基于流程的复合数据权限控制研究*)

霍晓丽[1] 陈嫄玲[2] 李瑞轩[1]

(华中科技大学计算机学院 武汉 430074)[1] (郑州轻工业学院计算机与通信工程学院 郑州 450002)[2]

摘 要 复合数据权限控制过去仅仅是通过角色来实现的,其权限在各个阶段是静态的,而在协同开发过程中数据却处于动态变化之中,因此用户权限也应该是动态的。本文分析了数据权限管理相关对象及其间的关系,介绍了基于角色的数据权限控制模型,并分析了优点和不足。在此基础上,文中提出了基于流程的复合数据权限模型,研究了流程中的动态权限控制,最后给出了数据权限的计算过程。

关键词 开发流程,权限控制模型,动态权限管理

Research of Compound Data Right Control Basde on Flow

HUO Xiao-Li[1] CHEN Yuan-Ling[2] LI Rui-Xuan[1]

(College of Computer, Huazhong University of Science and Technology, Wuhan 430074)[1]

(College of Computer and Communication Engineering, Zhenzhou University of Light Industry, Zhengzhou 450002)[2]

Abstract The compound data right control was realized merely by role in the past. Its right in each stage is the static state, but the data is in during the dynamic change in the coordination performance history. Therefore the user right also should be dynamic. This article has analyzed the data right management correlation objects and the relations between them, introduced based on the article proposed compound data right model based on the flow, has studied dynamic right control in the flow, and finally has produced the data right computation process.

Keywords Devlopment flow, Right control model, Dynamic right management

1 引言

计算机网络的发展使得用户对产品数据的存取不再受到地域限制,这给权限控制带来了新的难度。产品数据的管理也不再局限在某一个部门、某一个企业,而是协同开发团队中的所有人员。协同开发涉及不同部门甚至不同企业的大量开发成员,如何避免开发过程中产品数据的非法存取是一个重要问题[1]。传统的权限管理仅仅根据角色来控制权限,其权限在产品生命周期各个阶段是静态的,但开发过程中每个用户承担的任务各不相同,而且开发任务以及产品数据也处于动态变化之中,因此用户权限也应该是动态的,即用户的产品数据存取权限应该随着产品开发过程中的任务流而流动。作为协同产品开发过程的支撑技术,PDM 对协同产品开发中的数据流进行控制和管理,使开发成员能够快速获得所需要的产品数据,并对产品开发的协同过程提供支持。

2 产品数据权限管理对象及其相互关系

2.1 权限管理相关对象

(1)用户

用户是数据权限管理的主体对象,数据对象的存取控制最终都会归结到用户的权限计算。用户集可表示为 $USERs=\{u_i, i\in N\}$,其中 N 为自然数,每一个 u_i 表示具体的用户。协同开发过程涉及用户可以有很多种,不同的用户可以行使不同的功能。

(2)角色

角色集 $ROLEs$ 表示角色集合,记为 $ROLEs=\{r_i, i\in N\}$,r_i 称为角色集合中的一个角色。角色与权限是密切相关的,也可以将角色看作用户承担的岗位责任,按角色进行的人员组织有利于明确产品开发人员的责任、方便产品开发任务的分配、简化动态产品开发团队的组建和产品开发过程管理的权限控制。对角色的权限控制能方便地扩散到该拥有角色的所有产品开发人员。角色与用户的关系可表达映射为 $M_{RU}:ROLEs\rightarrow 2^{USERs}$,通过该映射可枚举出角色包含的所有用户。同理通过映射 $M_{UR}:USERs\rightarrow 2^{ROLEs}$ 可得出用户的所有角色。

(3)机构

机构是一种人员组织方式,可以分为动态组织和静态组织,静态机构中成员组成相对固定,因而便于管理,其缺点是灵活性较差。动态机构是根据某一产品开发任务临时组织起来的小组,它随着产品开发任务的完成而解体。在协同产品开发过程中存在大量的数据流动,因此对产品数据存取权限控制提出了更高的要求。一般来说,机构包含多个用户,用户也可以隶属于多个结构,即用户和机构是多对多的关系,可以表达为 $2^{USERs}\leftrightarrow 2^{DEPTs}$。用户的权限继承其所属机构的权限。

