软件学报 ISSN 1000-9825, CODEN RUXUEW
Journal of Software,2018,29(4):1094−1114 [doi: 10.13328/j.cnki.jos.005280]

E-mail: jos@iscas.ac.cn
http://www.jos.org.cn
Tel: +86-10-62562563

BPMN 2.0 编排的形式语义和分析*

代 飞[1,2,3], 赵文卓[2], 杨 云[2,3], 莫 启[2,3], 李 彤[2,3], 周 华[1]

[1](西南林业大学 大数据与人工智能工程学院,云南 昆明 650091)

[2](云南大学 软件学院,云南 昆明 650091)

[3](云南省软件工程重点实验室,云南 昆明 650091)

通讯作者: 杨云, E-mail: yangyan19@hotmail.com

摘 要: BPMN 2.0 编排已成为描述业务流程间交互事实上的标准.BPMN 2.0 编排面向流的特征,使之会产生控制流方面的语义错误.因此,检查编排语义正确性是 BPMN 2.0 编排建模工具所期望具有的功能.但是,BPMN 2.0 标准规约中的编排缺少形式语义及相应的分析技术,这阻碍了对 BPMN 2.0 编排的语义分析.提出了一种映射,用于将 BPMN 2.0 编排转换为工作流网,使用 Petri 网来形式化定义 BPMN 2.0 编排的语义.借助 Petri 网的分析技术,这种定义的语义可用来分析 BPMN 2.0 编排的结构和控制流方面的错误.该映射和语义分析已被实现为一种工具.实验结果表明,这种形式化可以识别 BPM AI 过程模型库中编排的语义错误.

关键词: 业务流程建模标注 2.0;编排;Petri 网;形式语义;语义分析

中图法分类号: TP311

中文引用格式: 代飞,赵文卓,杨云,莫启,李彤,周华.BPMN 2.0 编排的形式语义和分析.软件学报,2018,29(4):1094−1114. http://www.jos.org.cn/1000-9825/5280.htm

英文引用格式: Dai F, Zhao WZ, Yang Y, Mo Q, Li T, Zhou H. Formal semantics and analysis of BPMN 2.0 choreographies. Ruan Jian Xue Bao/Journal of Software, 2018,29(4):1094−1114 (in Chinese). http://www.jos.org.cn/1000-9825/5280.htm

Formal Semantics and Analysis of BPMN 2.0 Choreographies

DAI Fei[1,2,3], ZHAO Wen-Zhuo[2], YANG Yun[2,3], MO Qi[2,3], LI Tong[2,3], ZHOU Hua[1]

[1](Big Data and Artificial Intelligence College, Southwest Forestry University, Kunming 650091, China)

[2](School of Software, Yunnan University, Kunming 650091, China)

[3](Key Laboratory for Software Engineering of Yunnan Province, Kunming 650091, China)

Abstract: The Business Process Modelling Notation 2.0 (BPMN 2.0) choreography is a de factor standard for capturing the interactions between business processes. Flow-oriented BPMN 2.0 choreographies can exhibit a range of semantic errors in the control flow. The ability to check the semantic correctness of choreographies is thus a desirable feature for modelling tools based on BPMN 2.0 choreographies. However, the semantic analysis of BPMN 2.0 choreographies is hindered by the lack of formal semantic definition of its constructs and the corresponding analysis techniques in the BPMN 2.0 standard specification. This paper defines a formal semantics of BPMN 2.0 choreographies in terms of a mapping to WF-nets. This defined semantics can be used to analyze the structural errors and the

* 基金项目: 国家自然科学基金(61462095, 61663046, 61402397, 61379032); 云南省自然科学基金(2016FB102, 2016FB104); 云南省中青年学术和技术带头人后备人才培养项目(C6143002); 云南省软件工程重点实验室开放基金面上项目(2017SE201, 2016SE202, 2015SE102); 云南省教育厅科学研究基金重大专项项目(ZD2014001)

Foundation item: National Natural Science Foundation of China (61462095, 61663046, 61402397, 61379032); Natural Science Foundation of Yunnan Province, China (2016FB102, 2016FB104); Talent Project of Yunnan Province (C6143002); Open Fund Project of Key Laboratory of Software Engineering of Yunnan Province (2017SE201, 2016SE202, 2015SE102); Yunnan Provincial Department of Education Fund for Scientific Research of Major Special Projects (ZD2014001)

收稿时间: 2016-10-05; 修改时间: 2016-11-25; 采用时间: 2017-03-08; jos 在线出版时间: 2017-03-31

CNKI 网络优先出版: 2017-03-31 21:55:02, http://kns.cnki.net/kcms/detail/11.2560.TP.20170331.2155.013.html

control flow errors of BPMN 2.0 choreographies with analysis techniques of Petri nets. The proposed mapping and the semantic analysis have been implemented as a tool. The experimental results show this formalization can identify the semantic errors of choreographies from the BPM AI process model library.

Key words: business process modeling notation 2.0; choreography; Petri net; formal semantics; semantics analysis

业务流程建模标注 2.0(business process modeling notation 2.0,简称 BPMN 2.0)[1]是业务流程管理领域的一种标准符号.BPMN 2.0 存在 3 种基本类型的端到端的业务流程,分别是:编制(orchestration)、编排(choreography)和协作(collaboration).建模者不仅可以使用编制来建模组织内部的业务流程,还可以使用编排来建模组织间业务流程的交互.与编制相比,编排是在 BPMN 2.0 中首次提出的概念.它用于定义参与者(peer)之间的消息交互序列,其关注点是参与者间的消息交换,而非参与者内部的工作编制[1].

一方面,BPMN 2.0 具有面向图流程定义语言(graph-oriented process definition languages)的特征[2];另一方面,它还具有 Web 服务业务流程执行语言(Web service business process execution language,简称 BPEL)[3]的特征.这使得 BPMN 2.0 编排不仅会表现出由面向图流程定义产生的语义错误,例如流不一致(流增加或流减少)、死编排任务等,还会表现出整体编排与局部参与者间语义不一致的错误,例如可实现性(realizability)[4].其中,可实现性是指给定一个编排,参与者未必总能正确实现该编排[4].

由于上述原因,对 BPMN 2.0 编排进行语义分析成为了 BPMN 2.0 编排建模工具所期望具有的功能.但是,语义分析将面临下述挑战:一是,BPMN 2.0 标准规约用自然语言描述编排的语义,描述的语义存在不一致和二义性.二是,BPMN 2.0 标准规约缺乏分析技术,无法对编排进行语义分析.

面对上述挑战,本文关注 BPMN 2.0 编排的形式语义定义和语义分析,其主要贡献如下.

(1) 通过建立 BPMN 2.0 编排到工作流网(WF-nets)的映射,使用 Petri 网来准确定义 BPMN 2.0 编排的语义.首先,给出良构编排的概念,以消除构造结构的多样性,从而方便 BPMN 2.0 编排到工作流网的映射;其次,给出编排到工作流网映射的直观描述;最后,给出映射的形式定义.

(2) 借助 Petri 网的分析技术,把 BPMN 2.0 编排中存在的语义错误规约为工作流网的结构问题或性质问题,对 BPMN 2.0 编排进行语义分析.

(3) 本文实现了一种工具,用于将 BPMN 2.0 编排自动转换为工作流网,并借助现有的 Petri 网工具和自行设计的算法实现了对编排的语义分析.

本文第 1 节介绍编排和 Petri 网的相关概念.第 2 节定义良构编排.第 3 节讨论 BPMN 2.0 编排到 Petri 网的映射.第 4 节对编排进行语义分析.第 5 节是实验评估.第 6 节是相关工作的比较.最后是对本文进行总结.

1 相关基础

1.1 编 排

在 BPMN 2.0 中,编排由下述 4 种元素组成:编排活动(choreography activity)、事件(event)、网关(gateway)和序列流(sequence flow).编排的核心子集如图 1 所示,其中,编排活动、事件、网关统称为流对象(flow object);序列流又称为连接对象(connection object).本文将对图 1 所示的编排核心子集进行形式语义的定义.

编排活动是编排中执行的工作,分为编排任务和调用编排(call choreogprahy).编排任务又可分析为:标准编排任务(choreography task)和带内部标记(marker)的编排任务.标准编排任务是编排中的一个原子活动,用以描述两个参与者之间的一次或两次消息交换.带内部标记的编排任务用于描述编排任务的重复执行,包含 3 种类型:标准循环的编排任务(standard loop choreography task)、多实例并行的编排任务(parallel multi-instance choreography task)和多实例串行的编排任务(sequential multi-instance choreography task).调用编排是编排中的一个点,用以调用全局编排(global choreography).

事件是发生在编排中的事情,包含:开始事件(start event)、中间事件(intermediate event)和结束事件(end event).这些事件分别用以开始、中断、结束编排的流.其中,开始事件可分为 5 种类型:无(none)、计时器(timer)、

条件(conditional)、信号(signal)和多个(multiple);中间事件可分为 6 种类型:无(none)、计时器(timer)、条件(conditional)、链接(link)、信号(signal)和多个(multiple);结束事件可分为两种类型:无(none)和终止(terminate).

网关用于控制编排中序列流的收敛(converge)和发散(diverge),包含 3 种类型:排他数据网关(data-based exclusive gateway)、排他事件网关(event-based exclusive gateway)和并行网关(parallel gateway).其中,排他数据(事件)网关又包含:排他数据(事件)决策网关(data/event-based XOR decision gateway)和排他数据(事件)合并网关(data/event-based XOR merge gateway).排他数据(事件)决策网关用于从互斥的流中选择一个流;排他数据(事件)合并网关用于将一组互斥的流合成为一个流.并行网关又包含:并行分叉网关(parallel fork gateway)和并行汇聚网关(parallel join gateway).并行分叉网关用于产生并发流;并行汇聚网关用于同步并发流.

序列流用于连接流对象,表示编排中流对象间的控制流关系.

