基于角色的带时限的转授权与撤销模型的研究与实现

计算机系统结构专业
硕 士 生：杨虹轶
指导教师：汤庸 教授

摘　要

当前基于角色的访问控制系统完全依赖于管理者的集中管理方式，不能够满足分布式环境下的系统管理的需求，基于角色的转授权模型更适于分布式环境的授权管理。同时，时间是自然界无所不在的客观属性，所有的信息都具有相应的时间属性，通过在转授权模型中引入时间的因素，能够更好地控制用户之间的转授权与撤销的过程。

目前已有的转授权模型的研究都仅限于基于常规角色的转授权与撤销。RDM系列模型（Role-based Delegation Model）只有转授权的有效期到期将其自动撤销的概念，没有将时间因素引入到模型里，而 TRDM 模型（Temporal Role-based Delegation Model）是带时限的转授权模型，但是不支持角色的部分转授权，也没有详细讨论带时限的用户主动撤销转授权的机制。

本文将对 RDM 和 TRDM 中的基于常规角色的转授权与撤销机制进行扩展，扩展后的模型称为基于角色的带时限的转授权与撤销模型（Temporal Role-based Delegation and Revocation Model，TRDRM）。同时将时间因素、ARBAC（Administrative RBAC）的思想、SARBAC（Scoped ARBAC）的管理范围的定义引入到 TRDRM 中，从而在支持常规角色的转授权与撤销的同时，也支持管理角色的转授权与撤销，是集中管理方式和分布式管理方式的有效结合。

本文的最后对 TRDRM 中的转授权与撤销机制的核心功能进行了实现，是将 TRDRM 模型应用到本人所参与的科研机构协同工作平台项目里的一个探索。

关键词：访问控制，基于角色，转授权，时限，管理角色

Research and Implement of a Temporal Role-based Delegation and Revocation Model

Computer Architecture

Name: Yang Hongyi

Supervisor: Professor Tang Yong

ABSTRACT

Role-based Access Control is an enabling technology for managing and enforcing security in large-scale and enterprise-wide systems. Researchers have proposed many enhancements of RBAC models in the past decade, and delegation is an important factor for secure distributed computing environment.

Delegation models with capabilities to process temporal information is powerful. RDM(Role-based Delegation Model) and TRDM(Temporal Role-based Delegation Model) are recently published delegation models focused on regular role delegation. RDM describes a rule-based framework for role-based delegation and revocation without temporal factor. TRDM deals with temporal information based on RDM but omits user revocation.

This paper presents a Temporal Role-based Delegation and Revocation Model called TRDRM based on both RDM and TRDM. TRDRM not only supports regular role delegation and revocation, but also supports administrative role delegation and revocation. It is an effective way to build bridge between central management and distributed management.

A prototype implementation of TRDRM is presented in the last part of this paper. It is the first step to incorporate TRDRM into the Collaborative Work Platform Systems.

Keywords: access control, role-based, delegation, temporal, administrative role

第1章 绪论

1.1 访问控制的发展

访问控制技术的研究一直是信息安全研究的热点问题。访问控制技术起源于二十世纪七十年代，最初的需求是保护大型计算机系统的数据集的安全。1972年，Anderson首先提出了引用监视器（reference monitor）的概念，为访问控制成为一项关键的安全技术奠定了理论基础[1]。

在随后的三十多年中，先后出现了多种访问控制模型。其中，自主访问控制（DAC）、强制访问控制 （MAC）、基于角色的访问控制（RBAC）为其他的访问控制模型的发展提供了重要思路。

自主访问控制在操作系统和数据库系统中得到了广泛的应用。然而在DAC模型中，访问权限的授予是可以传递的。一旦访问权被传递出去将难以控制，会带来严重的安全问题。MAC 源于对信息机密性的要求以及防止特洛伊木马之类的攻击。虽然MAC 增强了信息的机密性，但不能实施完整性控制[2]。

1992 年 Ferraiolo 和 Kuhn 提出了 RBAC 模型[3]。随后，Sandhu 等人又给出了 RBAC 模型的一个比较完整的框架， RBAC96 模型[4]。到 2004 年初 RBAC 已成为 ANSI 标准。RBAC 模型的突出优点是简化了各种环境下的授权管理。基于角色的访问控制（RBAC）的本质是权限是通过角色来赋予的，而不是直接授予给独立的用户个体。也就是说，用户被赋予角色，而角色被授予了权限，从而实现对用户权限的控制，在这里，角色充当了用户和权限的中介。即给用户授权的动作可以分成两个逻辑上独立的部分：一是给用户指派一定的角色，二是给角色指派一定的权限。与用户相比角色是相对稳定的，角色实际上是与特定工作岗位相关的一个权限集，当用户改变时只需进行角色的撤消和重新分配即可。

基于角色的访问控制已经被广泛认可为传统的自主访问控制和强制访问控制的替代方案。就目前的应用情况来讲，基于角色的访问控制策略是在大型系统以及企业级应用系统中实施管理和执行安全策略的有效技术。

同时，基于角色的访问控制模型也衍生出来一系列扩展模型，主要包括扩展

RBAC的约束（RBAC Constraint）来增强其表达能力的研究[5，6，7，8，9]、基于角色的转授权模型[10，11，12，13，14，15，16]等方面的研究。当然还有继续研究和探索的空间。

