

文章编号:1006-5911(2010)08-1578-09

基于 workflow 模式的 BPMN 过程模型验证方法

蔡章利^{1,2}, 易树平¹, 高庆萱¹

(1. 重庆大学 机械工程学院, 重庆 400044; 2. 重庆大学 自动化学院, 重庆 400044)

摘要:针对业务流程建模符号过程模型语义验证问题,提出了一种新方法。该方法基于正确的业务流程建模符号过程模型是 workflow 模式的合理组合的理念,通过扩展和改进业务流程建模符号及其执行语义,设计了 workflow 模式的形式化编码规则;借鉴 Petri 网化简方法,设计了 workflow 模式组合化简规则;基于 workflow 模式编码及组合化简规则,给出了业务流程建模符号过程模型验证方法。应用示例表明了该方法的有效性。

关键词: workflow; 业务流程; 建模; 过程模型; 语义验证; Petri 网

中图分类号: TP391

文献标志码: A

Verification method for BPMN process model based on workflow patterns

CAI Zhang-li^{1,2}, YI Shu-ping¹, GAO Qing-xuan¹

(1. College of Mechanical Engineering, Chongqing University, Chongqing 400044, China;

2. College of Automation, Chongqing University, Chongqing 400044, China)

Abstract: To semantically validate Business Process Model Notation (BPMN) process model, a new method was put forward. This method was based on the idea that correct BPMN process model was the sound combination of workflow patterns. By improving the BPMN's notations and execution semantics, the formal encoding rules were designed. And the combinatorial simplification rules were designed by borrowing Petri net's simplification methods. The algorithms to validate the BPMN process model were also presented. Finally, this method was proved to be effective in validating BPMN process model by application examples.

Key words: workflow; business process; modeling; process model; semantic verification; Petri nets

0 引言

workflow 是部分或全部可以计算机实现的业务流程。workflow 过程模型是可以计算机实现的业务流程模型。为跨越业务流程设计与实现鸿沟,便于业务分析师、IT 人员和流程管理员共同设计、实现和监管业务流程,业务流程管理计划组织 (Business Process Management Initiative, BPMI) 于 2004 年提出了业务流程建模符号 (Business Process Model Notation, BPMN), 现已成为国际对象管理组织

(Object Management Group, OMG) 支持的一种业务流程建模语言^[1]。为有别于用 Petri 网等描述的工作流过程模型,本文将用 BPMN 描述的工作流过程模型称为 BPMN 过程模型。

语义验证 workflow 过程模型是 workflow 技术领域长期研究的一个热点和难点。文献[2]基于图形归约方法研究了 workflow 过程模型验证问题,提出的方法能对有向无环图 (Directed Acyclic Graph, DAG) 存储表示的工作流过程模型进行验证。文献[3]~文献[4]通过改进文献[2]的方法,提出了基于图形展

收稿日期:2010-01-08;修订日期:2010-03-08。Received 08 Jan. 2010; accepted 08 Mar. 2010.

基金项目:国家自然科学基金资助项目(70871127);高等学校博士学科点专项科研基金资助项目(20070611027);重庆市教育委员会科学技术研究资助项目(KJ08A03)。**Foundation items:** Project supported by the National Natural Science Foundation, China (No. 70871127), the Specialized Research Fund for Doctoral Program of Higher Education, China (No. 20070611027), and the Foundation of Chongqing Municipal Education Committee, China (No. KJ08A03).

开及图形归约和基于 Petri 网化简的工作流过程模型验证方法,能对有向有环图(Directed Cyclic Graph, DCG)存储表示的工作流过程模型进行验证。文献[5]研究了如何用源于 Petri 网的另一种工作流语言(Yet Another Workflow Language, YAWL)形式化文献[6]提出的工作流模式,文献[7]介绍了如何用 YAWL 方法验证工作流过程模型。与基于有向图或 Petri 网及其扩展表示的工作流过程模型相比,用 BPMN 符号描述的业务流程更易于业务人员理解,基于 BPMN 的业务流程建模方法比其他工作流建模方法具有更强的描述能力。BPMN 特有的描述能力使得现有验证方法无法直接用于 BPMN 过程模型。对此,文献[8]讨论了如何先将 BPMN 过程模型转换成工作流网(workflow net),然后用 Petri 网方法验证 BPMN 过程模型,但没有解决有异常处理的子过程及合并控制的容他网关(Inclusive Gateway)转换,不支持多实例不同步等工作流模式。文献[9]用通信顺序进程(Communicating Sequential Processes, CSP)形式化 BPMN 过程模型,但没有说明如何识别错误类型并报告错误所在。Puhmann 在文献[10]中用 Pi -演算形式化文献[6]提出的工作流模式,并在文献[11]给出了基于 Pi -演算的形式化验证 BPMN 过程模型算法,但没有支持任意循环等工作流模式。

针对 BPMN 1.2 规范存在的过程语义不明等问题,为支持业务分析师创建 BPMN 过程模型,笔者研究了基于 BPMN 的业务流程一体化建模方法,并用语义有向图(Directed Graph with Semantics, DGS)存储 BPMN 过程模型。为满足 BPMN 过程模型语义验证需要,本文基于正确的 BPMN 过程模型是工作流模式的合理组合的理念,借鉴 Petri 网化简规则,提出了基于工作流模式编码、识别、组合与化简的 BPMN 过程模型验证方法。