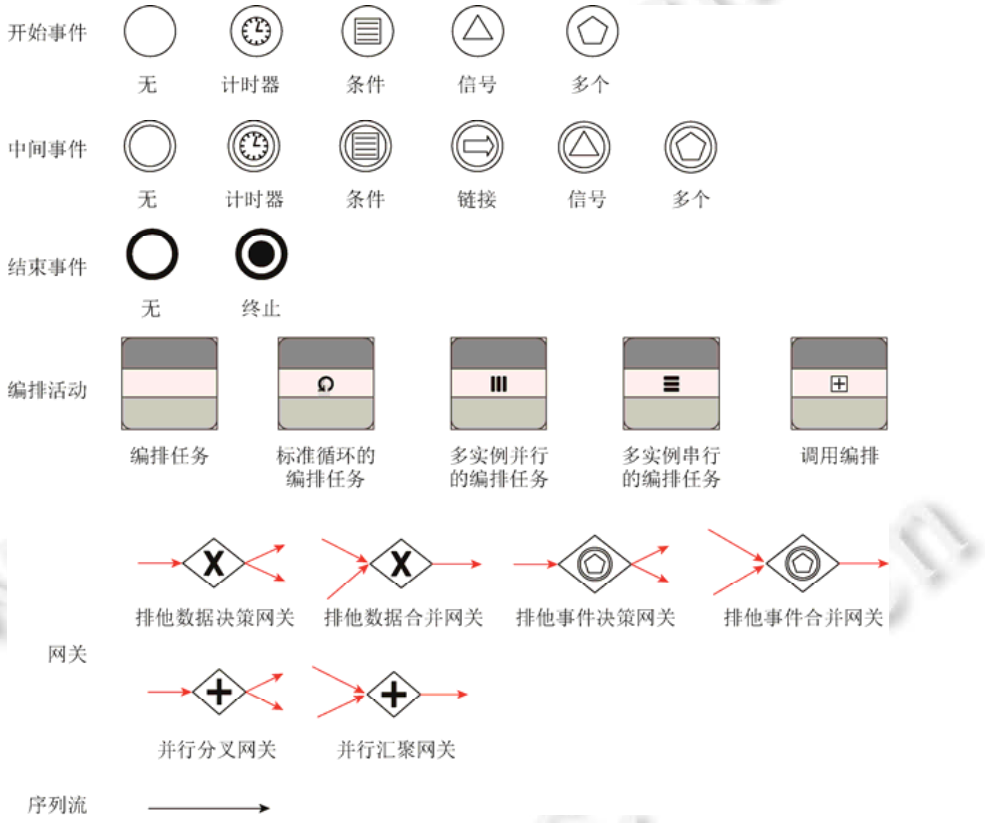

Fig.1 A core subnet of BPMN 2.0 choreographies

图 1 BPMN 2.0 编排的核心子集

1.2 Petri网

Petri 网是 1962 年由德国科学家 Petri 博士在他的博士论文《用自动机通信》中创立的一种网状结构[5].它是并发系统的一种形式模型,通过流来建模系统的行为.此外,Petri 网拥有大量的分析技术和工具,可以方便我们对编排进行分析和验证.

Petri 网是一个有向二分图,由库所和变迁组成.其图形化的表示方式使得 Petri 网直观、易懂.通常,库所使用圆圈表示,变迁使用方框表示,流关系使用有向线段表示,托肯使用实心小黑点表示.

定义 1[6]**(Petri 网)**. Petri 网是一个 4 元组 $\Sigma=(P,T;F,M)$, 其中,

(1) $P\cup T\neq\varnothing$, 习惯称 P 为库所集,T 为变迁集;

(2) $P\cap T=\varnothing$;

(3) $F\subseteq(P\times T)\cup(T\times P)$, 称 F 为流关系;

(4) 映射 $M:P\to\{0, 1, 2, 3,...\}$ 称为 Petri 网的一个标识.通常用 M_0 表示 Petri 网的初始标识.

定义 2[6](**前集和后集**). 设 $\Sigma=(P,T;F,M)$ 是一个 Petri 网,若 $x\in P\cup T,\dot{x}=\{y|(y,x)\in F\wedge y\in P\cup T\}$, 则称 $\dot{x}$ 为 x 的前集;若 $x\in P\cup T,\dot{x}=\{y|(x,y)\in F\wedge y\in P\cup T\}$, 则称 $\dot{x}$ 为 x 的后集.

定义 3[7](**工作流网**). 一个 $\Sigma=(P,T;F,M)$ 是工作流网(WF-net)当且仅当:

(1) 存在一个源库所 $i\in P$, 使得 $\dot{i}=\varnothing$;

(2) 存在一个汇聚库所 $o\in P$, 使得 $\dot{o}=\varnothing$;

(3) 每一个节点 $x\in P\cup T$ 都位于从 i 到 o 的一条路径上.

本质上,工作流网是 Petri 网的一类子网,要求只有一个源库所和一个汇聚库所,且每个变迁都在从源库所到汇聚库所的路径上.

定义 4[7](**合理性**). 设 $\Sigma=(P,T;F,M)$ 是一个工作流网,该工作流网是合理的当且仅当:

(1) 对于每一个从状态 i 可达的状态 M,存在一个实施序列,从状态 M 通往状态 o,形式化表示为

$$\forall_M(i\xrightarrow{*}M)\Rightarrow(M\xrightarrow{*}o);$$

(2) 状态 o 是从状态 i 可达的唯一最终状态,且结束时其中至少会有一个标记,形式化表示为

$$\forall_M(i\xrightarrow{*}M\wedge M\geqslant o)\Rightarrow(M=o);$$

(3) 无死变迁

$$\forall_{t\in T}\exists_{M,M'}i\xrightarrow{*}M\xrightarrow{t}M'.$$

2 良构编排

2.1 良构编排的定义

由于 BPMN 2.0 编排是面向流的,所以使用 Petri 网来定义 BPMN 2.0 编排的语义是非常合适的.本节将提出良构编排的概念,用于将结构不同但语义相同的编排转换为结构统一的编排,以消除构造结构的多样性.这样做的好处是:在定义编排的形式语义时,只需考虑良构编排到工作流网的映射.此外,良构编排的提出主要是为了方便 BPMN 2.0 编排到工作流网的映射,不会损失编排的一般性.

良构编排具有下述特征.

(1) 开始事件有一个,且只有一个输出流,而无输入流;

(2) 结束事件有一个,且只有一个输入流,而无输出流;

(3) 编排活动或中间事件均只有一个输入流和输出流;

(4) 编排活动只表示单向的发送或接收消息;

(5) 并行分叉网关、排他数据决策网关、排他事件决策网关只能有一个输入流,多个输出流;

(6) 并行汇聚网关、排他数据合并网关、排他事件合并网关只能有一个输出流,多个输入流;

(7) 网关必须与编排活动相连.

定义上述特征的目的在于规范编排的结构.其中,特征(1)~(3)、(5)、(6)用于确保编制中只有网关能够控制流的收敛和发散,而编排活动和事件只有一个输入流和输出流;特征(4)用于确保编排中每个活动只描述单向消息发送;特征(7)用于确保编排中不允许网关间直接嵌套.

为定义良构编排,下面先定义核心编排过程和编排的语法.

使用图 2 所示编排核心子集定义的编排过程,称为核心编排过程,参见定义 5.

定义 5(核心编排过程). 一个核心编排过程是一个 5 元组 CP={O,CA,E,G,F},其中,

(1) O 是流对象的集合, $O=(CA\cup E\cup G)\wedge(CA\cap E\cap G=\varnothing)$, 即划分为 3 个不相交的集合 CA,E 和 G;

(2) CA 是编排活动的集合, $CA=(CT\cup Call)\wedge(CT\cap Call=\varnothing)$, 即划分为两个不相交的集合,分别是编排任

务的集合 CT 和调用编排的集合 $Call$;

(3) E 是事件的集合, $E=(E^S\cup E^I\cup E^E)\wedge(E^S\cap E^I\cap E^E=\varnothing)$, 即划分为3个不相交的集合,分别是开始事件的集合 E^S、中间事件的集合 E^I、结束事件的集合 E^E.其中,E^I 又可划分为 6 个不相交的集合,分别是正常事件的集合 E^{IN}、计时器事件的集合 E^{IT}、计时器事件的集合 E^{IC}、链接事件的集合 E^{IL}、信号事件的集合 E^{IS}、多事件的集合 E^{IM};

(4) G 是网关的集合,

$$G=(G^{PF}\cup G^{PJ}\cup G^{XD}\cup G^{DM}\cup G^{ED}\cup G^{EM})\wedge(G^{PF}\cap G^{PJ}\cap G^{XD}\cap G^{DM}\cap G^{ED}\cap G^{EM}=\varnothing),$$

即可划分为 6 个不相交的集合,分别是并行分叉网关的集合 G^{PF}、并行汇聚网关的集合 G^{PJ}、排他数据决策网关的集合 G^{XD}、排他数据合并网关的集合 G^{DM}、排他事件决策网关的集合 G^{ED}、排他事件合并网关的集合 G^{EM};

(5) $F\subseteq O\times O$ 是连接对象的集合.

编排由一系列核心编排过程组成,参见定义 6.

定义 6(编排). 编排是一个 4 元组 $Chor=\{CPs,Calls,map,ER\}$, 其中,

(1) CPs 是核心编排过程的集合;

(2) $Calls=\cup_{cp\in CPs}cp.Call$ 是调用编排的集合;

(3) $map:Calls\to CPs$ 是一个单射函数,用于将调用编排映射为一个核心编排过程;

(4) $ER=\{(cp,cp')\subseteq CP\times CP\,|\,\exists calls\in cp.Call, map(call)=cp'\}$ 是一个嵌入关系,其中,cp'是 cp 的一个子核心编排过程.

函数 map 和关系 ER 用于描述 BPMN 2.0 编排中嵌入的编排过程.在 BPMN 2.0 编排中,建模者可以使用不同的调用编排调用同一个核心编排过程.

为了定义良构编排过程,对于编排过程中的流对象元素 $o\in O$,需引入两个辅助函数.o 的输入节点定义为 $in(x)=\{x\,|\,x\in O,(x,o)\in F\}$; o 的输出节点定义为 $out(x)=\{x\,|\,x\in O,(o,x)\in F\}$. 此外,编排活动 x 的接收消息表示为 $rec(x)$; x 的发送消息表示为 $send(x)$.

定义 7(良构编排过程). 设 $CP=\{O,CA,E,G,F\}$ 是一个核心编排过程,CP 是良构核心编排过程需同时满足下述条件.

(1) $\forall e\in E^S, in(e)=\varnothing\wedge|out(e)|=1$, 即开始事件有且只有一个输出流,而无输入流;

(2) $\forall e\in E^E, |in(e)|=1\wedge out(e)=\varnothing$, 即结束事件有且只有一个输入流,而无输出流;

(3) $\forall x\in CA\cup E^I, |in(x)|=1\wedge|out(x)|=1$, 即编排活动或中间事件只有一个输入流和输出流;

(4) $\forall x\in CA, rec(x)\neq\varnothing\wedge send(x)=\varnothing\cup rec(x)=\varnothing\wedge send(x)\neq\varnothing$, 即编排活动只表示单向的发送或接收消息;

(5) $\forall x\in G^{PF}\cup G^{XD}\cup G^{ED}, |in(x)|=1\wedge|out(x)|>1$, 即并行分叉网关、排他数据决策网关、排他事件决策网关只能有一个输入流,多于一个的输出流;

(6) $\forall x\in G^{PJ}\cup G^{DM}\cup G^{EM}, |in(x)|>1\wedge|out(x)|=1$, 即并行汇聚网关、排他数据合并网关、排他事件合并网关只能有一个输出流,多于一个的输入流;

(7) $\forall x\in G,\exists y\in CA\wedge(x,y)\in F\cup\exists y\in CA\wedge(y,x)\in F$, 即网关必须与编排活动相连.

定义 8(良构编排). 设 $Chor=\{CPs,Calls,map,ER\}$ 是一个编排,$Chor$ 是良构编排当且仅当 CPs 的每个核心编排过程是良构的,且 ER 是反对称的.

2.2 转换规则

若一个编排不满足良构编排的特征,可使用图 2 所示的 7 条规则对其转换,使之变成良构编排.

