ISSN 1000-9825, CODEN RUXUEW
Journal of Software, Vol.19, No.6, June 2008, pp.1363−1378
DOI: 10.3724/SP.J.1001.2008.01363

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

一种基于对象网的多视角软件过程模型*

葛季栋[1,2+], 顾 庆[1,2], 胡 昊[1,2], 吕 建[1,2]

[1](南京大学 计算机软件新技术国家重点实验室,江苏 南京 210093)

[2](南京大学 计算机科学与技术系,江苏 南京 210093)

A Multi-View Software Process Model Based on Object Petri Nets

GE Ji-Dong[1,2+], GU Qing[1,2], HU Hao[1,2], LÜ Jian[1,2]

[1](State Key Laboratory for Novel Software Technology, Nanjing University, Nanjing 210093, China)

[2](Department of Computer Science and Technology, Nanjing University, Nanjing 210093, China)

+ Corresponding author: E-mail: gjd@ics.nju.edu.cn

Ge JD, Gu Q, Hu H, Lü J. A multi-view software process model based on object Petri nets. *Journal of Software*, 2008,19(6):1363−1378. http://www.jos.org.cn/1000-9825/19/1363.htm

Abstract: According to the principle of "Separation of Concerns", by investigating the similarity between multi-view software process modeling and object Petri nets, this paper proposes the MOPN-SP-net model which is a multi-view software process model based on object Petri nets and enhances the reusability of software process model. During process modeling, MOPN-SP-net is a multi-dimensional Petri net, which is difficult to analyze directly. So, this paper provides a translation rule from an object Petri net to an equivalent traditional flat Petri net. The translation preserves the soundness property. According to the translation rule, the soundness property of the MOPN-SP-net can be indirectly analyzed by its translated flat net.

Key words: software process; software process modeling; PSEE; object Petri nets

摘 要: 基于关注点分离原则,结合多视角软件过程建模与对象网之间的相似性,提出了一种基于对象网的多视角软件过程模型 MOPN-SP-net,可以提高软件过程模型的可复用性(reusability).在使用 MOPN-SP-net 建模时,得到的多视角软件过程模型是一种多维网.直接分析这种多维网比较困难,为此,提供了一种从对象网到平面网的等价转换规则,且转换前后的模型在合理性准则上保持等价.根据这个转换规则,可以通过分析转换后平面网模型的合理性得知转换前的对象网的合理性.

关键词: 软件过程;软件过程建模;以过程为中心的软件工程环境;对象网

中图法分类号: TP311 **文献标识码**: A

软件过程管理是保障软件质量的重要手段[1−4].软件质量标准如 CMM/CMMI,ISO/IEC 15504,Bootstrap 等,

* Supported by the National Natural Science Foundation of China under Grant Nos.60233010, 60403014, 60603034 (国家自然科学基金); the National High-Tech Research and Development Plan of China under Grant Nos.2004AA112090, 2005AA113160, 2006AA01Z159, 2006AA01Z177 (国家高技术研究发展计划(863)); the National Basic Research Program of China under Grant No.2002CB312002 (国家重点基础研究发展计划(973))

Received 2006-08-23; Accepted 2007-03-07

都是通过规范软件过程来保障软件产品的质量并提高生产率的.在质量管理的现代理念中,过程、人和技术是保障软件质量的3个重要因素[3,5],软件过程的实施细节决定着软件产品的质量,因此,通过改善软件过程的管理来提高软件质量是一个重要途径.

软件开发是由多人参与的协同工作过程,依靠计算机辅助管理的以过程为中心的软件工程环境 PSEE (process-centered software engineering environment),为软件开发组织提供了一种协同工作环境,可以提高软件开发过程的工作效率,并通过监控软件过程的执行来达到保障软件质量的效果,还可以通过对过程数据的分析与反馈达到改进软件过程的目的.一般来讲,PSEE 系统中都包含一个软件过程建模工具和一个过程引擎.软件过程建模工具一般包含一个软件过程建模语言 PML(process modeling language),过程引擎的设计以软件过程建模语言为依据,因此,软件过程建模语言是 PSEE 系统的核心成分[4,6].软件过程建模一般分为形式化与非形式化两种,为了能够让 PSEE 系统解释执行,一般采用形式化建模的方法.在软件过程的形式化建模方法上存在着两个流派:一个流派侧重于预定义过程的完整描述[4,7,8],另一个流派注重提高软件过程的自适应[9,10].两个流派各有优势,侧重点有所不同.侧重于预定义的软件过程模型可以通过 PSEE 系统规范软件过程在预定义的要求下得到比较严格的执行,这种方式适用于步骤流程比较明确的过程;另一方面,注重提高自适应的建模方法,为过程的执行提供了一种自适应的机制,这种方式应用于流程易变的场景.本文的工作侧重于预定义过程的完整描述.

软件过程模型是对软件过程的抽象描述,由于软件过程的特殊性和复杂性,为了使 PSEE 系统能够有效地管理和调度软件企业的各种资源,软件过程建模需要考虑活动(activity)、产品(product)和角色(role)等多个视角[7,8].传统的软件过程建模语言一般直接将多个视角揉在一起描述,使得复用基于各视角建立的过程模块比较困难.本文基于“关注点分离”原则,结合多视角软件过程建模与对象网[11]之间的相似性,采用多视角分离定义的方法,提出了一种基于对象网的多视角软件过程模型 MOPN-SP-net(MOPN-based software process net).该模型既可以分离定义软件过程的各个视角,又可以通过对象网的交互集将各视角联系起来,各视角之间是一种松耦合的关系,由此可以提高各视角模块的可复用性(reusability)和软件过程建模的灵活性.为了确保软件过程模型能够被正确而有效地执行,合理性(soundness)是需要考虑的重要属性.在使用 MOPN-SP-net 建模时得到的是一种多维网,直接分析这种多维网的合理性比较困难,为此,本文提供了一种从对象网到平面网的等价转换规则,并且转换前后的模型在合理性准则上保持等价.根据这个转换规则,可以通过分析转换后平面网模型的合理性,得知转换前的对象网的合理性.

本文第1节介绍多视角软件过程建模与对象网结构的相似性.第2节提出一种基于对象网的多视角软件过程模型,并给出一个具体的应用实例.第3节给出从对象网到平面网的转换规则以及一个转换实例,并给出定理阐明转换前后的模型在合理性准则上保持等价.最后是相关工作与总结.

1 多视角软件过程建模与对象网结构的相似性

软件过程模型一般包括活动、产品和角色等多个视角[7,8].各视角之间的关系如图1所示.

(1) 活动:活动是软件过程的一个执行步骤.活动可以分为两种,一种是自动化执行的,另一种是需要手工干预执行的.活动可以有子结构,可以分解为子活动的活动称为复合活动,不能分解的活动称为原子活动.活动之间通过控制流明确相互关系,并与产品视角和角色视角有联系.活动视角主要关注软件开发过程中的活动组成及其相互间的约束关系等.

(2) 产品:产品是软件过程的执行过程中产生或修改的中间“原材料”或最终产品.软件过程的产品主要包括各种文档和配置项,如计划文档、设计文档和软件代码等.产品视角主要关注产品文档的操作流程及各个操作下的文档状态等.

(3) 角色:角色描述了完成软件过程中的某个特定活动所要求的职责、权限以及技能.角色视角主要关注每个角色参与的活动集合及不同活动中的权限分配等.在实际的软件过程执行中,执行者(actor)被赋予某个角色.

一个完整的软件过程模型需要包含上述多个视角,为此,本文从考察多视角软件过程建模的特点入手,结合

多元对象网 MOPN(multi-object petri nets)(定义 3)的特点,提出了一种基于对象网的多视角软件过程模型 MOPN-SP-net.

在具体的软件过程建模中,活动视角在整个软件过程模型中处于核心位置[7,8,12],其他视角相对于活动视角处于从属位置,而且活动视角与其他视角之间有交互.根据软件工程中“关注点分离(separation of concerns)”的原则,需要将多个视角分离定义,并定义不同视角之间的交互关系.这个特点与对象网模型的结构[11]相似,对象网模型在结构上分为系统网(system net)与子网(object net)*两层,两层网之间的交互集描述系统网与子网之间的交互关系,在整体上系统网处于核心位置,而子网相对于系统网处于从属位置.因此,基于对象网模型结构与多视角软件过程建模的特点之间的相似性,可以将对象网模型应用于多视角软件过程建模,建立基于对象网的多视角软件过程模型 MOPN-SP-net.如图 2 所示,以活动视角(activity view)与产品视角(product view)的组合建模为例,可以用系统网描述活动视角,用子网描述产品视角,系统网的托肯对象代表着某个产品视角的行为,用两层网之间的交互集描述活动视角与产品视角之间的交互关系.基于这个设想,既可以将活动视角的建模与产品视角的建模分离开来,又可以通过两层网之间的交互集将活动视角与产品视角联系起来,这样使得建立起来的模型结构清晰,层次分明,符合用户的思维习惯.基于关注点分离原则,采用多视角分离定义的方法可以提高各视角模块的可复用性(reusability)和软件过程建模的灵活性.此外,基于这种方法还可以将角色视角加入进来.

