第35卷 第23期 计 算 机 工 程 2009年12月
Vol.35 No.23 Computer Engineering December 2009
·开发研究与设计技术· 文章编号：1000—3428(2009)23—0274—04 文献标识码：A 中图分类号：TP393

基于Pi演算的BPMN编排模式

杨鹏玉，邱锦伦
(上海大学计算机工程与科学学院，上海 200072)

摘 要：针对业务流程建模标记(BPMN)无法依靠自身对编排进行形式化分析的问题，提出用Pi演算描述BPMN编排模式，实现对BPMN编排的描述。BPMN编排模式是服务交互模式的BPMN表达。实验结果表明，该方法能够找到并排除BPMN编排中的死锁。
关键词：业务流程建模标记；Pi演算；编排

BPMN Choreography Pattern Based on Pi-calculus

YANG Peng-yu, QIU Jin-lun
(School of Computer Engineering and Science, Shanghai University, Shanghai 200072)

【Abstract】Aiming at the problem that Business Process Modeling Notation(BPMN) does not allow for formal analysis, this paper proposes a method to describe BPMN choreography pattern by Pi-calculus to realize describing BPMN choreography. BPMN choreography pattern is service interaction pattern in BPMN. Experimental result shows that this method can find and eliminate the deadlock of BPMN choreography.
【Key words】Business Process Modeling Notation(BPMN); Pi-calculus; choreography

1 概述

业务流程管理系统的实现是以业务流程模型为基础的，因此，业务流程建模对业务流程系统起着关键作用。业务流程建模分为编制与编排。编制与编排的区别在于：编制基于控制流，主要用来对单个企业或实体能够集中控制的流程进行建模；而编排是基于消息交换的，并且有2个以上企业或实体在编排中协作[1]。

业务流程建模技术可以分为形式化方法(如Petri Net、进行代数和有限状态机等)和非形式化方法(如IDEFO等)。业务流程建模标记(Business Process Modeling Notation, BPMN)是一种图形化的业务流程建模工具，应用比较广泛。然而，BPMN不是形式化语言，不能依靠其自身进行形式化分析。因此，文献[2-3]对BPMN模型中的编制进行Pi演算描述。但目前尚没有对BPMN模型编排进行Pi演算形式化的工作。文献[4]提出的服务交互模式是一组细粒度的可用于组合成流程编排的交互模式。

本文通过对服务交互模式的BPMN实现进行Pi演算描述，从而探究Pi演算描述BPMN的编排能力。

2 相关知识和工作

2.1 BPMN及其扩展介绍

BPMN中有4个基本的元素类：流对象，连接对象，泳道，附件。

为了加强对复杂交互场景的建模能力，文献[5]对基本BPMN进行了扩展，增加了图1描述的4种图形元素。

扩展后对以下3种场景建模能力有所加强：

(1)一对多交互。

(2)一对多交互时，“一”方收到来自“多”方的消息，“一”方在稍后的交互中根据发送方分别回复。

(3)三方或更多方交互时，一方把第三方的交互地址转发至另一方，使另一方与第三方能进行交互。

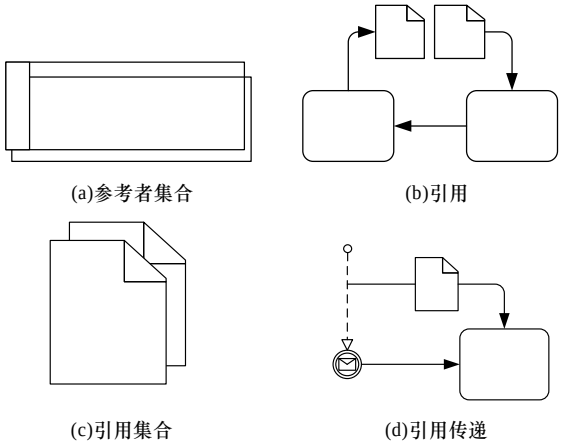


图1 扩展的BPMN图形元素

2.2 Pi演算

Pi演算是在CCS基础上开发的传名演算，它是描述通信拓扑结构动态变化的分布式通信系统。基本计算实体为名字和进程，进程之间的通信是通过传递名字来完成的。它允许进程之间传送和接收通道名，并且可以引入和输出局部名。因此，作为一种形式化方法，Pi演算可以用来描述结构不断变化的并发系统。