(4)权限

权限用来表达用户可对产品数据对象进行的操作。若用户的可能操作用 $METHs$ 表示,一般包括删除、修改、浏览、复制、圈阅等;开发过程中的数据对象集为 $OBJs$,则权限表达为:$PERMs\subseteq 2^{OBJs}\times 2^{METHs}$。

(5)会话

*)国家"十五"攻关项目(编号:2002BA103A04)。霍晓丽 博士生,主要方向:软件工程。陈嫄玲 讲师,主要方向:软件工程、数据库技术。李瑞轩 博士,副教授,副博士生导师,研究方向主要包括分布式异构系统集成,分布式系统安全。

会话(Session)表示权限控制主体与产品数据对象的一次连接,例如在 PDM 系统中,每当客户端运行时与服务端连接时就产生会话,且这个运行的客户端根据窗口区分与服务端的会话。一个运行的客户端只有一个控制主体,但可以有多个会话,每个窗口最多只能有一个会话。会话主要用来映射系统中处于活动状态的用户和它所拥有角色的子集。

2.2 权限管理对象之间的关系

产品数据权限管理的基本对象之间的关系如图 1 所示。开发成员是权限控制的核心对象,同一用户可以同时参加几个不同的机构,在不同的机构内扮演不同的角色、拥有不同的权限。权限由特定数据对象以及相关操作组成、权限既可授予特定的开发成员,也可授予角色和机构某种权限,但权限最终是赋予特定成员。

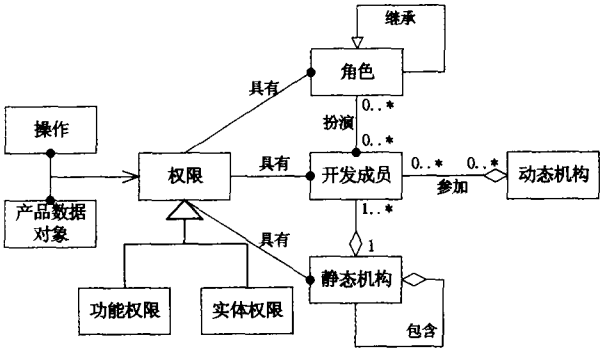

图 1 产品数据权限管理对象之间的关系

权限管理对象及其之间的关联构成了一个以权限管理为中心,开发成员为权限施控对象的数据安全模型。开发成员即用户分属于动态机构和静态机构,在系统中以某种角色出现,角色拥有某种权限,而角色所具有的读、写、删、改、拷贝等操作权限最终要落实到开发成员所拥有的对产品数据对象的存取权限,从而将图 1 中的人员组织机构、角色、权限、操作集等对象关联起来。

3 基于角色的产品数据权限控制模型

与强制访问控制将系统的权限直接赋给用户的方式不同,基于角色的权限管理模型将角色作为中间媒介把用户集合和权限集合联系起来,用户通过角色间接地访问系统资源。角色作为中间媒介把用户集合和权限集合联系起来,用户通过角色间接地访问系统资源,角色的概念是基于角色的访问控制模型的核心部分,一个角色和权限关联可以看作是该角色拥有一组权限的集合,与用户关联又可以看作是若干具有相同身份的用户的集合。在基于角色的权限管理模型中,一个用户可以被赋予多个角色,一个角色也可以被赋予多个用户,用户和角色之间是多对多的关系,同样,一个角色也可以具有多项权限,一项权限也可以被赋予多个角色,权限和角色之间也是多对多的关系。协同开发成员可以通过其所拥有角色的权限来判断其可以访问的产品数据库资源和系统资源,这是基于角色的权限管理模型的基本原理。

设用户、角色和权限集分别为 $USERs=\{u_1,u_2,\cdots,u_n\}$, $ROLEs=\{r_1,r_2,\cdots,r_k\}$, $PERMs=\{p_1,p_2,\cdots,p_m\}$,则它们之间的关系可用一个函数映射来 $Y=f(X,Z)$表示,其中 $X,Y\in\{USERs,ROLEs,PERMs\}$, Z 是对应的某种约束,f 是特定映射规则,例如权限 $p_i\in PERMs$,按照约束 Z_i,通过一定的规则 f,赋给角色 $r_k\in ROLEs$。图 2 为基于角色的权限控制模型,$USERs$,$ROLEs$,$PERMs$ 相互两两之间存在对应关系。

