主动授权管理中的关联继承机制

翟治年[1]，奚建清[1]，卢亚辉[2]，汤德佑[1]，郭玉彬[3]，韦凯[1]
(1.华南理工大学计算机科学与工程学院，510006，广州；2.深圳大学计算机与软件学院，518062，广东深圳；
3.华南农业大学信息学院，510640，广州)

摘要：为解决主动授权管理模型中普遍和长期存在的任务间重复授权问题，提高工作流授权的伸缩性，提出一种基于职能关联的权限继承机制.通过对传统的角色任务指派关系进行深入挖掘，定义了职能关联概念及其特化与管理关系；根据职能关联的分量和属性给出了2种关系的推导规则，并按2种关系上的继承性区分业务性和管理性授权，从而建立了可自动配置的授权继承结构.论文评审工作流上的验证结果表明，较之现有各类模型，文中提出的模型以10%以下的额外手工操作减少了53%以上的授权操作，较好地克服了任务间重复授权这一缺陷，取得了明显的授权伸缩性优势.同时，该模型还提供对任务内多角色协作的支持.

关键词：访问控制；授权；工作流；任务；角色；伸缩性；继承
中图分类号：TP309 **文献标志码**：A **文章编号**：0253-987X(2012)04-0024-08

Association Inheritance Mechanism in Active Authorization Management

ZHAI Zhinian[1], XI Jianqing[1], LU Yahui[2], TANG Deyou[1], GUO Yubin[3], WEI Kai[1]
(1. School of Computer Science and Engineering, South China University of Technology, Guangzhou 510006, China;
2. School of Computer and Software, Shenzhen University, Shenzhen, Guangdong 518062, China;
3. College of Informatics, South China Agriculture University, Guangzhou 510640, China)

Abstract: A permission inheritance mechanism based on function associations is proposed to address the widespread and long-stand issue of repetitive authorizations among tasks in active authorization management models and to enhance the authorization scalability in workflow systems. The definition of function association with specialization and management relations is given by analyzing the traditional relation of role-task assignment. Inference rules about the two relations on function associations are obtained from the components and attributes of function associations, then authorizations are divided into two types of business and management based on the inheritance on the two relations. Thus auto-configurable authorization inheritance structures are established. Comparisons with existing models for a paper review workflow show that authorization operations are reduced by more than 53% with only 10% or less extra manual operations. Therefore, repetitive authorizations among tasks are effectively reduced, and distinct superiority in authorization scalability is achieved. Meanwhile, the collaboration of multiple roles within a task can be supported.

Keywords: access control; authorization; workflow; task; role; scalability; inheritance

---

收稿日期：2011-08-25. 作者简介：翟治年(1977—)，男，博士生；奚建清(通信作者)，男，教授，博士生导师. 基金项目：国家自然科学基金资助项目(60903114，60973100)；广东省自然科学基金资助项目(10351806001000000)；粤港关键领域重点突破资助项目(2006B80407001)；广东省教育部产学研结合计划资助项目(2008B090500193)；深圳市科技计划基础研究资助项目(JC201005280402A，JC200903120046A)；中央高校基本科研业务费专项资金资助项目(2009ZM0162).
网络出版时间：2012-02-22 网络出版地址：http://www.cnki.net/kcms/detail/61.1069.T.20120222.1012.003.html

主动授权管理(active authorization management, AAM)是在工作流技术驱动下发展起来的访问控制范型[1]，其基本思想是将最小特权原则贯彻到业务执行过程中，随着任务的开始和结束动态地授予和撤销有关权限[1-2]. 企业级业务过程环境中存在大量的任务、权限和用户，如何降低数据规模带来的管理开销是 AAM 面临的重要挑战. 易管理与可伸缩系统中的一大障碍是重复信息，对访问控制而言主要体现为重复授权. RBAC(role based access control)模型[3]为被动环境访问控制提供了可伸缩的解决方案，现已广泛应用于大型系统和企业环境，并形成了 NIST 标准. 以角色为间接层可简化权限与用户的多对多关联，根据角色层次关系可有效化简角色间的重复授权. 然而，主动环境以业务过程为中心，从中分解出大量任务，在工作流管理系统调度下有序执行. 由于用户不能根据身份随意访问对象，任务是最基本和自然的授权依据. 在同一过程或子过程中，任务的权限需求往往存在交集，从而引起重复授权，维护时又不得不重复修改和删除，造成了严重的管理负担. AAM 出现于 20 世纪 90 年代，发展至今已取得了丰硕的成果，然而已有的模型多强调灵活性而忽视了伸缩性. 以 TBAC[1]为代表的一类模型以任务为间接层进行用户授权，但是由于任务数量庞大而且较不稳定，所以授权和任务分配负担仍然很严重. T-RBAC[4]为此类模型引入角色，建立了广为沿用的权限-任务-角色-用户三步授权模式，但它只能简化用户的任务分配，不能消除任务间的重复授权. 很多按任务授权的模型定义了任务层次，但是未见以此来消除重复授权的讨论. 事实上它们的这种作用非常有限，原因在于任务层次是一种结构组成关系，而一组任务间的重复授权往往是多样的，需要以某种泛化的职责来描述. 作为父任务的组成部分，子任务的职责必须更为具体，因而很难胜任这一要求. 例如，某软件项目包括框架、组件 1～k 等多个开发任务，它们均需要(计划，读)、(需求，读)、(设计，读)、(代码，读)、(编译脚本，执行)和(软件部署，写)等权限，由于这些权限的多样性，很难将它们分离到一个粗粒度的公共子任务中. 若以读计划、读需求、读设计、读代码、构建等多个子任务来描述，则只是将原来的重复授权变成了略少的重复子任务，实际上并未有效解决问题. 还有一类 AAM 模型，授予权限时考虑任务和角色两个因素，可支持任务内多角色协作，即团队任务[5]. 例如：文献[6-7]根据角色-任务指派关系授权；文献[8]在授权许可中根据任务和权限(集)指定可授权角色集；文献[9]通过角色授权策略对满足情景约束的角色授予任务权限；等等. 其中，文献[7-8]允许任务执行期间有关角色的授权被其上级角色继承，有利于消除任务内部的重复授权. 但是，这些模型都缺乏更有效的继承机制，不能化简更常见的任务间重复授权. 总的来说，由于未克服任务间重复授权的缺陷，AAM 的授权伸缩性还相当薄弱，这将导致系统难以商用，从而阻碍其他特性的发挥. AAM 在实用和标准化程度上落后于 RBAC，与此也有很大关系.