(1)语法介绍

设N为无限名字集，用u, v, w, x, y, z等小写字母名字集上的名字；进程标识符用A, B, C等大写字母表示；进程表达式用P, Q, R等大写字母表示，进程表达式如下：

基金项目：上海市学科建设基金资助重点项目(J50103)
作者简介：杨鹏玉(1981－)，男，硕士研究生，主研方向：业务流程建模；邱锦伦，副教授、硕士
收稿日期：2009-06-18 **E-mail：**lionback@163.com

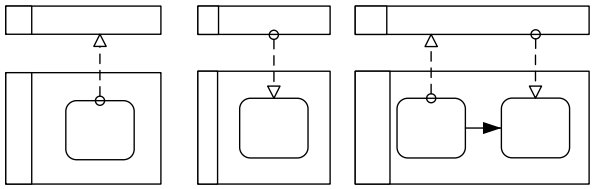
0	空进程
$P_1 \mid P_2$	并发表达式
P_1+P_2	和式表达式
$(v \cdot x)P$ 或 $(x)P$	限制表达式
$[x=y]P$	匹配表达式
$x(y).P$ 或 $x.P$	输出前缀表达式
$\bar{x}<y>.P$ 或 $\bar{x}.P$	输入前缀表达式
$t.P$	哑前缀
$!P$	重复表达式
$A(x_1,x_2,\cdots,x_n)$	进程标志符

3 Pi演算描述服务交互模式的BPMN实现

编排建模是编制流程交互的建模。服务交互模式描述了一组反复出现的编排场景。这些模式涵盖了从简单消息交换到多参与者多消息交换。实际上，BPMN 编排建模是对这些交互模式建模并不断重复组合。下文对各种模式的 BPMN 模型进行 Pi 演算描述。

3.1 单传送双方交互模式

单传送双方交互模式如图 2 所示。



(a)发送　(b)接收　(c)发送/接收

图 2　单传送双方交互模式

单传送双方交互模式具体描述如下：

(1)发送模式，如图 2(a)所示。发送模式指以发送者的角度来描述 2 个参与者间的交互。表达式中 X' 表示后续编排，Pi 演算描述为

$$X(s)=\bar{s}.X'$$

(2)接收模式，如图 2(b)所示。接收模式指以接收者的角度来描述 2 个参与者间的交互。Pi 演算描述为

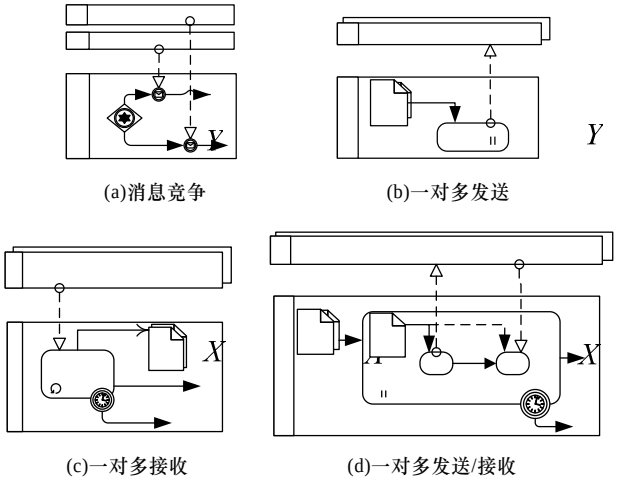
$$X(r)=r.X'$$

(3)发送/接收模式，如图 2(c)所示。Pi 演算描述为

$$X(s,r)=\bar{s}.r.X'$$

3.2 单传送多方交互模式

单传送多方交互模式如图 3 所示。



(a)消息竞争　(b)一对多发送

(c)一对多接收　(d)一对多发送/接收

图 3　单传送多方交互模式

单传送多方交互模式的具体描述如下：

(1)消息竞争

一个参与者 X 等待消息，其他参与者发送多个消息给 X。这些消息竞争到达，只有处理第 1 个到达的消息。如果 r_1 先收到消息则执行 X_1'，并且 r_2. X_2' 不再执行。反之亦然。Pi 演算描述为