1 语义验证 BPMN 过程模型的设计理念

关于工作流模式,目前尚无文献给出明确定义。文献[12]对“模式”的定义是,“对特定上下文中反复出现的具体形式的一种抽象”。“模式”最初在面向对象设计领域被成功应用,后来 W. M. P. van der Aalst 等人将“模式”概念引入工作流技术领域,提出“工作流模式”,目的是提供一个系统且实用的方法来处理工作流描述语言的多样性,并作为评价不同工作流描述语言或工作流管理系统的标准,该方

法强调工作流模式的抽取与具体的工作流描述语言无关,但其实现依赖于具体的工作流描述语言^[6]。因此,笔者认为工作流模式可定义为“对特定上下文中反复出现的,满足特定业务需要的,业务活动之间某种行为方式的一种抽象”。

上述定义体现了工作流模式的三个重要特性:

①每个工作流模式有其特定的上下文环境,离开该环境,工作流模式就无法满足业务需要;②构成工作流模式的业务活动之间有特定行为方式,如业务活动之间是顺序或并行执行等;③每个工作流模式只能满足某类业务需要。据此,工作流模式可形式化定义为 $WCP=(C, S, F)$ 。其中: C 是实现 WCP 的上下文环境; S 是构成 WCP 的业务活动之间的行为方式; F 是 WCP 可以满足的业务需要。

众所周知,程序通过主函数调用其他函数共同实现系统功能,函数间可以相互调用。如果用函数来理解工作流模式, WCP 可视为函数名, C 是函数接口, S 是函数体, F 是函数功能描述。同理,工作流过程模型可视为多个 WCP 实例的组合,即 $WPM=(Patterns, Context, Function)$ 。其中: $Patterns=\{wcp_1, wcp_2, \dots, wcp_n\}$,且 wcp_i 是 WCP 实例, $Context$ 是 WCP 实例之间组合调用方式, $Function$ 是 WPM 要完成的业务目标。工作流模式之间相互组合,形成了实现工作流模式的上下文环境。

2 BPMN 过程模型正确性评价标准选择

目前,研究人员至少提出了 Soundness, Weak Soundness, Relaxed Soundness 和 Lazy Soundness 四种^[11]工作流过程模型正确性评价标准。这四种标准的差异在于是否允许工作流过程模型存在死锁(dead locks)、活锁(live locks)、死活动(dead activities)和同步丢失(lack synchronization)。死锁指计算机执行工作流过程模型时,由于某个活动使能条件无法满足而导致计算机中止执行工作流;活锁指工作流过程模型执行期间,某个活动的使能条件始终满足,导致计算机不断循环执行该活动而无法执行完成工作流;死活动指工作流过程模型中没有路径让计算机可以从开始节点执行到该活动;同步丢失指当计算机执行到工作流过程模型的结束点时,该工作流过程模型中还有活动在执行。

工作流过程模型是否正确,业务人员关注的是能否实现预定业务目标,而 IT 人员关注的是计算机能否顺利执行完成模型所描述的过程。对计算机

而言,在工作流过程模型中出现死锁和活锁是不允许的;死锁会让计算机无法执行,活锁会让计算机陷入死循环;死活动和同步丢失对计算机执行完成工作流没有影响。但对业务人员来讲,死活动意味着有冗余活动存在,同步丢失意味着业务流程设计得不合理。

四种标准的存在,说明工作流过程模型是否正确没有一个绝对标准,与工作流描述语言对工作流模式的支持有关。理论上讲,人们很难将实际管理中所有的工作流模式全部列举出来,文献[13]将文献[6]总结的 20 种工作流模式扩展为 43 种,正好说明了这一点。但仔细分析文献[13]提出的工作流模式不难发现,最简单、最完整的工作流模式是只包含一个开始节点、一个活动节点、一个结束节点的 WCP_1 (以下称作 Well- WCP_1)。除 Well- WCP_1 可以被视为一个完整的、正确的工作流过程模型外,其余工作流模式并不完整,需要合理组合在一起才能构成一个完整的、正确的工作流过程模型。因此,将 BPMN 过程模型能否转换为一个等价的 Well- WCP_1 ,作为语义验证 BPMN 过程模型的标准。

BPMN 过程模型中的错误包括语法错误和语义错误两种。语法错误指 BPMN 过程模型中的元素属性设置不符合要求;语义错误指 BPMN 过程模型中的工作流模式组合不当。对于语法错误,文献[14]介绍的基于 XPath/Xquery 方法值得借鉴。