针对上述问题，本文提出一种基于关联继承的主动授权(association inheritance based active authorization, AIBAA)模型，以任务和角色构成的职能关联为授权依据，借助其特化和管理关系，为化简任务之间和内部的重复授权提供了有效手段，可显著提高 AAM 的伸缩性，并支持团队任务特性.

1　AIBAA 模型

任务和角色是大家熟知的概念，也是两种最基本的分工方式. 通过对其属性和关系的挖掘，AIBAA 将角色-任务指派关系发展为职能关联这一概念，得出特化和管理两种关联继承机制，以此为中心建立起如图 1 所示的访问控制框架. 权限和用户均被指派给职能关联，其中授权有业务性和管理性两种方式. 当某任务实例化时，通过指定在该任务的实例中担任某角色的用户，完成相应职能关联的实例化，用户在实例执行时可用的权限取决于职能关联的授权和继承情况. 在访问阶段，用户发起会话，通过激活所分配的实例和职能关联来激活可用权限，在激活权限的控制下进行对象访问.

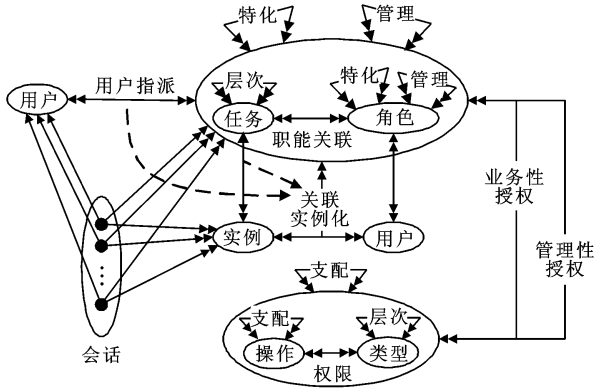

图 1　AIBAA 模型组件

1.1　授权管理

授权管理是访问控制模型的基本功能，也是运

行阶段进行访问检查的依据.本节将阐述职能关联的概念及其特化和管理关系在授权管理中的作用.

1.1.1 角色及其关系 本小节将通过划分角色特化和管理关系,为建立相应的关联继承机制奠定基础.

定义1 角色 $r(e,m)$是职责与能力的统一.

e:Boolean 表示角色的实体性. true 表示实体角色,可以直接分配用户;false 表示抽象角色,不可直接分配用户.例如,项目成员是抽象角色,分析员、设计师、程序员等是实体角色,前者不能直接分配用户,因为任何参与项目的人员都有具体分工,因而必须分配给后者之一.类似地,也不存在将用户分配给部门经理而不指定具体部门的情况.实体性是相对的,例如任何开发人员均掌握某种语言技能,在强调这一点时,应将其分配给 C++、Java 程序员等实体角色,而非程序员这一抽象角色.

m:Boolean 表示角色的管理职能. true 表示管理性角色,例如项目经理;false 表示业务性角色,例如程序员.某些角色可能有双重性质,例如开发组长负责开发管理,也承担编程工作.这种情况可通过角色约束来避免,例如将开发组长设置为管理性角色,但以程序员为前提角色[3].

若用户 u 是角色 r 的直接成员,记为 $u\in r$.

定义2 角色管理 $RM\subseteq R\times R$,偏序,记为$\geqslant_m$,表示角色之间一种具有命令、监督和奖惩等性质的关系. $r'\geqslant_m r$ 表示 r' 和 r 是管理上下级关系,称 r' 为 r 的上司角色,r 为 r' 的下属角色.例如,项目经理是分析员和开发组长的上司,销售经理是销售员和销售助理的上司.管理关系不同于职位或行政级别,例如销售部经理是比财务部出纳更高级别的职位,但两者没有管理和被管理的关系. $r'>_m r\Rightarrow r'$. $m=$true,即只有管理性角色才能成为其他角色的上司(但一个角色可视为自身的上司).

定义3 角色特化 $RS\subseteq R\times R$,偏序,记为$\geqslant_s$,表示角色之间的类属关系. $r'\geqslant_s r$ 表示 r' 和 r 的特殊一般关系,称 r' 为 r 的特化角色,r 为 r' 的泛化角色.例如,框架程序员、Java 程序员和高级程序员都是程序员的特化.特化关系可以从多种角度构造,例如在软件公司中,根据组织机构可定义程序开发部和公司员工,根据工龄可定义 1~3 年和不足 5 年的员工,根据薪级可定义 band5 以下和 band6~10 之间的员工,根据任务层次可定义测试相关人员和项目成员,等等. $r'>_s r\Rightarrow r$. $e=$false,即抽象角色才能特化为自身之外的角色.