$$X(r_1,r_2)=r_1.X_1'+r_2.X_2'$$

(2)一对多发送

流程 X 有 n 个通道$(s_1,s_2,\cdots,s_n)$，这些通道并行发送。设置信号通道 x，发送完成的数量与收到信号 x 的数量相等。后续编排会在所有发送结束后继续。Pi 演算描述为

$$X(s_1,s_2,\cdots,s_n)=(\prod_{j=1}^{n}(\bar{s}_j.\bar{x}.0)) \mid (x.)^n.X'$$

(3)一对多接收

一对多接收时，一个流程接收多个流程发送的消息，发送流程只发送一次。消息必须在要求时间内到达。通道 *enough* 用来判断接收结束。Pi 演算描述为

$$X(r_1,r_2,\cdots,r_n,enough,timeout)=$$
$$(v\,x,y,enough)(\prod_{j=1}^{n}(r_j.0+timeout.\bar{y}0+enough.\bar{x}.0) \mid (x.X'+y.C))$$

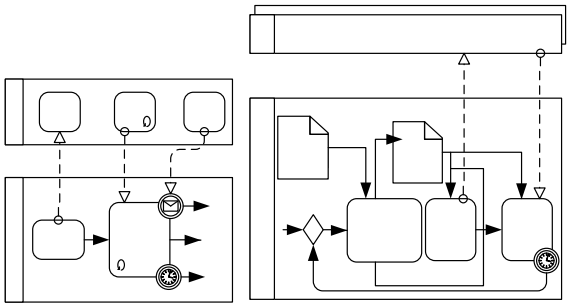
(4)一对多发送/接收

一对多接收/发送与一对多接收类似，只是在接收响应前要先对其他流程发送请求。Pi 演算描述为

$$X(s_1,s_2,\cdots,s_n,r_1,r_2,\cdots,r_n,enough,timeout)=$$
$$(v\,x,y,enough)(\prod_{j=1}^{n}\bar{s}_j.(r_j.0+timeout.\bar{y}0+enough.\bar{x}.0) \mid (x.X'+\mathrm{y}.C))$$

3.3 多传送交互模式

多传送交互模式如图 4 所示。



(a)多响应　(b)随时请求

图 4　多传送交互模式

多传送交互模式具体描述如下：

(1)多响应模式。多响应模式指的是流程 X 向流程 Y 发送一个请求，随后在一定时间范围内流程 X 将从流程 Y 处接收响应直到流程 Y 向 X 发送响应结束信息或超时。Pi 演算描述为

$$\begin{cases} A(r,over,timeout)=(r.A+over.X'+timeout.C) \\ X(s,r,over,timeout)=\bar{s}.A \end{cases}$$

(2)随机请求。随机请求批流程 X 向流程 Y 发送一个请求。如果在一定时间内 Y 没有回应，则再向其他流程发送此请求，直到在规定时间内收到响应。Pi 演算描述为

$$\begin{cases} X(s_1,s_2,\cdots,s_n,r_1,r_2,\cdots,r_n,timeout)=\bar{s}_1.(r_1.X'+timeout.A_2) \\ A_i(s_i,s_{i+1},\cdots,s_n,r_i,r_{i+1},\cdots,r_n,timeout)=\bar{s}_i.(r_i.X'+timeout.A_{i+1}) \\ A_{n+1}=A \quad (1\leqslant i\leqslant n) \end{cases}$$