3 实现 BPMN 过程模型语义验证的基础

3.1 BPMN 符号扩展及语义改进

BPMN 源于流程图(flow charts),能为业务人员图形化设计业务流程提供支持;但要计算机实现 BPMN 1.2 描述的业务流程,必须解决其 Process 元素执行语义不明等问题。为此,OMG 于 2007 年 6 月发布了制订 BPMN 2.0 规范的 RFP 文件,并于 2009 年 8 月发布了 BPMN 2.0 FTF Bata 1^[15]。与 BPMN 1.2 相比,在 BPMN 2.0 FTF Bata 1 中,Process 元素执行语义有所增强,但存在的问题,特别是 Event 元素类型由 BPMN 1.2 的 26 种扩为 54 种,对业务人员来讲,无疑增加了学习和使用难度。因此,笔者以业务人员为本,通过权衡业务人员图形化设计和计算机实现业务流程两方面需求,研究了基于 BPMN 的业务流程一体化建模方法,对 BPMN 1.2 的符号及语义做了改进,主要有:

改进 1 Activity 元素类型由 {Task, SubPro-

cess} 扩为 {Task, Block, SubProcess}, Block 是元素 Task, Event, Gateway 的偏序集合;取消 SubProcess 的原 Ad Hoc 属性。

改进 2 在 Process 和 SubProcess 中,任何 Activity 元素有且只有一个输入流和一个输出流,但不包括 Compensate 属性非空的 Activity 元素;在 Block 中,任何 Activity 元素要么有一个输入流和一个输出流,要么没有。

定义 1 元素 Block 执行语义。当 Block 使能时,总是从 Block 中无输入流的 Task 开始执行;没有结束指令时,Block 执行完毕,当且仅当 Block 中所有无输出流的 Task 执行完毕。

改进 3 任何 Activity 元素执行完毕以后,有且只有一个令牌 Token 触发其直接后继;具有多实例属性的 Activity 元素执行完毕,当且仅当其生成的全部实例执行完毕。

改进 4 任何 StartEvent 元素无输入流,有且只有一个输出流;任何元素无输出流,有且只有一个输入流;任何 IntermediateEvent 元素有且只有一个输入流和一个输出流,但当 IntermediateEvent 与元素绑定时,无输入流,有且只有一个输出流。

改进 5 任何用于分叉控制的 Gateway 元素有且只有一个输入流,但至少要有 2 个及以上输出流;任何用于合并控制的 Gateway 元素有且只有一个输出流,但至少要有 2 个及以上输入流。

改进 6 当元素 InclusiveGateway 用于分叉控制时,其执行语义是如果有 m 个直接后继,至少有 n 个直接后继被使能,且 $1 \leq n \leq m$;当元素 InclusiveGateway 用于合并控制时,其使能条件是如果有 m 个直接前驱,则至少有 n 个直接前驱已使能,且 $0 \leq n \leq m$ 。

改进 7 元素 ComplexGateway 仅限用于合并控制,其执行语义是如果有 m 个直接前驱,要求 m 个直接前驱已使能,当且仅当第一个使能完毕的直接前驱触发 ComplexGateway 时,ComplexGateway 可以使能并触发直接后继;当且仅当 ComplexGateway 的其余直接前驱全部使能完毕并已触发 ComplexGateway 时,ComplexGateway 才能再次被使能。

改进 8 Activity 元素新增 AttachedEvent 属性指向绑定在该 Activity 元素上的零个或多个 IntermediateEvent 元素;取消 IntermediateEvent 元素的 Target 属性。

限于篇幅,本文未将改进全部列出,只给出了与后面内容相关的几条。与 BPMN 1.2 规范相比,业务人员用改进后的 BPMN 符号设计业务流程, BPMN 过程模型也许没有改进前的简洁,但执行语义更明确,没有增加业务人员理解难度。表 1 是笔者采用的主要建模符号,其余建模符号同 BPMN 1.2 规范^[1]。

表 1 改进后的部分 BPMN 符号

符号名称	图示	编码
原子任务(Task)		a
活动集(Block)		b
子过程(SubProcess)		c
开始事件(Start Event)		d
中间事件(Intermediate Event)		e
结束事件(End Event)		f
基于数据的异或网关 (Data-Based Exclusive Gateway)		h
基于事件的异或网关 (Event-Based Exclusive Gateway)		k
容他网关(Inclusive Gateway)		m
复杂网关(Complex Gateway)		n
并行网关(Parallel Gateway)		o

为便于叙述,本文用集合 $A = \{a, b, c\}$ 表示全部 Activity 元素,用集合 $E = \{d, e, f\}$ 表示全部 Event 元素,用集合 $G = \{h, k, m, n, o\}$ 表示全部 Gateway 元素,小写字母 a, b, c 等为表 1 中 BPMN 符号编码。

3.2 工作流模式编码规则设计

仔细分析 BPMN 图形表示的工作流模式(见文献[16]),不难发现:有些工作流模式不能嵌套其他工作流模式,如 WCP_1, WCP_2 等;而有些工作流模式可以分解为其他工作流模式,如 WCP_{10}, WCP_{18} 等。本文将前者称为原子工作流模式,后者称为复合工作流模式。为形式化编码原子工作流模式,本文用 $|X|$ 表示取集合 X 中元素个数,符号 \wedge 表示逻辑与,符号 \vee 表示逻辑或,函数 $Node-$

$Type(x)$ 表示取元素 x 的类型,值域为 $A \cup E \cup G$, 并做如下定义:

