生成,如过程实例和活动实例的运行时属性:开始时间、结束时间、当前状态、启动人等,workflow 的运行时属性:实例数、并发执行的活动数等。workflow 控制数据主要用于审计和失败恢复。本文允许 workflow 控制数据作为系统变量使用,参与控制依赖(详见第 3 节)条件表达式的构造,增强了系统的动态适应性和柔性。

workflow 相关数据在操作 workflow 应用时使用,此时需要定义应用参数与这些数据的映射关系,通过传值或引用的方式传递给 workflow 应用或从 workflow 应用返回。workflow 相关数据也是 workflow 客户端与引擎交互的产物,同时它还用于构造控制依赖条件表达式。这种数据的生成和消费,指引着 workflow 引擎更改过程实例和活动实例的状态,最终使得 workflow 按特定的路线自动执行。因此,从较高的层次来看,这就像数据在活动间的流动,称为数据流,数据流模型是对 workflow 相关数据在 workflow 活动中生成和消费关系的形式化描述,应保证数据流模型语义正确,不存在数据冗余、缺失、不匹配、不一致等错误^[9]。

workflow 应用数据由 workflow 应用系统(如 CAD、PDM、Word 等系统)控制和管理,不能被 WfMS 读写。但是,在某些情况下,它也可以作为 workflow 相关数据使用。此时,它一般是非结构化数据,在 workflow 运行时传递给特定的活动,WfMS 只控制它的流向和访问机制。所以,workflow 应用数据也可以是数据流的一部分,workflow 模型中应对 workflow 应用数据和相关数据的类型和使用、变化情况详细定义。我们将在第 4 节详细讨论数据流建模。

2.2 数据存储机制

面向产品开发过程的 WfMS 中的数据,按照类型不同分别存储:

(1) workflow 控制数据,存储于 workflow 模型库和运行实例库。

(2) workflow 相关数据,存储于变量(详见第 3 节)数据库,该数据库保存变量值和变量读写历史。

(3) workflow 应用数据,由 workflow 应用系统独立管理,存储机制因具体应用而定,与 WfMS 无关。

值得注意的是,对于那些需要转换成 workflow 相关数据使用的 workflow 应用数据,在 WfMS 中应建立与其对应的变量,通过传值或传址的方式将 workflow 应用数据赋予该变量,并存储到变量数据库中。

2.3 数据结构

面向产品开发过程的 WfMS 中的数据可以是结构化的,半结构化的,或非结构化的. 结构化数据指能够存储在关系数据库中的数据,如过程实例创建时间(datetime 类型)、零件号(string 类型)、库存

量(long 类型)等. 半结构化数据指 XML 格式数据,例如,将订单描述成 XML. 非结构化数据一般指二进制文件,WfMS 不关心非结构化数据的内容,只是通过传值或传址控制其流转. 表 1 总结了面向产品开发过程的工作流管理系统中数据的特征.

表 1 面向产品开发过程的工作流管理系统中的数据

数据类型		数据来源	数据结构	存储地点	举例
工作流控制数据	静态数据	WfMS 内部. 一般来自工作流模型(build-time)	结构化或半结构化	工作流模型库和运行实例库	过程模型作者、创建时间等
	动态数据	WfMS 内部. 一般来自工作流引擎(run-time)			过程/活动实例启动时间、实例数、当前状态等
工作流相关数据	操作数据	WfMS 内部或外部. 参与执行人工活动或调用工作流应用的数据,或由这些数据重新加工而来	结构化、半结构化或非结构化	变量数据库	产品名称、零件号、库存量、重量、BOM 表等
	决策数据	是操作数据的子集,用于构造过程实例路由由控制条件			
工作流应用数据	需转换成工作流相关数据的数据	WfMS 外部. 由工作流应用生成,但是被 WfMS 用于确定过程或活动的状态变更,或传递给其他工作流应用使用	非结构化	因具体工作流应用而定	CAD 模型、Word 文档、数控程序代码文件等
	不需转换成工作流相关数据的数据	WfMS 外部. 完全由工作流应用生成和管理, WfMS 无法看到它	结构化、半结构化或非结构化		产品开发应用系统(CAD、CAM、PDM、ERP 等)内部的数据

由以上论述可知,在工作流创建时(build-time),对数据的描述应把重点放在工作流相关数据方面,特别是建立数据流模型. 由于数据流建模与工作流模型的功能方面、行为方面和操作方面都有密切关系,所以在具体研究数据流模型之前,有必要对工作流模型整体加以简单讨论.

3 面向产品开发过程的工作流模型

基于活动(任务)的工作流建模方法^[12]在 WfMS 中得到广泛应用^[13-14],它具有简单、直观,符合人们使用习惯等优点. 在描述产品开发过程时,我们提出的工作流模型属于这种类型.