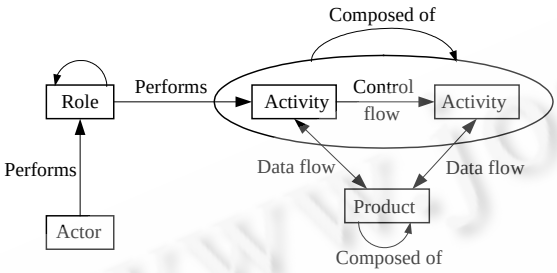


Fig.1 Basic views of software process model

图 1 软件过程模型的基本视角

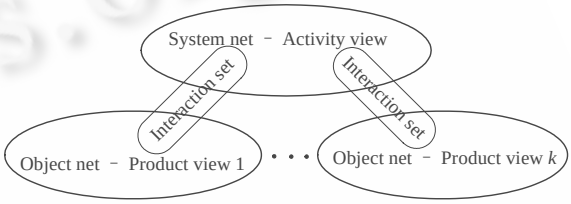


Fig.2 The similarity between multi-view software process modeling and object Petri nets

图 2 多视角软件过程建模与对象网的相似性

2 基于对象网的多视角软件过程建模

2.1 基于Petri网的软件过程建模

Osterweil 提出“Software Processes are Software too”的思想[13,14],即开发软件过程类似于开发软件本身,其中的相似在于,软件过程的控制结构类似于程序设计语言的控制结构.而在控制结构方面,Petri 网具有与程序设计语言同等的表达能力,常用于描述并发分布式系统,而 PSEE 是一种特殊的分布式应用系统,因此,Petri 网可以作为软件过程的建模工具.基于 Petri 网进行过程建模具有一些优点[15]:形式化的语义、直观的图形表示、易于理解、坚实的数学理论基础和成熟的分析技术等.基于 Petri 网的软件过程建模已有一些代表性的研究工作[16−21].SLANG[16]是一种基于特殊的 Petri 网(ER nets)软件过程建模语言,在 SPADE 软件过程支撑环境中得到应用.文献[19]提出了一种 Production Net 模型,用于描述软件配置项(software configuration items)的并发控制.FUNSOFT Net[20]是一种基于谓词网(predicate/transition nets)软件过程建模语言,在 MELMAC 系统中得到应用.文献[21]利用 Petri 网描述一种可视化过程建模语言 VPML(visual process modeling language),在 EPMS (enterprise process modeling system)系统中得到应用.在某种程度上,软件过程是一种特殊的工作流程[4,6,8,20,22],

* 在 Valk 的模型中,Object Petri Nets 模型结构包括 System net 与 Object net 两层,本文为了避免把 Object Petri Nets 与模型结构中的一层 Object net 混淆起来,约定在用中文表述时,整体模型名称 Object Petri Net 译为对象网,模型结构中的一层 Object net 译为子网.

可以用与工作流网类似的模型对软件过程进行建模.下面先介绍本文使用的一些基本概念和符号.

Petri 网是一个三元组 $PN=(P,T,F)$,其中,P 是库所(place)集合,T 是变迁(transition)集合,且 $P\cap T=\varnothing$,$F\subseteq(P\times T)\cup(T\times P)$是库所与变迁之间的弧线集合.$\bullet x=\{y\in P\cup T|(y,x)\in F\}$表示一个库所或者变迁的前集,$x\bullet=\{y\in P\cup T|(x,y)\in F\}$表示一个库所或者变迁的后集,相关的基本概念可以参考文献[23,24].工作流网 WF-net (workflow net)[15]是 Petri 网的一种应用特例,一个 Petri 网 $PN=(P,T,F)$是 WF-net,需要满足以下条件:

(1) PN 有两个特殊的库所:i 与 o.库所 I 是起始库所:$\bullet i=\varnothing$.库所 o 是结束库所:$o\bullet=\varnothing$.

(2) 如果在 PN 中加入一个变迁 t^*,且满足$\bullet t^*=\{o\}$与 $t^*\bullet=\{i\}$,那么所得到的扩展后的 Petri 网 $PN^*=(P,T\cup\{t^*\},F\cup\{(o,t^*),(t^*,i)\})$是强联通的(strongly connected).

将 Petri 网应用于软件过程建模,根据 Petri 网的运行规则,变迁表示软件过程的动作步骤,而托肯及其在库所中的分布状况表示动作步骤能够发生的条件.

基于传统平面 Petri 网的 WF-net 模型具有较强的表达能力,但是,单个传统平面 Petri 网不能直接分离定义软件过程的多个视角,难以直接而清晰地描述多个视角之间的联系,因此,单个平面 Petri 网不能直接描述软件过程的多视角建模的情形.为此,本文结合对象网[11]的特点,提出了一种基于对象网的多视角软件过程模型 MOPN-SP-net.

对象网[11]的结构包含系统网(system net)与子网(object net)两层,系统网的库所相当于存放子网实例的容器,系统网中的托肯指向子网.对象网模型中最重要的概念是“Petri nets as token objects”,将子网看成系统网的一个托肯对象,即用子网去替换系统网中的托肯对象,或者理解为系统网中的托肯带有一个指针指向(reference)某个子网,因此,系统网中的托肯有一些自治行为,而这些行为通过与之对应的子网来描述,这是本文中使用的对象网与传统 Petri 网模型的重要区别.对象网模型中含有层次的概念,层次的概念源于 Petri 网中一些元素的替换,但是关于层次的概念,对象网模型与传统的层次网(hierarchical Petri nets)模型之间有不同的含义.在对象网中,是用子网去替换系统网的托肯对象,而在传统的层次网中,是用一个 Petri 网去替换上一层网的库所或者变迁,所以,为了区别这两种关于“层次”的不同概念,本文将对象网中的层次概念称为维(dimension),这是一种高阶 Petri 网模型,或者叫做多维 Petri 网(multi-dimensional Petri nets),而包括谓词网、着色网和层次网在内的传统 Petri 网模型可以被称为平面网.关于对象网,Lakos[25]也曾经提出过一种模型,虽然都使用“Object Petri Nets”的名词术语,但是,Valk 的模型定义与 Lakos 的对象网模型有严格的区别,本文采用 Valk 的模型和思想.

2.2 一元对象网

关于对象网[11,26],最简单的形态是一元基本对象系统 UEOS(unary elementary object system).定义 1 中关于 UEOS 的形式定义可以体现对象网的上述思想.

定义 1. 一个一元基本对象系统是一个三元组 $UEOS=(SN,ON,\rho)$[11],其中:

(1) $SN=(P,T,F,M_0)$是一个 Petri 网系统,称为 UEOS 的系统网.M_0 为系统网的初始状态,$|M_0|=1$,即在初始状态,系统网中只有一个托肯对象,故称为一元对象网.

(2) $ON=(\hat{P},\hat{T},\hat{F},\hat{M}_0)$是一个 Petri 网系统,称为 UEOS 的子网.$\hat{M}_0$ 为子网的初始状态.

(3) $\rho\subseteq T\times\hat{T}$ 是 SN 与 ON 之间的交互集,用于描述 SN 与 ON 之间的同步交互关系.

图 3 所示为 UEOS 的一个示例.网结构分为两层:下面是系统网 SN,上面是子网 ON,其中,系统网 SN 的初始托肯标于库所 p_1,子网 ON 的初始托肯标于库所 $\hat{p}_1$,虚线箭头表示系统网 SN 中的托肯指引着子网 ON,即当前子网在系统网中所处的状态为库所 p_1.在 SN 和 ON 之间有一组交互集 $\rho=\{(t_2,\hat{t}_1),(t_4,\hat{t}_3),(t_5,\hat{t}_4)\}$.

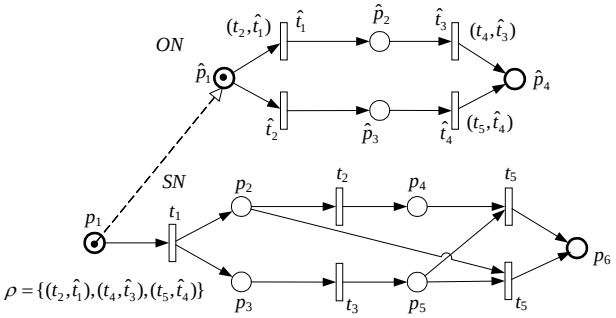


Fig.3 An example of UEOS

图 3 一个 UEOS 示例

UEOS 系统是一种多维 Petri 网,其点火规则(定义 2)不同于传统 Petri 网.

定义 2. *UEOS*=(*SN*,*ON*,ρ)在某个时刻的状态包括两个方面:系统网的状态与子网的状态,采用双标记(bi-marking)表示,记为 $(M,\hat{M})$,这里,M 表示 *SN* 的状态,$\hat{M}$ 表示 *ON* 的状态.设点火后的状态为 $(M',\hat{M}')$,*UEOS* 的点火动作表示为 $(M,\hat{M})\xrightarrow{[t,\hat{t}]}(M',\hat{M}')$,其中,$[t,\hat{t}]$ 是一个序偶,t 表示来自系统网变迁的动作,$\hat{t}$ 表示来自子网变迁的动作,在 $[t,\hat{t}]$ 组合中,允许 t 与 $\hat{t}$ 有一个为空动作 λ,因此,对象网共有以下 3 种点火方式[11]:

(1) 系统网自治点火:对于 $t\in T$,如果 $\rho(t)=\varnothing$ 且 t 在状态 M 下是可激活的(enabled),那么 t 在状态 $(M,\hat{M})$ 下也是可激活的,在 t 单独点火后的状态变化记为 $(M,\hat{M})\xrightarrow{[t,\lambda]}(M',\hat{M}')$,这里,$M\xrightarrow{t}M'$ 且 $\hat{M}=\hat{M}'$.

(2) 子网自治点火:对于 $\hat{t}\in\hat{T}$,如果 $\rho(\hat{t})=\varnothing$ 且 $\hat{t}$ 在状态 $\hat{M}$ 下是可激活的,那么 $\hat{t}$ 在状态 $(M,\hat{M})$ 下也是可激活的,在 $\hat{t}$ 单独点火后的状态变化记为 $(M,\hat{M})\xrightarrow{[\lambda,\hat{t}]}(M,\hat{M}')$,这里 $M=M'$ 且 $\hat{M}\xrightarrow{\hat{t}}\hat{M}'$.

(3) 同步点火:对于一对变迁 $(t,\hat{t})\in T\times\hat{T}$,如果 $(t,\hat{t})$ 是一对交互,即 $(t,\hat{t})\in\rho$,且 t 和 $\hat{t}$ 在状态 M 和 $\hat{M}$ 下分别可激活,那么 $[t,\hat{t}]$ 在状态 $(M,\hat{M})$ 下也是可激活的,在 $[t,\hat{t}]$ 同步点火后的状态变化记为 $(M,\hat{M})\xrightarrow{[t,\hat{t}]}(M',\hat{M}')$,这里,$M\xrightarrow{t}M'$ 与 $\hat{M}\xrightarrow{\hat{t}}\hat{M}'$ 同时发生.