1.1.2 任务及其关系 本小节将阐述关联继承涉及的任务属性和关系.

定义4 任务 $t(e)$是工作的基本单元.

e:Boolean 表示任务的实体性,即是否可执行. true 表示可执行任务,可以实例化并分配给用户;false 表示不可执行任务,不可实例化和分配给用户.例如,在订单处理流程中,信用审核是可执行任务,而订单处理本身也可视为一个任务,只是其具体工作被分解到库存检查、信用审核等等可执行任务中,因而自身是不可执行的.为了支持流程中的集中式管理,某些已被分解的任务仍然是可以执行的.例如,软件项目已分解为分析、设计、开发和测试等,但仍可以被执行,通过它项目经理可以管理下面的各项工作.

若实例 i 是任务 t 的一次执行,记为 $i\in t$.实例 i 在时刻 τ 的执行状态记为 $s(i,\tau)$.为简单起见,可以取两种状态——valid 表示正在执行,invalid 表示未开始或已完成.

定义5 任务层次 $TH\subseteq T\times T$,偏序,表示任务之间的组成关系. $t>t'$ 表示 t 和 t' 的整体部分关系,称 t 为 t' 的外层或上级任务,t' 为 t 的内层或下级任务.直接上、下级任务也称为父、子任务.

1.1.3 职能关联及其关系 本小节将角色-任务指派关系发展为完整的职能关联概念,并给出其特化与管理关系的推导规则.

定义6 职能关联 $a(t,r,e,m,i_m)$简称关联,表示何种角色参与哪个任务,即角色到任务的指派关系.虽然任务和角色均表示某种职责,并需要某种能力,但二者的倾向不同.任务是某种事项,或根据事项内容划分的职责,而角色是人员的抽象,或根据人员能力划分的职责.指派角色去完成任务,可以进一步明确工作内容及其人员资格,二者共同描述了一种完整的职能,所以职能关联也称为复合职能.

t:T 是关联的任务分量,表示其工作职责或内容.同一任务可能由不同的角色进行协作.

r:R 是关联的角色分量,表示其所需的人员能力或资格.同一角色可能在不同的任务中承担职责.

e:Boolean 表示关联的实体性,即是否可执行.由于一个任务可能由多个角色协作,因此在一般意义上,用户执行的是特定角色在该任务中的工作,即关联,而非整个任务. true 表示可执行关联,false 表示不可执行关联,例如 C++程序员参与通信组件开发是可执行关联,而程序员参与视频服务开发是不可执行关联. $a.e=$true$\Rightarrow a.t.e=$true,即可执行关

联的任务分量必须可执行. 对可执行任务可指派任何(实体或抽象)角色,但由此形成的关联不一定是可执行的,如项目总结会议是可执行任务,项目中的各种角色均需参加会议并承担职能——设计师介绍架构,测试员演示系统,项目经理作总结报告,等等,均为可执行关联;会议成员角色参与该会议则无具体职能,从而不是可执行关联. 不可执行任务及其角色指派完全是根据授权管理需要引入的,没有实际工作内容,因此不能生成可执行关联.

m:Boolean 表示关联有无管理职能. true 表示有管理性的关联,例如将项目经理指派给项目任务;false 表示纯业务性关联,例如将设计师指派给概要设计. 管理性角色在某些任务中不一定承担管理职能,例如开发组长在项目例会中只是普通参与者.

i_m:Boolean 表示关联的管理继承性. true 表示若下属角色可以执行该关联,则上司角色也有权执行,例如销售员在信用审核任务中的工作可由销售经理代替执行;false 表示上司角色不能代替下属角色执行的关联,例如测试员在测试任务中的工作不能由项目经理代替执行. 需要注意,管理性关联未必是管理可继承的,例如开发组长在组件开发任务中进行的管理工作,未必是项目经理可以代替执行的.

定义 7　关联管理 $AM \subseteq A \times A$,偏序,记为 $\geqslant_m$,表示职能关联之间一种具有命令、监督和奖惩等性质的关系. $a' \geqslant_m a$ 表示 a' 和 a 的管理上下级关系,称 a' 为 a 的上司关联,a 为 a' 的下属关联,例如(项目,项目经理)$\geqslant_m$(组件开发,程序员). 职能关联是一种复合职能,所以彼此可以存在管理关系. 关联管理和角色管理的区别在于它有作用域限制,例如组件 1 开发任务由高级程序员带领若干实习程序员完成,(组件 1 开发,高级程序员)是(组件 1 开发,实习程序员)的上司关联,但脱离这一任务背景,两个角色之间并没有管理关系.

定义 8　关联特化 $AS \subseteq A \times A$,偏序,记为 $\geqslant_s$,表示职能关联之间的类属关系. 职能关联不仅是一种关系,而且是一个实体概念,因此职能关联集上必然存在类属关系. $a' \geqslant_s a$ 表示 a' 和 a 的特殊一般关系,称 a' 为 a 的特化关联,a 为 a' 的泛化关联,例如(框架开发,高级程序员)$\geqslant_s$(开发,程序员).