3.1 工作流模型的形式化定义

定义 1(工作流). 工作流 W 定义为三元组 (A, Q, L) , 其中 A 是各种活动组成的集合, Q 是起止标记的集合, L 是活动间的控制依赖关系, 称为链接. W 满足以下关系:

- (1) $A \cap Q = \emptyset, A \neq \emptyset$;
- (2) $Q = \{s\} \cup E$, 其中 s 为起始标记, E 为结束标记集合, $E \neq \emptyset$;
- (3) $L \subseteq A \times A \cup \{s\} \times A \cup A \times E$;
- (4) 设 $dom(L) = \{a | \exists b; (a, b) \in L\}$, $ran(L) = \{b | \exists a; (a, b) \in L\}$, 则 $dom(L) \cup ran(L) = A \cup Q$.

由定义 1 可以看出,一个工作流包含一个起始标记,一个或多个结束标记和活动,它们由链接相互关联,并且不存在孤立的活动和起止标记(由关系(4)

保证). 另外,还有一些属性需定义,如名称、标识号、优先级、作者等,这属于工作流控制数据中的静态数据.

定义 2(活动). 活动 $a_i \in A$ 可以定义为一个七元组 (I, O, Ag, R, BR, Ac, S) . 其中 I 是该活动的输入数据集合,即该活动使用的变量集合; O 是该活动输出的数据集合,即该活动生成的变量和/或更新的变量的集合; Ag 是执行该活动的代理组成的集合; R 是该活动执行时使用资源的集合; BR 为商业规则集合; Ac 为动作集合; S 为活动具有的状态集合.

活动的输入和输出用变量来表示,活动对变量的读写关系体现了工作流模型中的数据流(见第 4 节).

定义 3(变量). 变量 $v \in V$ 是工作流中所有活动的执行代理或资源操纵或生成的工作流相关数据的一个逻辑指代. 变量可以取得不同的值,变量值 $vl \in VL$. 变量与任务的输入、输出集合的关系是

$$\bigcup_{i=1}^n (I(a_i) \cup O(a_i)) \subseteq V,$$

其中 $n \in \mathbb{N}$ 为过程中的活动数, $I(a_i)$ 和 $O(a_i)$ 分别表示活动 a_i 的输入和输出变量集合.

定义 4(变量类型). 变量类型 vT 是一个受到限制的“类”的表达式,它表示了变量的一个聚集. 也就是说一个变量类型是具有相同特性的变量组成的集合的名称, $vT \in \{\text{Boolean}, \text{String}, \text{Integer}, \text{Double}, \text{Datetime}, \text{Array}, \text{Enumeration}, \text{File}, \text{XML}\}$. 一个变量的类型声明表示为 $v: vT$, 如果把每个变量类

种表示方法,数据流的形式化定义如下.

定义 8(数据链接/数据流). 数据链接 dl 表示 workflow 中活动与变量的读写关系, dl 可表示成以下 3 种形式之一:

(1) $dl=(a, v(a_i), \mathbf{R})$, 表示执行活动 a 时需读取变量 v 在活动 a_i 处生成的版本;

(2) $dl=(a, v, \mathbf{R})$, 表示执行活动 a 时需读取变量 v 当前的最新版本;

(3) $dl=(a, v, \mathbf{W})$, 表示执行活动 a 后变量 v 有新版本生成.

\mathbf{R} 和 \mathbf{W} 分别为读操作和写操作算子. workflow W 的数据流定义为 $D_W := \bigcup_{j=1}^m dl_j$, 其中 $m \in \mathbb{N}$, 为 W 中数据链接的数目.

产品开发过程会产生大量的结构化、半结构化和非结构化数据,而且对数据特别是那些作为 workflow 相关数据使用的应用数据应保持良好的可追溯性,因此以上数据流的定义是基于版本控制的. WfMS 区分变量在不同活动执行后生成的历史版本,使其能参与到对象级协调和管理中. 由 CAD、CAE、Word 等应用系统生成的文件,若作为 workflow 相关数据使用,其版本控制方案有 3 种:(1)通过传值给变量交由 WfMS 处理;(2)将文档管理系统与 WfMS 集成,由文档管理系统管理版本,不同版本的文件传址给变量;(3)应用系统天生支持版本控制(例如 MS Word),WfMS 通过变量指明版本. 本文提出的数据流描述机制支持这 3 种方案.

注意到,活动执行时对变量的读写操作是通过算子 \mathbf{R} 和 \mathbf{W} 进行的,如果变量属于复杂数据类型,如 XML 类型或对象类型,若需要读取或更改变量的一部分,则通过特定算法完成. 例如,对于 XML 类型变量可以通过 XQuery 或 XPath 语言读写其局部内容;对于 Excel 文件,可以通过标准 API 进行读写.

4.2 变量影响列表

本文规定变量的作用域为 workflow 实例级,即在工作流模型中定义变量,而每个过程实例在运行时都拥有自己的一份变量值拷贝,同一过程实例中的活动可以共用它们,不同过程实例不能交叉使用各自的变量. 变量由 WfMS 维护,存储在变量数据库中. 为了在 WfMS 中描述数据流,我们引入变量影响列表 (Variable Influence Table, VIT).

定义 9(变量影响列表). 变量影响列表 T_v 是一个二维表格,它的列对应不同的变量,行对应不同的活动. 每个活动占据两行,分别表示读取和写入.

