一种基于活动序列的协同设计访问控制模型*

安毅生[1]，罗冰洁[1]，赵祥模[1]，李人厚[2]

（1. 长安大学 信息工程学院，西安 710064；2. 西安交通大学 系统工程研究所，西安 710049）

摘　要：为了确保在多用户协同设计环境中对文档及视图访问权限的动态分配与回收、提出了基于活动序列的访问控制模型，采用赋色 Petri 网描述并实现了模型中活动序列依赖关系约束、角色构造、分配与回收，权限冲突检测等功能。该模型将共享的文档和视图空间按照活动序列划分，把角色分配、回收与活动序列相关联，解决了使用单用户设计软件协同设计中授权用户对访问对象具有持久权限的问题。最后，以协作角色申请过程为例，说明了模型是如何实现访问权限动态分配与回收，以此说明该模型能够适应协同设计中权限随活动变化的访问控制需求。

关键词：访问控制；活动序列；角色；赋色 Petri 网

中图分类号：TP301.6　　**文献标志码**：A　　**文章编号**：1001-3695(2012)09-3324-06

doi:10.3969/j.issn.1001-3695.2012.09.033

Activity sequence based access control model in collaborative design

AN Yi-sheng[1], LUO Bing-jie[1], ZHAO Xiang-mo[1], LI Ren-hou[2]

(1. *School of Information Engineering, Chang'an University, Xi'an* 710064, *China*; 2. *Institute of Systems Engineering, Xi'an Jiaotong University, Xi'an* 710049, *China*)

Abstract: To assure activity based dynamic assignment and retrieve of cooperative permission in multi-user environment, this paper proposed an activity sequence based access control model, and provided CPNs based modeling and analyzing issues. In this model, it divided the shared cooperative document and view space according to the sequence of interdependent activities, and associated the cooperative activities with the assignment and retrieve of role. Therefore, it solved the problem of authorized users with persistent permissions to specific object in traditional access control. Finally, gave an example of applying cooperative roles to indicate how to implement the dynamic assignment and retrieve of cooperative permission.

Key words: access control; activity sequence; role; colored Petri nets(CPNs)

0　引言

随着网络技术迅速发展和广泛应用，协作环境中的访问控制成为计算机支持协同设计[1]领域的一个重要问题。现有单用户设计软件不支持多用户、多角色、多权限访问控制，无法实现重要数据、文档多视图浏览与操作，以及由此带来的访问权限动态分配与回收问题。

协同设计环境中的访问权限管理源于访问控制技术[2,3]，基于应用程序共享[4,5]的多用户协作，对协作环境中的访问控制提出了一些新要求，如访问权限动态管理与设计活动紧密相关、协作需求与文档访问权限相互渗透、用户协作权限（包括数据访问权限）具有生命周期、访问权限管理需要灵活性和合法性检查等。为此，研究人员提出了多种用于协作环境的访问权限管理方法。例如，文献[6]提出了一种用于协同编辑环境的权限管理模型，在访问控制矩阵基础上增加了细粒度权限设置群组管理和负权限管理等内容；文献[7]提出一种基于空间划分的访问权限管理模型，将大规模的协作环境划分为若干个细小的可控区域，每一个可控区域应用一个相应层次的访问控制策略；文献[8]提出了一种基于 MAC 和 RBAC[2]扩展的多用户同步协作环境，实现了面向三维模型的安全访问控制机制；文献[9]针对协作计算环境中潜在的用户安全与私有性保护问题，提出了一种组合认证多层次访问控制模型，确保用户生命周期内的动态授权认证；文献[10]提出了适用于分布式 CSCW 和工作流系统的访问控制模型，确保实时和非实时环境下用户对三维设计模型的访问；文献[11]提出了一种精细粒度的动态访问控制模型，该模型对角色、权限、子任务进行层次划分，由子任务选择必要的角色和权限，并进行合理配置。以上方法主要用于设计模型的多视图协同观察，其安全特征划分、角色及安全层次标志等工作预先定义，普遍存在授权用户对访问对象具有持久权限的问题，不能很好地支持访问权限的动态调整。

本文在分析现有访问控制技术及其在协作环境中初步应用的基础上，提出了基于活动序列的访问控制模型，并采用赋色 Petri 网对其进行描述和仿真，验证了文档及视图访问权限

收稿日期：2012-01-31；**修回日期**：2012-03-09　**基金项目**：国家教育部长江学者和创新团队发展计划资助项目(IRT0951)；国家自然科学基金资助项目(50978030)；陕西省自然科学基金资助项目(2009-JM8002-1)

作者简介：安毅生(1972-)，男，陕西西安人，副教授，博士，主要研究方向为计算机支持协同设计、Petri 网理论与应用等(aysm@chd.edu.cn)；罗冰洁(1986-)，女，陕西咸阳人，硕士研究生，主要研究方向为协作信息系统；赵祥模(1966-)，男，重庆大足人，教授，博导，主要研究方向为交通系统工程；李人厚(1935-)，男，浙江宁波人，教授，博导，主要研究方向为 Petri 网理论与应用.