函数 $\rho(t)$ 与 $\rho(\hat{t})$ 的定义分别为 $\rho(t)=\{\hat{t}\in\hat{T}\mid(t,\hat{t})\in\rho\}$ 与 $\rho(\hat{t})=\{t\in T\mid(t,\hat{t})\in\rho\}$.

在定义 2 的点火规则中,系统网自治点火表示只有系统网 *SN* 的变迁发生动作,子网自治点火表示只有子网 *ON* 的变迁发生动作,同步点火表示两层网 *SN* 与 *ON* 之间的同步交互关系.在图 3 的例子中,可能的点火序列有 $[t_1,\lambda],[\lambda,\hat{t}_2],[t_3,\lambda],[t_5,\hat{t}_4]$ 或者 $[t_1,\lambda],[t_2,\hat{t}_1],[t_3,\lambda],[t_4,\hat{t}_3]$.

由上面的介绍可以看出,对象网最重要的特点是可以用系统网与子网分离定义两层的过程模型,并通过系统网与子网之间的交互集将两层的过程模型联系起来,而这个特点与软件过程建模中的多个视角之间的关系相似.因此,基于上述想法,本文提出了一种基于对象网的多视角软件过程模型.

2.3 多元对象网

上面提到的一元对象网 UEOS 的模型定义约定了在系统网中只有一个子网实例,然而在实际软件过程建模中,会遇到一个活动视角对应多个产品视角的情形,为了使对象网能够支持多个产品视角的建模需求,需要定义一种模型,使得一个系统网可以对应多个子网实例,称这种模型为多元对象网 MOPN(multi-object Petri nets).

在定义多元对象网时,本文借鉴 Valk 关于对象网模型的基本思想,将网结构分为两层:系统网与子网.系统网的初始状态有多个托肯对象,且分别映射到多个子网,子网之间没有直接交互.以活动视角与产品视角的组合建模为例,多元对象网模型与多视角软件过程建模的对应关系是:在一个软件过程中,有一个处于核心位置的活动视角和多个处于从属位置的产品视角,活动视角与产品视角之间有交互,各个产品视角相对独立运行.用系统网描述活动视角,用嵌于系统网中的多个子网描述多个产品视角,系统网的托肯对象代表某个产品视角的行为,通过系统网与子网之间的交互集描述活动视角与多个产品视角之间的交互关系.

定义 3. 多元对象网(multi-object Petri net)是一个三元组,*MOPN*=(*SN*,ON_S,ρ).

(1) $SN=(P,T,F)$表示系统网.

(2) $ON_S=\{ON_1,ON_2,\ldots,ON_n\}$表示由 n 个 Petri 网组成子网的集合,n 是有限自然数,$ON_k=(\hat{P}_k,\hat{T}_k,\hat{F}_k)$ 表示编号为 k 的子网,同时也是系统网中编号为 k 的托肯所指引(reference)的对象.子网中所有变迁集合的并集记为 $\hat{T}=\bigcup_{k=1}^{n}\hat{T}_k$,子网中所有库所集合的并集记为 $\hat{P}=\bigcup_{k=1}^{n}\hat{P}_k$.

(3) $\rho_k\subseteq T\times\hat{T}_k$ 表示系统网 SN 与子网 ON_k 的交互集合,$\rho=\bigcup_{k=1}^{n}\rho_k$ 表示系统网与所有子网的交互全集.

(4) $\rho_k(\hat{t})=\{t\in T\mid(t,\hat{t})\in\rho_k\}$ 表示在系统网 SN 中所有与 ON_k 的变迁 $\hat{t}$ 有交互关系的变迁集合,$\rho(t)=\{\hat{t}\in\hat{T}_k\mid(t,\hat{t})\in\rho_k\}$表示在子网 ON_k 中所有与系统网 SN 的变迁 t 有交互关系的变迁集合.

定义 4. 多元对象网 MOPN 的状态表示.

(1) MOPN 模型的结构分为系统网与子网两层.表示这个模型的运行状态,需要同时描述系统网的状态与多个子网的状态,而子网是内嵌于系统网的基本单元.因此,为了便于理解,可以从刻画每个子网的状态的角度描述整个模型的运行状态.在某个时刻,处于系统网中的某个子网 ON_k 的状态同时包含系统状态与自身状态,采用双标记的方式,记为 $ON_k.(M,\hat{M})=(ON_k.M,ON_k.\hat{M})$,其中,$ON_k.M$ 表示子网 ON_k 在系统网中的系统状态,即 ON_k 在系统网中的分布情况;$ON_k.\hat{M}$ 表示子网 ON_k 的自身状态,通过 ON_k 自身的托肯分布来刻画.

(2) 以子网 ON_k 的状态作为基本单元,可以刻画某个时刻 MOPN 的全局运行状态,记为

$$MOPN.Mark=(ON_1.(M,\hat{M}),ON_2.(M,\hat{M}),\ldots,ON_k.(M,\hat{M}),\ldots,ON_n.(M,\hat{M})).$$

定义 1 的 *UEOS* 为 n=1 时的特例情况.

定义 5. 多元对象网 MOPN 的可激活条件与点火规则.

$MOPN.Mark=(ON_1.(M,\hat{M}),ON_2.(M,\hat{M}),\ldots,ON_k.(M,\hat{M}),\ldots,ON_n.(M,\hat{M}))$ 表示点火前的某个状态,

$MOPN.Mark'=(ON_1.(M',\hat{M}'),ON_2.(M',\hat{M}'),\ldots,ON_k.(M',\hat{M}'),\ldots,ON_n.(M',\hat{M}'))$ 表示点火后的状态.

点火前后状态的变化记为 $MOPN.Mark\xrightarrow{([t,\hat{t}],k)}MOPN.Mark'$,$([t,\hat{t}],k)$ 表示点火动作,其中,k 表示这个点火动作与子网 ON_k 有关,$[t,\hat{t}]$ 是一个序偶且 $(t\in T)\wedge(\hat{t}\in\hat{T}_k)$,$t$ 表示来自系统网变迁的动作,$\hat{t}$ 表示来自子网 ON_k 变迁的动作.设每次点火的动作只与 n 个子网中的一个子网 ON_k 有关,即除 ON_k 以外的其他子网的系统状态与自身状态都不发生改变,表示为 $\forall j\neq k:ON_j.(M',\hat{M}')=ON_j.(M,\hat{M})$.对于 SN 中的库所 p,$ON_k.M(p)$表示当子网 ON_k 的系统状态为 $ON_k.M$ 时,子网 ON_k 在 SN 的库所 p 中的个数.对于 ON_k 中的库所 $\hat{p}\in\hat{P}_k$,$ON_k.\hat{M}(\hat{p})$ 表示子网 ON_k 的自身状态为 $ON_k.\hat{M}$ 时,在其内部库所 $\hat{p}$ 的托肯个数.MOPN 有以下 3 种点火方式:

(1) 系统网对子网 ON_k 的自治点火:在 SN 中,对于 $t\in T$ 且$\rho_k(t)=\varnothing$,$\exists k.Enabled(ON_k.M,t)$当且仅当$\forall p\in\bullet t$:$ON_k.M(p)\geq1$,$\exists k.Enabled(ON_k.M,t)$表示存在子网 ON_k 对于系统网 SN 中的变迁 t 是可激活的,其条件是当且仅当 ON_k 在变迁 t 的每个输入库所中都分布着至少 1 个,即覆盖了变迁 t 的每个输入库所.这些系统网中的映射到同一个子网 ON_k 的托肯对象在系统网的变迁 t 点火时,会同时分别传输到 t 的每个输出库所中(如图 4 所示).点火动作记为 $MOPN.Mark\xrightarrow{([t,\lambda],k)}MOPN.Mark'$,表示系统网的变迁 t 对子网 ON_k 的自治点火,这里,

$$ON_k.(M,\hat{M})\xrightarrow{[t,\lambda]}ON_k.(M',\hat{M}'),$$

其中,$ON_k.M\xrightarrow{t}ON_k.M'$,$ON_k.\hat{M}'=ON_k.\hat{M}$,即 $ON_k.M'=ON_k.M-\bullet t(ON_k)+t\bullet(ON_k)$,表示将子网 ON_k 从 t 的每个输入库所各取一个传输到 t 的每个输出库所,$ON_k.\hat{M}'=ON_k.\hat{M}$ 表示在系统网对 ON_k 自治点火时,ON_k 的自身状态不发生变化.图 4 表示 t 的一个输入库所中含有子网 ON_k 与 ON_j,另一个输入库所中只含有 ON_k,此时,ON_k 对于系统网 SN 中的变迁 t 可激活,而 ON_j 对于变迁 t 不可激活.当系统网的变迁 t 对子网 ON_k 做自治点火后,其状态变化如图 4 所示.

(2) 子网 ON_k 的自治点火:在子网 ON_k 中,$\exists\hat{t}\in\hat{T}_k$ 且 $\rho(\hat{t})=\varnothing$,$Enabled(ON_k.\hat{M},\hat{t})$ 当且仅当 $\forall\hat{p}\in\bullet\hat{t}$:$ON_k.\hat{M}(\hat{p})\geq1$,$Enabled(ON_k.\hat{M},\hat{t})$表示子网 ON_k 满足对变迁 $\hat{t}$ 做子网的自治点火的可激活条件是变迁 $\hat{t}$ 的每个

输入库所中都分布着至少 1 个托肯. $MOPN.Mark \xrightarrow{([\lambda,\hat{t}],k)} MOPN.Mark'$ 表示子网 ON_k 对其自身的变迁 $\hat{t}$ 做自治点火的状态变化,这里, $ON_k.(M,\hat{M}) \xrightarrow{[\lambda,\hat{t}]} ON_k.(M',\hat{M}')$,其中, $ON_k.M'=ON_k.M$, $ON_k.\hat{M} \xrightarrow{\hat{t}} ON_k.\hat{M}'$,即 $ON_k.\hat{M}'=ON_k.\hat{M}-ON_k.\bullet\hat{t}+ON_k.\hat{t}\bullet$, $ON_kM'=ON_kM$ 表示在 ON_k 做子网的自治点火时, ON_k 的系统状态不发生变化.