T_v 中位于活动 a_i 所在的读取行和变量 v_j 所在的列的交叉点处的值用 $t_v(a_i^r, v_j)$ 表示,同理,相应的写入行上对应的值用 $t_v(a_i^w, v_j)$ 表示. 则 $t_v(a_i^r, v_j) \in \{\text{null}, \text{"initial"}, \text{"uptodate"}\} \cup A_{id}$, $t_v(a_i^w, v_j) \in \{\text{null}, \text{"w"}\}$, 其中, A_{id} 是 W 中所有活动的标识符构成的集合.

各种表格值的含义如下:

(1) $t_v(a_i^r, v_j) = \text{null}$, 表示活动 a_i 不对变量 v_j 进行读取操作;

(2) $t_v(a_i^w, v_j) = \text{null}$, 表示活动 a_i 不对变量 v_j 进行写入操作;

(3) $t_v(a_i^r, v_j) = \text{"initial"}$, 表示活动 a_i 读取变量 v_j 的初始值;

(4) $t_v(a_i^r, v_j) = \text{"uptodate"}$, 表示活动 a_i 读取变量 v_j 当前的最新版;

(5) $t_v(a_i^r, v_j) = a_k.id$, 表示活动 a_i 读取活动 a_k 写入变量 v_j 的值;

(6) $t_v(a_i^w, v_j) = \text{"w"}$, 表示活动 a_i 对变量 v_j 进行写入操作.

VIT 清楚地描述了数据在不同的活动间传递、流动和访问的关系. 由于同一变量的值可能随其“经过”的活动发生改变,即产生不同的版本,而 VIT 记录了这种版本的变更,所以能保证数据访问的正确性.

变量的读写具有先后次序,即设 $t_v(a_i^r, v_j) = a_k.id$, 则必须保证在工作流实例中活动 a_k 先于 a_i 执行完毕. 然而,在环结构中某些活动的初次执行与重复执行需要读取或写入的变量是不一致的. 例如图 2 所示的例子中,活动 a_1 初次执行时,设计人员使用 CAD 软件进行零件结构设计,并将设计结果存入变量 $prtFile$: File 引用的文件中,设计文件若经过 a_2 审核不合格 ($check$: Boolean = false),则产生说明文件 $checkFile$: File, 进入下一轮迭代,此时 a_1 的输入变量变成 $prtFile$ 和 $checkFile$. a_1 修改设计结果后,将修改说明存入 $modifyFile$: File 引用的文件中,使得 a_1 写入的变量变成 $prtFile$ 和 $modifyFile$. 相应地, a_2 读取的变量也随之变成 $prtFile$ 和 $modifyFile$.

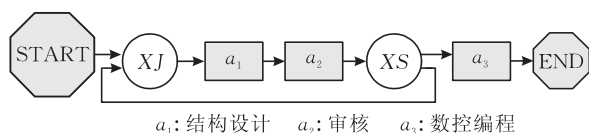


图 2 环结构对 VIT 的影响

为了保持 VIT 对数据传递的指导作用,本文规

定环结构中活动对变量的读写按环结构重复执行时的情形处理,并总是读取变量的最新版本. 这样可以用表 2 表示图 2 所示过程的数据流,其中 $ncFile$: File 是数控程序清单文件, $check$: Boolean 是代表审核结果的变量.

表 2 图 2 示例工作流的 VIT

	a_1		a_2		a_3	
	r	w	r	w	r	w
$priFile$	a_1	w	a_1		a_1	
$modifyFile$		w	a_1			
$checkFile$	a_2			w		
$ncFile$						w
$check$				w		

活动 a_1 读取的变量是 $priFile$ 和 $checkFile$, 它们在 a_1 初次执行时实际值并未产生. 所以在运行时, a_1 第一次执行应避免读取它们, 这通过如下的算法处理.

算法 1. 工作流运行时环结构中数据流的执行算法.

输入: 工作流 W 及其 VIT

输出: 无

1. $\forall a_i \in W$
2. IF $a_i \in I(cn_p, cn_q)$
3. IF a_i 是初次执行
4. FOR $t_v(a_i^r, v_j), j=1, 2, \dots, m, m$ 为变量数
5. IF $t_v(a_i^r, v_j) = a_k.id$
6. IF $a_i = a_k \parallel a_i \in \{cn_p \rightarrow cn_q\} \wedge a_k \in I(cn_p, cn_q) \parallel a_k \in Successor(a_i) \parallel a_i \in \{cn_q \rightarrow cn_p\} \wedge a_k \in \{cn_q \rightarrow cn_p\} \parallel a_k \in Successor(a_i)$
7. 不读取 v_j ;
8. ELSE
9. 读取 v_j ;
10. END IF
11. END IF
12. NEXT
13. END IF
14. END IF

对于环结构中活动初次执行不需写入, 而重复执行需写入的变量, 只要在活动初次执行时将该变量重复置初值一次即可, 不会影响活动的实际运行, 如前面例子中的 $modifyFile$.