定义 2 符号 \cdot 表示元素间的顺序关系,如 $x \cdot y$ 表示元素 x 执行完毕再执行元素 y 。

定义 3 符号 τ 表示空活动,语义是 $x \cdot \tau = \tau \cdot x = x$, 且 $NodeType(x) \in A \cup E \cup G$ 。

在 BPMN 过程模型中,当两个类型同属集合 G 的元素直接相连时,可视作中间有一个空活动 τ 存在,以便统一原子工作流模式编码。

定义 4 符号 $|$ 表示元素间的并行关系,如 $x | y$, 且 $x | y$ 等价于 $y | x$ 。

定义 4 中的元素 x 和元素 y 是同时执行,还是异或执行,取决于它们的直接前驱元素;如果它们的直接前驱元素是一个并行网关,则元素 x 和元素 y 将并行执行;如果它们的直接前驱元素是一个基于数据或事件的异或网关,则元素 x 和元素 y 将异或执行。

定义 5 式子 $x \cdot (y_1 | y_2 | \dots | y_n)$ 有效,当且仅当 $(NodeType(x) \in G) \wedge (NodeType(y_i) \in A \cup \{e, \tau\})$ 成立,语义是元素 x 执行完毕,元素 $y_1 \sim y_n$ 中至少有一个被触发。

定义 6 式子 $(x_1 | x_2 | \dots | x_n) \cdot y$ 有效,当且仅当 $(NodeType(x_i) \in A \cup \{e, \tau\}) \wedge (NodeType(y) \in G)$ 成立,语义是元素 x_i 执行完毕就触发元素 y , 但元素 y 何时使能取决于元素 y 语义及已使能的 x_i 。

以 $(x | y | z) \cdot u$ 为例,如果元素 u 为表 1 中的并行网关(Parallel Gateway),则只有当元素 x 、元素 y 和元素 z 都使能并触发元素 u 后,元素 u 才使能;如果元素 u 为表 1 中基于数据的异或网关(data-based exclusive gateway),只要元素 x, y, z 中有一个使能并触发元素 u ,元素 u 立即使能。

定义 7 符号 $@$ 表示元素间的绑定关系,符号 $\&$ 用作元素前缀,如 $\&x$ 表示 x 是绑定元素; $@$ 和 $\&$ 的语义满足定义 8。

符号 $@$ 和 $\&$ 用于形式化编码 BPMN 过程模型中有异常处理的 Activity 元素,因异常处理结构在执行时,元素之间既非顺序执行,也非并行执行,无法用定义 6、定义 7 中式子表示。

定义 8 式子 $x @ (y | \&z_1 | \&z_2 | \dots | \&z_n)$ 有效,当且仅当 $(NodeType(x) \in A) \wedge (NodeType(y) \in A \cup \{e\}) \wedge (NodeType(z_i) \in \{e\})$ 成立;语义是元素 x 执行完毕就执行元素 y ,但在元素 x 执行

期间,如果元素 z_i 的触发条件被满足,就中止执行元素 x ,转而执行元素 z_i 。

广度优先搜索有向图存储表示的 BPMN 过程模型,容易得到类似定义 2~定义 8 中的式子;如果将式子中每个元素赋予明确语义,则可用形式化编码表示原子工作流模式。

以图 1 所示的工作流模式为例, WCP_1 的形式化编码为 $x \cdot y \cdot z$, 且 $(NodeType(x) \in A \cup \{e\}) \wedge (NodeType(y) \in A \cup \{e\}) \wedge (NodeType(z) \in A \cup \{e\})$; WCP_4 的形式化编码为 $x \cdot y \cdot (z|u)$, 且 $(NodeType(x) \in A \cup \{d, e\}) \wedge (NodeType(y) \in \{h\}) \wedge (NodeType(z) \in A \cup \{e\}) \wedge (NodeType(u) \in A \cup \{e\})$; WCP_5 的形式化编码为 $(x|y) \cdot z \cdot u$, 且 $(NodeType(x) \in A \cup \{e\}) \wedge (NodeType(y) \in A \cup \{e\}) \wedge (NodeType(z) \in \{h\}) \wedge (NodeType(u) \in A \cup \{e, f\})$ 。

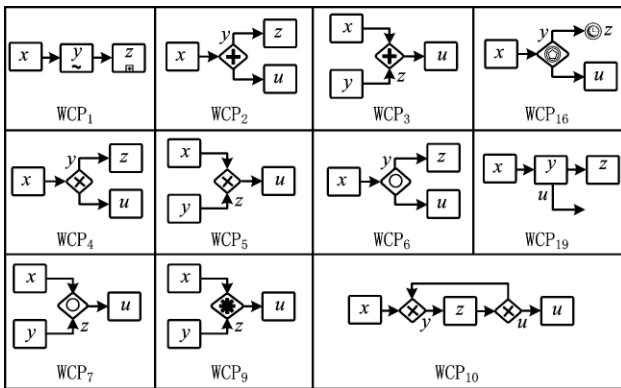


图1 部分工作流模式BPMN图形表示

复合工作流模式可以分解为多个原子工作流模式,如图 1 中的 WCP_{10} , 其形式化编码为 $(x|\tau) \cdot y \cdot z \cdot u \cdot (\tau|v)$, 且 $(NodeType(x) \in A \cup \{d, e\}) \wedge (NodeType(y) \in \{h\}) \wedge (NodeType(z) \in A \cup \{e\}) \wedge (NodeType(u) \in \{h\}) \wedge (NodeType(v) \in A \cup \{e, f\})$; 将其拆分成 $(x|\tau) \cdot y \cdot z$ 和 $z \cdot u \cdot (\tau|v)$, 不难发现两段编码分别表示 WCP_4 和 WCP_5 。