(3) 系统网与子网 ON_k 的同步点火:对于一对交互关系 $(t,\hat{t})\in\rho_k$, $Enabled(ON_k.(M,\hat{M}),[t,\hat{t}])$ 当且仅当 $Enabled(ON_k.M,t)\wedge Enabled(ON_k.\hat{M},\hat{t})$, $MOPN.Mark \xrightarrow{([t,\hat{t}],k)} MOPN.Mark'$ 表示系统网 SN 的变迁 t 与子网 ON_k 的变迁 $\hat{t}$ 同步点火,这里, $ON_k.(M,\hat{M}) \xrightarrow{[t,\hat{t}]} ON_k.(M',\hat{M}')$,其中, $ON_k.M \xrightarrow{t} ON_k.M'$ 与 $ON_k.\hat{M} \xrightarrow{\hat{t}} ON_k.\hat{M}'$ 同时发生. □

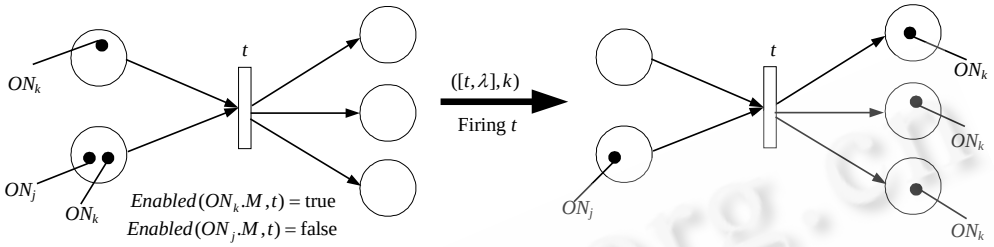


Fig.4 System-Autonomous firing in MOPN

图 4 多元对象网 MOPN 的系统网的自治点火

2.4 基于多元对象网的多视角软件过程建模

为了将多元对象网应用于多视角软件过程建模,本文借鉴工作流网模型的概念,并在定义 3 的多元对象网模型的基础上,定义基于多元对象网的多视角软件过程模型 MOPN-SP-net(定义 6),以活动视角与产品视角的组合建模为例(如图 5 所示),用系统网描述软件过程的活动视角,用多个子网分别描述多个产品视角.

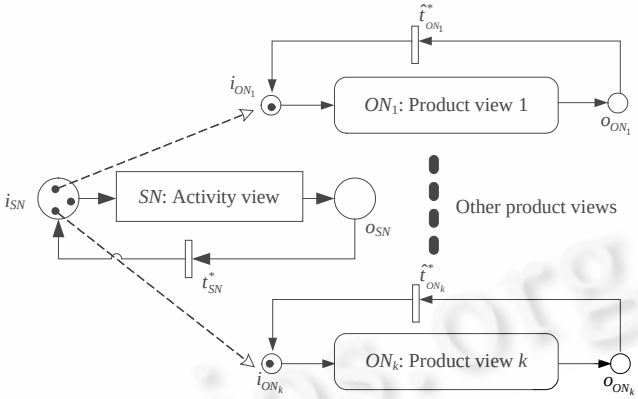


Fig.5 Modeling multi-view software process with by MOPN-SP-net

图 5 MOPN-SP-net 用于多视角软件过程建模

定义 6. 一个 $MOPN=(SN,ON_S,\rho)$ 是 MOPN-SP-net 的需要满足以下条件:

(1) 系统网 $SN=(P,T,F)$ 是 WF-net,用于描述软件过程的活动视角, i_{SN} 是起始库所: $\bullet i_{SN}=\varnothing$; o_{SN} 是结束库所: $o_{SN}\bullet=\varnothing$.

(2) $ON_S=\{ON_1,ON_2,\ldots,ON_n\}$ 表示由 n 个子网组成的集合, n 是有限自然数,用于描述软件过程的多个产品视角,其中的每个子网 $ON_k=(\hat{P}_k,\hat{T}_k,\hat{F}_k)$ 所描述的产品视角都是 WF-net,在 ON_k 中, i_{ON_k} 是起始库所: $\bullet i_{ON_k}=\varnothing$; o_{ON_k} 是结束库所: $o_{ON_k}\bullet=\varnothing$.

(3) $\rho_k \subseteq T \times \hat{T}_k$ 表示活动视角 SN 与产品视角 ON_k 的交互集合, $\rho = \bigcup_{k=1}^{n} \rho_k$ 表示活动视角与所有产品视角之间的交互关系的全集. □

在这个有 n 个子网的多元对象系统中,MOPN-SP-net 的初始状态为:所有子网都分布在系统网的起始库所 i_{SN},即每个子网的初始状态分别为 $\forall k \in \{1,...,n\}: ON_k.(M_0, \hat{M}_0) = ON_k.(i_{SN}, i_{ON_k})$,MOPN-SP-net 的全局初始状态表示为 $MOPN.Mark_0=(ON_1.(i_{SN}, i_{ON_1}),...,ON_k.(i_{SN}, i_{ON_k}),...,ON_n.(i_{SN}, i_{ON_n}))$,相应地,MOPN-SP-net 的全局结束状态表示为 $MOPN.Final=(ON_1.(o_{SN}, o_{ON_1}),...,ON_k.(o_{SN}, o_{ON_k}),...,ON_n.(o_{SN}, o_{ON_n}))$.

图 6 是一个用 MOPN-SP-net 描述的针对设计文档变更过程进行多视角软件过程建模的例子,其中,系统网 SN 用来描述变更控制过程的活动视角,与这个变更过程相关的产品视角有两个:子网 ON_1 与子网 ON_2.子网 ON_1 描述设计文档变更的产品视角.如图 6 所示,设计文档在检出(checkout)后进入可操作状态 op,然后经过修改(modify)之后进行评审(review),评审的结果是取消变更(cancel),或者是评审通过(OK),进入评审后状态“rv”,再录入(checkin)配置库结束一次流程.子网 ON_2 描述对应的变更报告流转的产品视角,报告提出(originate)后首先通知(notify)相关人员,随后由负责变更控制的配置管理委员会 CCB(configuration control board)评审(review),评审的结果或者是取消变更(cancel),或者是接受(accept),修改后核对(justify).活动视角与各产品视角之间的交互关系如图 6 所示.

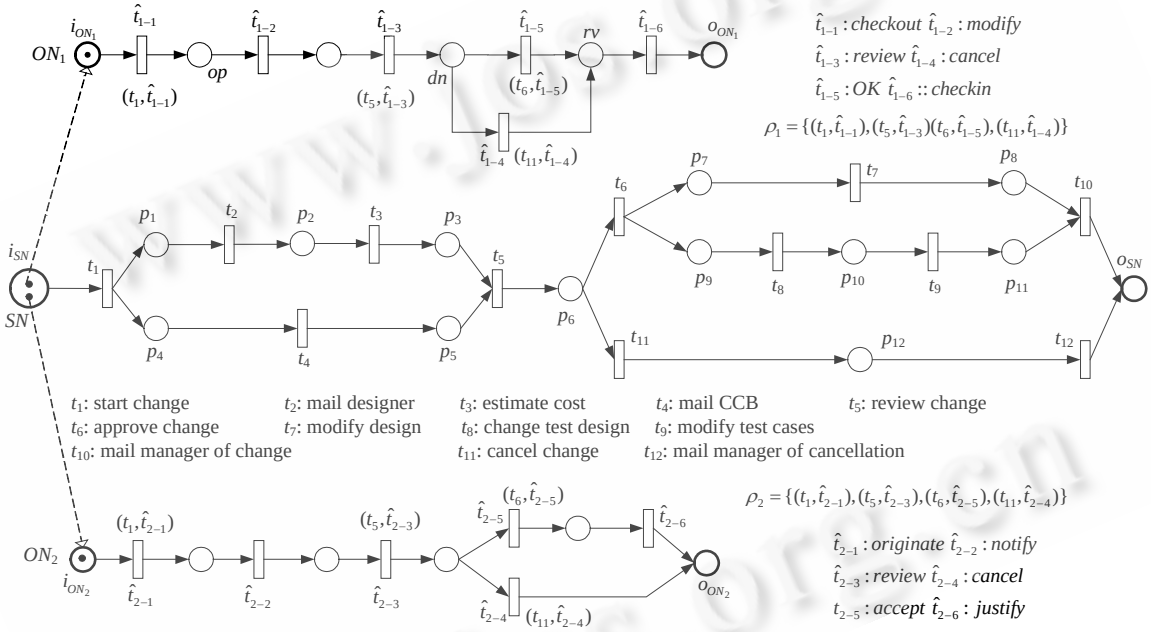


Fig.6 An example of multi-view software process modeled by MOPN-SP-net

图 6 例子:一个基于 MOPN-SP-net 的多视角软件过程模型

从图 6 可以看出,MOPN-SP-net 分离定义软件过程的活动视角和产品视角,各视角之间是一种松耦合的关系,使得结构清晰、层次分明.当上述软件过程发生改变需要修改建模时,只需对发生改变的视角作局部的修改,然后重新配置交互集.因此,基于 MOPN-SP-net 建模可以提高各视角模块的可复用性,并使得软件过程模型的修改变得灵活而简单.

3 从对象网到平面网的转换及其合理性等价

由计算机程序辅助执行的软件过程可以规范软件企业的工作流程,为了使建立的软件过程模型能够被

PSEE 系统的过程引擎正确而有效地执行,合理性(soundness)是需要考虑的重要属性,并对其合理性作相应的分析[15,27,28].对象网模型有很强的过程建模能力[29],但是作为一种多维 Petri 网,直接分析 MOPN-SP-net 的合理性比较困难,而对于传统的平面网有很多成熟的分析技术和分析工具[30].为此,我们设计了一套转换规则,将对象网转换到平面网,并且阐明转换前后的模型在合理性准则上保持等价.这个转换在对象网与平面网之间建立了一个等价的桥梁,可以用现有的分析技术分析转换后的平面网的合理性,以此来得知转换前的对象网的合理性,从而避免了直接分析的难度.