4.3 变量影响列表的存储

由表 2 可知, VIT 可以看作一个稀疏矩阵, 当活动数和变量数较多时, 其存储结构并不合理, 因此本文采用如下方法存储它:

对活动 $a \in W$, 定义 String 型二维数组 $r[p][2]$, 其中 p 为 a 读取的变量数, $r[j][0] = version(v_j)$, $r[j][1] = name(v_j)$, $j = 0, 1, \dots, p-1$, 其中 $version(v_j) \in \{\text{"initial"}, \text{"uptodate"}\} \cup A_{id}$, $name(v_j)$ 是变量 v_j 的名称; 定义 String 型一维数组 $w[q]$, 其中 q 为 a 写入的变量数, $w[k] = name(v_k)$, $k = 0, 1, \dots, q-1$.

这样, 就可以通过活动的数组 r 和 w 获得它的数据访问情况, 指导数据在活动间传递.

5 数据流模型正确性分析

数据流模型正确性分析是工作流分析领域的重要研究课题之一, 其目的是在创建时找出数据在工作流执行中可能发生的缺失、类型不匹配、不一致等错误. 由于本文采用面向版本控制的数据流模型, 并且变量作用域为工作流实例级, 通过由 WfMS 管理的中心工作流变量数据库存储, 所以数据的类型不匹配和不一致问题不会存在, 关键是保证数据流不存在数据缺失, 避免工作流执行发生死锁.

数据流模型正确性分析的思路是: 每个活动需读取的变量的版本与写入该版本变量的活动在过程实例中的执行顺序一致. 使用紧凑格式存储 VIT 时, 按上述思路进行数据流分析, 只需检查每一个活动 a_i 中定义的读变量数组 r_i 的值 $r_i[k][0]$ 所指向的活动 a_j 的写变量数组 w_j 的值 $w_j[l]$ 与 $r_i[k][1]$ 的一致性, 以及活动 a_j 在任一过程实例中的执行是否先于 a_i , 其中 $k=0, 1, \dots, p-1$, p 为 a_i 读取的变量数, $l=0, 1, \dots, q-1$, q 为 a_j 写入的变量数. 注意到环结构中活动的执行先后次序可能在不同的循环中交替变化, 而我们在 3.2 节针对环结构已经对 VIT 进行了特殊处理, 所以 a_i 和 a_j 属于同一个环结构时在逻辑上不会产生数据读写错误问题, 不需考虑, 只需考虑 a_i 和 a_j 分属不同环结构的情形. 另外, 当 $r_i[k][0]$ 等于“initial”或者“uptodate”时, 显然不会发生数据读写错误, 所以也无需考虑.

算法 2. 数据流正确性检验算法.

输入: 工作流 W 和变量读写数组 r, w

输出: 错误提示信息

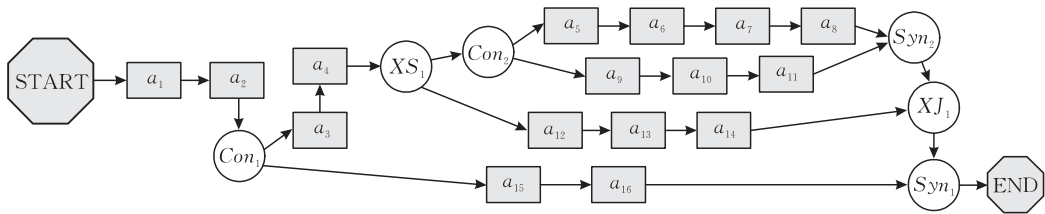
1. FOR $a_i \in W, i=1, 2, \dots, n, n$ 为 W 中的活动数
2. FOR a_i 读取的每个变量 $r_i[k][1], k=0, 1, \dots, p-1$
3. 查找 $r_i[k][0]$ 所指向的活动 a_j ;
4. $m := 0$;
5. FOR a_j 的写变量数组 w_j 中的每个值 $w_j[l], l=0, 1, \dots, q-1$

6. IF $r_i[k][1]=w_j[l]$
7. $m++$;
8. END IF
9. NEXT
10. IF $m>0$
11. IF $a_i \in I_1$
12. IF $a_j \notin I_1 \wedge a_j \notin I_2 \wedge \dots \wedge a_j \notin I_t$, 其中 I_1, I_2, \dots, I_t 是 W 中相互联结的 t 个迭代结构
13. IF a_j 不属于以 I_1 的某个入口点为终点的路径
14. 发出错误消息“变量读写顺序错误: a_i 读取了 a_j 尚未写入的变量 $r_i[k][1]$ ”;
15. END IF
16. END IF
17. ELSE
18. IF $a_j \in Successor(a_i) \parallel a_i$ 和 a_j 属于同一异或结构的不同分支
19. 发出错误消息“变量读写顺序错误: a_i 读取了 a_j 尚未写入的变量 $r_i[k][1]$ ”;
20. END IF
21. IF $a_j \in Predecessor(a_i) \wedge a_j$ 属于某个异或结构的分支而 a_i 不属于此异或结构
22. 发出错误信息“变量版本可能未定义: 如果 a_j 所属路径不执行, 则 a_i 读取的变量版本不存在”;
23. END IF
24. END IF
25. ELSE
26. 发出错误消息“变量版本未定义: a_j 未定义 a_i 读取的变量 $r_i[k][1]$ ”;

27. END IF
28. NEXT
29. NEXT

以上算法第 14 和 19 行找出的错误指出在 W 的任一实例中,某活动读取的变量版本没有在该活动完成之前生成;第 22 行找出的错误指出,由于异或结构的单选效应,某活动读取的变量版本可能不存在;第 26 行找出的错误指出某活动读取的变量版本根本不会生成,即该版本未定义. 3 种错误都会造成数据流的死锁,从而使工作流执行中断.