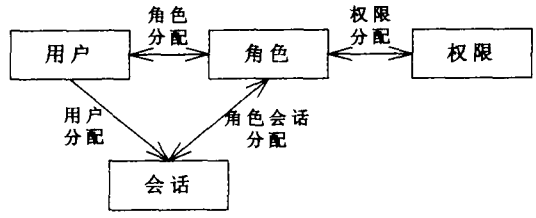

图 2 基于角色的权限控制模型

根据基于角色的权限控制模型中用户、权限和角色的关系,可定义以下 4 种分配关系[2]:

(1)权限分配:分配角色完成工作所需要的特定权限,表示为:$RPA:ROLEs\rightarrow 2^{PERMs}$。

(2)角色分配:用于指定用户能够扮演的角色,表示为:$URA:USERs\rightarrow 2^{ROLEs}$。

(3)用户分配:用于绑定特定用户到会话,表示为。$SUA:SESSs\rightarrow USERs$。

(4)角色会话分配:用于将特定角色与会话关联,表示为:$SRA:SESSs\rightarrow 2^{ROLEs}$。

其中 RPA,URA,RSA 是一种多对多的映射关系,而 SUA 是一对多关系。RSA 建立了在特定会话中被用户激活的有效角色集合,该集合是为建立会话所需要分派给用户的所有角色的子集,即 $RSA(s)\subseteq URA(SUA(s))$。用户在一次会话中可得到的存取权限(Available Session Permissions, ASP)可表达如下:

$$ASP:SESSs\rightarrow 2^{PERMs}=\bigcup_{r\in RSA(s)}RPA(r)$$

上式表明,在一次会话中用户可得到的权限是针对这次会话分派给该用户所有角色的权限之和。

4 基于流程的产品数据权限管理模型

4.1 协同开发过程中产品数据权限管理的特点

协同开发链涉及不同部门甚至不同企业的大量开发成员,其开发成员也处于不断的动态变化中,因此数据存取权限也应该随着其承担的任务不同而不断发生变化。与一般系统的权限控制不同,协同开发链的数据权限管理具有以下特点:

(1)开发团队的动态性:如前所述,协同开发团队是因应一定的市场需求和开发任务而动态建立起来的,在开发过程中可能会有成员的退出或加入,因此成员权限应该被不断地修改变更。

(2)权限的多重性:协同开发过程中权限类型有多种,按照权限客体性质可分为功能性权限、实体性权限、流程权限等,按照主体类型可分为角色权限、机构权限以及成员权限等。这些权限仅仅通过基于角色的存取控制模型难于有效管理。

(3)基于任务的权限流动:在协同开发过程中,开发成员承担一定的任务,从而对任务中的相关数据对象具有权限,一旦任务完成,则权限自动流动到任务的下一执行者,这是基于角色的权限模型无法实现的。

(4)工作流程与权限控制密切相关:产品开发流程涉及大量开发成员、角色、机构以及众多的产品数据对象,而且开发团队以及产品数据均处于动态变化之中,因此权限控制模型非常复杂。在工作流管理中融合了对任务执行者的存取权限管理,因此权限管理必须考虑到开发工作流程对数据安全控

制的需求。

根据协同开发过程的特点，在基于角色的权限管理模型的基础上，本节提出了基于任务和角色的复合数据权限管理模型。该模型以任务和角色为核心，建立多角度权限视图，实现复合立体权限交叉控制体系，从而确保在协同产品开发过程中正确的数据在正确的时间以正确的权限传递给正确的用户。

4.2 产品数据权限的类型

按照控制对象的不同，权限可以分为数据权限和功能权限：①数据权限也称为实体性权限，是指授予权限主体一个具体的产品数据对象相应的操作许可，例如授予用户 A 对零部件“发动机”的修改权；②功能权限是对系统中具有相似特性的一类操作的统称，它与具体的数据对象无关，例如产品结构编辑权，它对任何零部件都适用。按照权限控制的方式不同，权限又可分为静态权限和动态权限，如图 3 所示。