3.1 转换规则

根据 MOPN 中的状态表示(定义 4)和点火规则(定义 5),我们设计了一套从对象网到平面网的等价转换规则.从 MOPN 转换后的平面网记为 TMOPN(translated MOPN).对于一个给定的 $MOPN=(SN,ON_S,\rho)$,令转换后的平面网为 $TMOPN=(TP,TT,TF)$,TP 是 TMOPN 的库所集,TT 是 TMOPN 的变迁集,TF 是 TMOPN 的弧线集合,转换规则的本质就是构造一个与 MOPN 等价的平面网 TMOPN.因为要把来自不同子网的结构与状态融合到同一个平面网 TMOPN 中,所以需要对来自不同子网的变迁与托肯作颜色(colour)区分[31],表示该变迁允许通过哪些颜色的托肯.TMOPN 有很多结构是从 MOPN 的系统网或者子网继承而来,源自系统网的托肯若指向不同编号的子网,则在 TMOPN 中看成不同颜色的托肯;源自不同子网的托肯,在 TMOPN 中看成不同颜色的托肯.本文用子网的编号表示 TMOPN 中托肯的颜色分类,把源自系统网中指向子网 ON_k 的托肯与源自子网 ON_k 中的托肯看成同一种颜色 ON_k.引入颜色的概念,使得 TMOPN 中的变迁对可以通过的托肯的颜色有所限制,称 TMOPN 中的变迁 t_x 允许通过的托肯的颜色类别为变迁 t_x 的颜色集合(colour set),记为 $t_x.Colour$.具体规则如下:

规则 1(Rule 1). 从对象网到平面网的转换规则.

(1) $TP=P\cup\hat{P}$,继承系统网与子网的库所集,组成 TMOPN 的库所集合.

(2) $TT=T\cup\hat{T}$,继承系统网与子网的变迁集,组成 TMOPN 的变迁集合.

(3) 参照定义 5 的点火规则,设置变迁的颜色集以及构造 TMOPN 的弧线集合 TF,由以下步骤组成:

(a) 对应于系统网的自治点火,$\forall t\in T$,那么 $t\in TT$,对变迁 t 赋以初始颜色集 $t.Colour=\{ON_1,ON_2,\ldots,ON_n\}$,与变迁 t 相关的弧线的连接表示如下:

$\forall p_x\in\bullet t$ in SN, then $(p_x,t)\in TF$ in TMOPN

$\forall p_y\in t\bullet$ in SN, then $(t,p_y)\in TF$ in TMOPN

(b) 对应于子网 ON_k 的自治点火,$\forall\hat{t}\in\hat{T}_k$,$\hat{t}\in\hat{T}$,那么 $\hat{t}\in TT$,对变迁 $\hat{t}$ 赋以初始颜色集 $\hat{t}.Colour=\{ON_k\}$,与变迁 $\hat{t}$ 相关的弧线的连接表示如下:

$\forall\hat{p}_x\in\bullet\hat{t}$ in ON_S, then $(\hat{p}_x,\hat{t})\in TF$ in TMOPN

$\forall\hat{p}_y\in\hat{t}\bullet$ in ON_S, then $(\hat{t},\hat{p}_y)\in TF$ in TMOPN

(c) 对应于系统网与子网 ON_k 的同步点火,$\forall t\in T.\forall\hat{t}\in\hat{T}_k:(t,\hat{t})\in\rho_k$,对于每一对交互关系 $(t,\hat{t})\in\rho_k$,将子网变迁 $\hat{t}$ 改名(rename)为 TMOPN 中的同步变迁 $t\hat{t}$,即 $t\hat{t}\in TT$ 且 $t\hat{t}.Colour:=\{ON_k\}$,并增加弧线,将变迁 t 的输入库所与输出库所分别加入到 $t\hat{t}$ 的输入库所集与输出库所集($\bullet t\hat{t}:=\bullet t\cup\bullet\hat{t}$,$t\hat{t}\bullet:=t\bullet\cup\hat{t}\bullet$),另外修改 $t.Colour$,$t.Colour:=t.Colour\backslash\{ON_k\}$,即使得 $ON_k\notin t.Colour$.这些操作步骤,表示与系统网的变迁 t 有同步交互的颜色为 ON_k 的托肯不能直接通过源自系统网的变迁 t,而需要选择与子网 ON_k 合并后的同步变迁 $t\hat{t}$.与变迁 $t\hat{t}$ 相关的弧线的连接表示如下:

$\forall p_x\in\bullet t$ in SN, then $\forall(p_x,t\hat{t})\in TF$ in TMOPN

$\forall p_y\in t\bullet$ in SN, then $\forall(t\hat{t},p_y)\in TF$ in TMOPN

$\forall\hat{p}_x\in\bullet\hat{t}$ in ON_S, then $\forall(\hat{p}_x,t\hat{t})\in TF$ in TMOPN

$\forall\hat{p}_y\in\hat{t}\bullet$ in ON_S, then $\forall(t\hat{t},\hat{p}_y)\in TF$ in TMOPN □

图 7 表示了一个简单的多元对象网到平面网的转换,图中标注了转换后的平面网的变迁的颜色集合,其中粗线条表示为每一对交互关系 $(t,\hat{t})\in\rho_k$ 新增加的弧线.

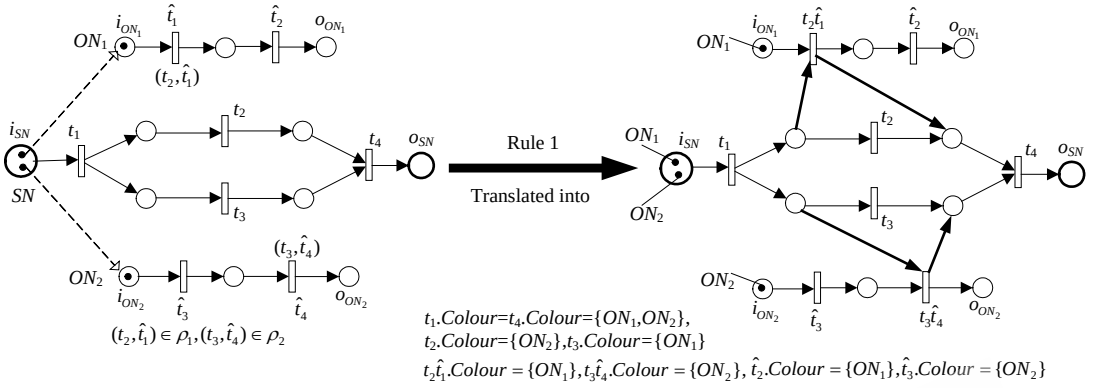


Fig.7 An example: Translate an MOPN into a flat net with configuration of transitions' colour set

图 7 例子:从多元对象网到平面网的转换及其变迁的颜色集的设置

因为在 TMOPN 中不同颜色的托肯代表着不同子网,某个颜色托肯的分布代表着某个子网的运行状态.而在点火条件上,由于颜色的约束,不同颜色的托肯即使同时覆盖某个变迁的输入库所,也不能通过这个变迁进行同步点火,即不同颜色的托肯之间没有交互,也代表了不同子网之间没有直接交互.因此,我们可以将 TMOPN 的状态根据不同颜色的托肯作分割,即在某个时刻的 TMOPN 的状态由不同颜色的托肯的分布状态叠加组成.设某个时刻 TMOPN 的状态为 $TMOPN.TMark$,设抽取其中颜色为 ON_k 的托肯的分布状态为 $ON_k.TMark$,那么 $TMOPN.TMark=ON_1.TMark+ON_2.TMark+\ldots+ON_n.TMark$.在 $TMOPN$ 中,托肯与变迁的颜色概念对点火条件的约束表现为:$t_x\in TT$ 是可激活的当且仅当$\exists ON_k\in t_x.Colour:\forall p\in\bullet t_x:ON_k.TMark(p)\geq 1$.其含义是某个变迁 t_x 可以激活的条件是,要求在状态 $TMOPN.TMark$,在变迁 t_x 的颜色集合中存在某个颜色为 ON_k 的托肯覆盖了 t_x 的所有输入库所.在点火时,将 t_x 的每个输入库所中的同一个颜色为 ON_k 的托肯各取一个传输到 t_x 的每个输出库所中,并且在传输的过程中托肯的颜色不发生变化.MOPN 的全局状态与 TMOPN 的全局状态存在着一组对应关系,在某个时刻,设 MOPN 中的子网 ON_k 的状态为 $ON_k.(M,\hat{M})$,那么在与之对应的 TMOPN 中,颜色为 ON_k 的托肯的分布状态为 $ON_k.TMark=ON_k.M+ON_k.\hat{M}=ON_k.(M+\hat{M})$;设 $MOPN$ 的全局状态为 $TMOPN.TMark=(ON_1.(M,\hat{M}),ON_2.(M,\hat{M}),...,ON_n.(M,\hat{M}))$,那么与之对应的 $TMOPN$ 的全局状态为

$$TMOPN.TMark=(ON_1.M+ON_1.\hat{M})+(ON_2.M+ON_2.\hat{M})+,...,+(ON_n.M+ON_n.\hat{M}).$$