下面我们通过一个例子说明上述算法的执行情况. 图 3 是某型零件叶片精锻模设计过程的过程模型,其中活动 a_3 根据叶片锻件的几何特性和物理特性评估是否需要两次热态锻造,若是则应设计预锻模和终锻模,否则只需设计一套热态模具. 锻件评估的结果写入变量 $evaluation$: Boolean. 工作流运行时,活动 a_4 根据 $evaluation$ 变量的值,确定预锻余量和预锻件收缩率($evaluation=true$)或终锻余量和终锻件收缩率($evaluation=false$). 异或控制点 XS_1 在链接 $a_4 \rightarrow a_5$ 和 $a_4 \rightarrow a_9$ 上的分量条件表达式为“ $evaluation$ ”,在链接 $a_4 \rightarrow a_{12}$ 上的分量条件表达式为“ $\neg evaluation$ ”. 活动 a_5 确定终锻余量和终锻件收缩率, a_7 、 a_{13} 设计终锻模分模面, a_{10} 、 a_{15} 分别设计预锻模和校正模分模面. 该锻模设计过程的数据流用各活动中的两个数组 r 、 w 描述,如表 3 所示.



a_1 : 叶身实体造型; a_2 : 叶片锻件造型; a_3 : 锻件评价; a_4 、 a_5 : 确定加工余量和锻件收缩率; a_6 、 a_{12} : 终锻件造型; a_7 、 a_{10} 、 a_{13} 、 a_{15} 确定分模面; a_8 、 a_{14} : 设计终锻模; a_9 : 预锻件造型; a_{11} : 设计预锻模; a_{16} : 设计校正模

图 3 锻模设计过程模型

可见,对产品开发过程而言,数据流模型十分复杂,靠人工很难找出数据缺失等错误,但是通过算法 2 可以迅速对数据流模型进行检验. 如上例中, $\forall a_{12}$ 读取的变量 $r_{12}[1][1]=finalEnlarge$, $\exists r_{12}[1][0]=“a4”$, 但 a_4 的 w 数组 $w_4[0]=enlarge$, 即 $r_{12}[1][1] \notin w_4$, 所以算法 2 抛出错误消息“变量版本未定义: a_4

未定义 a_{12} 读取的变量 $finalEnlarge$ ”. $\forall a_{14}$ 读取的变量 $r_{14}[0][1]=finalForge$, $\exists r_{14}[0][0]=“a6”$, 且 a_6 的 w 数组 $w_6[0]=r_{14}[0][1]=finalForge$, 但是 a_{14} 和 a_6 属于同一异或结构(XS_1, XJ_1)的不同分支, 所以算法 2 抛出错误消息“变量读写顺序错误: a_{14} 读取了 a_6 尚未写入的变量 $finalForge$ ”.