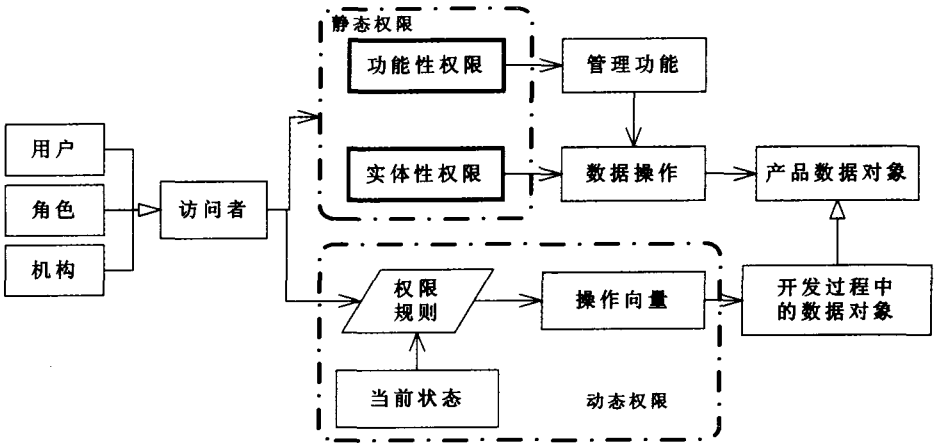

图 3 静态权限和动态权限

(1)静态权限

包括三种类型：①功能权限，主要指用户对系统管理功能的使用权限，如数据类型定义、数据归档、人员组织定义等系统管理功能；②用户对静态数据的操作权限，静态数据是指已经定型，不会发生更改的产品对象，包括已归档的文档、零部件以及支持产品开发的各种规范、标准等资料；③针对具体产品数据对象，对用户、角色或机构所进行的专门授权。

静态权限管理属于一种被动的权限控制策略，用户对功能或数据的使用权限按照“先设置后使用”的方式进行，静态权限管理可通过专门授权的方式进行管理。

(2)动态权限

指对处在不同工作状态的产品数据的操作设定的权限，它依赖于系统在此时的状态，如用户的角色、数据的状态、正在执行的任务等。动态权限是一种主动的权限控制策略，即权限的生效或失效是由系统根据一定的管理规则自动控制的。协同产品开发流程中的数据对象在不同用户之间传递，用户权限因此需要不断发生改变。动态权限的控制方式比较复杂，是协同开发过程的数据安全管理技术难点。

功能权限对象是系统访问者和产品数据管理功能之间的关系，数据对象可以是文档对象或零部件对象，数据操作是指特定数据对象的某个操作，静态数据权限是访问者和数据操作间的关系。访问者对象可以看作用户、角色、机构等对象的集合。通过引入访问者对象，可以简化权限的设置，可以对用户集合进行授权。

动态权限管理中，对数据的访问控制和数据所处的生命周期状态有关。动态权限的管理由权限规则和规则处理器组成，将访问控制要求以规则的形式定义在产品开发生命周期各个阶段上，由规则处理器根据当前的情况进行权限鉴别，实现数据权限的自动控制。权限规则用于描述对象进入特定状态后，哪些人员能对该对象进行什么样的操作。权限规则是由访问者对象和权限操作向量组成。权限操作向量描述访问者可以对数据对象进行哪些操作。对于数据对象生命周期中的每个状态，可以定义多条权限规则，这些权限规则共同构成对该状态下数据的动态权限控制。

4.3 产品开发流程中的动态权限控制

工作流以任务为单位，任务的执行需要使用的一定的数据资源，若用户获得了该任务的执行权，就应自动获得了完成该任务所需要资源的权限。任务是工作流程的基本单元，任务之间的偏序关系构成工作流，即 $WF=(TASK,\leqslant)$。任务的角色分派是任务集合到角色集合的一个映射，表达为 $TRA:TASKs\rightarrow 2^{ROLEs}$，任务的权限分派 TPA 是任务集到权限集的映射，可表达为 $TRA:TASKs\rightarrow 2^{PERMs}$。图 4 为面向工作流程的动态权限控制模型。

在面向过程的权限控制模型(Process-oriented Permission Control，PPC)中，数据存取权限分配给任务，然后任务分派给角色，而基于角色的权限模型则是直接将权限分配给角色。在实际产品开发过程中，只有当一个用户执行某种任务时才需要授予他相应权限，因此将存取权限授予任务是合理的。在 PPC 模型中，工作流中各个节点的角色始终保持稳定，但随着过程的推进，用户与角色之间的关系在发生着变化，但这种变化由系统自动控制。