(1) 规则 1:若开始事件具有多个输出流,可通过与具有多个输出流的并行分叉网关相连,使之转换成一个具有一个输出流的开始事件;

(2) 规则 2:若结束事件有多个输入流,可通过与具有多个输入流的排他数据(事件)合并网关相连,使之转换

成一个具有一个输入流的结束事件;

(3) 规则 3:若编排活动或中间事件具有多个输入流,可通过与具有多个输入流的排他数据(事件)合并网关相连,使之转换成一个具有一个输入流的编排活动或中间事件;

(4) 规则 4:若编排活动或中间事件具有多个输出流,可通过与具有多个输出流的并行分叉网关相连,使之转换成一个具有一个输出流的编排活动或中间事件;

(5) 规则 5:若编排任务表示双向消息,可将该编排任务转换为两个紧邻的编排任务,其中,前驱编排任务表示单向发送消息,后继编排任务表示单向接收消息;

(6) 规则 6:若网关具有多个输入流和多个输出流,可将该网关转换为两个紧邻的同类网关相连,其中,前驱网关与多个输入流相连,后继网关与多个输出流相连;

(7) 规则 7:若网关间通过序列流直接相连,可在网关间增加一个编排任务.通常,网关间的直接相连可以分为 4 种情况:并行网关与并行网关间的直接相连、排他网关与排他网关间的相连、并行网关与排他网关间的相连、排他网关与并行网关间的相连.为了简化和理解的方便,图 2 所示规则 7 中并没有显示区分排他数据网关和排他事件网关,两者均用菱形框表示.

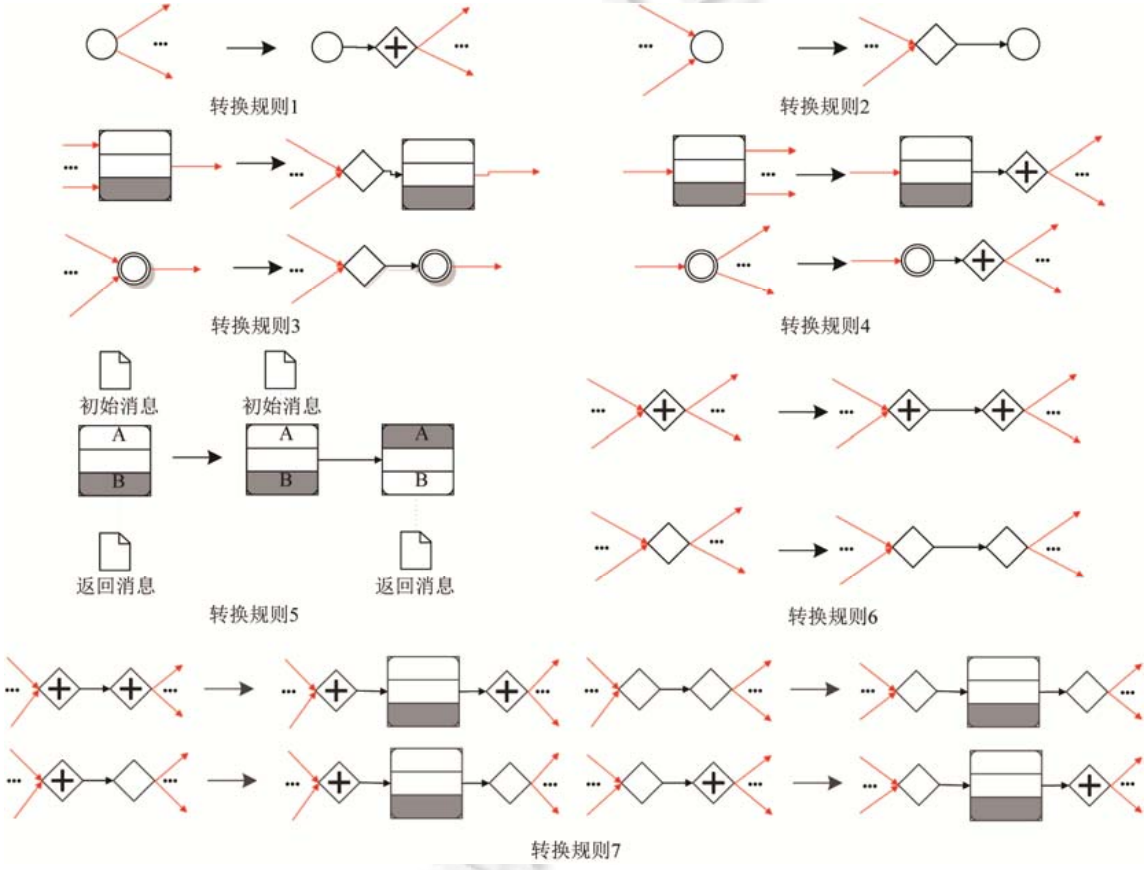

Fig.2 Transformation rules

图 2 转换规则

由定义 4 可知,构成编排的基本建模元素是:开始事件、结束事件、编排活动(包含编排任务)、中间事件和网关.也就是说,任意一个编排均是由这些建模元素组合而成;结构不同的编排体现为建模元素的组合方式不同.另一方面,上述 7 条转换规则的作用是将这些基本建模元素的输入流或输出流进行规范化处理.因而,对任何

一个编排,我们总能唯一求出组成该编排的基本建模元素集,并按照上述规则对这些基本建模元素依次进行转换,最终得到良构编排.

3 编排到工作流网的映射

下面将定义良构编排到工作流网的映射.在工作流网中,我们把变迁元素分为两类:标号变迁和空变迁.其中,标号变迁用于建模编排中的编排活动和事件;空变迁用于建模编排中的网关路由行为,记为 tau.这些路由行为在工作流网执行时将被视为空变迁,不产生任何可视效果.

3.1 编排任务、事件、网关和序列流

图 3 描述了 BPMN 2.0 编排中事件(开始事件、中间事件、结束事件)、编排任务、序列流及网关(并行网关、排他数据网关、排他事件网关)到 Petri 网片段的映射.

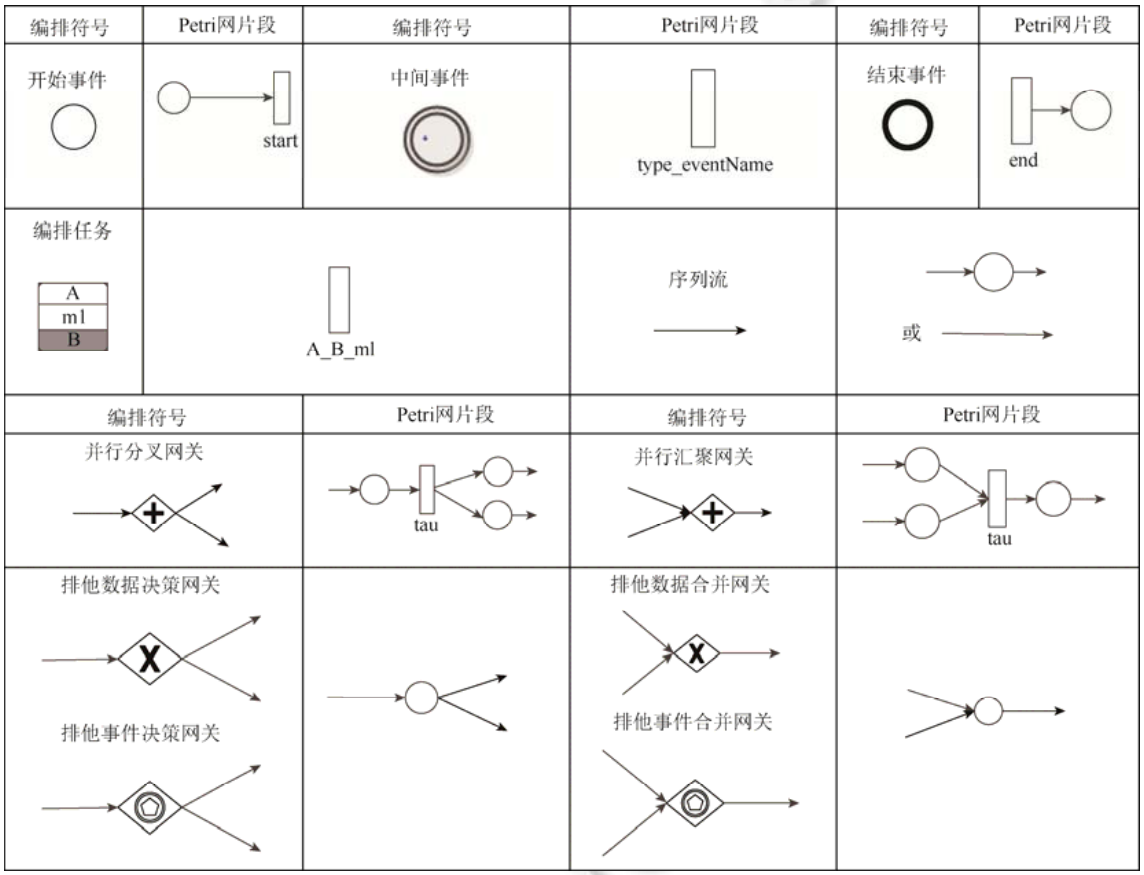

Fig.3 Mapping events, choreography tasks, sequence flow and gateways onto Petri-net segments

图 3 事件、编排任务、序列流和网关到 Petri 网片段的映射

事件的映射包含 3 种情况:(1) 开始事件被映射为 Petri 网中的一个库所、一个标号变迁及一个流关系,且将该标号变迁的标号命名为 start.映射过程不区分开始事件的类型,均将其映射为标号变迁.(2) 中间事件被映射为 Petri 网中的一个标号变迁,且将该标号变迁的标号命名为:类型_中间事件名.映射过程不区分中间事件的类型,均将其映射为标号变迁.(3) 结束事件被映射为 Petri 网中的一个库所、一个标号变迁及一个流关系,且该标号变迁的标号命名为 end.映射过程不区分结束事件的类型,均将其映射为标号变迁.

编排任务被映射为 Petri 网中的一个标号变迁,且该变迁的标号命名采用:发送者_接收者_编排任务名的格式.

序列流的映射分为两种情况:(1) 与排他数据(事件)网关相关的序列流被映射为 Petri 网中的一个库所和两个流关系;(2) 与排他数据(事件)网关无关的序列流被映射为一个流关系.

网关映射分两种情况:(1) 并行分叉网关和并行汇聚网关分别被映射为 Petri 网中带空变迁的 Petri 网片段,以描述该网关的并行路由行为.(2) 排他数据(事件)决策网关和排他数据(事件)合并网关被分别映射为 Petri 网中的一个库所,以描述该网关的选择路由行为.此外,映射过程不区分排他数据网关和排他事件网关.