3.3 工作流模式化简规则设计

从图的角度来看,每个原子工作流模式的形式化编码,实际上对应所属 BPMN 过程模型的一个子图。文献[2]~文献[3]提出的工作流过程模型验证方法,实际上是将复杂子图化简为等价的简单子图,满足性质不变要求,即模型原来存在死锁等错误,化简以后仍存在相同错误。文献[4]、文献[5]和文献

[7]提出的基于 Petri 网的工作流过程模型验证方法同样满足性质不变要求。借鉴文献[4]和文献[7]的工作流网化简规则,本节设计了如下工作流模式组合化简规则:

规则 1 当且仅当 $(NodeType(x) \in A \cup \{d, e\}) \wedge (NodeType(y) \in A \cup \{e\})$ 成立时,式子 $x \cdot y$ 可直接化简为 x 或 y ,即 $x \cdot y \rightarrow x$ 或 y 。

规则 1 是将顺序执行的 Activity 元素或 IntermediateEvent 直接化简为一个元素。

规则 2 当且仅当 $NodeType(y') \in \{k\}$ 时,式子 $x \cdot y @ (z|&u)$ 可等价转换为 $x \cdot y' \cdot (z|u)$, 即 $x \cdot y @ (z|&u) \xrightarrow{NodeType(y') \in \{h\}} x \cdot y' \cdot (z, u)$ 。

规则 2 是将异常处理结构转义为异或选择结构;从语义执行角度来讲,这种转变是不合理的;但由于元素 y 即使绑定有多个中间事件,最多只有一个绑定事件或直接后继被触发,不会同时触发多个;因此,这种转换用于 BPMN 过程模型验证是可行的,满足性质不变要求。

规则 3 设集合 $Z = \{z_1, z_2, \dots, z_n\}$, 集合 $U = \{u_1, u_2, \dots, u_m\}$, 集合 $X = Z - U = \{z'_1, z'_2, \dots, z'_n\}$, 集合 $Y = U - Z = \{u'_1, u'_2, \dots, u'_m\}$, 条件 $Cond = ((NodeType(y) \in \{o\}) \wedge (NodeType(v) \in \{n, o\})) \vee ((NodeType(y) \in \{h, k\}) \wedge (NodeType(v) \in \{h\})) \vee ((NodeType(y) \in \{m\}) \wedge (NodeType(v) \in \{m\}))$; 当且仅当 $(Z=U) \wedge Cond$ 成立时,式子 $x \cdot y \cdot (z_1|z_2|\dots|z_n)$ 和 $(u_1|u_2|\dots|u_m) \cdot v$ 可组合化简为 x 或 w ; 当且仅当 $(Z \subset U) \wedge Cond$ 成立时,式子 $x \cdot y \cdot (z_1|z_2|\dots|z_n)$ 和 $(u_1|u_2|\dots|u_m) \cdot v \cdot w$ 可组合化简为 $(z'_1|z'_2|\dots|z'_n|x) \cdot v \cdot w$; 当且仅当 $(Z \supset U) \wedge Cond$ 成立时,式子 $x \cdot y \cdot (z_1|z_2|\dots|z_n)$ 和 $(u_1|u_2|\dots|u_m) \cdot v \cdot w$ 可组合化简为 $x \cdot y \cdot (u'_1|u'_2|\dots|u'_m|w)$ 。

规则 3 用来化简两个组合成全闭合或半闭合结构的原子工作流模式;但需要特别说明的是,图 1 中的工作流模式 WCP_7 与 WCP_2 , WCP_4 , WCP_6 , WCP_{16} 及 WCP_{19} 中任何一个组成闭合结构,计算机都能实现,但不必要地增加了计算机处理负担。因此本文将 WCP_7 与 WCP_6 组合形成的闭合结构视为合理,与其他工作流模式组合形成的闭合结构视为不合理。

规则 4 当且仅当 $(NodeType(y) \in G) \wedge (NodeType(v) \in G) \wedge (NodeType(y) = NodeType(v))$

(v)成立时,式子 $x \cdot y \cdot (z_1 | z_2 | \dots | z_n | u)$ 和 $u \cdot v \cdot (w_1 | w_2 | \dots | w_m)$ 可组合化简为 $x \cdot y \cdot (z_1 | z_2 | \dots | z_n | w_1 | w_2 | \dots | w_m)$ 。