3.4 路由模式

路由模式具体描述如下：

(1)请求/响应第三方模式

此模式指流程 X 向流程 Y 发送请求，同时发送引用集。流程 Y 按引用集中的通道分别向其他流程 $Z_1,Z_2,\cdots,Z_n$ 进行响应。

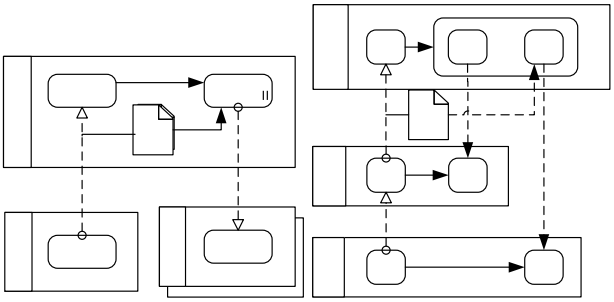
$$\begin{cases} X(s,r_1,r_2,\cdots,r_n)=\overline{s}<r_1,r_2,\cdots,r_n>.0 \\ Y(s)=s(r_1,r_2,\cdots,r_n).(\prod_{j=1}^{n}\overline{r}_j.0) \\ Z_i(r_i)=r_i.0 \quad (1\leqslant i\leqslant n) \end{cases}$$

(2)延迟请求

延迟请求是指流程 X 向 Y 发送请求，流程 Y 为完成 X 的请求，将请求委托给流程 Z。流程 Z 同时响应 X 与 Y。Pi 演算描述为

$$\begin{cases} X(b,p)=(v\ a)\overline{b}<a,p>.a.X' \\ Y(b,x,q)=b(a,x).\overline{x}<a>.q.0 \\ Z(p,q)=p(a).(\overline{q}.0\,|\,\overline{a}.0) \end{cases}$$

路由模式如图 5 所示。



(a)请求/响应第三方　　(b)延迟请求

图 5　路由模式

4　应用

拍卖场景的 BPMN 编排描述如图 6 所示。将上文所述各种模式应用于委托拍卖的编排，会发现委托拍卖的编排是多种模式的组合。将 BPMN 的编排模型分解成后，很容易对它进行 Pi 演算描述，从而对 BPMN 模型进行形式化分析和检查。

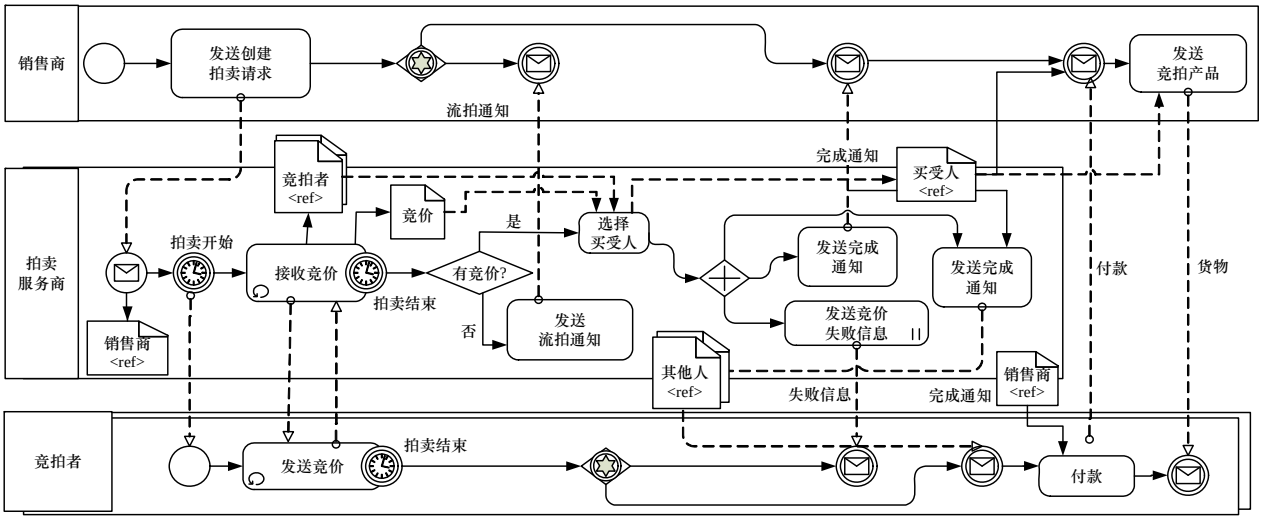


图 6　拍卖场景的 BPMN 编排描述

4.1　编排分析

从图 6 可以看出，由销售商(seller)的发送创建拍卖请求、竞拍者(bidder)竞争价和竞争拍者付款(payment)是一个延迟请求模式。

竞争拍者付款与销售商供货是一个简单的发送/请求模式。销售商收到流拍消息或竞争拍成功消息与竞争拍者收到遗憾消息或竞价成功消息都是消息竞争模式。拍卖服务商(auctionservice)向竞拍者发送当前最高价(currentprice)并等待接收竞价是一对多发送接收模式。