3.2 标准循环的编排任务、多实例并行的编排任务和多实例串行的编排任务

标准循环的编排任务用以表示编排任务的重复执行,存在两种情况:一种表示编排任务 0 至多次的重复执行,称为 While-do 循环,如图 4(a2)所示;另一种表示编排任务 1 至多次的重复执行,称为 do-until 循环,如图 4(a3)所示.While-do 循环映射为的 Petri 网片段如图 4(a4)所示;do-until 循环映射为的 Petri 网片段如图 4(a5)所示.

多实例并行的编排任务用以表示编排任务实例的并行执行,其实例数目(n)应在设计时指定.事实上,一个多实例并行的编排任务可由同一编排任务的 n 个拷贝及并行分叉网关与并行汇聚网关与之相连所代替,如图 4(b2)所示.多实例并行的编排任务映射为的 Petri 网片段如图 4(b3)所示.

多实例串行的编排任务用以表示多个编排任务实例的串行执行,其实例数目(n)应在设计时指定.事实上,一个多实例串行的编排任务可由同一编排任务的 n 个拷贝及序列流与之相连所代替,如图 4(c2)所示.多实例串行的编排任务映射为的 Petri 网片段如图 4(c3)所示.

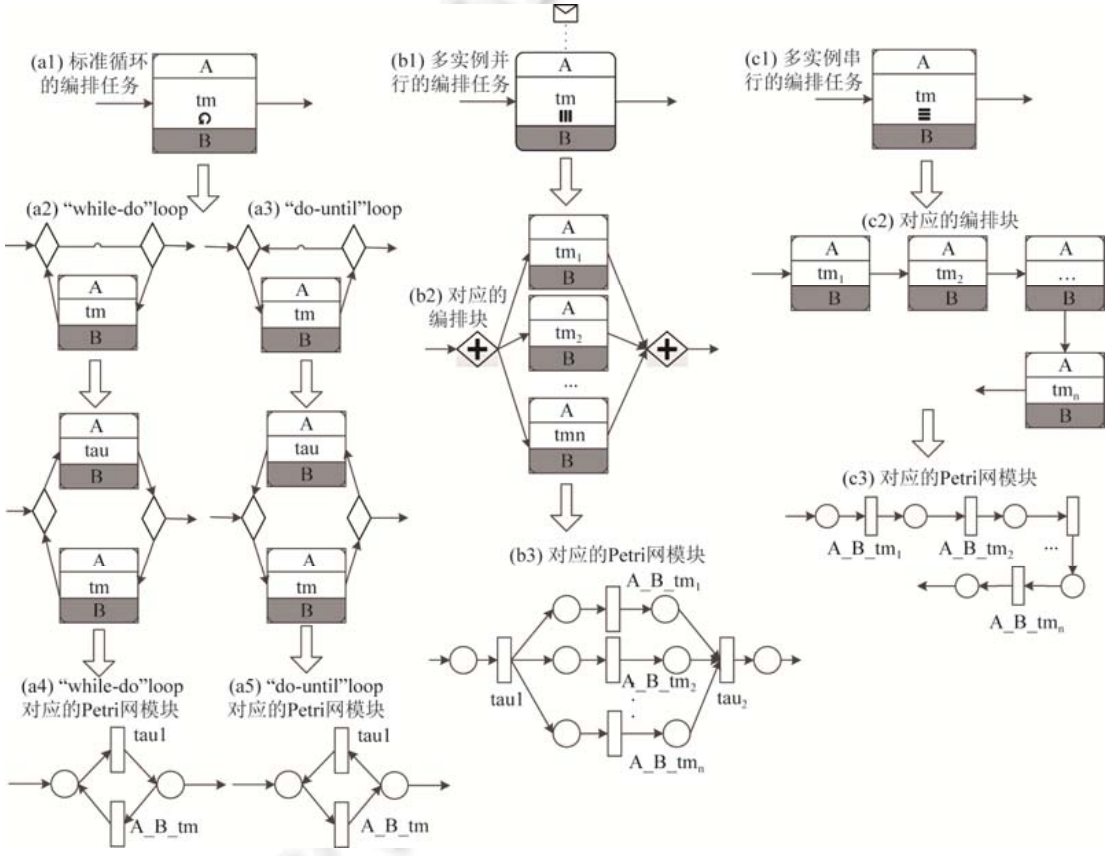

Fig.4 Mapping standard loop choreography task, parallel multi-instance choreography task, and sequential multi-instance choreography task onto Petri-Net segments

图 4 标准循环的编排任务、多实例并行的编排任务和多实例串行的编排任务到 Petri 网片段的映射

3.3 子编排

子编排可视为一个独立的编排.图 5 描述了将一个具有单一开始事件和单一结束事件的子编排映射为 Petri 网片段的过程.在 Petri 网片段中,sub_start 变迁用于建模子编排的开始,sub_end 变迁用于建模子编排的结束,变迁 A_B_m1 用于建模编排任务 m1,变迁 B_A_m2 用于建模编排任务 m2.

子编排可以具有多个开始事件和结束事件,但为了简化,我们将子编排限制为只能具有单一的开始事件和结束事件.

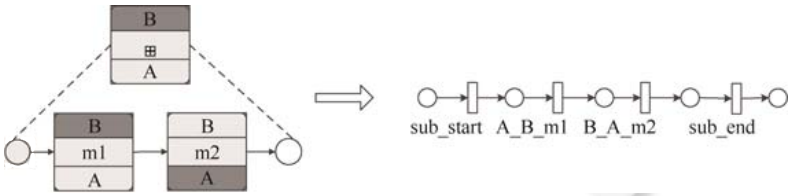

Fig.5 Mapping a sub-choreography onto Petri-net segments

图 5 子编排到 Petri 网片段的映射

3.4 初始标识配置

BPMN 2.0 编排的初始状态可用对应 Petri 网的初始状态来描述.配置初始状态的基本思想是:对顶层编排中每个开始事件对应的库所中添加一个托肯,对子编排中每个开始事件对应的库所中不添加托肯,如图 6 所示.

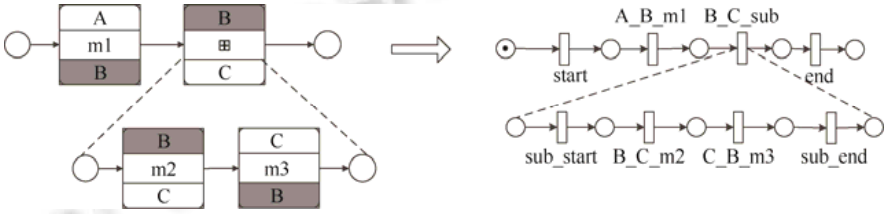

Fig.6 Configuring initial marking of choreographies

图 6 配置编排初始标识

3.5 映射的形式定义

本节将形式化地定义 BPMN 2.0 编排到工作流网的映射,见定义 9.

定义 9(BPMN 2.0 编排的形式语义). 设 $Chor=\{CPs,Calls,map,ER\}$是一个良构编排,$Chor$被映射为一个工作流网$(P,T;F,M)$,其中,

$$
\begin{aligned}
P=\cup_{cp\in CPs}\ (&\{p_s \mid e\in cp.E^s\}\cup && -\text{开始事件}\\
&\{p_e \mid e\in cp.E^E\}\cup && -\text{结束事件}\\
&\{p_{(x,y)} \mid (x,y)\in cp.F\wedge x\notin cp.G^{XD}\wedge x\notin cp.G^{DM}\wedge x\notin cp.G^{ED}\wedge x\notin cp.G^{EM}\wedge y\notin cp.G^{XD}\\
&\wedge y\notin cp.G^{DM}\wedge y\notin cp.G^{ED}\wedge y\notin cp.G^{EM}\}\cup && -\text{与排他数据(事件)网关无关的序列流}\\
&\{p_g \mid g\in cp.G^{XD}\vee g\in cp.G^{DM}\vee g\in cp.G^{ED}\vee g\in cp.G^{EM}\}) && -\text{排他数据(事件)网关}
\end{aligned}
\tag{1}
$$

$$
\begin{aligned}
T=\cup_{cp\in CPs}\ (&\{t_{start} \mid e\in cp.E^s\}\cup && -\text{开始事件}\\
&\{t_{end} \mid e\in cp.E^E\}\cup && -\text{结束事件}\\
&\{t_{type\_e} \mid e\in cp.E^I\}\cup && -\text{中间事件}\\
&\{t_{tau} \mid g\in cp.G^{PF}\cup cp.G^{PJ}\}\cup && -\text{并行网关}\\
&\{t_{sender\_receiver\_a} \mid a\in cp.CT\}\cup && -\text{编排任务}\\
&\{t_{call} \mid g\in cp.Call\}) && -\text{调用编排}
\end{aligned}
\tag{2}
$$

$$F=\cup_{cp\in CPs}(\{(p_s,t_{\text{start}})\mid e\in cp.E^S\}\cup \quad -\text{开始事件}$$
$$\{(t_{end},p_e)\mid e\in cp.E^E\}\cup \quad -\text{结束事件}$$
$$\{(x,p_{(x,y)}),(p_{(x,y)},y)\mid (x,y)\in cp.F\wedge x\notin cp.G^{XD}\wedge x\notin cp.G^{DM}\wedge x\notin cp.G^{ED}\wedge x\notin cp.G^{EM}$$
$$\wedge y\notin cp.G^{XD}\wedge y\notin cp.G^{DM}\wedge y\notin cp.G^{ED}\wedge y\notin cp.G^{EM}\}\cup \quad -\text{与排他数据(事件)网关无关的序列流}$$
$$\{(x,y)\mid ((x,y)\in cp.F\wedge y\in cp.G^{XD}\cup cp.G^{DM}\cup cp.G^{ED}\cup cp.G^{EM})\vee$$
$$((x,y)\in cp.F\wedge x\in cp.G^{XD}\cup cp.G^{DM}\cup cp.G^{ED}\cup cp.G^{EM})\}) \quad -\text{与排他数据(事件)网关相关的序列流} \tag{3}$$

$$M(Ps)=1 \quad -\text{每个开始事件对应生成的库所中添加1个托肯} \tag{4}$$

在上述的形式定义中,(1)定义库所将由开始事件、结束事件、与排他数据(事件)网关无关的序列流及排他数据(事件)网关映射而来;(2)定义变迁将由开始事件、结束事件、中间事件、并行网关、编排任务及调用编排映射而来,其中,并行网关映射产生是空变迁;(3)定义流关系将由开始事件、结束事件、与排他数据(事件)网关无关的序列流及与排他数据(事件)网关有关的序列流映射而来.(4)定义托肯将由开始事件对应生成的库所映射而来.需要特别注意的是,良构编排中不会出现网关与网关直接相连.所以,(3)中在定义与排他数据(事件)网关有关的序列流时,只需说明序列流的一端与排他数据(事件)网关有关即可.

4 分析 BPMN 2.0 编排

第 4.1 节描述工具的设计与实现,用于将 BPMN 2.0 编排自动映射为工作流网.第 4.2 节讨论 BPMN 2.0 编排中存在的问题.第 4.3 节讨论如何使用 Petri 网的分析技术对编排进行语义分析.