规则 4 实际上是将两个相连的同种类型分叉结构(即分叉控制网关类型相同)组合化简为一个相同类型的分叉结构。

规则 5 当且仅当 $(Node\ Type(y) \in G) \wedge (Node\ de\ Type(v) \in G) \wedge (Node\ Type(y) = Node\ Type(v))$ 成立时,式子 $(x_1 | x_2 | \dots | x_n) \cdot y \cdot z$ 和 $(z | u_1 | u_2 | \dots | u_m) \cdot v \cdot w$ 可组合化简为 $(x_1 | x_2 | \dots | x_n | u_1 | u_2 | \dots | u_m) \cdot v \cdot w$ 。

规则 5 实际上是将两个相连的同种类型合并结构(即合并控制网关类型相同)组合化简为一个相同类型的合并结构。

规则 6 设集合 $Z_i = \{z_{i1}, z_{i2}, \dots, z_{in}\}, U_j = \{u_{j1}, u_{j2}, \dots, u_{jm}\}, 1 \leq i \leq m, 1 \leq j \leq n, m \leq n$; 当且仅当 $(Node\ Type(y_i) \in \{o\}) \wedge (Node\ Type(v_j) \in \{h\}) \wedge (Z_i \cap Z_i' = \emptyset) \wedge (U_j \cap U_j' = \emptyset) \wedge (\bigcup_{i=1}^m Z_i = \bigcup_{j=1}^n U_j)$

成立时,式子 $\sum_{i=1}^m (x_i \cdot y_i \cdot (z_{i1}, z_{i2}, \dots, z_{in}))$ 和 $\sum_{j=1}^n ((u_{j1}, u_{j2}, \dots, u_{jm}) \cdot v_j \cdot w_j)$ 可组合化简为 $(x_1, x_2, \dots, x_m) \cdot y_1 \cdot \tau$ 和 $\tau \cdot v_1 \cdot (w_1, w_2, \dots, w_n)$ 。

规则 6 将多个 WCP_2 和多个 WCP_5 组合形成的交叉重叠结构化简为一个 WCP_2 相连一个 WCP_5 , 理论依据是文献[4]中的规则 6 及文献[7]中的定义 6.3。

规则 7 设集合 $X = \{x_1, x_2, \dots, x_n\}$, 集合 $W = \{w_1, w_2, \dots, w_m\}$, 集合 $X' = X - W = \{x'_1, x'_2, \dots, x'_n\}$, 集合 $W' = W - X = \{w'_1, w'_2, \dots, w'_m\}$, $1 \leq n' < n, 1 \leq m' < m$; 当且仅当 $(|X \cap W| \geq 1) \wedge (z \notin X) \wedge (Node\ Type(y) \in \{h\}) \wedge (Node\ Type(v) \in \{h\})$, 式子 $(x_1 | x_2 | \dots | x_n) \cdot y \cdot z$ 和 $u \cdot v \cdot (w_1 | w_2 | \dots | w_m)$ 可化简为 $(x'_1 | x'_2 | \dots | x'_n) \cdot y \cdot z$ 和 $u \cdot v \cdot (w'_1 | w'_2 | \dots | w'_m)$ 。

规则 7 将多个组合在一起构成非死循环的原子 workflow 模式,以消掉循环的方式化简复合 workflow 模式 WCP_{10} 和 WCP_{21} , 理论依据是文献[4]中的规则 8 及文献[7]中的定义 6.13。注意:规则 7 的特例是当 $n'=1$ 时,式子 $(x_1 | x_2 | \dots | x_n) \cdot y \cdot z \rightarrow x'_1 \cdot y \cdot z \rightarrow x'_1$ 或 z 。

利用规则 1~规则 7 消掉元素时,如果两个元素可任选一个,则要求保留非 τ 元素,以便确定错误

所在。

4 BPMN 过程模型语义验证方法设计

4.1 模型编码

通过遍历语义有向图存储表示的 BPMN 过程模型,可得模型中原子 workflow 模式的形式化编码;这些形式化编码构成的集合 M 就是该 BPMN 过程模型编码,其实现算法如下。