设 MOPN-SP-net 是一个基于对象网的多视角软件过程模型,TMOPN-SP-net 是其通过规则 1 转换得到的平面网模型,设 MOPN-SP-net 全局初始状态为 $MOPN.Mark_0=(ON_1.(i_{SN},i_{ON_1}),ON_2.(i_{SN},i_{ON_2}),...,ON_n.(i_{SN},i_{ON_n}))$,转换后的 TMOPN-SP-net 的初始状态是在库所 i_{SN} 中存放着 n 种颜色代表 n 个子网的托肯各一个,即此时的 $i_{SN}.token=\{ON_1,ON_2,\ldots,ON_n\}$,而对于源自某个子网 ON_k 的起始库所 i_{ON_k} 存放着一个颜色为 ON_k 的托肯,因此,对于颜色为 ON_k 的托肯,在 TMOPN-SP-net 中的初始分布状态为 $ON_k.TMark_0=ON_k.i_{SN}+ON_k.i_{ON_k}=ON_k.(i_{SN}+i_{ON_k})$,TMOPN-SP-net 的全局初始状态为 $TMOPN.TMark_0=ON_1.TMark_0+ON_2.TMark_0+\ldots+ON_m.TMark_0=(ON_1.i_{SN}+ON_1.i_{ON_1})+(ON_2.i_{SN}+ON_2.i_{ON_2})+...+(ON_n.i_{SN}+ON_n.i_{ON_n})$.相应地,子网 ON_k 的结束状态为 $ON_k.TFinal=ON_k.o_{SN}+ON_k.o_{ON_k}=ON_k.(o_{SN}+o_{ON_k})$,TMOPN-SP-net 的全局结束状态为 $TMOPN.TFinal=ON_1.TFinal+ON_2.TFinal+\ldots+ON_n.Tfinal=(ON_1.o_{SN}+ON_1.o_{ON_1})+(ON_2.o_{SN}+ON_2.o_{ON_2})+...+(ON_n.o_{SN}+ON_n.o_{ON_n})$.图 8 表示从图 6 所示的 MOPN-SP-net 经过规则 1 转换得到的 TMOPN-SP-net.TMOPN-SP-net 保留了源自 MOPN-SP-net 的库所和变迁集合,其中粗线条表示为每一对交互关系 $(t,\hat{t})\in\rho_k$ 新增加的弧线.图 8 中各变迁的颜色集为

$t_2.Colour=t_3.Colour=t_4.Colour=t_7.Colour=t_8.Colour=t_9.Colour=t_{10}.Colour=t_{12}.Colour=\{ON_1,ON_2\}$,

$t_5.Colour=t_6.Colour=t_{11}.Colour=\varnothing$,

$t_1\hat{t}_{1-1}.Colour=t_5\hat{t}_{1-3}.Colour=t_6\hat{t}_{1-5}.Colour=t_{11}\hat{t}_{1-4}.Colour=\{ON_1\}$,

$\hat{t}_{1-2}.Colour=\hat{t}_{1-6}.Colour=\{ON_1\}$,

$t_1\hat{t}_{2-1}.Colour=t_5\hat{t}_{2-3}.Colour=t_6\hat{t}_{2-5}.Colour=t_{11}\hat{t}_{2-4}.Colour=\{ON_2\}$,

$\hat{t}_{2-2}.Colour=\hat{t}_{2-6}.Colour=\{ON_2\}$.

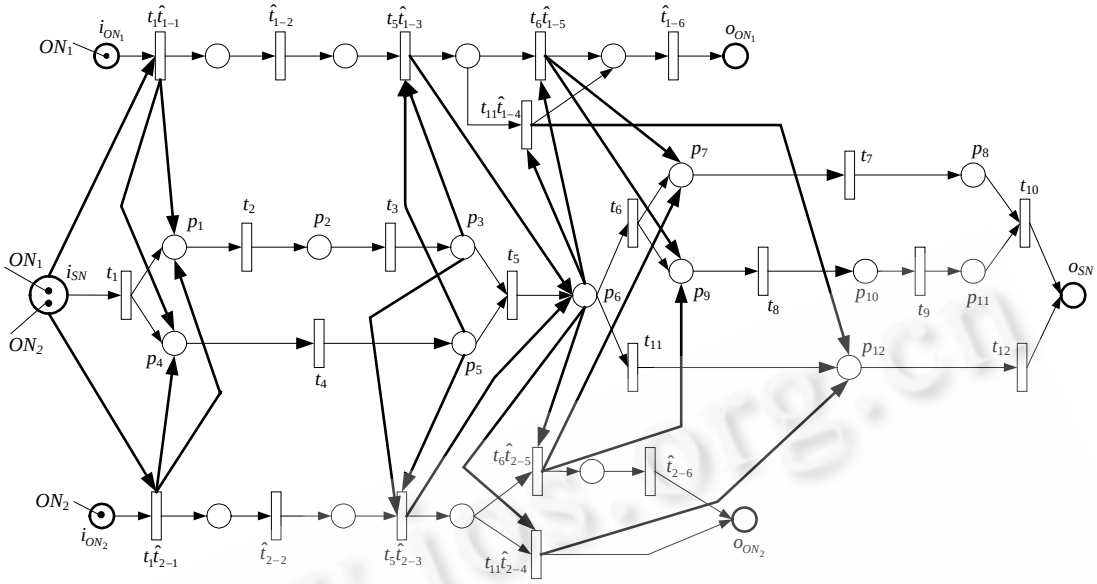


Fig.8 The flat net translated from Fig.6 by Rule 1

图 8 从图 6 经过规则 1 转换后得到的平面网

3.2 合理性等价

为了确保基于MOPN-SP-net建立的多视角软件过程模型能够被PSEE系统的过程引擎正确而有效地执行,合理性是过程模型在建模时需要遵循的重要性质.我们将阐明转换规则 1 保持了合理性等价(定理 1).因为MOPN-SP-net与TMOPN-SP-net在点火条件和点火规则上与传统Petri网有所不同,本文对照WF-net的合理性的含义,对MOPN-SP-net与TMOPN-SP-net的合理性作了相关的界定(定义8与定义9).

定义 7. WF-net 的合理性 [15].

对于一个用 WF-net 描述的过程 $PN=(P,T,F)$是合理的(sound),当且仅当同时满足以下条件:

(1) $\forall M([i]\xrightarrow{*}M)\Rightarrow(M\xrightarrow{*}[o])$.

(2) $\forall M([i]\xrightarrow{*}M\wedge M\geq[o])\Rightarrow(M=[o])$.

(3) $\forall t\in T:\exists M,M',[i]\xrightarrow{*}M\xrightarrow{t}M'$.

前两个条件表示WF-net描述的过程能够正确地结束,而最后一个条件表示以$[i]$为初始状态的Petri网系统不存在死变迁,即每个变迁所代表的动作都有可能被执行.从本质上讲,WF-net 的合理性是 Petri 网的活性(liveness)与有界性(boundedness)两个基本性质的组合,即 PN 是合理的当且仅当与之对应的扩展后的模型$PN^*=(P,T\cup\{t^*\},F\cup\{(o,t^*),(t^*,i)\})$是活的(live)且是有界的(bounded)[15].

定义 8. 设 TMOPN-SP-net 是从某个 MOPN-SP-net 转换得到的平面网模型,TMOPN-SP-net 描述的过程是合理的,考察颜色为 ON_k 的托肯的分布状态 $ON_k.TMark$,当且仅当同时满足以下条件:

(1) $\forall ON_k.(\forall ON_k.TMark):((ON_k.i_{SN}+ON_k.i_{ON_k})\xrightarrow{*}ON_k.TMark)\Rightarrow ON_k.TMark\xrightarrow{*}(ON_k.o_{SN}+ON_k.o_{ON_k})$.

(2) $\forall ON_k.(\forall ON_k.TMark):((ON_k.i_{SN}+ON_k.i_{ON_k})\xrightarrow{*}ON_k.TMark)\wedge(ON_k.TMark\geq(ON_k.o_{SN}+ON_k.o_{ON_k}))$
$\Rightarrow ON_k.TMark=(ON_k.o_{SN}+ON_k.o_{ON_k})$.

(3) $\forall ON_k.(\forall t\in TT)\wedge(ON_k\in t.Colour)$,

$\exists ON_k.TMark, ON_k.TMark':((ON_k.i_{SN}+ON_k.i_{ON_k})\xrightarrow{*}ON_k.TMark\xrightarrow{t}ON_k.TMark')$

定义 9. MOPN-SP-net 是合理的,当且仅当同时满足以下条件:

(1) 对于任意 ON_k, $\forall ON_k.(M,\hat{M}):ON_k.(i_{SN},i_{ON_k})\xrightarrow{*}ON_k.(M,\hat{M})\Rightarrow ON_k.(M,\hat{M})\xrightarrow{*}ON_k.(o_{SN},o_{ON_k})$, $ON_k.(i_{SN},i_{ON_k})$ 是子网 ON_k 的初始状态, $ON_k.(o_{SN},o_{ON_k})$ 是子网 ON_k 的结束状态.

(2) 对于任意 ON_k, $\forall ON_k.(M,\hat{M}):ON_k.(i_{SN},i_{ON_k})\xrightarrow{*}ON_k.(M,\hat{M})\wedge(ON_k.(M,\hat{M})\geq ON_k.(o_{SN},o_{ON_k}))\Rightarrow ON_k.(M,\hat{M})=ON_k.(o_{SN},o_{ON_k})$.

(3) $\forall t\in\{1,\ldots,n\}.\forall t\in T$,

a) 如果 $\rho_k(t)=\varnothing$,那么

$\exists ON_k.(M,\hat{M}),ON_k.(M',\hat{M}'):ON_k.(i_{SN},i_{ON_k})\xrightarrow{*}ON_k.(M,\hat{M})\xrightarrow{[t,\lambda]}ON_k.(M',\hat{M}')$.

b) 如果 $\rho_k(t)\neq\varnothing$,那么

$\forall\hat{t}\in\rho_k(t).\exists ON_k.(M,\hat{M}),ON_k.(M',\hat{M}'):ON_k.(i_{SN},i_{ON_k})\xrightarrow{*}ON_k.(M,\hat{M})\xrightarrow{[t,\hat{t}]}ON_k.(M',\hat{M}')$.

(4) $\forall t\in\{1,\ldots,n\}.\forall\hat{t}\in\hat{T}_k$,

a) 如果 $\rho_k(\hat{t})=\varnothing$,那么

$\exists ON_k.(M,\hat{M}),ON_k.(M',\hat{M}'):ON_k.(i_{SN},i_{ON_k})\xrightarrow{*}ON_k.(M,\hat{M})\xrightarrow{[\lambda,\hat{t}]}ON_k.(M',\hat{M}')$.

b) 如果 $\rho_k(\hat{t})\neq\varnothing$,那么

$\forall t\in\rho_k(\hat{t}).\exists ON_k.(M,\hat{M}),ON_k.(M',\hat{M}'):ON_k.(i_{SN},i_{ON_k})\xrightarrow{*}ON_k.(M,\hat{M})\xrightarrow{[t,\hat{t}]}ON_k.(M',\hat{M}')$.