4.1 工具设计与实现

图 7 是我们实现的转换工具 Chor2PetriNet 的结构.其中,YaoQiang BPMN Editor 是一个图形化的编排编辑器,用以创建编排模型;Pre-processor 用以对编排模型进行预处理,将语义相同但结构不同的编排模型处理为良构编排模型;Transformer 根据第 3 节提出的映射,用以将良构编排模型转换为工作流网.该工具以基于 XML 格式的 BPMN 文件作为输入,以基于 XML 格式的 PNML 文件(Petri net markup language,简称 PNML)[8]作为输出.PNML 是存储 Petri 网模型的标准文件格式,其好处是可被多个 Petri 网的建模和分析工具所支持.

Fig.7 Structure of Chor2PetriNet

图 7 Chor2PetriNet 的结构

4.2 编排中存在的问题

在将 BPMN 2.0 编排映射为工作流网的过程中,我们发现了 BPMN 2.0 编排中存在的结构方面和控制流方面的问题.结构方面的问题是指编排无开始和结束或违背了 BPMN 2.0 标准规约中对编排的结构约束;控制流方面的问题是指编排会出现死锁、活锁和死任务.下面对这些问题进行讨论.

(1) 编排缺少开始事件或结束事件.BPMN 2.0 标准规约规定:开始事件用来表示编排的开始;结束事件用来表示编排的结束.若编排缺少开始事件,则编排不能直观、显式地描述业务流程间的交互从何处开始;若编排缺少结束事件,则编排不能直观、显式地描述业务流程间的交互在何处结束.图 8 所示为没有开始事件和结束事件的编排.

(2) 编排违反了编排活动序列的基本规则(the basic rule of choreography activity sequencing).由于编排缺少集中机制,因而编排中的参与者只能通过消息的接收或消息的发送来知道编排的进展.为此,BPMN 2.0 标准规约规定:编排中的编排任务必须满足编排活动序列的基本规则,即每个后继编排任务(除首编排任务外)的消息发送者必须为前驱编排任务的参与者.否则,该编排任务将不知道前驱编排任务是否已经执行.图 9 所示为违反

了编排活动序列基本规则的编排.

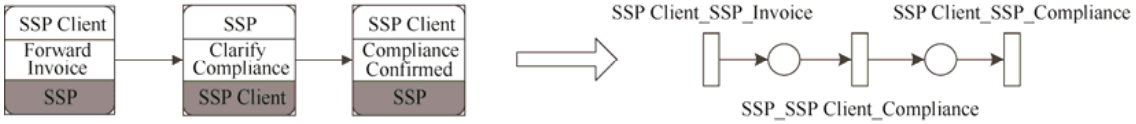

Fig.8 A choreography without start events or end events and the corresponding WF-net

图 8 无开始事件或结束事件的编排及对应的工作流网

Fig.9 A choreography violating the basic rule of choreography activity sequencing and the corresponding WF-net

图 9 违反了编排活动序列的基本规则的编排及对应的工作流网

(3) 编排中存在网关类型不匹配问题.在 BPMN 2.0 编排中,并行网关与排他网关的混用将带来流的增加或流的减少.图 10(a)所示为并行网关和排他网关混用带来流的增加.图 10(b)所示为排他网关和并行网关混用带来流的减少.

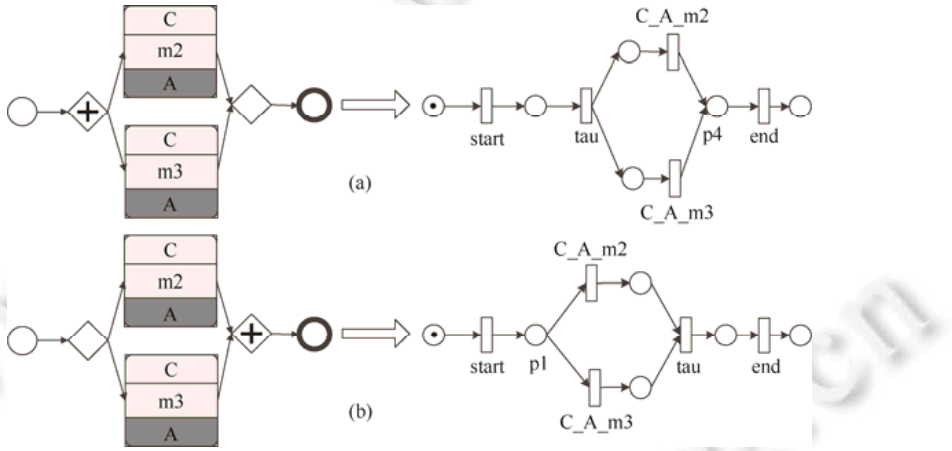

Fig.10 Choreographies with mixed gateways and the corresponding WF-nets

图 10 网关混用的编排及其对应的工作流网

(4) 编排存在死锁、活锁和死任务.编排存在永远无法执行的编排任务,即死编排任务.图 11 所示的编排存在无法执行的编排任务.对于编排中出现死锁和活锁的情况,这里不再举例.

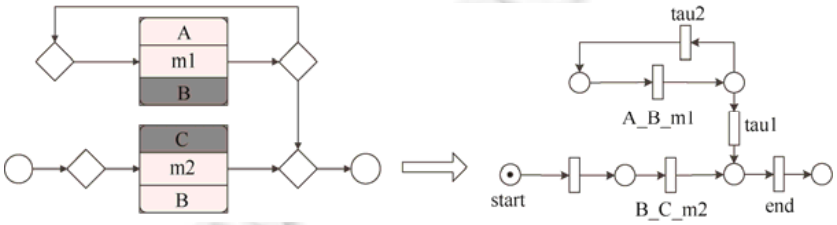

Fig.11 A choreography with a dead choreography task and the corresponding WF-net

图 11 存在死编排任务的编排及其对应的工作流网

4.3 分 析

针对上述编排中存在的 4 类问题,我们将这些问题规约为工作流网的结构问题或性质问题,并使用 Petri 网的分析技术对其进行检测.

针对问题(1):编排缺少开始事件或结束事件,将此类编排转换映射为工作流网后,可把此类问题规约为检测对应的工作流网中是否存在变迁的前集为空或变迁的后集为空的问题.在图 8 所示的工作流网中,变迁 SSP Client_SSP_Invoice 的前集为空,则表示该编排缺少开始事件;变迁 SSP Client_SSP_Compliance 的后集为空,则表示该编排缺少结束事件.

针对问题(2):编排违反了编排活动序列的基本规则,将此类编排转换映射为工作流网后,可把此类问题规约为检测对应的工作流网中相邻标号变迁间的可见性问题,即后继标号变迁的标号名字中的发送者必须出现在前驱标号变迁的标号名字中,或是发送者,或是接收者.在图 9 所示的工作流网中,后继标号变迁 C_A_m2 的发送者为 C,而前驱标号变迁的参与者是 A 和 B,则表示该编排的编排任务违反了编排活动序列的基本规则.

针对(3):编排中网关类型不匹配,将此类编排转换映射为工作流网后,可把此类问题规约为检测对应的工作流网中是否存在 P-T 结构或 T-P 结构的问题.在图 10 所示的工作流网(a)中,变迁 tau 和库所 p4 形成了 T-P 结构,则表示该编排中存在网关类型不匹配;在图 10 所示的工作流网(b)中,库所 p1 和变迁 tau 形成了 P-T 结构,则表示该编排中存在网关类型不匹配.

针对(4):编排存在死锁、活锁和死任务,将此类编排转换映射为工作流网后,可把此类问题规约为检测对应工作流网的合理性问题.在图 11 所示的工作流网中,变迁 A_B_m1 是死变迁,则表示该编排中存在死编排任务,即为不合理的编排.

进一步地,可将上述规约为的工作流网的结构问题或性质问题转换为对工作流网的结构分析或性质分析问题.其中,结构分析包括:检测工作流网中是否存在变迁的前集为空或变迁的后集为空、检测工作流网中相邻标号变迁间的可见性及检测工作流网中是否存在 P-T 结构或 T-P 结构.性质分析包括:检测对应的工作流网是否合理.

对于检测工作流网中是否存在 P-T 结构或 T-P 结构及工作流网是否合理这一问题,可把 BPMN 2.0 编排经 Chor2PetriNet 自动映射产生的 PNML 文件作为一个 Petri 网验证工具:WoPeD[9]的输入,使用 WoPeD 对其进行检测.

对于检测工作流网中是否存在变迁的前集为空或变迁的后集为空这一问题,可把 BPMN 2.0 编排经 Chor2PetriNet 自动映射产生的 PNML 文件作为算法 1 的输入,使用算法 1 对其进行检测.

对于检测工作流网中相邻标号变迁间的可见性问题,可把 BPMN 2.0 编排经 Chor2PetriNet 自动映射产生的 PNML 文件作为算法 2 的输入,使用算法 2 对其进行检测.

算法 1. 检测变迁.给定一个 PNML 文件,输出该文件中是否存在前集为空的变迁或后集为空的变迁.

算法名:checking_transition.

输入:PNML 文件;

输出:该工作流网中存在前集为空的变迁或该工作流网中存在后集为空的变迁或该工作流网不存在前集和后集为空的变迁.

```
BEGIN
    解析 PNML 文件;
    读取标签为⟨transition⟩的元素放入集合 T 中;
    读取标签为⟨arc⟩的元素放入集合 ARC 中;
    WHILE T 中存在未被标识的元素
      BEGIN
        从 T 中任选一个未被标识的元素 t 并标识它;
        WHILE ARC 中存在未被标识的元素
```

```
BEGIN
  从 ARC 中任选一个未被标识的元素 arc 并标识它;
  IF arc 的源点为 t THEN
    source_flag=true;
  IF arc 的目标点为 t THEN
    target_flag=true;
END
IF source_flag!=true THEN
   BEGIN
      输出:该工作流网中存在后集为空的变迁;
      Break;
   END
IF target_flag!=true THEN
   BEGIN
      输出:该工作流网中存在前集为空的变迁;
      Break;
   END
END
IF source_flag==true∧target_flag==true THEN
  输出:该工作流网不存在前集和后集为空的变迁;
END
```

例 1:以图 8 右边所示的工作流网作为算法 1 的输入,算法 1 将检测出:变迁 SSP Client_SSP_Invoice 的前集为空,变迁 SSP Client_SSP_Compliance 的后集为空.因而,原编排缺少开始事件和结束事件.

算法 2. 检测相邻标号变迁间的可见性.给定一个 PNML 文件,输出该文件中是否存在违反编排活动序列基本规则的变迁.

算法名:checking_ visibility.