基本的关联管理和特化关系可以根据以下规则自动配置.

规则 1(关联管理)

$$a'.t \geqslant a.t \wedge a'.r \geqslant_m a.r \wedge a'.m = \text{true} \Rightarrow a' \geqslant_m a$$

式中:a'、a 表示上司角色在外层任务中行使管理职能和下属角色参与内层任务. 由于后者在前者管辖范围内,角色管理关系必然延续到职能关联之间.

例如,项目经理在项目任务中进行整个项目的集中管理,因此程序员在组件 1 开发任务中从事编程工作、开发组长负责开发工作的管理、项目经理在测试任务中对某些 bug 的处理进行决策,都是前者的下属情形.

规则 2(关联特化)

$$a'.t \leqslant a.t \wedge a'.r \geqslant_s a.r \wedge a'.m = a.m = \text{false} \wedge a.e = \text{false} \Rightarrow a' \geqslant_s a$$

式中:a'、a 分别特化角色在内层任务中承担业务职能和泛化角色在外层任务中承担业务职能,且后者为不可执行关联,从而前者是后者的特殊情形.

例如,C++程序员负责通信组件开发和 Java 程序员负责播放插件开发均为程序员参与视频服务开发的特殊情形. 此规则要求 a'、a 均无管理职能且 a 不可执行,这样可以避免一些语义上的不确定性,例如 a'(组件 1 开发,开发组长)与 a(组件开发,开发组长)的关系:如果开发组长在组件 1 和组件 3 开发任务中对代码进行技术审查,由此抽象出关联 a,则 $a'.m = a.m = \text{false}$,且 $a.e = \text{false}$,a' 必然是 a 的特化关联,然而如果开发组长在组件开发任务中进行计划制定与进度检查等管理工作,即 $a.m = \text{true}$,则无论 $a'.m$ 取值如何,a' 都是 a 的下属关联. 规则 2 的定义方式避免了在后一种情况下错误地将 a' 识别为 a 的特化关联. 另外需要强调的是,职能关联的角色分量应具有独立含义,例如(设计,经理)表示经理角色参与设计任务这一关系,而非两个分量共同构成设计负责人这一角色. 在这里,经理可以是抽象角色,或根据上下文解释为项目经理、部门经理和总经理等,但均是一个独立完整的角色.

此外,也可手动配置一些特殊的关联关系.

1.1.4　权限及其关系　本小节借鉴文献[9]给出权限的相关定义,这样可以避免对按层次组织的大量客体重复定义权限.

定义 9　类型集 Y. 类型是用同样一组属性来描述的数据对象.

定义 10　类型层次 $YH \subseteq Y \times Y$,偏序,$y \geqslant y'$ 表示 y 和 y' 的整体部分关系.

定义 11　操作支配 $XD \subseteq X \times X$,偏序,$x \geqslant x'$ 表示 x 以 x' 为步骤或前提. 例如,文档标注(l)是写入(w)的可选步骤,写入和标注均以读取(r)为前提,故 $w > l > r$.

定义 12 权限集 $P\subseteq Y\times X$. 权限是对一类对象的操作许可,如$(y,x)\in P$ 表示对 y 类型的对象执行 x 操作的许可. 这里将权限定义在类型而非对象上,因为工作流定义阶段只能确定所涉及的类型,到案例执行时,才能将类型解释为具体对象.

定义 13 权限支配 $\mathrm{PD}\subseteq P\times P$. $((x',y'),(x,y))\in\mathrm{PD}\Leftrightarrow x'\geqslant x\wedge y'\geqslant y$,易证 PD 为 P 上的偏序.

1.1.5 *授权与资源分配* 本小节将阐述基于职能关联的授权和资源分配(为实例执行分配人力资源).

定义 14 授权 $\mathrm{PA}=\mathrm{BPA}\cup\mathrm{MPA}$.

业务性授权 $\mathrm{BPA}\subseteq P\times A$,多对多关系. $(p,a)\in\mathrm{BPA}$ 表示根据业务需要将权限 p 授予职能关联 a,可被其特化关联继承. 例如,将(需求文档,读)作为项目成员参与项目的业务性授权,则它可被程序员、设计师等在各自的任务中继承. 管理性关联也可进行业务性授权. 例如,项目经理、设计负责人和开发组长均为管理人员,分别在项目、设计和开发任务中行使管理职能,可将各种管理人员公有的权限,如(绩效考核细则,读),作为管理人员参与项目这一管理性关联的业务性授权,并将前述各种管理关联配置成此关联的特化,以免重复授权.

管理性授权 $\mathrm{MPA}\subseteq P\times A$,多对多关系. $(p,a)\in\mathrm{MPA}$ 表示根据管理需要将权限 p 授予职能关联 a,可被其上司关联继承. 例如,将(开发计划,读)作为开发组长在开发任务中的管理性授权,则它可被项目经理在项目任务中继承. 业务性关联也可进行管理性授权,例如将(bug,读)作为测试员在测试任务中的管理性授权,则它可被项目经理在项目任务中继承.

称 $\mathrm{DPA}=\mathrm{BPA}\cap\mathrm{MPA}$ 为双重授权.