动态调整中的一些约束条件,以确保协作环境中访问控制的可靠性和灵活性。

1　基于活动序列的访问控制模型

基于活动序列的访问控制模型如图 1 所示,它建立在角色与设计活动动态绑定基础上,引入了活动及活动序列化的概念,扩展了对象集合,使其不但包括文档、模型等传统的操作对象,还增加了设计工具共享视图等协作对象。在此基础上按照活动序列将协作过程中使用的共享文档和共享视图进行划分,进而为其关联用户、角色、操作和操作对象。角色的分配和撤销依赖于活动序列。该机制适应于设计过程的流程化与协作化,也为各种协作对象提供灵活、安全的访问策略。

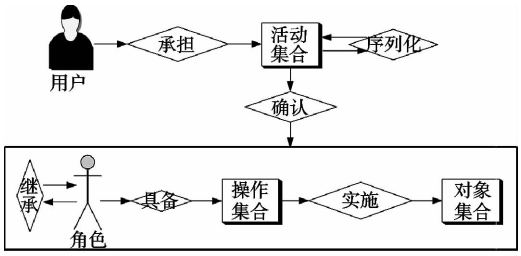


图1　基于活动序列的访问控制模型

1.1　模型元素定义

定义 1　对象集。对象集是协同设计环境中被操作的客体的集合,包括设计文档、设计工具两大类,记为 OBS = (docs, tools)。

定义 2　操作集。操作集是可以对对象实施操作的集合。由于对象集被分为设计文档和设计工具两类,操作也分为两类{dOPS,tOPS},其中:dOPS = {open, close, read, write, create};tOPS = {share, view, erase, drag, scrollbar, changeview, draw}。

定义 3　文档访问权限集。$dPRMS \subseteq docs \times 2^{dOPS}$,表示对设计文档的多种操作。

定义 4　协作权限集。$cPRMS \subseteq tools \times 2^{tOPS}$,表示基于设计工具界面的协同操作。

后文中不作明确说明时,协作权限包含定义 3 和 4。

定义 5　角色集。角色集由文档访问权限集和协作权限集的子集构成,记为 $ROLES = 2^{dPRMS \cup cPRMS}$。

定义 6　活动集。活动集是由设计者承担的一系列具有内在逻辑关系的针对特定对象的动作集合,记为 ACTIVITY = (aName, aType, aInputdoc, aOutputdoc)。其中:aName 表示活动的名称;aType 表示活动的类型;aInputdoc 表示活动的输入文档集;aOutputdoc 表示活动的输出文档集。活动既可以由一个设计者单独承担,也可以由一个设计者邀请多个设计者共同承担。

定义 7　用户集。协同设计中承担某个活动具备相应角色所规定操作集的用户,通常是由设计者的客户端 agent 承担。

定义 8　角色继承。$INHERTIS \subseteq ROLES \times ROLES$,表示建立在角色集偏序关系上的角色继承关系。$\forall r_1, r_2 \in ROLES$,如果 $r_1 \subseteq r_2$,则称 r_2 继承 r_1,r_1 与 r_2 之间满足继承关系,记做 $r_1 \rightarrow r_2$,r_1 称为父角色,而 r_2 称为子角色。因为角色继承关系是建立在偏序关系上的,因此角色继承具有传递性。

定义 9　操作冲突。对于任意的文档对象 $d_1, d_2 \in OBS$,$\forall op_1, op_2 \in \{dOPS, tOPS\}$。如果任何用户在 d_1 上执行 op_1 操作的同时,不能在 d_2 上执行 op_2 操作,则称 d_1 上的 op_1 和 d_2 上的 op_2 是冲突操作,记为 $op_{1(d_1)} \mid op_{2(d_2)}$。

显然,对于权限 prms = (obs, ops), $ops \subseteq \{dOPS, tOPS\}$,若 $\exists op_1, op_2 \in ops$,则不允许 op_1 和 op_2 在对象 obs 上冲突。