输入:PNML 文件;

输出:该工作流网中存在违反编排活动序列基本规则的变迁或该工作流网不存在违反编排活动序列基本规则的变迁.

```
BEGIN
   解析 PNML 文件;
   读取标签为⟨transition⟩的元素放入集合 T 中;
   WHILE T 中未被标识的元素
     BEGIN
      从 T 中任选一个未被标识的元素 t 并标识它;
计算变迁 t 的后集的后集,记为 t••;
WHILE t••中未被标识的元素
       BEGIN
         从 t••中任选一个未被标识的元素 t′并标识它;
         IF t′是标号变迁∧t′的发送者不是 t 的发送者或接收者 THEN
         BEGIN
           输出:该变迁 t′违反了编排活动序列的基本规则;
```

```
                t_flag=true;
                BREAK;
            END
        END
        IF t_flag!=true THEN
            输出:该工作流网不存在违反编排活动序列基本规则的变迁;
    END
```

例 2:以图 9 右边所示的工作流网作为算法 2 的输入,算法 2 将检测出:变迁 C_A_m2 的发送者为 C,其前驱变迁 A_B_m1 的发送者为 A、接收者为 B,变迁 C_A_m2 的发送者不是前驱变迁 A_B_m1 的参与者.因而,原编排违反了编排活动序列的基本规则.

5 实 验

5.1 转换工具的性能及语义分析结果

本文将使用 BPM AI 过程模型库[10](http://bpt.hpi.uni-potsdam.de/BPMAcademicInitiative)作为实验评估编排模型的数据源.BPM AI 是由工业界和学术界合作创建的用于科学研究的流程建模平台,合作组织的研究人员和学生使用平台提供的在线建模工具创建了大量的真实编排模型.我们共获得 422 个编排模型,通过将相似和相同的编排模型过滤掉及人工方式的筛选,最终选取了 10 个具有代表性的编排模型.

使用 Chor2PetriNet 工具对上述 10 个编排模型进行映射转换,得到了对应的工作流网,见表 1.运行该工具的笔记本电脑的软/硬件配置如下.

(1) 内存 4G;

(2) 操作系统:Windows 7 旗舰版 64;

(3) 处理器:Intel(R) Core(TM) i5-3470 CPU @ 3.20GHz;

(4) JDK:1.8.0_65;

(5) Eclipse: Mars.1 Release (4.5.1).

Table 1 Evaluation results of Chor2PetriNet

表 1 Chor2PetriNet 的实验评估结果

序号	编排				工作流网			转换时间(ms)	分析结果
	编排任务	事件	排他数据(事件)网关	并行网关	库所	标号变迁	空变迁		
1	6	1	1	0	6	7	0	98	无结束事件
2	11	3	2	7	29	14	11	127	网关类型不匹配
3	17	2	1	0	21	19	2	136	无开始事件
4	4	0	0	0	9	8	2	1003	无开始和结束事件
5	8	5	3	0	32	19	10	119	正确的编排
6	29	14	14	4	49	43	13	196	网关类型不匹配
7	7	2	1	0	9	9	0	104	违反编排活动序列的基本规则
8	8	2	3	0	21	13	8	109	正确的编排
9	11	4	6	1	22	15	9	136	网关类型不匹配
10	10	3	1	0	17	17	0	129	无开始事件;违反编排活动序列的基本规则

表 1 第 1 列为序号.该序号所对应的 BPM AI 过程模型中的编排文件名,见表 2.表 1 第 2 列显示了每个测试编排中编排任务、事件、排他数据(事件)网关及并行网关的数量.表 1 的第 3 列显示了经工具映射转换后得到的工作流网,该工作流网具有的库所、标号变迁及空变迁的数量.表 1 的第 4 列显示了映射转换的时间(单位为 ms).表 1 的第 5 列显示了对编排的分析结果.

Table 2 The mapping between the numbers in the Table 1 and choreography file names

表 2 表 1 中的序号与编排文件名间映射

序号	对应的文件名	序号	对应的文件名
1	2012-02-23_ABC RFQ choreography	2	2011-05-10_Plan and book trip choreography shared
3	2011-03-02_Fulfillment process-choreography diagram	4	2011-04-15_Test choreography diagram
5	2011-03-30_Choreography diagram scenario	6	2011-10-22_Assignment 2-choreography diagram
7	2012-02-23_ABC RFQ choreography diagram	8	2011-08-08_Choreography diagram scenar
9	2011-10-24_SSP choreography	10	2011-10-24_Chorigraph_withall

需要注意的是,(1) 转换时间主要包括两部分:预处理时间和映射转换时间.其中,预处理时间用于将编排转换为良构编排;映射转换时间用于根据映射定义,将编排映射为 Petri 网片段.(2) 在 BPM AI 过程模型库中,经分析我们发现,编排容易产生的语义是:缺少开始事件或结束事件、违反了编排活动序列的基本规则及编排中网关类型不匹配,存在死编排任务的编排不常见.例如,序号 1、3、4、10 对应的编排缺少开始事件或结束事件;序号 2、6、9 对应的编排存在网关类型不匹配;序号 7、10 对应的编排违反了编排活动序列的基本规则.

例 3:图 12 中黑色箭头左边所示是表 1 中序号 7 对应的编排,图 12 中黑色箭头右边所示是映射产生的工作流网.在工作流网中,变迁 start 对应的是开始事件,变迁 end 对应的是结束事件,变迁 Customer_Sales_verfiy 对应的是编排任务 Veriy RFQ,变迁 Sales_Customer_ask 对应的是编排任务 Ask for More Information,变迁 Customer_Sales_provide 对应的是编排任务 Provide Clarification,变迁 Engineering_Sales_analyze 对应的是编排任务 Analyze the RFQ and Provide L&M cost Estimates,变迁 Finance_Sales_add 对应的是编排任务 Add overhead costs and generate pricing portions,变迁 Sales_Customer_generate 对应的是编排任务 Generate Sales Quote,变迁 Customer_Sales_receive 对应的是编排任务 Receive Sales Quote_,库所 p2 由排他决策网关生成.经语义分析后,该编排违反了编排活动序列的基本规则,其原因在于:在工作流网中,变迁 Engineering_Sales_analyze 的发送者为 Engineering,其前驱变迁 Customer_Sales_verfiy 的发送者为 Customer、接收者为 Sales,变迁 Engineering_Sales_analyze 的发送者不是前驱变迁的参与者.

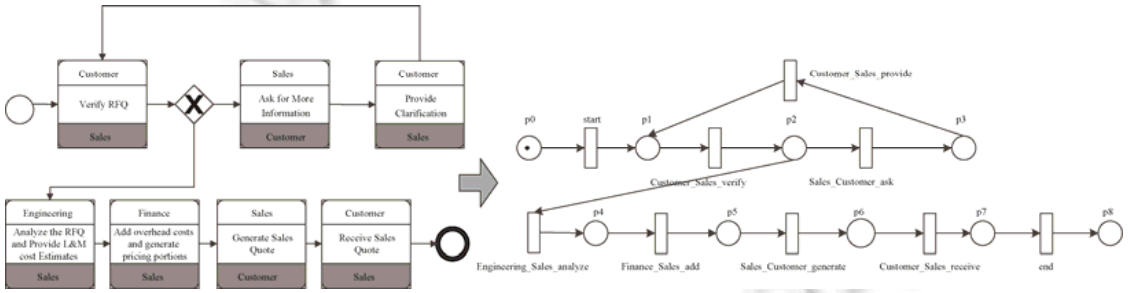

Fig.12 An example of BPMN 2.0 choreography and transformation to WF-net

图 12 BPMN 2.0 编排到工作流网转换的例子

5.2 测试映射规则的一致性

本文第 3 节给出了编排到工作流网映射的直观描述和形式定义,但没有讨论映射规则的正确性.讨论映射规则的正确性的关键是证明 BPMN 2.0 编排与映射生成的工作流网间行为等价.本节将对映射规则的一致性进行测试,具体思路如图 12 所示.第 1 步,将 BPMN 2.0 编排使用 Chor2PetriNet 工具生成对应的工作流网;第 2 步,根据 BPMN 2.0 标准规约中对编排的语义描述,采用人工方式生成 BPMN 2.0 编排对应的标号迁移系统;第 3 步,将映射生成的工作流网,使用 WoPeD 工具生成对应的可达图;第 4 步,将标号迁移系统和可达图作为第 5 步的输入;第 5 步,测试标号迁移系统与可达图间是否满足弱互模拟关系,若满足,则该编排和工作流网是行为等价的;否则,行为不等价.更多关于标号迁移系统和弱互模拟的内容,请参考文献[11].

上述测试映射规则一致性的思路是从弱互模拟关系的角度,检测 BPMN 2.0 编排与映射生成的工作流网间

是否满足行为等价.之所以选择弱互模拟关系作为行为等价检测的标准,而不采用迹等价或强互模拟关系,原因有二:一是,映射产生的工作流网中具有 tau 变迁,对观察者而言,该变迁是透明的、不可见的.例如,将并行分叉网关和并行汇聚网关映射产生的 Petri 网片段中便会出现 tau 变迁.由于强互模拟关系无法比较不可见动作,因此,本文不采用强互模拟关系.二是,迹等价不区分选择的时机,会将不确定性的可达图等价为确定性的可达图,未考虑编排在某状态下的所有分支结构.因此,本文也不采用迹等价.

针对表 1 中列举的编排及其映射生成的工作流网,我们通过人工分析的方式对映射规则进行了一致性测试.测试结果表明,第 3 节提出的映射规则是一致性的.

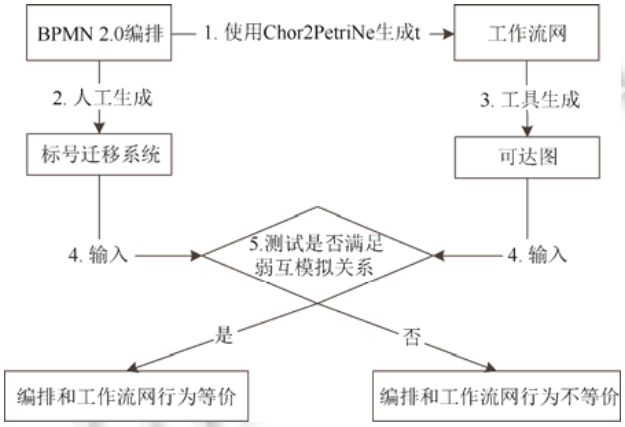

Fig.13 Testing conformance of the mapping rules

图 13 测试映射规则的一致性

6 相关工作

相关工作将从 BPMN 标准规约中编制与编排的形式语义定义、编排建模语言的形式语义定义及编排的语义分析这 3 个方面进行对比.

6.1 编制与编排的形式语义定义

BPMN 2.0 标准规约使用自然语言定义了编制的执行语义[1].但是,用自然语义定义的编制的执行语义,无法直接实现为工具,用于对编制进行模拟、验证和执行[12,13].为此,文献[2,12−18]使用不同的形式化方法定义了 BPMN 编制的形式语义.

文献[2]使用 Petri 网形式定义了 BPMN 1.0 编制子集的语义.它提出了 BPMN 中建模符号到 Petri 网元素的映射关系.这样做的好处是:可以使用 Petri 网丰富的分析技术对 BPMN 进行合理性检测.

文献[12]使用图重写规则(graph rewriting rule)形式定义了 BPMN 2.0 编制子集的执行语义.该文献建立了 BPMN 2.0 标准中非形式定义编制的执行语义与用图重写规则形式定义的编制的执行语义间的直接联系.这样做的好处是:每个形式定义规则的正确性都可以通过 BPMN 2.0 标准中非形式的规约来验证.此外,该形式化的执行语义还可视作一个测试套件,用来测试实现 BPMN 执行语义的工具的一致性.作为该文献工作的扩展,文献[13]使用图重写规则形式定义了更大的 BPMN 2.0 编制子集的执行语义.