定义 15 资源分配 $\mathrm{RA}\subseteq U\times A$,多对多关系. $(u,a)\in\mathrm{RA}$ 表示 u 是 a 的候选执行者,即有可能指派用户 u 以角色 $a.r$ 执行任务 $a.t$ 的实例. $(u,a)\in\mathrm{RA}\Rightarrow a.e\wedge\exists r\geqslant_{\mathrm{s}}a.r(u\in r)\vee a.i_{\mathrm{m}}\wedge\exists r\geqslant_{\mathrm{m}}a.r(u\in r)$,即 a 必须是可执行的,而其执行者应来自 $a.r$ 或其特化的成员,或者当 a 管理可继承时,也可来自 $a.r$ 上司的成员.

定义 16 关联实例化 $\mathrm{AI}\subseteq A\times I\times U$. $(a,i,u)\in\mathrm{AI}$ 表示由用户 u 在实例 i 中执行关联 a. $(a,i,u)\in\mathrm{AI}\Rightarrow i\in a.t\wedge(u,a)\in\mathrm{RA}$,即 i 必须是 $a.t$ 的实例,且 u 是 a 的候选执行者.

关联实例化为实例中的关联指定了执行用户,而实例执行权限由关联的授权状态决定,其规则如下.

规则 3(关联继承)

$$p\in\mathrm{perms}(a)\Leftrightarrow\exists a'\leqslant_{\mathrm{s}}a(p,a')\in\mathrm{BPA}\vee\exists a'\leqslant_{\mathrm{m}}a(p,a')\in\mathrm{MPA}$$

即权限 p 可通过关联 a 激活,当且仅当它是 a 或其特化的业务性授权,或者是 a 或其上司的管理性授权,其中 $\mathrm{perms}(a:A)\rightarrow 2^P$ 表示可通过关联 a 激活的权限集. AIBAA 的权限激活机制将在下面阐述.

1.2 访问检查

为了避免每次访问都进行权限计算,AIBAA 引入了基于职能关联的权限激活机制. 在会话中,用户首先激活自己分配的实例,然后激活实例上的职能关联,根据职能关联激活授权. 用户可将已激活的实体去活. 若实例失效或去活,则其激活的职能关联随之去活;若职能关联去活,则由其激活的权限随之去活. 用户可随时激活/去活各种实体. 处于激活状态的权限可高效访问.

以下函数描述了会话的瞬时授权状态.

$\mathrm{insts}(s:S,\tau:\Gamma)\rightarrow 2^I$,表示会话 s 在时刻 τ 激活的实例集. $i\in\mathrm{insts}(s,\tau)\Rightarrow s(i,\tau)=\mathrm{valid}\wedge\exists a(a,i,u(s))\in\mathrm{AI}$,即会话中处于激活状态的实例必须是正在执行的,且该实例中某个关联被分配给会话用户来执行.

$\mathrm{dir\_assos}(s:S,\tau:\Gamma)\rightarrow 2^A$,表示会话 s 在时刻 τ 直接激活的关联集. $a\in\mathrm{dir\_assos}(s,\tau)\Rightarrow\exists i\in\mathrm{insts}(s,\tau)(a,i,u(s))\in\mathrm{AI}$,即会话中直接激活的关联必须是在已激活实例中分配给会话用户执行的关联.

$\mathrm{gen\_assos}(s:S,\tau:\Gamma)\rightarrow 2^A$,表示会话 s 中在时刻 τ 根据特化关系激活的关联集,$a\in\mathrm{gen\_assos}(s,\tau)\Rightarrow\exists a'\in\mathrm{dir\_assos}(s,\tau)a'\geqslant_{\mathrm{s}}a$.

$\mathrm{man\_assos}(s:S,\tau:\Gamma)\rightarrow 2^A$,表示会话 s 中在时刻 τ 根据管理关系激活的关联集,$a\in\mathrm{man\_assos}(s,\tau)\Rightarrow\exists a'\in\mathrm{dir\_assos}(s,\tau)a'\geqslant_{\mathrm{m}}a$.

$\mathrm{perms}(s:S,\tau:\Gamma)\rightarrow 2^P$,表示会话 s 中在时刻 τ 激活的权限集. $p\in\mathrm{perms}(s,\tau)\rightarrow 2^P\Rightarrow\exists a\in\mathrm{gen\_assos}(s,\tau)(p,a)\in\mathrm{BPA}\vee\exists a\in\mathrm{man\_assos}(s,\tau)(p,a)\in\mathrm{MPA}$,即被激活的权限必须是根据特化关系所激活关联的业务性授权,或者根据管理关系所激活关联的管理性授权.

2 相关工作比较

办公自动化是典型的工作流应用,这里选择一论文评审流程来检验各 AAM 模型的授权伸缩性.

2.1 场景描述

某期刊的审稿流程如图 2 所示. 作者提交初稿

后,经过收稿、初审、外审和终审 4 个阶段的审查才能录用. 其中:外审并行送 2 名专家,编辑部分别跟踪,若有缺审则另选专家;收稿和终审阶段可能要作者修改后重审. 有关角色在任务中的职责如表 1 所示,其中主编(m_editor)和责编(c_editor)均为编辑(editor)的特化,且前者为后者的上司. 涉及的对象类型如下:模板(t);稿件(p),稿签(i);收稿通知(n_g),退稿通知(n_r),录用通知(n_a);初审决定(d_1),终审决定(d_3);审稿单(f),审稿单 1(f_1),审稿单 2(f_2);审改稿(r),审改稿 1(r_1),审改稿 2(r_2);收稿意见(o_0),初审意见(o_1),返修意见(o_3);终审建议(s);修改说明(m);专家列表(l);处理时间表(t_p). 其中,审稿单一式两份($f>f_1,f_2$),由 2 名专家(reviewer)独立填写意见. 类似地,审改稿 $r>r_1,r_2$. 修改后复审时,专家、主编和责编分别在原审稿单、终审决定和终审建议上填写新的意见. 处理时间表记录每个任务的开始、预期和实际结束时间,由系统自动维护,供用户了解进度或掌握时间. 对象上的操作