合理性是本文关注的重要性质,图 6 中,MOPN-SP-net 描述的多视角软件过程模型是合理的,与之对应的图 8 中转换后的 TMOPN-SP-net 也是合理的.对于转换规则 1,我们将阐明这个转换规则保持了合理性等价,即 MOPN-SP-net 是合理的当且仅当 TMOPN-SP-net 是合理的(定理 1).为了证明该定理,需要定义 10 和引理 1.

定义 10. 合拍的点火序列(compatible firing sequence).

令 $MOPN=(SN,ON_S,\rho),SN=(P,T,F),ON_S=\{ON_1,ON_2,\ldots,ON_n\}$, $ON_k=(\hat{P}_k,\hat{T}_k,\hat{F}_k)$ 是一个多元对象网,经过规则 1 转换后得到的平面网为 $TMOPN=(TP,TT,TF)$.对于 MOPN,令 $\sigma_1=a_1a_2a_3\ldots a_n$ 是 MOPN 的一个点火序列, $\sigma_2=u_1u_2u_3\ldots u_n$ 是与之对应的 TMOPN 点火序列.称 σ_1 与 σ_2 是合拍的(compatible),当且仅当:

(1) 在 MOPN 中,若 a_x 是系统网对子网 ON_k 的自治点火,点火变迁为 $t\in T, a_x=[t,\lambda]$

⇔在 TMOPN 中, $ON_k\in t.Colour$ 且 $u_x=t$.

(2) 在 MOPN 中,若 a_x 是子网 ON_k 的自治点火,点火变迁为 $\hat{t}\in\hat{T}_k$, $a_x=[\lambda,\hat{t}]$

⇔在 TMOPN 中, $ON_k\in\hat{t}.Colour$ 且 $u_x=\hat{t}$.

(3) 在 MOPN 中,若 a_x 是系统网与子网 ON_k 的同步点火,点火变迁分别为 $t\in T$ 与 $\hat{t}\in\hat{T}_k$, $a_x=[t,\hat{t}]$

⇔在 TMOPN 中, $ON_k\in t\hat{t}.Colour$ 且 $u_x=t\hat{t}$.

根据定义 10,一个点火序列 σ,无论是在 MOPN(或者是在 TMOPN)中,将唯一决定一个在 TMOPN(或者是在 MOPN)中的点火序列 σ',使得 σ' 与 σ 是合拍的.

引理 1. 令 TMOPN 是根据转换规则 1 从 MOPN 转换得到的平面网,设 MOPN 的初始状态为 $MOPN.Mark=(ON_1.(M,\hat{M}),\ldots,ON_k.(M,\hat{M}),\ldots,ON_n.(M,\hat{M}))$,其中,子网 ON_k 的初始状态为 $ON_k.(M,\hat{M})$,与之对应的 TMOPN 的初始状态为 $TMOPN.TMark=(ON_1.M+ON_1.\hat{M})+\ldots+(ON_k.M+ON_k.\hat{M})+\ldots+(ON_n.M+ON_n.\hat{M})$,其中,颜色为 ON_k 的托肯的初始分布状态为 $ON_k.TMark=ON_k.M+ON_k.\hat{M}$. $\sigma_1=a_1a_2a_3\ldots a_n$ 是 MOPN 的点火序列, $\sigma_2=u_1u_2u_3\ldots u_n$ 是 TMOPN 的点火序列,如果 σ_1 与 σ_2 是合拍的,那么考察 MOPN 中子网 ON_k 的状态变化以及 TMOPN 中颜色为 ON_k 的托肯的分布状态的变化,有 $ON_k.(M,\hat{M})\xrightarrow{\sigma_1}ON_k.(M',\hat{M}')\Leftrightarrow ON_k.TMark\xrightarrow{\sigma_2}ON_k.TMark'$,这里, $ON_k.TMark'=ON_k.M'+ON_k\hat{M}'$.

引理 1 的证明从略.引理 1 揭示了对象网的点火序列与转换后平面网的点火序列之间的映射关系.

定理 1. 设 TMOPN-SP-net 是根据规则 1 从某个 MOPN-SP-net 转换得到的平面网.MOPN-SP-net 是合理的当且仅当 TMOPN-SP-net 是合理的.

定理 1 的证明从略.根据定理 1,转换规则 1 保证转换前后的模型在合理性准则上保持等价,所以,MOPN-SP-net 的合理性可以通过分析转换后的 TMOPN-SP-net 的合理性得知.

这里需要说明的是,虽然基于 MOPN-SP-net 建立的多视角软件过程模型可以等价转换为平面网,但是在建模时直接采用平面网会比较繁琐,且不符合用户的思维习惯,也不利于提高各视角模块的独立性和可复用性.因此,我们的方法是建模时采用 MOPN-SP-net 建立多视角软件过程模型,在分析时将其转换为等价的平面网,便于分析.对于生成的平面网,可以采用一些计算机辅助的自动化的分析方法[30].

在上述研究工作的基础上,我们基于 MOPN-SP-net 模型设计并实现了一个多视角软件过程建模的原型工具(MOPN-SP-net designer),该工具提供了一种图形化的软件过程建模环境,可以分离定义软件过程的活动视角、产品视角等,通过系统网与子网之间的交互集将活动视角与其他视角联系起来.该工具还集成了一些基于传统平面网的分析技术,如不变量分析、状态空间分析、可达性分析等功能.此外,我们还设计了一个解释执行 MOPN-SP-net 模型的过程引擎,并实现了一个 PSEE 系统,为软件企业提供了一种协同工作环境,可以达到对软件过程执行、监控、仿真等有效管理的目标.

4 相关工作与总结

软件过程建模是 PSEE 系统中的核心技术,受到国内外很多学者的普遍关注,其中有代表性的工作是文献[9]将软件过程描述为一组相互独立且对等的实体——软件过程 Agent,针对软件过程难以对所处环境的变化作出自适应的调整,提出了一种基于 Agent 的自适应软件过程模型.文献[32]通过分析 CMM 软件过程,提出了一个基于 SPEM(软件过程工程元模型)的 CMM 软件过程元模型(SPM-CMM),利用模型驱动架构 MDA 来支持 CMM 模型转换.文献[33]提出了一种基于模型融合的 CMM 实施过程建模方法,用 SPEM 建立 CMM 过程模型 CPM 和企业过程模型 EPM,通过融合 CPM 和 EPM 来获得 CMM 实施过程模型 CIPM.文献[34]针对软件过程事务处理的特殊性,提出了一个基于规则的软件过程事务模型,能够刻画长周期、迭代式过程和数据共享的多用户协作等特征.文献[35]提出了一个柔性的形式化过程建模语言 FLEX,其具有灵活性、可扩充性、可重用性和分布性等特点,允许用户通过自定义多种抽象级别的语言元素来扩充语言的描述能力.文献[12]提出了一种反应式(reactive)软件过程模型,并建立了图形化的软件过程建模语言,用时序逻辑语言 XYZ/E 表示它的行为视图动态语义,采用反应式控制方式,达到支持软件过程演化的目的.文献[36]基于抽象过程的复用机制,提出了 P-F 建模方法,使用 3 层复用结构:过程模板、模式和元模式,利用元模式来构造软件过程.文献[37]开发了一种支持多视角软件过程管理的工具,并且用 CPRG(change propagation and response graphs)方法解决软件过程的多个视角的不一致问题.

在基于 Petri 网的软件过程建模方面:SLANG[16]是软件过程支撑环境 SPADE 的过程建模语言,用于支持软件过程建模、分析与执行.SLANG 基于一种特殊的 Petri 网(ER nets),其中托肯可以赋予属性值,变迁可以约定执行时间.它采用变迁表示活动,变迁可以层次展开为子网以描述子过程,用不同类型的库所表示活动的输入、输出和中间事件,并提供了一种基于反射机制的过程演化方法.FUNSOFT Net[20]采用一种特殊的基于谓词网的模型,用于描述软件过程.它支持复杂的类型定义和变迁的谓词定义,并提供了过程计划定量评估和关键路径的计算以及验证软件过程的某些性质等功能.文献[21]基于面向对象 Petri 网,开发了一套建模工具 OOPN- IDE 用于 EPMS 系统的建模与仿真.

与上述相关工作相比,本文的工作主要体现在:基于关注点分离原则,结合多视角软件过程建模与对象网之间的相似性,采用多视角分离定义的方法,提出了一种基于对象网的多视角软件过程模型(MOPN-SP-net),该模型既可以分离定义软件过程的各个视角,又可以通过对象网的交互集将各视角联系起来,各视角之间是一种松耦合的关系,从而提高各视角模块的可复用性(reusability)和软件过程建模的灵活性.由此,使得建立起来的模型

结构清晰、层次分明、符合用户的思维习惯.基于 MOPN-SP-net 建立的多视角过程模型是一个多维网,针对目前关于 Petri 网的分析技术,多维网的直接分析比较困难,而对于传统平面网有很多成熟的分析技术[30].为此,本文提供了一种从多元对象网到平面网的转换规则,并且该规则保证转换前后的模型在合理性准则上保持等价,该规则在对象网与平面网之间建立了一种等价的桥梁,可以通过分析转换后平面网的合理性来得知转换前 MOPN-SP-net 的合理性,从而为分析 MOPN-SP-net 的合理性提供一种有效的策略,避免了直接分析的难度.

另外,多元对象网可以描述系统网与多个子网的同步交互,而这种同步交互可以用于描述和仿真(simulation)软件过程中多个涉众(stakeholders)之间的协同(coordination)与协商(negotiation),这个特点已经在 Boehm 提出的基于价值(value-based)的软件工程及其软件过程仿真中得以应用[38].