算法 1

```
void EnCodeProcessModel(Graph G', CodeSet &M){
    //输入语义有向图存储表示的 BPMN 过程模型 G',
    要求该 BPMN 过程模型已通过语法验证;输出集合 M
    InitQueue(Q1); M={}; //初始化队列 Q1 和集合 M
    num=CountVexNum(G'); //求 G' 中顶点数
    for(i=0; i<num; i++){
        j=din(vi); //求顶点 vi 的入度,即输入流数
        if(j==0) EnQueue(Q1, vi); //顶点 vi 入度为零就
        入队 Q1
    }
    while(QueueEmpty(Q1) == FALSE){ //队列 Q1
        非空
        DeQueue(Q1, u); //Q1 执行出队操作,队头元素
        放入变量 u
        v = NextAdjustVerx(G', u); //取顶点 u 的邻接
        点 v
        code1=u.ID+"."+v.ID; //ID 为顶点 u、v 的
        标志号
        while((Node Type(v) ∈ E) ∨ ((Node Type(v) ∈ A)
        ∧ (v.AttachedEvent == None))){//元素 v 没有异常处理
            v = NextAdjustVerx(G', v); //取顶点 v 的邻接
            点 w
            code1=code1+"."+v.ID;
        } //循环结束时,要么遇到网关,要么活动有异常
        处理

        if(Node Type(v) ∈ A){ //带异常处理的活动
            //找正常输出流的邻接点并入队
            w=NextAdjustVerx(G', v);
            code2=w.ID;
            EnQueue(Q1, w);
            //找绑定的中间事件节点并入队
            w=GetAttachedEvent(v); //获取活动 v 绑定的中
            间事件

            while(w){ //w 存在就循环
                code2=code2+"|&."+w.ID;
                EnQueue(Q1, w);
            }
        }
    }
}
```

```

        w=GetAttachedEvent(v);
    }
    code1=code1+"@"+code2+""; //生成异常
    处理结构编码
}
else{//元素类型为网关
    num=dout(v);//求顶点 v 的出度,即输出流数
    w=NextAdjustVerx(G',v);//取网关 v 的邻接点
    if(NodeType(w)∈G){//邻接点 w 是网关
        w=CreateNullAction(v,w);//生成特殊活动 τ,
        暂存 τ 的前驱 v、后继 w 及编号 ID
    }
    if(num==1){
        code1=code1+" "+w.ID;
        EnQueue(Q1,w);
    }
    else{//网关用于分叉时有多个邻接点
        code2=w.ID;
        EnQueue(Q1,w);
        for(int i=1;i<num;i++){
            w=NextAdjustVerx(G',v);//取网关节点的
            第二个邻接点
            if(NodeType(w)∈G) w=CreateNullAction
            (v,w);
            code2=code2+"|"+w.ID;//循环结束时遇
            到了网关
            EnQueue(Q1,w);
        }
        code1=code1+"("+code2+"";
    }
    M=M∪{code1};//将编码段 code1 并入集合 M
}
M=MergeCode(M);//合并集合 M 中每个编码段最
后两部分相同的编码,即将式子  $x \cdot y \cdot z$ 
//和  $u \cdot y \cdot z$  合并为  $(x|u) \cdot y \cdot z$ ,
并将合并后的编码集合返回给集合 M
}

```

当 BPMN 过程模型中有 Block 和 SubProcess 元素时,根据 3.1 节的改进 2,算法 1 同样适用于 Sub-Process 中工作流模式形式化编码;基于算法 1 稍作改动,可得 Block 中工作流模式形式化编码算法。

4.2 模型化简

循环使用 3.3 节设计的规则 1~规则 7,可以组合化简 BPMN 过程模型的编码集合 M ,直到不能继续组合化简为止,其实现算法如下。

算法 2

```

void PatternReduction(CodeSet M, CodeSet &M') {
    //输入 BPMN 过程模型编码集合 M,输出已组合化
    简的编码集合 M'
    M'=M;    M''={}; //集合 M''初值为空
    While(M'≠M''){//可以继续组合化简集合 M'
        M''=M';
        ExcuteRule01(M'); //对 M'执行规则 Rule 1
        ExcuteRule02(M'); //对 M'执行规则 Rule 2
        ExcuteRule03(M'); //对 M'执行规则 Rule 3
        ExcuteRule04(M'); //对 M'执行规则 Rule 4
        ExcuteRule05(M'); //对 M'执行规则 Rule 5
        ExcuteRule06(M'); //对 M'执行规则 Rule 6
        ExcuteRule07(M'); //对 M'执行规则 Rule 7
    }
}

```

算法 2 化简编码集合 M 时,被消掉的元素如果在集合 M 的其他编码中存在,这些元素将被替换为留下的元素;如果化简得到的编码只剩一个元素,且在其他编码中存在,则可以直接消掉。例如,设编码集合 $M=\{x \cdot y \cdot (z_1|z_2), (z_1|z_2) \cdot v \cdot w, w \cdot t \cdot (r,s)\}$,且前两个编码满足化简规则 5 要求,则执行算法 2 以后,集合 $M'=\{x \cdot t \cdot (r,s)\}$ 。

4.3 错误识别

如果集合 M' 中只有一个形式化表示工作流模式 Well-WCP₁ 的编码,则可判定对应 BPMN 过程模型语义正确,否则有错。此时,分析集合 M' 中编码,可确定错误所在及错误原因。例如:设编码集合 $M'=\{x \cdot y \cdot (z_1|z_2), (z_1|z_2) \cdot v \cdot w\}$,且 $NodeType(y) \in \{h\}$, $NodeType(v) \in \{o\}$;此时 $x \cdot y \cdot (z_1|z_2)$ 表示工作流模式 WCP₄, $(z_1|z_2) \cdot v \cdot w$ 表示工作流模式 WCP₃,用本文 3.3 节的规则 1~规则 7 无法继续化简 M' ,当计算机执行到该 BPMN 过程模型的元素 v 时, v 的使能条件无法满足,出现死锁错误。

5 语义验证 BPMN 过程模型方法示例

下面以文献[16]提供的澳大利亚技术移民签证申请流程为例,检验第 4 章的验证方法是否有效。文献[16]用 YAWL 语言描述该流程,内含 WCP₁, WCP₂, WCP₂₅ 等工作流模式,笔者用 BPMN 符号重新进行了描述,如图 2 所示。除 WCP₂₅ 外,该流程在文献[16]中有的工作流模式图 2 全有。