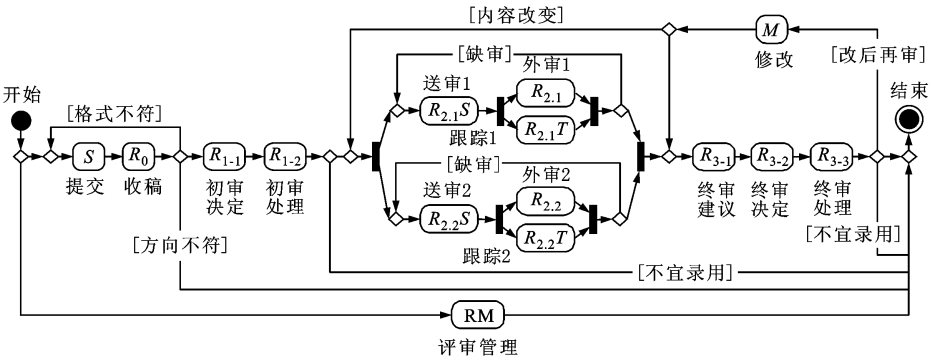

图 2　论文评审流程

表 1　论文评审的角色分工

角色	任务	职责
author	提交	提交匿名稿件并在稿签中填写基本信息;根据模板和收稿意见调整稿件格式.
	修改	根据返修意见修改稿件,写出修改说明.
c_editor	收稿	检查已提交论文,对不在收稿范围者发退稿通知,对格式不符者给出收稿意见,经修改符合要求后发收稿通知.
	初审处理	根据初审意见,对不合格者发退稿通知.
	送审	根据稿签和稿件,从列表中选择相关专家,发审稿单送审,与专家沟通后商定返回时间;修改后复审时必须送原专家.
	跟踪	在返回意见或确定缺审之前,若专家在预定期限内未返回意见,则发信催促.
	终审建议	根据稿件、稿签和 2 份专家审稿单给出处理建议.
	终审处理	根据主编决定,对需要修改者综合外审和终审情况给出返修意见,达到标准者发录用通知,不合格者发退稿通知.
		终审修改后需要根据修改稿、修改说明等材料进行复审.
m_editor	初审决定	根据稿件、稿签,决定是否可以外审.
	终审决定	根据稿件、稿签、2 份专家审稿单和责编建议给出处理决定. 终审修改后需要复审.
	评审管理	查看投稿和处理情况,推动工作进展.
reviewer	外审	评审稿件并填写审稿单,有必要时另附审改稿. 终审修改后,如果稿件有内容变动,需进行复审.

有读(r)、写(w)和创建(c),且 $w,c>r$.

2.2　对比模型

选择表 2 列出的模型为比较对象,其中 T-RBAC 是三步授权模式的代表,其余均为综合任务与角色进行授权的模型:WAM 是早期的 AAM 代表模型,其授权模板中允许使用角色;文献[6-7]模型是较早根据角色任务指派关系授权的模型;文献[8-9]模型是近几年提出的模型. 设角色 r_1 和 r_2 分别在任务 t_1 和 t_2 中有 p_1、p_2、p_3 和 p_4、p_5 权限,各模型的授权方式如表 2 所示,其授权次数均等于角色-任务-权限三元关联的数量,事实上,若每个任务均由单个角色执行,则此关系总是成立.

2.3　对比模型授权方案

论文评审流程符合前述假设,各种对比模型的授权方案可统一用三元关联形式来表达. 由于比较的主要指标是不同任务之间的授权重复程度,故采用此类场景对所有模型都是公平的. 对比模型的授权方案如表 3 所示,共 81 次授权,有 47 次重复,重复率为 58%. 若每份稿件送 k 个外审,则总授权次数为 $12k+57$.

2.4　本文模型授权方案

建立如图 3 左下和右下所示的任务层次和角色关系,进而得到图 3 上方的职能关联关系,其中 R 是可执行的,m_editor 在其中进行评审管理. 然后,

表 2 现有各种模型的授权方式示例

模型	授权示例
WAM[2]	$AT_1(t_1)=(r_1,(p_1.y,-),p_1.x),\cdots,AT_3(t_1)=(r_1,(p_3.y,-),p_3.x)$; $AT_1(t_2)=(r_2,(p_4.y,-),p_4.x),AT_2(t_2)=(r_2,(p_5.y,-),p_5.x)$
T-RBAC	TRA=$\{(r_1,t_1),(r_2,t_2)\}$; PTA=$\{(p_1,t_1),(p_2,t_1),(p_3,t_1),(p_4,t_2),(p_5,t_2)\}$
文献[6]模型	RoleTaskAssignment=$\{(r_1,t_1),(r_2,t_2)\}$; PermissionAssignment2=$\{((r_1,t_1),p_1),((r_1,t_1),p_2),((r_1,t_1),p_3),((r_2,t_2),p_4),((r_2,t_2),p_5)\}$
文献[7]模型	TUA=$\{(t_1,r_1),(t_2,r_2)\}$; AA=$\{((t_1,r_1),p_1),((t_1,r_1),p_2),((t_1,r_1),p_3),((t_2,r_2),p_4),((t_2,r_2),p_5)\}$;$A=\{\{\mathrm{id}_1,\mathrm{id}_{t1},p_1,k_{r1},\text{necessity},\varnothing\},\cdots\}$
文献[8]模型	AA=$\{(t_1,p_1,k_{r1},\text{inheritability},\varnothing),\cdots,(t_2,p_4,k_{r2},\text{inheritability},\varnothing),\cdots\}$
文献[9]模型	R_AUTHP=$\{\text{role\_name}=r_1\rightarrow(t_1,p_1,+,\text{inheritability}),\cdots,\text{role\_name}=r_2\rightarrow(t_2,p_4,+,\text{inheritability}),\cdots\}$