定义 10　权限冲突。对于任意 prms1 = ($obs1$, $ops1$),及 prms2 = ($obs2$, $ops2$),其中 $obs1, obs2 \in OBS$, $ops1, ops2 \subseteq \{dOPS, tOPS\}$,如果有 $op_1 \in ops_1$, $op_2 \in ops_2$,且 $op_{1(obs1)} \mid op_{2(obs2)}$,则称 prms1 和 prms2 是冲突权限。

定义 11　角色冲突。$\forall r1, r2 \in ROLES$,如果 $r1$, $r2$ 的某个权限存在冲突,则称 $r1$ 和 $r2$ 是冲突角色,记为 $r1 \mid r2$。

定义 12　操作依赖。若任意的权限 prms = (obs, ops), $\exists op1, op2 \in \{dOPS, tOPS\}$,如果针对 obs 的操作 op_2 必须在操作 op_1 执行完之后才可以执行,则称 op_2 在 obs 上依赖于 op_1。

定义 13　活动依赖关系。活动间存在的依赖关系,根据活动之间是否存在条件,可以分为以下七类关系:

a)跟随关系。$\forall \alpha, \beta \in ACTIVITYS$,如果 α 的触发完成是 β 触发的前提,则称这两个活动存在跟随关系,记为(α, β)。

b)并行关系。$\forall \alpha, \beta \in ACTIVITYS$,如果 α 的触发不会对 β 的触发造成任何影响,反之亦然,则称这两个活动间存在并发关系,记为$(\alpha; \beta)$。

c)条件分支。$\forall \alpha, \beta \in ACTIVITYS$ 及 Cond 为任意的条件表达式,如果 Cond 为真则活动 α 触发,否则活动 β 触发,称这种关系为条件分支,记为$(Cond ? \alpha \vee \beta)$。

d)与汇集。$\forall \alpha, \beta \in ACTIVITYS$,只有当并发的活动 α 及 β 被触发并完成,后续的活动才能触发,记为$((\alpha; \beta)/AND)$。

e)或汇集。$\forall \alpha, \beta \in ACTIVITYS$,只需要并发的活动 α 和 β 中的一个被触发并完成,后续的活动就可以触发,记为$((\alpha; \beta)/OR)$。

f)异或汇集。$\forall \alpha, \beta \in ACTIVITYS$,当且仅当并发的活动 α 和 β 中的一个被触发并完成,后续的活动就可以触发,记为$((\alpha; \beta)/XOR)$。

g)循环关系,$\forall \alpha \in ACTIVITYS$ 及 Cond 为任意的条件表达式,活动 α 重复执行,直到测试条件 Cond 不成立,记为$(Cond ? \alpha)^*$。

假设某个设计活动可以分解为如下活动序列:$\alpha1$, $\alpha2$, …, $\alpha8$,活动序列间的依赖关系可以表示为:$((\alpha1; \alpha2; \alpha3)/AND, \alpha4, Cond1 ? \alpha5 \vee (\alpha6, (Cond2 ? \alpha6)^*, \alpha7), \alpha8)$,如图 2 所示。

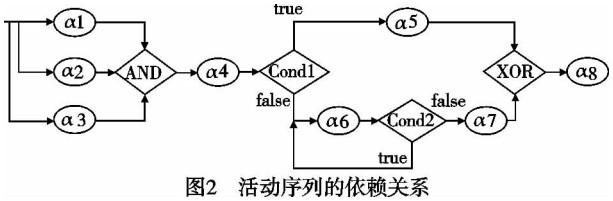


图2　活动序列的依赖关系

1.2　操作集合

协同设计访问控制模型中,用户与活动的绑定操作定义为:assigned_user:$(u: USERS) \rightarrow 2^{ACTIVITYS}$,它在用户与活动之间

建立了一对多的映射关系,表示一个用户可以异步地承担多个活动。活动与角色的绑定操作定义为:bound_role:(a:ACTIVITY)→2^{ROLES},它在活动和角色之间建立的一对多的映射关系,表示一个活动的完成所必须具备的角色集合。