4.2　编排模型的Pi演算描述

为了简化，竞价者人数限定为 2。对各个流程的 Pi 演算描述如下：

seller 的 Pi 演算描述如下：

seller=(v paychannel)$\overline{\text{creatauction}}$<paychannel>.(unsucmess.
seller+compmesss.paychannel(delivadd).$\overline{\text{delivadd}}$.seller)

auctionservice 的 Pi 演算的描述如下：

auctionservice=(v finish)creatauction(paychannel).
(bidreceive|finish.($\overline{\text{unsucmess}}$.auctionservice+
$\overline{\text{compmess}}$.auctionservice|($\overline{\text{fail1}}$.auctionservice+
$\overline{\text{done1}}$<paychannel>.auctionservice)|($\overline{\text{fail2}}$.
auctionservice+$\overline{\text{done2}}$<paychannel>.auctionservice))

bidreceive=$\prod_{i=1}^{n}\overline{\text{start}_i}$.biding(start$_i$,currentprice$_i$,
bid$_i$,fail$_i$,done$_i$,timeout)

biding=$\overline{\text{currentprice}}$.(bid.biding+timeout.$\overline{\text{finish}}$.biding)

bidder 的 Pi 演算的描述如下：

bidder=start.bid
bid=currentprice.$\overline{\text{bid}}$.bid+timeout.result
result=(v delivadd)fail.bidder+done(paychannel).
$\overline{\text{paychannel}}$<delivadd>.delivadd.bidder

为了统一时间，还要创建一个全局时间控制器：

timer='timeout.0|'timeout1.0|'timeout2.0

整个委托拍卖的编排模型描述如下：

auction=(v creatauction,unsucmess,complete,timeout
start1,currentprice1,bid1,fail1,done1,timeout1,
start2,currentprice2,bid2,fail2,done2,timeout2)
(timer|seller|auctionservice
bidder(start1,currentprice1,bid1,fail1,done1,timeout1)|
bidder(start2,currentprice2,bid2,fail2,done2,timeout2))

4.3　验证

业务流程编排的应用是以其正确性为前提，必须保证各个参与者间交互不存在死锁。Pi 演算的分析工具 MWB'99 (Mobility WorkBench)可以通过命令行交互的方式对 Pi 演算

所描述的模型进行分析检查。将对销售商、拍卖服务商、竞拍者和整个系统的 Pi 演算描述转换成 MWB 的语法在 MWB 中进行验证如图 7 所示。

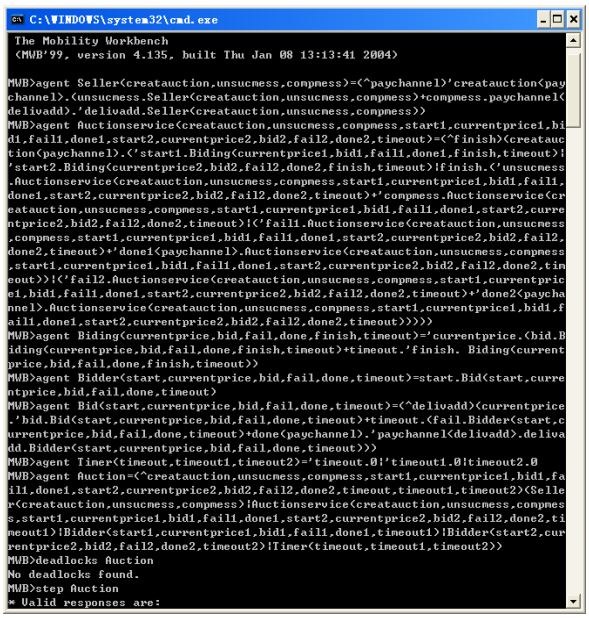


图 7　实验结果

用死锁命令 deadlocks 检查拍卖系统 auction 是否有死锁：结果没死锁。也可用 step 命令检查进程运行过程。这有利于分析、检查整个流程的每一步交互是否与 BPMN 中的编排相符。