表 3 论文评审实例的现有授权方案

角色	任务	权限	N
author	S	$(i,w),(p,w),(t,r),(o_0,r)$	4
	M	$(i,w),(p,w),(m,w),(t,r),(o_3,r),(r,r)$	6
c_editor	R_0	$(o_0,w),(n_g,w),(n_r,w),(t,r),(i,r),(p,r),(p_t,r)$	6
	$R_{1\text{-}2}$	$(n_r,w),(d_1,r),(i,r),(p,r),(t_p,r)$	5
	$R_{2.1}S$	$(l,r),(f_1,c),(i,r),(p,r),(t_p,r)$	6
	$R_{2.2}S$	$(l,r),(f_2,c),(i,r),(p,r),(t_p,r)$	5
	$R_{2.1}T$	$(l,r),(f_1,r),(t_p,r)$	3
	$R_{2.2}T$	$(l,r),(f_2,r),(t_p,r)$	3
	$R_{3\text{-}1}$	$(s,w),(f,r),(r,r),(i,r),(p,r),(m,r),(o_3,r),(t_p,r)$	8
	$R_{3\text{-}3}$	$(o_3,w),(n_a,w),(n_r,w),(d_3,r),(f,r),(r,r),(i,r),(p,r),(m,r),(t,r),(s,r),(t_p,r)$	12
m_editor	RM	$(i,r),(t_p,r)$	2
	$R_{1\text{-}1}$	$(d_1,w),(i,r),(p,r),(t_p,r)$	4
	$R_{3\text{-}2}$	$(d_3,w),(s,r),(f,r),(r,r),(i,r),(p,r),(m,r),(o_3,r),(t_p,r)$	9
reviewer	$R_{2.1}$	$(f_1,w),(r_1,w),(p,r),(m,r)$	4
	$R_{2.2}$	$(f_2,w),(r_2,w),(p,r),(m,r)$	4

给出表 4 所示的授权方案，其效果与对比方案相同(获权数量未必相等，例如$(R_{3\text{-}1},\text{c_editor})$同时获得$(s,w)$和$(s,r)$，对比方案中 c_editor 在 $R_{3\text{-}1}$ 中只获得(s,w))，但只需 38 次授权，有 6 次重复，重复率为 16%，较之对比方案，授权减少了 53%，重复率降低了 42%. 对 k 个外审的情况，则需 $4k+30$ 次授权，较对比方案减少了$(8k+27)/(12k+57)>53\%$. AIBAA 引入的额外开销是 8 个不可执行关联、1 个关联管理和 19 个关联特化关系，每个关系都需要 1 次创建操作，其中的关联管理/特化关系可根据规则 1、2 自动配置，故额外的手工操作为 8 次，仅为对比方案授权次数的 10%. 对 k 个外审的情况，额外开销是$(3k+22)$次操作，其中手工操作仍为 8 次，为对比方案授权次数的 $8/(12k+57)<10\%$. 这样，AIBAA 方案以低于 10%的额外手工开销，消除了超过 53%的多余授权，在总体手工建模开销上仍然显著优于对比方案. 通常，管理员只需维护任务层次、角色特化和职能关联关系，职能关联的管理和特化关系可根据规则 1 和规则 2 自动调整，而授权部分的低重复显著减少了维护过程中的无效劳动，以及操作失误的可能. 因此，AIBAA 不仅优化了授权，降低了总体手工建模开销，在授权方案的整个生命周期中，同样有着易于管理和维护的优势.

表 4 论文评审实例的 AIBAA 授权方案

职能关联	BPA-MPA	DPA	N
(W,author)	$(t,r),(i,w),(p,w),(r,r)$		4
(S,author)	(o_0,r)		1
(M,author)	$(m,w),(o_3,r)$		2
$(R,\text{m\_editor})$			0
(R,editor)		$(i,r),(p,r),(t_p,r)$	3
$(P,\text{c\_editor})$	(n_r,w)		1
$(R_0,\text{c\_editor})$	$(t,r),(o_0,w),(n_g,w)$		3
$(R_{1-1},\text{m\_editor})$	(o_1,w)		1
$(R_{1-2},\text{c\_editor})$	(o_1,r)		1
$(R_2,\text{reviewer})$	$(p,r),(m,r)$		2
$(R_2,\text{c\_editor})$	(l,r)		1
$(R_{2.1}S,\text{c\_editor})$	(f_1,c)		1
$(R_{2.2}S,\text{c\_editor})$	(f_2,c)		1
$(R_{2.1},\text{reviewer})$	$(f_1,w),(r_1,w)$		2
$(R_{2.2},\text{reviewer})$	$(f_2,w),(r_2,w)$		2
$(R_{2.1}T,\text{c\_editor})$	(f_1,r)		1
$(R_{2.2}T,\text{c\_editor})$	(f_2,r)		1
(R_3,editor)	$(f,r),(r,r),(m,r),(s,r),(o_3,r)$		5
$(R_{3-1},\text{c\_editor})$	(s,w)		1
$(R_{3-2},\text{m\_editor})$	(d_3,w)		1
$(R_{3-3},\text{c\_editor})$	$(o_3,w),(n_a,w),(d_3,r),(t,r)$		4