此外,在模型元素集合上还定义了以下操作:

a) AddUser:

if $u \notin$ USERS then USERS = USERS $\cup \{u\}$

b) DelUser:

if $u \in$ USERS and assigned_user(u) $\neq \varnothing$ then

if assigned_user(u) is executing then waiting

elseif assigned_user(u) is ready or finish

then release user-activity binding and

assigned_user(u) $\Leftarrow \varnothing$

else USERS = USERS $\setminus \{u\}$

c) AddRole:

if $r \notin$ ROLES then ROLES = ROLES $\cup \{r\}$

d) DelRole:

if r $\in$ ROLES and $\forall$ p $\in$ ROLE, $\neg$ ($r \rightarrow p$) and

$\forall a \in$ ACTIVITYS, $\neg$ ($r \in$ bound_role(a))

then ROLES = ROLES $\setminus \{r\}$

e) InhRole:

if($\forall$ prms1 $\in$ PRMS or $\forall$ coops1 $\in$ COOPS) and

($\neg$ (r. prms. op | prms1. op) or

$\neg$ (r. coops. op | coops1. op)) and $r \in$ ROLES

then(r1. prms = r. prms $\cup$ prms1 or

r1. coops = r. coops $\cup$ coops1) and $r \rightarrow r$1

f) bindUA:

if $\forall u \in$ USERS and $\forall a \in$ ACTIVITYS,

$\neg$ ($\exists u \in$ USERS, assigned_user(u) = a)

then assigned_user(u) $\Leftarrow a$

g) bindAR:

if $\forall a \in$ ACTIVITYS and $\exists r \subseteq$ ROLES and

$\bigcup_{r \in role} r$. prms meet the extcution of a

then bound_role(a) = r

1.3 操作流程

基于活动序列的访问控制模型与设计活动不断推进相适应,把协同设计过程中的共享文档和共享协作空间进行划分,使得访问权限与活动阶段紧密耦合,提高了管理的灵活性。模型的应用分为三个阶段,即权限划分、活动—角色绑定、协作角色申请。

1)权限划分

协同设计中的操作对象分为文档和设计视图两大类,与此相适应,操作集、权限集乃至角色也分为两大类。一类用于对共享文档操作,另一类用于处理共享视图空间。因此,权限分配要严格区分两类操作,即操作对象是共享文档则不能采用tOPS 中的操作;若操作对象是共享视图则只能采用 tOPS 中的操作。另外,dOPS 中 open、close 操作是 read、write 操作的基础,而在 tOPS 中,view 和 share 也分别是视图操作集中的两个基础操作,构造权限、角色的过程中必须满足这一约束。

2)活动—角色绑定

设计活动被触发后,即某个用户承担了该活动并开始执行,就需要根据活动属性,为其关联角色。如果有输入文档,则给它分配的角色集合中至少有一个权限中包括 read 操作;如果有输出文档,则给它分配的角色集合中至少有一个权限中包含 write 操作。

3)协作角色申请

设计活动执行过程中,如果承担该活动的用户有协作需求,还需考虑协作的安全性。协作邀请方首先要在本地申请新角色,以支持协同工作。申请新角色的权限中至少要包括 share 操作(表示允许生成本地屏幕的共享视图),获得新角色后便可以向其他用户发送协作邀请。接收方在收到协作邀请后,根据协作邀请的程度在本地申请新角色,且新申请角色的权限中至少要包括 view 操作(表示允许查看共享视图),接收方获得新角色后即向邀请方发送应答,表示准备就绪。由此可知,协作视图仅在授权用户间可见,对于无关用户及未授权用户则是透明的。

2 模型仿真实现

为了描述并分析基于活动序列的协同设计访问控制模型的静态结构和动态行为,本章采用赋色 Petri 网描述并实现该模型,这是开发适用于协同设计中权限管理子系统的一个必要阶段。该模型能够表述协作环境中用户、活动、角色、权限、对象之间的绑定关系、权限管理策略,通过对活动序列依赖关系的建模,使其成为配置用户、申请角色、操作对象的触发器,实现多用户对共享文档及协作视图的安全访问。

赋色 Petri 网(colored Petri nets,CPNs)[12]是一种图形化的数学建模语言,将 Petri 网图形界面和 CPN ML 编程语言相融合,既便于对系统结构模型进行直观描述,同时也适合对模型进行计算机仿真和形式化分析。

2.1 自定义颜色集

模型仿真实现中定义的主要颜色集:

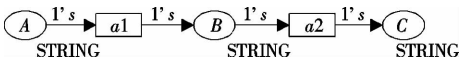
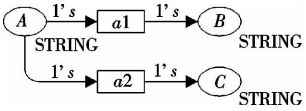
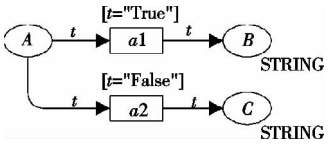
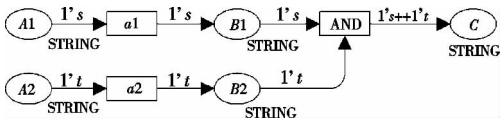
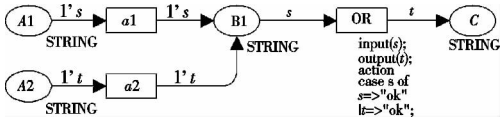
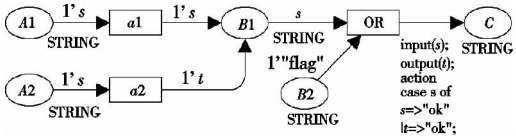
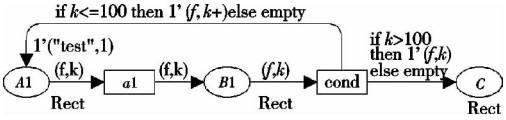
```
colset  Operation = with open | close | read | write | create | view | share |
drag | draw | erase | changeview | scrollbar | execute | mutex1 | mutex2;
colset  Permission = list Operation;//操作权限定义
colset  Role_Name = string;
colset  Role_Number = int;
colset  Role_Type = with Info_seeker | Designer |
        Requester | Viewer | Fullcontrol | Modifier |
        Userdefined | Participator;
colset  Role_pre = product Role_Type * Permission;
colset  Role = product Role_Name * Role_Type * Permission;
        //角色定义
colset  AName = string; colset  AInput = string;
colset  AOutput = string;
colset  AType = with prepare | design | evaluate;
colset  Activity = product AName * AType * AInput * AOutput;
        //活动定义
```

```
colset  User = with user1 | user2 | user3 | user4 | user5 | user6;
        //用户定义
colset  cooptype = with View | Modify | Fullcontrol;//定义协作类型
```

2.2　活动序列间的依赖关系建模

在基于活动序列的访问控制模型中,设计过程中的共享文档和共享视图按照活动序列进行划分,每一个活动包含一组相关的输入/输出文档以及可能的视图共享需求,活动之间存在可能的跟随、并发等依赖关系。采用 CPNs 描述,可以把活动间的依赖关系转换为 CPNs 变迁的依赖关系。定义 13 中七种依赖关系的 CPNs 模型如表 1 所示,图 2 所示的活动依赖关系所对应的 CPNs 模型如图 3 所示。

表 1　活动间关系的 CPNs 表示

类型	活动序列的赋色 Petri 网模型
跟随关系	
并行关系	
条件分支	
与汇集	
或汇集	
异或汇集	
循环	

活动间依赖关系涉及到活动使能条件、活动—用户绑定、用户—活动绑定、活动属性处理等,因此采用分层 CPNs 可有效地屏蔽实现细节。图 3 给出的活动依赖关系顶层 CPNs 模型中,活动用双框变迁表示,每一个活动对应一个 CPNs 子网。库所 START1、START2 和 START3 分别表示活动 1、2 和 3 的使能条件。初始阶段这三个库所中都有一个托肯,因此前三个活动可以并发执行,与这三个活动相关的访问权限也可以并发审核。普通变迁 And Join 表明当前三个活动执行完毕,并释放了各自的权限时,系统才能为活动 4 分配它所需要的权限。这样可以确保没有两个用户同时对一个协作文档进行读写操作。同时,用户仅在它所承担的活动执行期对协作对象内具有权限,一旦活动终止,它所具有的权限也就停止。其他活动的触发机理与此相同。

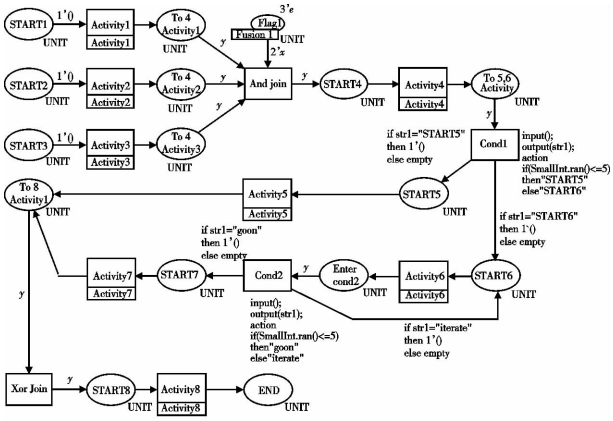


图3　活动依赖关系的赋色Petri网模型

2.3　操作—权限—角色建模

模型中对象集分为两类,相应的操作、权限和角色也都分为两类。此外,由于活动序列是所有访问权限的触发器,因此不同于传统的角色访问控制模型中直接在角色中把操作对象与权限相关联,仅用对象类型与权限相关联,并且不允许两类不同的操作、权限混合使用。图 4 创建了八类角色,分别是 Info_seeker、Designer、Requester、Viewer、Modifier、Fullcontrol、Participator,其中前两类角色用于协作文档的操作,后五类用于共享视图的操作。Userdefined 建模了角色的继承,如图 5 所示。