PPC 模型通过对任务节点执行角色的控制来增强工作流程中数据对象的安全性。任务的角色分派 TRA 确定其执行角色集合，如任务 T_i 的执行者表达为 $R_{T_i}=(r_1,r_2,\cdots,r_m)$，角色的权限分派 RPA 确定角色的执行权限，角色存在一个访问控制列表，对于不同的角色，它的访问控制列表中包含了授权访问此角色的用户标识符。因此在判断用户使用系统的权限时，必须保证用户的标识或所在机构的标识符在角色的访问控制列表上，同时根据任务角色分派来判断当前角色执行的任务，这意味着通过一个二阶段的过程来证实一个授权。在这个二阶段的过程中，主体被授权为一个角色，角色包含了访问客体所需要的相关权限，这些权限称为具有某种

角色的可操作性.在相关的权限中所指定的操作模式称之为通过该角色的合法访问模式。在基于角色的工作流操作控制模式中，角色的执行权限和角色的管理权限是相分离的，即主体不应同时拥有二类权限，否则会出现权限管理的失控，这称为职责分离原则(Separation of duty, SOD)。

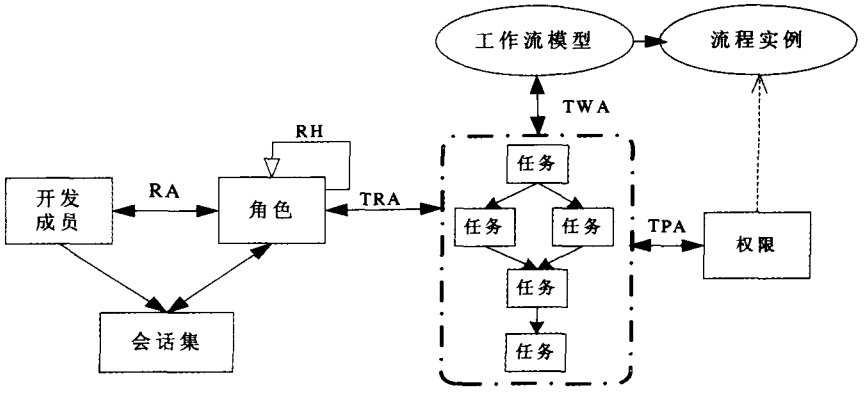

图 4　面向工作流程的动态权限模型

我们将赋予了执行角色的任务称为角色任务 T^{ROLE}，它是权限控制工作流模型的基本元素，T^{ROLE} 是一个三元组，即 $T^{ROLE}=(T,R^T,C)$，其中 R^T 表示能够执行任务 T 的角色集合，C 表示执行该任务的动态约束条件。如果将产品开发活动视为一项复杂过程，则可定义为角色任务的有限集 $A=\{T_i^{ROLE}\mid i=1,\cdots,n\}$。

过程执行单元为三元组，定义为 $PRU=(P,Exr,RD)$，其中 p 既可以是一项简单任务，也可以是任务的有限集 A，Exr 为过程的执行者集合，RD 为过程相关数据对象。工作流单元是一个四元组，即 $WFU=(PEU,WFEng,AC,Org)$，其中 $WFEng$ 为工作流引擎，AC 为数据存取控制机制，Org 为工作流单元的组织模型。WFU 是工作流执行的基本单位，将多个 WFU 组织在一起，就构成了工作流。

工作流过程 $A=(T_1^{ROLE},T_2^{ROLE},\cdots,T_n^{ROLE})$ 是角色任务的有限集，在实际工作流过程中，任务之间除了路由约束外，还存在一些授权约束。如上所述的 SOD 原则，即同一角色不能承担两个要求权限分离的任务，例如方案的设计者和审核者不能由同一人承担。SOD 原则实际上是描述了一种冲突关系，主要包括任务冲突和角色冲突：