际验证的需要,文献[11]也对文献[7]提出的验证方法提出了质疑。与早期没有标准评价不同工作流描述语言和工作流管理系统一样,目前也存在没有比较不同验证方法优劣的标准。工作流模式为评价不同工作流描述语言及工作流管理系统提供了标准,可以考虑以支持多少工作流模式作为评价语义验证方法的标准。

严格来讲,笔者目前还无法回答本文方法是否比现有方法好。但与现有方法相比,本文方法的不同之处至少有两点:①直接用于 BPMN 过程模型;②基于工作流模式编码实现。Petri 网和 Pi-演算是两种重要的形式化理论,有坚实的数学基础,有人在研究如何将两种理论融合起来用于业务流程管理,本文是笔者的一个尝试,希望能为 BPMN 过程模型语义验证提供新思路。本文方法还存在不支持 WCP₁₈ 和 WCP₄₃ 等高级工作流模式、不支持 BPMN 过程编排验证等不足,笔者将继续进行深入研究。

参考文献:

- [1] OMG. Business process modeling notation(BPMN) version 1.2 [EB/OL]. [2009-01-03]. <http://www.omg.org/spec/BPMN/1.2/PDF>.
- [2] SADIQ W, ORLOWSKA M. Analyzing process models using graph reduction techniques[J]. Information System, 2000, 25(2):117-134.
- [3] SONG Baoyan, WANG Juying, YU Ge. Verification method for process model based on graph-spreading and graph-reduction[J]. Mini-Micro System, 2005, 26(6):1073-1078(in Chinese). [宋宝燕,王菊英,于戈.基于图形展开及图形规约的过程模型验证方法[J].小型微型计算机系统,2005,26(6):1073-1078.]
- [4] LI Jianqiang, FAN Yushun. Research of Petri net based on workflow model reduction methods[J]. Information and Control, 2001, 30(6):492-497(in Chinese). [李建强,范玉顺.基于Petri网化简方法的工作流模型验证[J].信息与控制,2001,30(6):492-497.]
- [5] VAN DER AALST W M P, TER HOFSTEDE A H M. YAWL:yet another workflow language[J]. Information Systems, 2005, 30(4):245-275.
- [6] VAN DER AALST W M P, TER HOFSTEDE A H M, KIEPUSZEWSKI B, et al. Workflow patterns[J]. Distributed and Parallel Databases, 2003, 14(1):5-51.
- [7] WYNN M. Semantics, verification, and implementation of workflows with cancellation regions and OR-joins[D]. Brisbane, Australia:Queensland University of Technology, 2006.
- [8] DIJKMAN R M, DUMAS M, OUYANG C. Semantics and analysis of business process models in BPMN[J]. Information and Software Technology, 2008, 50(12):1281-1294.
- [9] WONG E Y H, GIBBONS J. A process semantics for BPMN [C]//Proceedings of the 10th International Conference on Formal Methods and Software Engineering. Berlin, Germany: Springer-Verlag, 2008:355-374.
- [10] PUHLMANN F, WESKE M. Using the Pi-calculus for formalizing workflow patterns[C]//Proceedings of the 3rd International Conference on Business Process Management. Berlin, Germany:Springer-Verlag, 2005:153-168.
- [11] PUHLMANN F, WESKE M. Investigations on soundness regarding lazy activities[C]//Proceedings of the 4th International Conference on Business Process Management. Berlin, Germany:Springer-Verlag, 2006, 4102:145-160.
- [12] RIEHLE D, ULLIGHOVEN H Z. Understanding and using patterns in software development[J]. Theory and Practice of Object Systems, 1996, 2(1):3-13.
- [13] RUSSELL N, TER HOFSTEDE A H M, VAN DER AALST W M P, et al. Workflow control-flow patterns:a revised view[R]. Eindhoven, the Netherlands:BPM Center, 2006.
- [14] CHINOSI M, TROMBETTA A. Modeling and validating BPMN diagrams[C]//Proceedings of 2009 IEEE Conference on Commerce and Enterprise Computing. Washington, D. C., USA:IEEE, 2009:353-360.
- [15] OMG. Business process modeling notation(BPMN) FTF bata 1 for version 2.0 [EB/OL]. [2009-08-14]. <http://www.omg.org/spec/BPMN/2.0>.
- [16] WOHEDE P, VAN DER AALST W M P, DUMAS M, et al. Pattern-based analysis of BPMN—an extensive evaluation of the control-flow, the data and the resource perspectives[R]. Eindhoven, the Netherlands:BPM Center, 2005.

作者简介:

蔡章利(1973—),男,四川大竹人,重庆大学机械工程学院博士研究生,重庆大学自动化学院讲师,研究方向:业务流程管理、智能信息处理, E-mail: czl@cqu.edu.cn;

易树平(1960—),男,四川富顺人,教授,博士生导师,研究方向:工业工程、先进制造与管理技术;

高庆萱(1970—),女,重庆人,讲师,博士,研究方向:工业工程、业务流程再造。