表 3 锻模设计过程的数据流及其存储方法

活动	<i>r</i> 数组	<i>w</i> 数组	变量定义
<i>a</i> ₁	<i>r</i> ₁ [0][0]="initial", <i>r</i> ₁ [0][1]= <i>dataFile</i>	<i>w</i> ₁ [0]= <i>body</i>	<i>dataFile</i> : File 叶身数据文件
<i>a</i> ₂	<i>r</i> ₂ [0][0]=" <i>a</i> 1", <i>r</i> ₂ [0][1]= <i>body</i> , <i>r</i> ₂ [1][0]="initial", <i>r</i> ₂ [1][1]= <i>bladeNote</i>	<i>w</i> ₂ [0]= <i>blade</i> , <i>w</i> ₂ [1]= <i>bladeDesign</i>	<i>body</i> : File 叶身实体模型 <i>bladeNote</i> : File 叶冠、榫头、工艺凸台等工艺说明文件
<i>a</i> ₃	<i>r</i> ₃ [0][0]=" <i>a</i> 2", <i>r</i> ₃ [0][1]= <i>blade</i> , <i>r</i> ₃ [1][0]=" <i>a</i> 2", <i>r</i> ₃ [1][1]= <i>bladeDesign</i>	<i>w</i> ₃ [0]= <i>evaluation</i>	<i>blade</i> : File 锻件实体模型
<i>a</i> ₄	<i>r</i> ₄ [0][0]=" <i>a</i> 3", <i>r</i> ₄ [0][1]= <i>evaluation</i> , <i>r</i> ₄ [1][0]=" <i>a</i> 2", <i>r</i> ₄ [1][1]= <i>blade</i> , <i>r</i> ₄ [2][0]=" <i>a</i> 2", <i>r</i> ₄ [2][1]= <i>bladeDesign</i>	<i>w</i> ₄ [0]= <i>enlarge</i>	<i>bladeDesign</i> : File 锻件设计说明
<i>a</i> ₅	<i>r</i> ₅ [0][0]=" <i>a</i> 2", <i>r</i> ₅ [0][1]= <i>blade</i> , <i>r</i> ₅ [1][0]=" <i>a</i> 2", <i>r</i> ₅ [1][1]= <i>bladeDesign</i> , <i>r</i> ₅ [2][0]=" <i>a</i> 4", <i>r</i> ₅ [2][1]= <i>enlarge</i>	<i>w</i> ₅ [0]= <i>finalEnlarge</i>	<i>evaluation</i> : Boolean
<i>a</i> ₆	<i>r</i> ₆ [0][0]=" <i>a</i> 2", <i>r</i> ₆ [0][1]= <i>blade</i> , <i>r</i> ₆ [1][0]=" <i>a</i> 5", <i>r</i> ₆ [1][1]= <i>finalEnlarge</i>	<i>w</i> ₆ [0]= <i>finalForge</i>	<i>enlarge</i> , <i>finalEnlarge</i> : File (终)锻造余量和收缩率说明文件
<i>a</i> ₇	<i>r</i> ₇ [0][0]="initial", <i>r</i> ₇ [0][1]= <i>dataFile</i> , <i>r</i> ₇ [1][0]=" <i>a</i> 6", <i>r</i> ₇ [1][1]= <i>finalForge</i>	<i>w</i> ₇ [0]= <i>finalPrtSurface</i>	<i>finalForge</i> , <i>preForge</i> : File 终锻件、预锻件实体模型
<i>a</i> ₈	<i>r</i> ₈ [0][0]=" <i>a</i> 6", <i>r</i> ₈ [0][1]= <i>finalForge</i> , <i>r</i> ₈ [1][0]=" <i>a</i> 7", <i>r</i> ₈ [1][1]= <i>finalPrtSurface</i>	<i>w</i> ₈ [0]= <i>finalDie</i> , <i>w</i> ₈ [1]= <i>finalDieDesign</i>	<i>finalPrtSurface</i> , <i>prePrtSurface</i> , <i>adjPrtSurface</i> : File 终锻模、预锻模、校正模分模面数据文件
<i>a</i> ₉	<i>r</i> ₉ [0][0]=" <i>a</i> 2", <i>r</i> ₉ [0][1]= <i>blade</i> , <i>r</i> ₉ [1][0]=" <i>a</i> 4", <i>r</i> ₉ [1][1]= <i>enlarge</i>	<i>w</i> ₉ [0]= <i>preForge</i>	<i>finalDie</i> , <i>preDie</i> , <i>adjDie</i> : File 终锻模、预锻模、校正模实体模型
<i>a</i> ₁₀	<i>r</i> ₁₀ [0][0]="initial", <i>r</i> ₁₀ [0][1]= <i>dataFile</i> , <i>r</i> ₁₀ [1][0]=" <i>a</i> 9", <i>r</i> ₁₀ [1][1]= <i>preForge</i>	<i>w</i> ₁₀ [0]= <i>prePrtSurface</i>	<i>finalDieDesign</i> , <i>preDieDesign</i> , <i>adjDieDesign</i> : File 终锻模、预锻模、校正模设计说明
<i>a</i> ₁₁	<i>r</i> ₁₁ [0][0]=" <i>a</i> 9", <i>r</i> ₁₁ [0][1]= <i>preForge</i> , <i>r</i> ₁₁ [1][0]=" <i>a</i> 10", <i>r</i> ₁₁ [1][1]= <i>prePrtSurface</i>	<i>w</i> ₁₁ [0]= <i>preDie</i> , <i>w</i> ₁₁ [1]= <i>preDieDesign</i>	
<i>a</i> ₁₂	<i>r</i> ₁₂ [0][0]=" <i>a</i> 2", <i>r</i> ₁₂ [0][1]= <i>blade</i> , <i>r</i> ₁₂ [1][0]=" <i>a</i> 4", <i>r</i> ₁₂ [1][1]= <i>finalEnlarge</i>	<i>w</i> ₁₂ [0]= <i>finalForge</i>	
<i>a</i> ₁₃	<i>r</i> ₁₃ [0][0]="initial", <i>r</i> ₁₃ [0][1]= <i>dataFile</i> , <i>r</i> ₁₃ [1][0]=" <i>a</i> 12", <i>r</i> ₁₃ [1][1]= <i>finalForge</i>	<i>w</i> ₁₃ [0]= <i>finalPrtSurface</i>	
<i>a</i> ₁₄	<i>r</i> ₁₄ [0][0]=" <i>a</i> 6", <i>r</i> ₁₄ [0][1]= <i>finalForge</i> , <i>r</i> ₁₄ [1][0]=" <i>a</i> 13", <i>r</i> ₁₄ [1][1]= <i>finalPrtSurface</i>	<i>w</i> ₁₄ [0]= <i>finalDie</i> , <i>w</i> ₁₄ [1]= <i>finalDieDesign</i>	
<i>a</i> ₁₅	<i>r</i> ₁₅ [0][0]="initial", <i>r</i> ₁₅ [0][1]= <i>dataFile</i> , <i>r</i> ₁₅ [1][0]=" <i>a</i> 2", <i>r</i> ₁₅ [1][1]= <i>blade</i>	<i>w</i> ₁₅ [0]= <i>adjPrtSurface</i>	
<i>a</i> ₁₆	<i>r</i> ₁₆ [0][0]=" <i>a</i> 2", <i>r</i> ₁₆ [0][1]= <i>blade</i> , <i>r</i> ₁₆ [1][0]=" <i>a</i> 15", <i>r</i> ₁₆ [1][1]= <i>adjPrtSurface</i>	<i>w</i> ₁₆ [0]= <i>adjDie</i> , <i>w</i> ₁₆ [1]= <i>adjDieDesign</i>	