(1)任务冲突。对于 T_i 和 T_j，如果要求用户不能既执行 T_i 又执行 T_j，则称为二者冲突，记为 $Collision(T_i,T_j)$。因此不存在一个角色 $r\in ROLEs$ 使其同时执行 T_i,T_j，表达为如下约束：

$$Collision(T_i,T_j)\Rightarrow\neg\exists(r\in ROLEs)(r\in R_{T_i})\wedge(r\in R_{T_j})$$

(2)角色冲突。对于 $r_i\in R_{T_i},r_j\in R_{T_i}$，如果 T_i 与 T_j 冲突，则称角色 r_i,r_j 冲突，记为 $Collision(r_i,r_j)$。在协同开发成员中，任何成员不得拥有两个冲突角色，表达为如下约束：

$$\forall u\in USERs,\neg\exists(r_i\in CR\wedge r_j\in CR)(r_i\in URA(u)\wedge r_j\in URA(u))$$

4.4　复合数据权限管理模型

协同开发过程的主要内容包括为了完成某项特定开发任务而在一起协同工作并具有一定组织结构的成员，以及他们在协同工作中所承担的职能、角色，协同工作中的各种信息资源，进行信息交流和对信息资源进行操作所应遵循的权限规则及各种协同机制。开发过程中产品数据存取权限控制受到多种因素的制约，而且每个用户根据所承担的任务不同其权限是动态的，即随着开发进程的向前推进，用户对某数据对象的权限也不断变化。基于前面的讨论，图 5 给出了面向开发过程的复合数据权限管理模型。

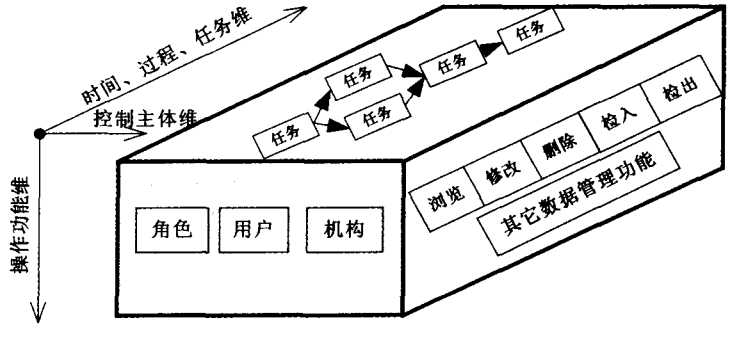

图 5　面向开发过程的复合数据权限管理模型

复合数据权限管理模型包括时间维、控制主体维以及操作功能维等三个方面。时间维表达用户对数据对象的存取权限是随着任务的执行而动态改变的；操作功能维表达控制主体所具有的功能性权限，例如对文档的浏览、修改、检入检出等；控制主体维是指可授予权限的主体对象，用于表达开发成员所具有的角色和所处的机构，用户可从角色和机构继承相应的权限。

复合数据权限管理模型为进行协同开发过程中的权限控

制提供了基本原则和方法：①定义权限控制主体，即开发过程涉及哪些用户、角色、机构，并建立主体层次关系；②对产品数据管理的主要功能进行分类，例如文档管理、产品结构管理、流程定义、编码定义等，并根据角色的职责分配相应功能权限；③在开发流程定义过程中，确定每个任务的执行者，可以是具体用户，也可以是角色或者机构，并根据流程任务的要求，确定执行者对相关数据对象的操作权限。

4.5 复合权限计算过程

面向过程的复合数据权限管理模型简化了系统的权限管理，采用任务和角色作为权限的承载对象，将角色与流程任务的岗位责任对应起来，实现了协同开发过程中权限的动态管理。在开发过程中，如何判断特定用户是否具有对某一数据对象的相关权限是一个复杂过程。本节给出了数据存取权限计算一般过程。