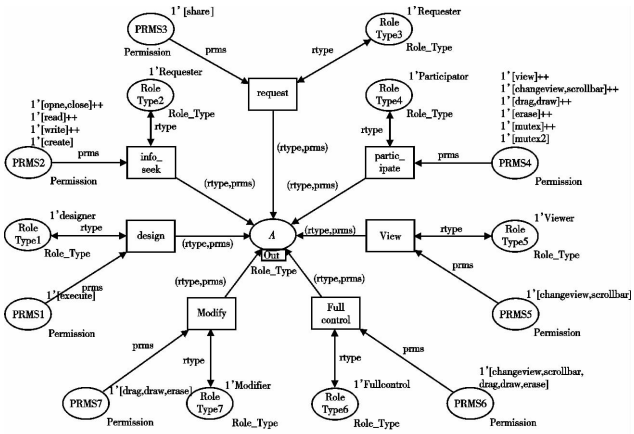


图4　基本角色生成的赋色Petri网系统

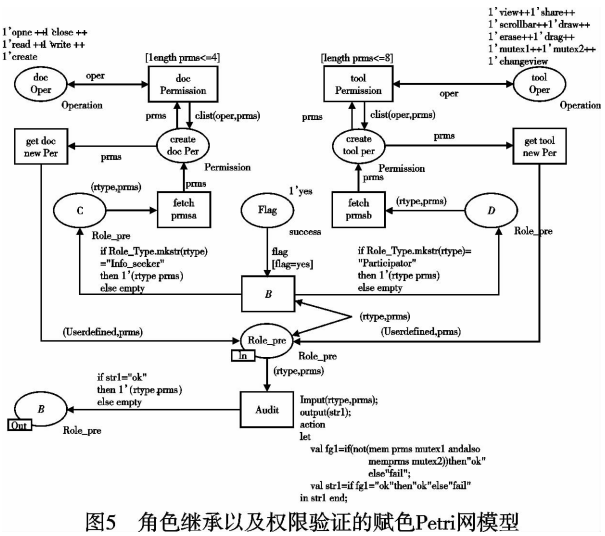


图5　角色继承以及权限验证的赋色Petri网模型

2.4 用户—活动绑定建模

当某个活动的所有先序活动均已完成,则这个活动处于就绪状态,为该活动绑定用户成为该活动触发的必要条件。用户—活动之间可以存在一对多的映射关系,表明一个用户可以承担多个活动,但是仅允许一个活动处于执行状态,这可以由活动对应变迁的偏序关系确保。例如,assigned_user($u1$,($a1$,$a2$))表明用户 $u1$ 绑定了具有跟随关系的活动序列($a1$,$a2$),根据活动 $a1$、$a2$ 在其所对应的 CPNs 模型中变迁的触发先后关系可知,这两个活动仅有一个处于执行状态,而且必须是 $a1$ 执行后 $a2$ 执行。图 6 建模了为 Activeity1 分配用户的过程,此外变迁 Activity1 的代码段还给出了该活动的输入、输出文档,分别是"$d1$、$d2$"和"$d3$"。

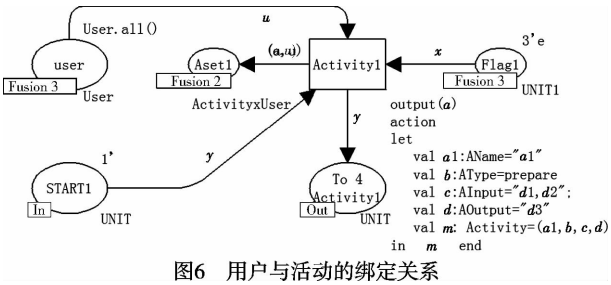


图6 用户与活动的绑定关系

2.5 活动—角色绑定建模

当活动处于执行状态时,用户需要具备一定的权限,对活动的输入/输出文档及视图进行操作。为此,关联角色就成为活动执行的一个重要步骤。活动的执行分为准备阶段和执行阶段。准备阶段为活动执行过程中使用到的共享文档分配权限;执行阶段则根据用户的需求判断是否建立协作视图,如果需要则进一步关联协作角色。在活动角色绑定的 CPNs 模型中必须满足:a)如果活动有输入文档,则关联的角色中必须有"read"权限;b)如果活动有输出文档,则关联的角色中必须有"write"权限;c)按照活动中协作对象的类型为其关联权限。图 7 给出了活动—角色绑定的 CPNs 模型,在这个模型中,库所 Aset 表示已被使能的活动和承担它的用户,库所 Role 表示系统的角色集合,变迁 Binding 根据角色、活动的类型进行绑定。变迁 Realease User 出发后,在将用户并入用户集合的同时,也会将角色返回角色集合库所 Role 中。