6 相关工作

工作流技术领域中 对数据(流)建模方面的研究不像控制流方面的研究那样广泛和深入,只有很少的学者对数据流模型正确性分析进行了有限的探索^[9-10].文献[16]对工作流系统中的数据进行了较详尽的研究,内容包括数据的表示和使用方法.该文的研究工作与具体建模方法无关,从数据可见性、数据交互、数据传递和基于数据的路由 4 个方面考察 WfMS 系统中的数据特征,归纳出 39 种数据模式,这些模式能够对创建工作流模型起到宏观指导作用.但是该文并未涉及数据结构、数据版本控制、数据流检验等方面的内容.WIDE 项目^[13]单独建立了信息模型,它由数据模型和表单模型组成.定义了工作流中使用的数据和数据上执行的操作,并控制数据的作用域和持久性.数据流使用参数传递的方式表达.总体而言,WIDE 中的数据流表示能力较强,但是它没有提供数据版本的控制能力.IBM MQ Workflow 产品通过活动的数据容器持久地存储输入参数和活动的输出,数据容器间用数据连接器连接,形成数据流.这种方法消除了对全局变量和集中数据库的需求,允许每个活动定义它们自己的参数,适于分布式应用.但是复杂的数据迁移机制不但增

加了系统运行负担,而且容易出错,不适于不需组件级分布的中小型应用.MENTOR 项目^[17]中使用活动图表示数据流,活动图是一个有向图,节点代表活动,数据条目标注在弧上.它将控制流与数据流分离,虽然有利于清楚地表示这两种模型视图,但是增加了数据不一致的危险,文中并未对此深入研究.Sadiq 等人^[9]研究了数据流检验的一般原则,但是并未给出具体的数据流模型和检验算法.Zhou 等人^[10]基于 Petri 网理论,提出一种三维工作流网,并给出其语义验证方法,涵盖了数据流方面的内容.但是由于数据流只是体现在对 Token 和弧表达函数的扩充上,数据流与控制流紧密耦合,导致数据流模型灵活性较差.另外该文并未考虑数据存储、数据版本等问题.文献[15]提出的数据流模型与我们的方法有相似之处,也使用了过程级变量作为数据传递的媒介,但是在活动中定义了局部参数,负责与变量进行数据交换,这使得信息传递效率变低.另外,文中并未给出变量版本控制的方法,而且对数据流的检验也仅局限于处理工作流模型的动态变化.在 PDM 研究领域,工作流技术主要用于产品相关的文档、数据的发布及其状态控制^[2],这属于工作流应用数据的研究范畴,一般并不涉及工作流控制数据和相关数据等内容,从工作流系统自身角度研究数据(流)建模和模型检验问题比较少见.

7 总结和展望

产品开发过程涉及的数据多样,控制复杂,需要对这种过程的信息管理系统(如 WfMS)中数据流建模及其正确性分析进行深入研究,但是该领域的研究成果较少.本文提出的面向版本控制的数据流模型和检验分析方法旨在满足面向产品开发过程的 WfMS 中数据管理的需求,研究了以下内容:(1)对产品开发过程管理系统中的数据进行了研究,包括数据类型、存储方式和数据结构,指出数据流模型所涉及的数据及其特征;(2)形式化地定义了工作流模型,并且为了适应产品开发的需求,使其支持任意环结构;(3)建立了面向版本控制的数据流模型,并使用变量影响列表描述它,给出其存储方式;(4)给出数据流模型的正确性检验分析算法.需要指出,本文的研究工作没有考虑各种数据的应用语义对数据流模型正确性的影响,即没有从应用逻辑的角度考虑数据之间的相互关系,无法解决诸如如何保证需求说明书和设计文档在数据流中的先后关系等问题,我们将在以后的工作中对此进行研究.另外,由于本文约定变量的作用域为过程实例级,所以数据流限制于单个过程实例内,在后继研究工作中,我们将把变量作用域扩大到过程之间,允许数据流跨越不同过程边界,增强其表达能力,这将大大增加数据流检验的研究难度.