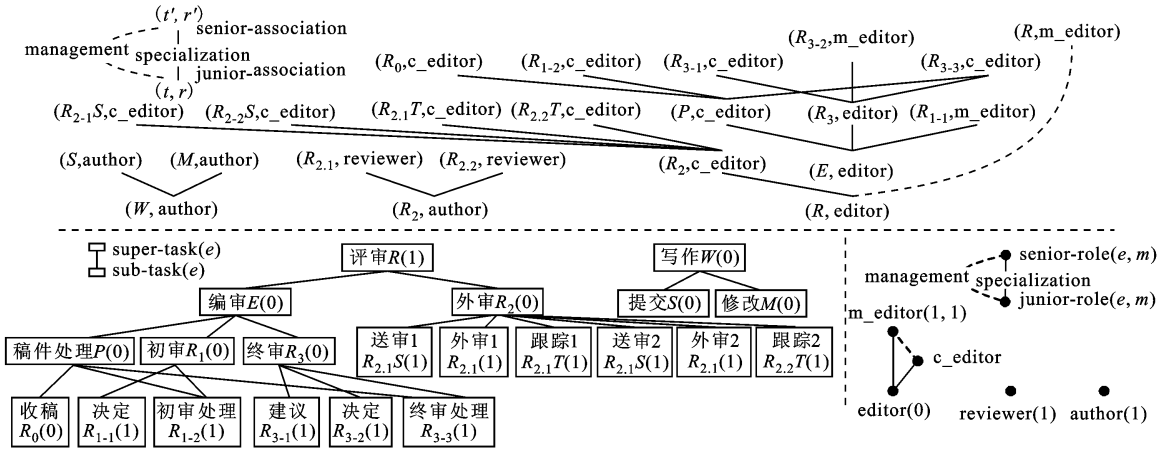

图 3　论文评审实例的职能关联关系

3　结论与未来工作

本文通过分析工作流授权中的角色-任务指派关系，建立起职能关联的概念，以及特化/管理两种关联继承机制. 应用验证和概念分析表明，职能关联及其继承机制有利于精确灵活地提取任务之间和之内的重复权限，可有效增强 AAM 模型的伸缩性，同时支持任务内的角色协作和差异化授权，兼顾了较强的普适性. 然而，本文模型的概念和应用也相对复杂一些，例如关联关系涉及某些语义辨析，各种属性均需配置默认值，等等. 下一步工作将重点考察模型的约束问题——基于职能关联可以表达哪些新型约束？它们与基于任务和角色的约束有何种关系？以及约束方案与资源分配的冲突与检测，等等.

参考文献：

[1] THOMAS R, SANDHU R. Task-based authorization controls(TBAC): a family of models for active and enterprise-oriented authorization management [C]// Proceedings of the IFIP 11th International Conference on Database Security. London, UK: Chapman & Hall, 1997: 166-181.

[2] ATLURI V, HUANG Wei-kuang. Petri net based safety analysis of workflow authorization models [J]. Journal of Computer Security, 2000, 8(2/3): 209-240.

[3] SANDHU R, COYNE E, FEINSTEIN H, et al. Role-based access control models [J]. IEEE Computer, 1996, 29(2): 38-47.

[4] OH S, PARK S. Task-role based access control model [J]. Journal of Information Systems, 2003, 28(6): 533-562.

[5] AALST W M P, KUMAR A. A reference model for team-enabled workflow management systems [J]. Data and Knowledge Engineering, 2001, 38(3): 335-363.

[6] WU Shengli, SHETH A, MILLER J, et al. Authorization and access control of application data in workflow systems [J]. Journal of Intelligent Information Systems, 2002, 18(1): 71-94.

[7] 裘炅，谭建荣，张树有，等. 应用角色访问控制的工作流动态授权模型[J]. 计算机辅助设计与图形学学报，2004，16(7)：992-998.
QIU Jiong, TAN Jian-rong, ZHANG Shu-you, et al. Workflow dynamic authorization model with role-based access control [J]. Journal of Computer-Aided Design and Computer Graphics, 2004, 16(7): 992-998.

[8] 马晨华，陆国栋，裘炅. 面向工作流系统的柔性策略访问控制模型[J]. 浙江大学学报：工学版，2008，42(12)：2112-2120.
MA Chenhua, LU Guodong, QIU Jiong. Flexible policy access control model for workflow systems [J]. Journal of Zhejiang University: Engineering Science, 2008, 42(12): 2112-2120.

[9] 马晨华，王进，裘炅，等. 基于情景约束的工作流柔性访问控制模型[J]. 浙江大学学报：工学版，2010，44(12)：2297-2308.
MA Chenhua, WANG Jin, QIU Jiong, et al. Flexible context-constraint-based access control model for workflows [J]. Journal of Zhejiang University: Engineering Science, 2010, 44(12): 2297-2308.

（编辑　葛赵青）