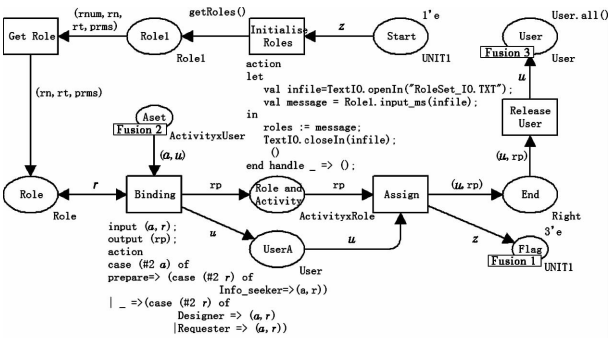


图7 活动—角色绑定的赋色Petri网模型

3 应用实例及分析

设计活动执行过程可能存在协作需求,但是在活动—角色绑定阶段对于可能出现的协作需求无法处理,因此仅考虑为共享文档设置相关的权限。当设计过程中出现协作需求时,则需按照以下步骤申请协作角色:

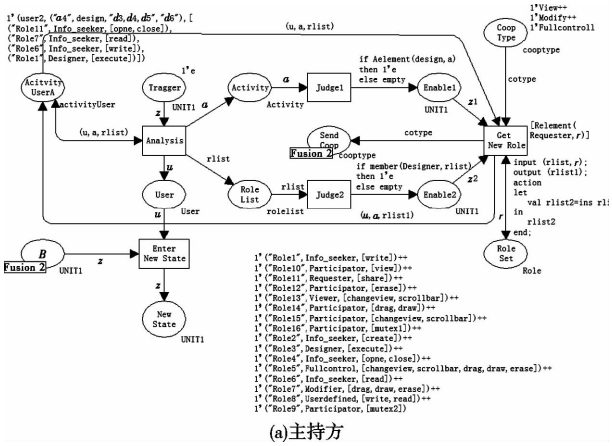
a)主持方申请协作角色,以协作工具界面为共享视图申请相应的权限,其中必须包括 share 操作,这表明主持方能够用己方设计工具界面创建协作视图,同时还应该给出协作等级(视图模式、部分修改模式、完全控制模式)。

b)寻找在线的用户,并发出协作请求。

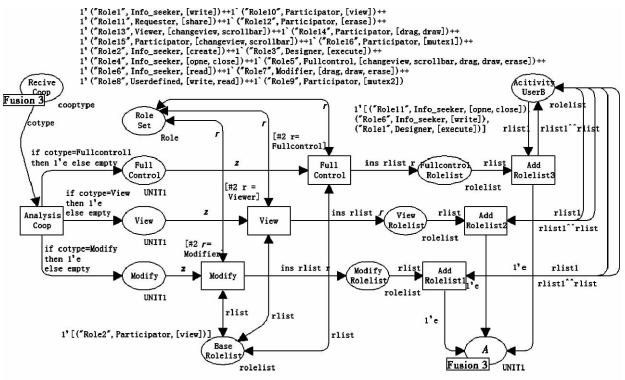
c)参与方根据协作请求的等级申请新角色,若是视图模式,仅申请具有 change view、scrollbar 等操作的权限;如果是部分修改模式则申请具有 drag、draw、erase 等操作的权限;如果是完全控制模式,则申请最大协作权限的角色。

d)参与方向主持方发送确认应答,表示申请的角色已经具备可以开始协同设计了。

主持方和参与方协作角色申请的 CPNs 模型如图 8 所示。



(a)主持方



(b)参与方

图8 协作角色申请的赋色Petri网模型

协作角色申请的 CPNs 模型涉及主持方和参与方的消息传递,为了确保协作角色申请的可靠性,需要进一步对可能导致状态不可达的结构死锁进行分析。

在 CPNs 形式化定义的基础上,给出其结构死锁的定义如下:

定义 14 设 D 为库所颜色绑定关系 $\cup_{s\in S}\{s\}\times C(s)$ 的非空子集,D 的前集定义为 $\cup_{(p,c)\in D}{}^{*}(p,c)$,$D$ 的后集定义为 $\cup_{(p,c)\in D}(p,c)^{*}$,若 ${}^{*}D\subset D^{*}$,则 D 为 CPNs 的结构死锁。