参 考 文 献

- [1] Huang G Q, Huang J, Mak K L. Agent-based workflow management in collaborative product development on the internet. *Computer-Aided Design*, 2000, 32(2): 133-144
- [2] Fan Wen-Hui, Ge Zheng-Yu, Xiong Guang-Leng et al. Study on PDM-based product development process management. *High-Tech Letters*, 2004, 10(3): 63-68(in Chinese)
(范文慧,葛正宇,熊光楞等.基于 PDM 的产品开发过程管理方法的研究与实现.高技术通讯, 2004, 10(3): 63-68)
- [3] Petkov S, Oren E, Haller A. Aspects in workflow management. National University of Ireland, Galway: Technical Report DERI-TR-2005-04-10, 2005
- [4] van der Aalst W M P, ter Hofstede A H M. Verification of workflow task structures: A Petri-net-based approach. *Information Systems*, 2000, 25(1): 43-69
- [5] Sadiq W, Orlovskaya M E. Analyzing process models using graph reduction techniques. *Information Systems*, 2000, 25(2): 117-134
- [6] van der Aalst W M P, ter Hofstede A H M, Kiepuszewski B et al. Workflow patterns. *Distributed and Parallel Databases*, 2003, 14(1): 5-51
- [7] ter Hofstede A H M, Orłowska M, Rajapakse J. Verification problems in conceptual workflow specifications. *Data and Knowledge Engineering*, 1998, 24(3): 221-238
- [8] Zhao Wen, Yuan Chong-Yi, Liu Gang et al. Workflow process model verification using reduction method based on P/T system. *Journal of Software*, 2004, 15(10): 1423-1430 (in Chinese)
(赵文,袁崇义,刘刚等.基于 P/T 系统化简方法的工作流过程模型验证.软件学报, 2004, 15(10): 1423-1430)
- [9] Sadiq S, Orłowska M E, Sadiq W et al. Data flow and validation in workflow modeling//*Proceedings of the 15th Australasian Database Conference*. Dunedin, 2004. Darlinghurst: Australian Computer Society Inc., 2004: 207-214
- [10] Zhou Jian-Tao, Shi Mei-Lin, Ye Xin-Ming. A method for semantic verification of workflow processes based on Petri net reduction technique. *Journal of Software*, 2005, 16(7): 1242-125(in Chinese)
(周建涛,史美林,叶新铭.一种基于 Petri 网化简的工作流过程语义验证方法.软件学报, 2005, 16(1): 1-9)
- [11] Hollingsworth D. The workflow reference model. *Workflow Management Coalition*, Brussels: Technical Report TC00-1003, 1995
- [12] Lei Y, Singh M P. A comparison of workflow metamodels//*Proceedings of the ER'97 Workshop on Behavioral Models and Design Transformations: Issue and Opportunities in Conceptual Modeling*. Los Angeles, 1997
- [13] Casati F, Grefen P, Pernici B et al. WIDE workflow model and architecture. Center for Telematics and Information Technology, University of Twente, Netherlands: Technical Report 96-19, 1996
- [14] WFMC. Interface 1: Process definition interchange—Process model. *Workflow Management Coalition*, Brussels: Technical Report TC-1016-P, 1999
- [15] Reichert M, Dadam P. ADEPT_{flex}: Supporting dynamic changes of workflow without losing control. *Journal of Intelligent Information Systems*, 1998, 10(2): 93-129
- [16] Russell N, ter Hofstede A H M, Edmond D et al. Workflow data patterns. *Queensland University of Technology*, Brisbane: Technical Report FIT-TR-2004-01, 2004
- [17] Muth P, Wodtke D, Weissenfels J et al. Enterprise-wide workflow management based on state and activity charts//Dogac A, Kalinichenko L, Ozsu T, Sheth A. *Workflow Management Systems and Interoperability*. Berlin: Springer-Verlag, 1998: 281-303



LI Wei-Gang, born in 1972, Ph. D. , associate professor. His main research interests include flexible workflow management and Service Oriented Computing (SOC).

WANG Wen-Bin, born in 1972, Ph. D. candidate. His main research interests include distributed computing and data mining.

SHEN Jun-Yi, born in 1939, professor, Ph. D. supervisor. His main research interests include software engineering and data mining.

Background

There is less research on data flow modeling and verification in the field of process management. Existing methods to model the data flow in workflow research field are not suitable to product development processes since they do not consider the version control of data in processes, which are very important for the product development yet. In the PDM (Product Data Management) field data flows are investigated, where data are digital documents and their states during their lifecycles. But few concerns are put on the control data and workflow relevant data in PDM. These make workflow management in PDM unsuitable to manage product development processes other than approval flow. Moreover, few researches were done in correctness problem of data flow model in either workflow or PDM field, which is a complex problem when we consider the version of data in the product development processes.

This paper proposes a novel approach to model data flow in product development processes, which covers control data, workflow relevant data and workflow application data. All these data can be controlled in version. It also proposes an algorithm for verifying this data flow model. At last, it is

shown that the results are practicable through an example.

The work in this paper is supported by the Science-Technology Project of Shaanxi Province of China and the Special Fund on Information Technology of Xi'an City under grant No. 2004k05-G45 and ZX06033 respectively. The name of this project is "Adaptable Business Process Management System".

The development processes for complicated mechatronical product are very complex. To coordinate and manage this kind of processes, workflow management system (WfMS) is often introduced. But, traditional WfMS is low in the flexibility of modeling and executing, expressive power of process model, and ability of adapting to the changes. In the project, the authors researched all this key technologies in adaptable business process management system with workflow model with high expressiveness. The data flow modeling and model verification is important issues, and the proposed version control based data flow model can integrate with the drawing document management module of the PDM system, therefore integrating process management with product data management organically.