文献[14,15]使用进程代数 CSP 形式定义了 BPMN 1.0 编制子集的语义.这样做的好处是:建模者可以使用形式语义检测不同抽象层次业务流程模型间的一致性.具体而言,一方面,建模者可以对业务流程模型领域相关的时序性质进行检测,例如订单提交后,一个响应必须在 24 小时内发送到客户端;另一方面,建模者可以对业务流程模型的通用性质进行检测,例如:无死锁(deadlock-freeness)和正常结束(proper completion).通常,在业务流程管理领域,把对无死锁性质和正常结束性质的检测称为合理性检测(soundness checking).

文献[16]使用 YAWL 定义了 BPMN 1.0 编制子集的语义.它提出了 BPMN 元素到 YAWL 元素的映射关系.

这样做的好处是:建模者可以使用 YAWL 的验证技术对业务流程模型的合理性和活性进行检测.

文献[17]使用进程代数 COWS(calculus of orchestration of Web services)形式定义了 BPMN 1.0 编制子集的语义.这样做的好处是:基于形式语义,建模者不仅可以对业务流程模型进行合理性检测,还可以对业务流程模型(只要业务流程模型具有模拟信息)进行定量模拟.与文献[2,14,15]相比,该文献从控制流和数据流两个方面定义了 BPMN 1.0 子集的语义.

文献[18]使用线性时序逻辑 LTL(linear temporal logic)形式定义了 BPMN 1.2 编制子集的执行语义.这样做的好处是:一方面,在 LTL 框架下,建模者可以对业务流程模型的时序性质进行检测;另一方面,还可以为 BPMN 1.2 标准提供形式的语义规约.

文献[19]提出了一种模型转换的方法,将企业流程建模语言 TiPLM(Tsinghua InfoTech product lifecycle management solution)定义的工作流流程转换为行为等价的 Petri 网,并使用 WoPeD 对 Petri 网进行了可靠性验证.TiPLM 是一种类似 BPMN 编制的流程建模语言.

与上述文献相比,本文关注的是 BPMN 2.0 编排的形式语义定义,而非编制的形式语义定义.需要特别注意的是,BPMN 1.X(BPMN 1.0、BPMN 1.1 和 BPMN 1.2)中只有编制模型的概念,没有编排模型.表 3 从对象、方法、用途、范围这 4 个方面,将本文工作与上述工作进行了对比和总结.本文定义的形式语义不仅可作为 BPMN 2.0 编排的语义规约,用于消除二义性和不一致,还可作为形式化分析的基础,支持编排的语义分析.具体比较如下.

与文献[12–17]相比,本文所做工作的区别在于:(1) 关注对象.本文关注的建模语言是 BPMN 2.0 编排,而这些文献关注的建模语言是编制.(2) 使用的形式化方法不同.本文使用 Petri 网定义编排的形式语义,而这些文献使用的是其他形式化方法定义编制的形式语义,如 CSP、YAWL、COWS、LTL、graph rewriting rule.

Table 3 Semantics of BPMN orchestrations

表 3 BPMN 编制的语义

文献	对象	方法	用途	范围
BPMN 2.0 标准[1]	BPMN 2.0 的编制	自然语言	语义规约	全部
文献[2]	BPMN 1.0 的编制	Petri nets	语义规约,合理性检测	控制流子集
文献[12,13]	BPMN 2.0 的编制	Graph rewriting rule	语义规约,一致性检测	控制流子集
文献[14,15]	BPMN 1.0 的编制	CSP	时序性质检测,合理性检测	控制流子集
文献[16]	BPMN 1.0 的编制	YAWL	合理性检测,活性检测	控制流子集
文献[17]	BPMN 1.0 的编制	COWS	合理性检测,定量模拟	控制流子集,数据流子集
文献[18]	BPMN 1.2 的编制	LTL	时序性质检测,语义规约	控制流子集
文献[19]	TiPLM 的工作流流程	Petri nets	合理性检测	控制流子集
本文	BPMN 2.0 的编排	Petri nets	语义规约,语义分析	控制流子集

与本文工作最为相似的是文献[2]和文献[19]所做的工作.文献[2]使用 Petri 网形式定义了编制的语义,与之相比,本文所做工作的区别在于:(1) 本文定义开始事件、结束事件、中间事件、排他数据(事件)网关和并发网关到 Petri 网片段的映射方式与文献[2]不同;(2) 文献[2]使用库所融合的方式将映射得到的 Petri 网模块组合为一个 Petri 网,而本文使用流关系将映射得到的 Petri 网片段组合为一个 Petri 网.这样做的好处是,本文映射产生的 Petri 网具有更少的库所.(3) 文献[2]中没有单独定义流关系到 Petri 网片段的映射,而本文考虑了这一点.文献[19]使用 Petri 网形式定义了 TiPLM 工作流流程的形式语义,并使用 WoPeD 对 Petri 网进行了合理性验证,与之相比,本文所做工作的区别在于:(1) 在 TiPLM 工作流流程中,建模元素有 4 种:开始事件、结束事件、任务和网关;而本文定义的编排中,建模元素有 5 种:开始事件、结束事件、编排活动、中间事件和网关,其中,编排活动包含编排任务和调用编排;(2) TiPLM 工作流流程没有层次任务(hierarchical task),不考虑层次性;而本文定义的编排可以具有层次结构的子编排.

6.2 编排建模语言的形式语义定义

为定义编排,大量编排建模语言被提了出来,例如:BPMN 2.0[1]、协作图(collaboration diagram,简称 CD)[20]、Web 服务编排规约语言(Web services choreography description language,简称 WS-CDL)[3]、Let's Dance[21]、IPN(interaction Petri nets)[22]、会话协议(conservation protocol,简称 CP)[23].这些编排建模语言可分为两类:具有

形式语义的编排建模语言和不具有形式语义的编排建模语言.具有形式语义的编排建模语言包括:CD、IPN 和会话协议.针对不具有形式语义的编排建模语言,文献[1,24–26]使用不同的方法定义了编排的形式语义.

为了支持编排的定义,与 BPMN 1.X(BPMN 1.0、BPMN 1.1 和 BPMN 1.2)相比,BPMN 2.0 首次提出了编排的概念,将其视为“头等公民”[1].BPMN 2.0 标准规约中用自然语言描述了 BPMN 2.0 编排的语义,但是用自然语义描述的语义存在二义性和不一致.

文献[24]使用 Pi 演算定义了 Let’s Dance 的形式语义.这样做的好处是:基于形式语义,建模者可以把对编排的可达性(reachability)进行分析的问题规约为判断两个 Pi 演算进程间是否满足弱互模拟关系的问题.

文献[25]使用时间自动机(timed automata)形式定义了 WS-CDL 子集的操作语义.这样做的好处是:基于操作语义,建模者可以使用 UPPAAL 工具来对编排进行各种性质的检测,例如:不变量、可达性、前条件和后条件(pre and post conditions)、隐含属性(implication properties)和时间限制(time restriction).

文献[26]针对互连式编排建模存在的不足,借鉴交互式编排建模的思想,将 BPMN 1.1 扩展为 iBPMN,并使用交互式 Petri 网(IPN)形式定义了 iBPMN 的执行语义.这样做的好处是:一方面,建模者可以避免对编排解释的二义性;另一方面,可以使用 Petri 网的分析技术对编排模型进行合理性分析.

文献[27]针对 BPMN 2.0 编排,使用进程代数 LOTOS NT 定义了其形式语义.这样做的好处是:基于形式语义,可以对 BPMN 2.0 进行可实现性(realizability)分析.

与上述文献相比,本文关注的是 BPMN 2.0 编排的形式语义定义.表 4 从编排建模语言、方法、用途、范围这 4 个方面,将本文工作与上述工作进行了对比和总结.文献[24–26]分别针对的编排建模语言是:Let’s Dance、WS-CDL、iBPMN,而本文关注的是 BPMN 2.0 编排.具体比较如下.

与文献[24–26]相比,本文所做工作的区别在于:关注的编排建模语言不同.本文关注的编排建模语言是 BPMN 2.0 编排,而这些文献分别关注的是其他编排建模语言:Let’s Dance、WS-CDL 和 iBPMN.

与本文工作最为相似的是文献[1]和文献[27]所做的工作,文献[1]使用自然语言描述 BPMN 2.0 编排的语义,会产生二义性和不一致;而本文使用 Petri 网,准确定义了 BPMN 2.0 编排的语义.文献[27]使用进程代数 LOTOS NT 定义了 BPMN 2.0 编排的语义,而本文使用 Petri 网定义 BPMN 2.0 编排的语义.与文献[27]所做工作的区别在于:(1) 文献[27]定义编排语义时,限制网关必须是平衡的,即若一个网关有 n 个输入分支,则应有一个对应的网关有 n 个输出分支,本文无此限制;(2) 文献[27]没有考虑构造结构的多样性.(3) 文献[27]未考虑标准循环的编排任务和子编排的形式语义定义.

Table 4 Semantics of choreographies

表 4 编排的语义

编排的语义	编排建模语言	方法	用途	范围
BPMN 2.0 标准[1]	BPMN 2.0	自然语言	语义规约	全部
文献[24]	Let’s Dance	Pi calculus	可达性检测	控制流子集
文献[25]	WS-CDL	Timed automata	不变量 可达性 前条件和后条件隐含属性 时间限制	控制流子集
文献[26]	iBPMN	Petri nets	语义规约 合理性检测	控制流子集
文献[27]	BPMN 2.0 编排	进程代数 LOTOS NT	语义分析	控制流子集
本文	BPMN 2.0 编排	Petri nets	语义规约 语义分析	控制流子集

6.3 编排的语义分析

编排作为参与者协同的全局规约,需将其合成为每个参与者,才能用于指导参与者的实现.但是,合成的参与者未必总能准确实现编排[28].进一步地,给定一个编排,能否将全局编排映射为每个参与者并确保这些参与者间的交互精确匹配流程编排,这就是可实现性分析的内涵[4].为此,文献[4,22,23,29–34]对编排进行了可实现性分析.这些方法大致可以分为 3 类:基于自动机的可实现性分析方法、基于进程代数的可实现性分析方法和基

于 Petri 网的可实现性分析方法.

基于自动机的可实现性分析方法.针对基于会话协议定义的编排,文献[23]以自动机作为形式化基础,最早提出了会话协议可实现需满足的 3 个充分条件:无损连接(lossless join)、同步兼容(synchronous compatible)、自治性(autonomous).在此基础上,文献[29−31]针对会话协议的子类,提出了同步性条件(synchronizability conditions),并将会话协议的可实现性分析转换为对同步性条件的检验.文献[32]针对任意发起者的会话协议(arbitrary-initiator protocols),提出了一个新的充分条件,用于检验会话协议的可实现性.文献[33]提出了会话协议可实现的充分必要条件,将可实现性分析转换为对同步性和合适性(well-formedness)的检测,从而可使用现有的行为等价检测工具和模型检测工具,对会话协议的可实现性进行自动检测.