5　结束语

本文通过用 Pi 演算描述 BPMN 编排模式来实现对 BPMN 编排的描述。BPMN 编排模式是服务交互模式的 BPMN 表达。具体实例按模式分解分析并进行 Pi 演算描述，实验结果表明，该方法能实现对 BPMN 编排的 Pi 演算描述并进行形式化分析。目前，流程编制的 Pi 演算描述与流程编排的 Pi 演算描述还没整合，如何将其有机整合是下一步研究的重点。

参考文献

[1] Weske M. Business Process Management[M]. Berlin, Germany: Springer-Verlag, 2007.

[2] Prandi D, Quaglia P, Zannon N. Formal Analysis of BPMN via a Translation into COWS[C]//Proc. of COORDINATION'08. Salerno, Italy: Springer-Verlag, 2008.

[3] 李向宁. 业务过程管理理论与若干关键技术研究[D]. 西安: 西北大学, 2007.

[4] Barros A, Dumas M, Hofstede A. Service Interaction Patterns[C]// Proc. of the 3rd International Conference on Business Process Management. Nancy, France: Springer-Verlag, 2005: 302-318.

[5] Decker G, Puhlmann F. Extending BPMN for Modeling Complex Choreography[C]//Proc. of the 15th International Conference on Cooperative Information Systems. Vienna, Italy: Springer- Verlag, 2007: 24-40.

编辑　陆燕菲

（上接第 273 页）

态，搜集由本分区应用引起的故障和错误，通过蓝图定义的策略进行故障与错误过滤，然后向分区内的错误管理模块和核心软件层 GSM 中的健康监控模块通告错误。分区操作系统中的错误管理模块根据配置管理处理、记录本分区的错误，并向核心软件层 GSM 中的错误管理模块通告，以便 GSM 能够对节点的多个故障进行分析和做出正确处理。

节点的安全由核心软件区 GSM 的安全管理(Security Management, SM)模块集中管理。主要负责经过加密、解密和审计的受保护数据能够安全传输，将检测破坏安全的事件写入安全日志[3]。

融合中另一个重要工作是蓝图的融合。ARINC653 使用配置表来管理系统[4]。配置表中包括系统初始化信息，分区间通信的配置信息，用来进行健康监控和错误管理的信息。为了让分区能在操作系统的核心上运行，还要配置分区的内存需求、运行周期、每个周期持续时间、用来发送消息的标识、用来接收消息的标识等信息。此类信息都是静态的。ASAAC 更是通过蓝图来对整个系统进行管理。系统在工作模式的切换、容错管理、地勤维护和测试、系统初始化和关机阶段都需要依靠动态运行蓝图来做出相应变化。本文需要将这种静态的配置表信息融入 ASAAC 的蓝图中以便节点的 GSM 可以通过读取蓝图中的信息对节点进行管理。

5　结束语

综合模块化航空电子系统是现代航空电子系统的发展方向，因此，国外的很多开放式标准得以引入并在国内航空电子系统的设计中得到应用。在学习这些开放式标准时对其加以优化和改进使其更实用、具有更好的性能是一项非常重要的工作。本文对 IMA 系统的可靠性改进的设计正是出于该目的。

参考文献

[1] 王运盛, 陈　颖. ASAAC 航空电子体系结构分析[J]. 电讯技术, 2007, 47(5): 159-162.

[2] 雷　清, 叶　宏. 嵌入式实时操作系统中的安全性考虑[J]. 航空计算技术, 2005, 35(4): 61-65.

[3] 施　刚, 钱泰来. 综合化航空电子的系统管理技术[J]. 计算机工程, 2008, 34(增刊): 43-45.

[4] ARINC Specification 653. AvionicsApplication Software Standard Interface[Z]. (2005-01-21). http://www.arinc.com.

编辑　陆燕菲