致谢 在此,我们感谢德国汉堡大学 TGI 研究组的 Valk 教授、Köhler 博士和 B. Farwer 博士以及英国 Warwick 大学计算机系的 Misra 博士,他们为本文提供了一些有价值的建议.

References:

[1] Yang FQ. Thinking on the development of software engineering technology. Journal of Software, 2005,16(1):1−7 (in Chinese with English abstract). http://www.jos.org.cn/1000-9825/16/1.htm

[2] Humphrey WS. Managing the Software Process. Boston: Addison-Wesley Publishing Co., 1989.

[3] Chrissis MB, Konrad M, Shrum S. CMMI: Guidelines for Process Integration and Product Improvement. Boston: Addison-Wesley Publishing Co., 2003.

[4] Gruhn V. Process-Centered software engineering environments: A brief history and future challenges. Annals of Software Engineering, 2002,14(1-4):363−382.

[5] Li MS. Expanding the horizons of software development processes: A 3-D integrated methodology. In: Li MS, Boehm BW, Osterweil LJ, eds. Proc. of the Unifying the Software Process Spectrum, Int'l Software Process Workshop (ISPW 2005). LNCS 3840, Berlin: Springer-Verlag, 2005. 54−67.

[6] Barthelmess P. Collaboration and coordination in process-centered software development environments: A review of the literature. Information & Software Technology, 2003,45(13):911−928.

[7] Acuna ST, Antonio AD, Ferre X, Lopez M, Mate L. The software process: Modeling, evaluation and improvement. In: Chang SK, ed. Proc. of the Handbook of Software Engineering and Knowledge Engineering. Singapore: World Scientific Publishing Co., 2001. 193−237.

[8] Derniame JC, Kaba BA, Wastell D. Software process: Principles, methodology, technology. LNCS 1500, Berlin, Heidelberg: Springer-Verlag, 1999.

[9] Zhao XP, Li MS, Wang Q, Chan K, Leung H. An agent-based self-adaptive software process model. Journal of Software, 2004,15(3):348−359 (in Chinese with English abstract). http://www.jos.org.cn/1000-9825/15/348.htm

[10] Jaccheri ML, Conradi R. Techniques for process model evolution in EPOS. IEEE Trans. on Software Engineering, 1993,19(12): 1145−1156.

[11] Valk R. Petri nets as token objects: An introduction to elementary object nets. In: Desel J, Silva M, eds. Proc. of the 19th Application and Theory of Petri Nets. LNCS 1420, Berlin: Springer-Verlag, 1998. 1−25.

[12] Dong GZ, Liu JF, Qi X. A kind of reactive SPM and the expression of its dynamic semantics with XYZ. Journal of Software, 2005, 16(11):1876−1885 (in Chinese with English abstract). http://www.jos.org.cn/1000-9825/16/1876.htm

[13] Osterweil LJ. Software processes are software too. In: Proc. of the 9th ICSE. ACM Press, 1987. 2−13.

[14] Sutton SM, Heimbigner D, Osterweil LJ. APPL/A: A language for software process programming. ACM Trans. on Software Engineering and Methodology, 1995,4(3):221−286.

[15] Aalst WMP, Hee K. Workflow Management—Models, Methods and Systems. Cambridge: MIT Press, 2002.

[16] Bandinelli S, Fuggetta A, Lavazza L, Loi M, Picco GP. Modeling and improving an industrial software process. IEEE Trans. on Software Engineering, 1995,21(5):440−454.

[17] Ambriola V, Conradi R, Fuggetta A. Assessing process-centered software engineering environments. ACM Trans. on Software Engineering and Methodology, 1997,6(3):283−328.

[18] Pohl K, Weidenhaupt K, Dömges R, Haumer P, Jarke M, Klamma R. PRIME—Toward process-integrated modeling environments: 1. ACM Trans. on Software Engineering and Methodology, 1999,8(4):343−410.

[19] Gunter CA. Abstracting dependencies between software configuration items. ACM Trans. on Software Engineering and Methodology, 2000,9(1):94−131.

[20] Deiters W, Gruhn V. Process management in practice applying the FUNSOFT net approach to large-scale processes. Process Technology/Automated Software Engineering (Special Issue), 1998,5(1):7−25.

[21] Ren AH, Zhou BS, Wang B, Huang LJ. The equivalence verify between visual process modeling language and Petri net. Computer Engineering and Design, 2001,22(6):11−18 (in Chinese with English abstract).

[22] Fan YS. Fundamentals of Workflow Management Technology. Beijing: Tsinghua University Press, 2001 (in Chinese).

[23] Yuan CY. Principals and Application of Petri Nets. Beijing: Publishing House of Electronics Industry, 2005 (in Chinese).

[24] Reisig W. An Introduction to Petri Nets. Berlin: Springer-Verlag, 1985.

[25] Lakos C. From coloured Petri nets to object Petri nets. In: Michelis GD, Diaz M, eds. Proc. of the 16th Application and Theory of Petri Nets. LNCS 935, Berlin: Springer-Verlag, 1995. 278−297.

[26] Valk R. Object Petri nets: Using the nets-within-nets paradigm. In: Desel J, Reisig W, Rozenberg G, eds. In: Proc. of the Lectures on Concurrency and Petri Nets. LNCS 3098, Berlin: Springer-Verlag, 2003. 819−848.

[27] Zhou JT, Shi ML, Ye XM. Formal verification techniques in workflow process modeling. Journal of Computer Research and Development, 2005,42(1):1−9 (in Chinese with English abstract).

[28] Lu P, Hu H. Lü J. On 1-soundness and soundness of workflow nets. In: Moldt D, ed. Proc. of the 3rd Workshop on Modelling of Objects, Components and Agents. Vol. PB-571. Denmark: Aarhus University, 2004. 21−36.

[29] Köhler M, Rölke H. Properties of object Petri nets. In: Cortadella J, Reisig W, eds. Proc. of the 25th Application and Theory of Petri Nets. LNCS 3099, Berlin: Springer-Verlag, 2004. 278−297.

[30] Girault C, Valk R. Petri Nets for System Engineering: A Guide to Modeling, Verification and Application. Berlin: Springer-Verlag, 2003.

[31] Jensen K. Coloured Petri nets. Basic Concepts, Analysis Methods and Practical Use, Vol. 1, Basic Concepts. 2nd Corrected Printing. Berlin: Springer-Verlag, 1997.

[32] Li J, Li MS, Wu ZC, Wang Q. A SPEM-based software process metamodel for CMM. Journal of Software, 2005,16(8):1366−1377 (in Chinese with English abstract). http://www.jos.org.cn/1000-9825/16/1366.htm

[33] Li J, Yuan F, Li MS, Wang Q. A model merging based approach for modeling the CMM implementation process. Chinese Journal of Computers, 2006,29(1):54−65 (in Chinese with English abstract).

[34] Shen BJ, Chen C, Ju DH. Criteria-Based software process transaction model. Journal of Software, 2002,13(1):24−32. http://www.jos.org.cn/1000-9825/13/24.pdf

[35] Chen C, Shen BJ, Gu YQ. A flexible and formalized process modeling language. Journal of Software, 2002,13(8):1374−1381. http://www.jos.org.cn/1000-9825/13/1374.pdf

[36] Zhou ZY. Reusability based on P-F method for software process modeling. Journal of Software, 2001,12(8):1258−1264 (in Chinese with English abstract).

[37] Grundy JC, Hosking JG, Mugridge WB. Inconsistency management for multiple-view software development environments. IEEE Trans. on Software Engineering, 1998,24(11):960−981.

[38] Huang LG, Boehm BW, Hu H, Ge JD, Lü J, Qian C. Applying the value/Petri process to ERP software development in China. In: Osterweil LJ, Rombach HD, Soffa ML, eds. Proc. of the 28th ICSE. ACM Press, 2006. 502−511.

附中文参考文献:

[1] 杨芙清.软件工程技术发展思索,软件学报,2005,16(1):1−7. http://www.jos.org.cn/1000-9825/16/1.htm

[9] 赵欣培,李明树,王青,陈振冲,梁金能.一种基于 Agent 的自适应软件过程模型.软件学报,2004,15(3):348−359. http://www.jos.org.cn/1000-9825/15/348.htm

[12] 董广智,柳军飞,齐璇.一种反应式 SPM 及其动态语义 XYZ 表示.软件学报,2005,16(11):1876−1885. http://www.jos.org.cn/1000-9825/16/1876.htm

[21] 任爱华,周伯生,王博,黄理骏.可视化过程建模语言 VPML 与 PETRI 网的等价性证明.计算机工程与设计,2001,22(6):11−18.

[22] 范玉顺.工作流管理技术基础.北京:清华大学出版社,2001.

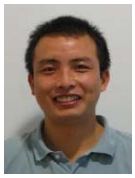
[23] 袁崇义.Petri 网原理与应用.北京:电子工业出版社,2005.

[27] 周建涛,史美林,叶新铭.工作流过程建模中的形式化验证技术.计算机研究与发展,2005,42(1):1−9.

[32] 李娟,李明树,武占春,王青.基于 SPEM 的 CMM 软件过程元模型.软件学报,2005,16(8):1366−1377. http://www.jos.org.cn/1000-9825/16/1366.htm

[33] 李娟,袁峰,李明树,王青.一种基于模型融合的 CMM 实施过程建模方法.计算机学报,2006,29(1):54−65.

[36] 周之英.基于 P-F 方法的软件过程建模的复用性.软件学报,2001,12(8):1258−1264.



葛季栋(1978－),男,江苏南通人,博士生,主要研究领域为软件过程建模,软件质量管理,形式化方法.

胡昊(1975－),男,博士生,讲师,CCF 会员,主要研究领域为软件过程,软件质量管理,软件 Agent 技术.

顾庆(1972－),男,博士,副教授,CCF 高级会员,主要研究领域为软件质量管理,软件测试.

吕建(1960－),男,博士,教授,博士生导师,CCF 高级会员,主要研究领域软件形式化与自动化,软件 Agent 技术及其应用.