类似地,根据上述定义,可进一步给出 CPNs 中结构死锁

的判定方法,即任意库所颜色绑定集合 $D \subset \cup_{s \in S}\{s\} \times C(s)$,若 D 是结构死锁,当且仅当 $\forall (t,c) \in [\cup_{t \in T}\{t\} \times C(t)]$,$^*(t,c) \subset (PC \backslash D) \Rightarrow (t,c)^* \subset (PC \backslash D)$ 成立。图 8(a)中,设 (t,c) = (Analysis,〈u = user2, a = ("$a4$", design, "$d3$, $d4$, $d5$", "$d6$"), rlist = ("role11", "role7", "role6", "role1"), $z = e$〉),则 $^*(t,c)$ = (ActivityUserA, Tragger),且 $(t,c)^{\cdot}$ = (ActivityUserA, User, Role, List, Activity)。假设该模型中存在结构死锁 D,若库所 User $\in D$,可以推出 $^*(t,c)$ 是 $(PC \backslash D)$ 的子集,但并不能推出 $(t,c)^*$ 是 $(PC \backslash D)$ 的子集,因此库所 User $\notin D$。类似地可以推出变迁 Analysis 的所有外延库所均不属于结构死锁 D,进而可以推出图 8 中所有变迁的外延库所都不在结构死锁 D 中,这表明协作权限申请的 CPNs 模型中没有结构死锁存在。

4　结束语

本文在分析现有访问控制技术及其在协作环境中初步应用的基础上,提出了基于活动序列的访问控制模型,通过引入活动以及活动序列化的概念,把协同设计过程中的共享文档和共享视图进行划分,采用赋色 Petri 网描述该访问控制模型,实现了活动间依赖关系约束、用户—活动绑定、活动—角色绑定以及活动执行过程中协作角色申请等功能,并对协作角色申请中的死锁预防进行了分析。结果表明,该模型能够满足协同设计变化频繁的访问控制需求。

参考文献:

[1] SHEN Wei-ming, HAO Qi, LI Wei-dong. Computer supported collaborative design: retrospective and perspective [J]. Computers in Industry, 2008, 59(9): 855-862.

[2] FERRAIOLO D F, SANDHU R, GAVRILA S, *et al*. Proposed NIST standard for role-based access control [J]. ACM Trans on Information and System Security, 2001, 4(3): 224-274.

[3] SEJONG O, SEOG P. Task-role-based access control model [J]. Information Systems, 2003, 28(6): 533-562.

[4] LU H C, CHU Y P, SHEU R K, *et al*. A generic application sharing architecture based on message-oriented middleware platform [C]// Proc of the 10th International Conference on Computer Supported Cooperative Work in Design. 2006: 1-5.

[5] 张鹏程, 李人厚, 秦明, 等. 混合式应用共享机制模型的研究 [J]. 小型微型计算机系统, 2003, 24(7): 1124-1127.

[6] DEWAN P, SHEN H H. Flexible Meta-access control for collaborative system [C]//Proc of ACM Conference on Computer Supported Cooperative Work. New York: ACM Press, 1998: 247-256.

[7] BULLOCK A, BENFORD S. An access control framework for multi-user collaborative environments [C]//Proc of International ACM SIGGROUP Conference on Supporting Group Work. New York: ACM Press, 1999: 140-149.

[8] CERA C D, KIM T, HAN J H, *et al*. Role-based viewing envelopes for information protection in collaborative modeling [J]. Computer Aided Design, 2004, 36(9): 873-886.

[9] 李俊青, 李新友, 谢圣献, 等. P2P 网络动态精细粒度访问控制研究 [J]. 计算机应用研究, 2009, 26(4): 1467-1450.

[10] AHMED T, TRIPATHI A R. Security policies in distributed CSCW and workflow systems [J]. IEEE Trans on Systems, Man, and Cybernetics Part A: Systems and Humans, 2010, 40(6): 1220-1231.

[11] PARK H, HONG J W, PARK J H, *et al*. Combined authentication-based multilevel access control in mobile application for daily life service [J]. IEEE Trans on Mobile Computing, 2010, 9(6): 824-837.

[12] JENSEN K, KRISTENSEN L M, WELLS L. Coloured Petri nets and CPN tools for modelling and validation of concurrent systems [J]. International Journal on Software Tools for Technology Transfer. 2007, 9(3-4): 213-254.