(1)功能性权限的计算

可以通过层次关系来组织功能性权限，下级权限是上级权限的细化，例如零部件库管理功能包括零部件库目录整理、零部件变更、零部件删除等子功能，上级功能是下级功能的前提，如用户必须具有零部件库管理的权限才能进行零部件的变更。因此，在进行权限判断时必须从上到下判断。下面的步骤用于计算当前用户的功能性权限集：

a)令当前用户的功能权限集 $FPs=\varnothing$；

b)遍历功能层次树，取得用户的所有功能权限 FP_u；

c)取得用户所具有的所有角色，循环检测每个角色所具有的功能权限，计算所有角色功能权限之和 FP_r；

d)根据当前用户所处的机构，包括静态机构和动态结构，计算其从机构继承来的功能权限 FP_o；

e)当前用户总的功能权限集 $FPs=FP_u\cup FP_r\cup FP_o$。

(2)实体性权限的计算

实体性权限用于控制具体数据对象的操作，假定用户要对产品数据对象 DO 进行 Op 操作，则计算用户是否有操作权限的步骤如下：

a)若 $Op\notin FPs$，则没有权限，终止权限计算；

b)计算用户本身是否具有对 DO 的操作权限，若有权限，终止权限计算，否则继续；

c)计算当前用户所处的机构，判断是否从它们继承对 DO 的权限，若有权限，终止权限计算，否则继续；

d)计算当前用户所充当的角色集，角色的权限分为两个部分：其一为静态权限，其二为流动权限。针对角色集中的每一可角色计算其对 DO 的静态权限，若有权限，终止权限计算，否则继续；若每个角色均没有权限，则根据角色所承担的任务计算动态权限。动态权限的计算详见文[3]。

(3)过程中的权限计算

动态权限主要指当前用户因承担某一项开发任务而获得的对任务相关数据对象的权限。其权限计算一般步骤如下：

a)计算当前用户所扮演的角色集 R_u；

b)对每一个 $r_i\in R_u$，计算其所承担的任务集以及每个任务所关联的数据对象集 $TDOs$；

c)若 $DO\in TDOs$ 则计算其权限，若具有预定操作权，则返回，否则对所有角色及其任务循环计算。

权限管理的最终目的是保证数据的安全性和保密性，简单地说就是保证每个用户能且只能执行其能够执行的操作。虽然从简化授权操作的角度可以对机构、团队、项目和角色授权，但任何权限控制最终都必须落实到具体用户。系统中需要进行权限控制的对象主要有数据对象和功能对象，也即上文讲的实体性权限和功能性权限。对数据对象授权操作可以针对某种集合(如文件夹、产品、项目等)进行，但最终必须落实到具体产品数据对象亦即文档和零部件上。

小结 本文通过分析产品数据权限管理相关对象及其它们之间的关系，介绍了基于角色的产品数据存取控制模型，并分析其优点和不足。在此基础上，提出了基于任务和角色的复合产品数据权限管理模型，详细讨论了数据存取权限的类型，研究了产品开发流程中的动态权限控制，最后给出了产品数据存取权限的计算过程。

参 考 文 献

1 Leong K K, Yu K M, Lee W B. A security model for distributed product data management system. Computers in Industry, 2003, 50:179～193

2 Botha R A, Eloff J H P. Access Control in Document-centric Workflow Systems- An Agent-based Approach. Computers & Security, 2001, 20(6):525～532

3 Cheng E C. An object-oriented organizational model to support dynamic role-based access control in electronic commerce. Decision Support Systems, 2000, 29:357～369

4 Castano S, Martella G, Samarati P. Analysis, comparison and design of role-based security specifications. Data & Knowledge Engineering, 1997, 21:31～55

5 Bellettini C, Bertino E, Ferrari E. Role Based Access Control Models. Information Security Technical Report, 2001, 6(2):21～29

6 Oh S, Park S. Task-role-based access control model. Information Systems, 2003, 28:533～562

7 Ahn G J, Hongb S P, Shin M E. Reconstructing a formal security model. Information and Software Technology, 2002, 44:649～657

(上接第 212 页)

结束语 在信息获取领域中，查全率和查准率一直是研究和关注的焦点。查询者不能给出完整或明确的查询目标是搜索引擎查准率低下的原因之一。研究和讨论弱信念 Agent 的知识交互模型，对于推测、获取查询者实际查询意图具有重要意义。

参 考 文 献

1 刘勇，蒲树祯，程代杰，等. BDI 模型信念特性研究. 计算机研究与发展，2004(4):54～59

2 Bratman M E. Intention, Plans, Practical Reason. Cambridge MA: Hardvard University Press, 1987

3 Marinacci. M Ambiguous Games. Games and Economic Behavior, 2000, 31(2): 191～219

4 候文华. 基于概率区间的信念均衡. 系统科学与数学，2002，22(3):381～384