基于进程代数的可实现性分析方法.针对基于协作图定义的编排子类,文献[4]以进程代数 LOTOS NT 作为形式化基础,提出一种将协作图定义的编排形式转换为基于 LOTOS NT 定义的进程表达式的方法,从强互模拟的角度,检验了同步通信和异步通信环境下协作图的可实现性.针对 BPMN 2.0 中的编排子类,文献[27]以进程代数 LOTOS NT 作为形式化基础,提出一种把编排形式转换为 LOTOS NT 定义的进程表达式的方法,从强互模拟的角度,对同步通信环境和异步通信环境下编排的可实现性进行检验.在此基础上,文献[34]提出一种模块化的框架,用于自动验证 WSDL、会话协议和 BPMN 2.0 编排的可实现性.

基于 Petri 网的可实现性分析方法.文献[22]以 Petri 网作为形式化基础,用 IPN 定义编排,并从分支互模拟的角度,对同步通信环境下 IPN 的可实现性进行了分析.但 Internet 环境下存在网络延迟,因此必须对异步通信环境下编排的可实现性进行分析.但是,该方法只能分析同步通信环境下 IPN 的可实现性,不能对异步通信环境下 IPN 的可实现性进行分析.

与上述文献相比,本文的语义分析更关注从控制流的角度,分析 BPMN 2.0 编排表现出由面向图流程定义产生的语义错误,如流增加、流减少、死编排任务等.这样做的原因在于:从控制流方面,确保编排的语义正确性是讨论可实现性的前提.因为,可实现性分析本质上是讨论整体编排与局部参与者间的语义一致性.若整体编排的正确性无法保证,则讨论整体编排与局部参与者间的语义将无任何实际意义.此外,本文的语义分析还考虑了 BPMN 2.0 标准规约中对编排的结构约束:编排活动序列的基本规则.

7 结束语

BPMN 2.0 标准规约中编排缺少形式语义及相应的分析技术,这将阻碍对 BPMN 2.0 编排的语义分析.首先,通过建立 BPMN 2.0 编排到工作流网的映射,使用 Petri 网准确定义编排的语义;其次,借助 Petri 网的分析技术,把 BPMN 2.0 编排中存在的语义错误规约为工作流网的结构问题或性质问题,对 BPMN 2.0 编排进行语义分析;最后,通过实验表明,这种形式化可以识别 BPMN 2.0 编排中存在的错误.

本文并没有考虑复杂网关和包含网关的语义定义.下一步的工作重点是:提出 BPMN 2.0 编排到参与者的映射算法、对 BPMN 2.0 编排进行可实现性分析以及根据 Petri 网中发现的问题,自动地将其对应到原编排.

References:

[1] OMG. Business Process Model and Notation (BPMN) Version 2.0. 2011. http://www.omg.org/spec/BPMN/2.0/

[2] Dijkman RM, Dumas M, Ouyang C. Semantics and analysis of business process models in BPMN. Information & Software Technology, 2008,50(12):1281–1294. [doi: 10.1016/j.infsof.2008.02.006]

[3] W3C. Web Service Choreography Description Language (WS-CDL). 2005. http://www.w3.org/TR/ws-cdl-10/

[4] Salaün G, Bultan T, Roohi N. Realizability of choreographies using process algebra encodings. IEEE Trans. on Services Computing, 2012,5(3):290–302. [doi: 10.1109/TSC.2011.9]

[5] Petri CA. Kommunikation mit automaten [Ph.D. Thesis]. Institut fur Instrumentelle Mathematik, Schriften des IIM 2, 1962.

[6] Yuan CY. The Principle and Application of Petri Nets. Beijing: Electronic Industry Publishing House, 2005 (in Chinese).

[7] van der Aalst WMP. The application of Petri nets to workflow management. Journal of Circuits, Systems, and Computers, 1998,8(1):21–66. [doi: 10.1142/S0218126698000043]

[8] Billington J, Christensen S, van Hee KV, Kindler E, Kummer O, Petrucci L, Post R, Stehno C, Weber M. The Petri net markup language: Concepts, technology, and tools. In: Proc. of the Int'l Conf. on Applications and Theory of Petri Nets. Berlin: Springer-Verlag, 2003. 483–505. [doi: 10.1007/3-540-44919-1_31]

[9] Eckleder A, Freytag T. WoPeD 2.0 goes BPEL 2.0. In: Proc. of the German Workshop on Algorithms and TOOLS for Petri Nets, Algorithmen Und Werkzeuge Für Petrinetze, Awpn 2008. Rostock, 2008. 75–80.

[10] BPM academic initiative. http://bpt.hpi.uni-potsdam.de/BPMAcademicInitiative/

[11] Milner R. Communicating and Mobile Systems: The Pi-Calculus. Cambridge: Cambridge University Press, 1999.

[12] Dijkman R, Gorp PV. BPMN 2.0 execution semantics formalized as graph rewrite rules. Lecture Notes in Business Information Processing, 2010,67:16–30. [doi: 10.1007/978-3-642-16298-5_4]

[13] Gorp PV, Dijkman R. A visual token-based formalization of BPMN 2.0 based on in-place transformations. Information & Software Technology, 2013,55(55):365–394. [doi: 10.1016/j.infsof.2012.08.014]

[14] Wong PYH, Gibbons J. A process semantics for BPMN. In: Proc. of the Int'l Conf. on Formal Engineering Methods. Berlin: Springer-Verlag, 2008. 355–374. [doi: 10.1007/978-3-540-88194-0_22]

[15] Wong PYH, Gibbons J. Formalisations and applications of BPMN. Science of Computer Programming, 2011,76(8):633–650. [doi: 10.1016/j.scico.2009.09.010]

[16] Ye JH, Sun SX, Song W, Wen L. Formal semantics of BPMN process models using YAWL. In: Proc. of the Int'l Symp. on Intelligent Information Technology Application. Washiongton: IEEE, 2008. 70–74. [doi: 10.1109/IITA.2008.68]

[17] Prandi D, Quaglia P, Zannone N. Formal analysis of BPMN via a translation into COWS. In: Proc. of the 10th Int'l Conf. on Coordination Models and Languages. Berlin: Springer-Verlag, 2008. 249–263. [doi: 10.1007/978-3-540-68265-3_16]

[18] Lam VSW. A precise execution semantics for BPMN. Int'l Journal of Computer Science, 2012,39(1):20–33.

[19] Zha H, Aalst WMPVD, Wang J, Wen L, Sun J. Verifying workflow processes: A transformation-based approach. Software & Systems Modeling, 2011,10(2):253–264. [doi: 10.1007/s10270-010-0149-9]

[20] Bultan T, Fu X. Specification of realizable service conversations using collaboration diagrams. Service Oriented Computing and Applications, 2008,2(1):27–39. [doi: 10.1007/s11761-008-0022-7]

[21] Zaha JM, Barros A, Dumas M, Hofstede AT. Let's Dance: A language for service behavior modeling. In: Proc. of the 14th Int'l Conf. on Cooperative Information Systems. Berlin: Springer-Verlag, 2006. 145–162. [doi: 10.1007/11914853_10]

[22] Decker G, Weske M. Local enforceability in interaction Petri nets. In: Business Process Management. Berlin, Heidelberg: Springer-Verlag, 2007. 305–319. [doi: 10.1007/978-3-540-75183-0_22]

[23] Fu X, Bultan T, Su J. Conversation protocols: A formalism for specification and verification of reactive electronic services. Theoretical Computer Science, 2004,328(1):19–37. [doi: 10.1016/j.tcs.2004.07.004]

[24] Decker G, Zaha JM, Dumas M. Execution semantics for service choreographies. Lecture Notes in Computer Science, 2006,34(3): 163–177. [doi: 10.1007/11841197_11]

[25] Cambronero ME, Díaz G, Valero V, Martínez E. Validation and verification of Web services choreographies by using timed automata. Journal of Logic & Algebraic Programming, 2011,80(1):25–49. [doi: 10.1016/j.jlap.2010.02.001]

[26] Decker G, Weske M. Interaction centric modeling of process choreographies. Information Systems, 2011,36:292–312. [doi: 10.1016/j.is.2010.06.005]

[27] Poizat P, Salaün G. Checking the realizability of BPMN 2.0 choreographies. In: Proc. of the 27th Annual ACM Symp. on Applied Computing. Riva del Garda, 2011. 1927–1934. [doi: 10.1145/2245276.2232095]

[28] Roohi N, Salaün G, France V. Realizability and dynamic reconfiguration of chor specifications. Informatica: An Int'l Journal of Computing and Informatics, 2011,35(1):39–49.

[29] Basu S, Bultan T. Choreography conformance via synchronizability. In: Proc. of the 20th Int'l Conf. on World Wide Web. Hyderabad, 2011. 795–804. [doi: 10.1145/1963405.1963516]

[30] Basu S, Bultan T, Ouederni M. Synchronizability for verification of asynchronously communicating systems. In: Proc. of the 13th Int'l Conf. on Verification, Model Checking, and Abstract Interpretation. Philadelphia, 2012. 56–71. [doi: 10.1007/978-3-642-27940-9_5]

[31] Güdemann M, Salaün G, Ouederni M. Counterexample guided synthesis of monitors for realizability enforcement. Lecture Notes in Computer Science, 2012,7561:238–253. [doi: 10.1007/978-3-642-33386-6_20]

[32] Hallé S, Bultan T. Realizability analysis for message-based interactions using shared-state projections. In: Proc. of the 18th ACM SIGSOFT Int'l Symp. on Foundations of Software Engineering. Seoul, 2010. 27–36. [doi: 10.1145/1882291.1882298]

[33] Basu S, Bultan T, Ouederni M. Deciding choreography realizability. In: Proc. of the 39th Annual ACM SIGPLAN-SIGACT Symp. on Principles of Programming Languages. Philadelphia, 2012,47(1):191–202. [doi: 10.1145/2103656.2103680]

[34] Güdemann M, Poizat P, Salaün G, Dumont A. A framework for verifying choreographies choreographies. In: Proc. of the 16th Int'l Conf. on Fundamental Approaches to Software Engineering, Rome, 2013. 226–230. [doi: 10.1007/978-3-642-37057-1_16]

附中文参考文献:

[6] 袁崇义.Petri 网原理与应用.北京:电子工业出版社,2005.

代飞(1982－),男,四川乐山人,博士,副教授,CCF 专业会员,主要研究领域为软件工程,业务过程管理.

莫启(1986－),男,博士,讲师,主要研究领域为业务过程管理.

赵文卓(1992－),女,学士,主要研究领域为业务过程管理.

李彤(1963－),男,博士,教授,博士生导师,CCF 高级会员,主要研究领域为软件工程.

杨云(1981－),男,博士,教授,CCF 专业会员,主要研究领域为机器学习,数据挖掘,智能软件工程.

周华(1963－),男,博士,研究员,博士生导师,主要研究领域为软件工程,系统分析与集成.