RBAC的约束，规定了为角色分配权限时，或者为用户分配角色时，以及当用户在某一时刻激活一个角色时所应遵循的强制性规则。

文献[5]中对授权约束的时间特性需求进行分析，提出一个带时间特性的约束描述语言RCLT（Role-based Constraints Language with Time-character）。文献[6]中形式化描述了一个引入时间后的角色访问控制模型，对原有授权约束进行了时间扩充，提出了相应算法解决了时间授权约束和会话的状态转变问题。

文献[7]给出了一个角色激活受时间约束的模型 TRBAC（Temporal Role-based Access Control），该模型通过角色触发器（role trigger）支持角色激活的依赖关系。文献[8]则在文献[7]的基础之上，支持角色继承关系下的时间约束。文献[9]主要探讨时间约束与职责分离等其他约束的依赖关系。

Barka 和 Sandhu 提出了基于角色的转授权模型[10]，称为 RBDM0。文献[11]中 Zhang 等人提出的 RDM2000，是对 RBAC 以及 RBDM0 的扩展。该模型可以支持角色继承关系下的多步转授权，而后 Zhang 等人又在 RDM2000 的基础上，对原有的模型进行了完善并做了概念原型的实现[12]。文献[13]提出了 PBDM（Permission-based Delegation Model）转授权模型，支持部分转授权。

Bandmann 在文献[14]中提出了一个受限的转授权模型，可以很好地将权限的授权和使用分离，并且可以有效地限制被授权权限的传播。文献[15]的 CRDM（Constrained Role-based Delegation Model）提出了基于角色的转授权的约束的需求，临时性约束、部分性约束等探讨得比较完善，不过它不是在角色继承关系下的研究，需要扩展多步转授权中存在的约束。

文献[16]中提出的X-RDR模型则是针对Web信息系统的应用，使用XML来描述转授权策略，以及XML相关的技术完成转授权的操作，从而实现对系统资源的细粒度访问。

那么除此之外，同时也有在RBAC的基础之上根据不同的应用需求所做出的扩展研究。

文献[17]提出的基于角色的CSCW系统访问控制模型，主要针对CSCW环境

下的多用户、交互、协作、实时、动态等特性，扩展RBAC使其能较好地满足CSCW系统对访问控制的需求。文献[18]从协作环境下对访问控制模型的要求，对几种模型的优缺点进行了对比分析。包括基于角色的访问控制模型RBAC，基于任务的访问控制模型（Task-Based Access Control，TBAC），基于用户组的访问控制模型（Team-based Access Control，TMAC），空间访问控制模型（Spatial Access Control），上下文感知的访问控制模型（Context-Aware Access Control）等等。

文献[19]中的基于任务和角色的双重Web访问控制模型，文献[2]的面向工作流和服务的基于角色的访问控制模型，都是在RBAC的基础上，针对工作流应用的需求所做的扩展。

1.2 与本文相关的研究工作

当前基于角色的访问控制系统完全依赖于管理者的集中管理方式，不能够满足分布式环境下的系统管理的需求，基于角色的转授权模型更适于分布式环境的授权管理。

RDM 系列模型（Role-based Delegation Model）[11，12]在 RBDM0[10]以及 RBAC[3，4]的基础上进行了扩展，但只有转授权的有效期到期自动撤销转授权关系的概念，没有将时间因素引入到模型里，而 TRDM 模型（Temporal Role-based Delegation Model）[20]虽然将时间因素引入了转授权模型，但是不支持角色的部分转授权，也没有详细讨论带时限的用户主动撤销转授权的机制。PBDM模型[13]则主要讨论了如何灵活地支持部分转授权的问题，但该模型的理论研究过于复杂，实现起来十分困难。

目前已有的转授权模型的研究都仅仅限于基于常规角色的转授权与撤销，不支持管理角色的转授权与撤销。虽然关于基于角色的访问控制模型 RBAC 的研究相当多，但是利用 RBAC 的原理去管理 RBAC 系统的研究却为数不多。其实早在 RBAC96 就引入了管理角色的概念，到后来的 ARBAC97（Administrative Role-based Access Control）[21]、ARBAC02[22]，以及 SARBAC（Scoped Administration of Role-based Access Control）模型的不断补充完善[23，24，25]，为本文扩展支持管理角色的转授权与撤销提供了研究思路。

1.3 本文所作的工作及意义

1.3.1 项目背景

本文的研究工作始于本人参与开发的项目——科研机构协同工作平台项目。

项目开发的目的在于研究与开发基于中小型科研机构管理的解决方案，实现科研机构里的各项工作，比如教学、实验室管理、论文管理等活动的电子信息化。该项目的成果将是极大的提高中小型科研机构的管理水平，提高效率，解决某些中小型科研机构地点分散，人员众多，教务繁忙，各种工作不能协调开展的问题。

目前，此项目的访问控制模块的设计与开发是以 RBAC96 模型为指导的，是一种静态地访问控制方式。不支持转授权与撤销，也没有引入时间的因素进行有效的动态控制，势必不能满足协同工作的需求。

1.3.2 本文所做的工作

时间是自然界无所不在的客观属性，所有的信息都具有相应的时间属性，通过在转授权模型中引入时间的因素，能够更好地控制用户之间的转授权与撤销的过程。

因此本文将对 RDM 和 TRDM 中的基于常规角色的转授权与撤销机制进行扩展，同时将时间因素引入到扩展之后的模型中，并且利用 ARBAC 的思想以及 SARBAC 的管理范围的定义，使扩展后的模型在支持常规角色的转授权与撤销的同时，也支持管理角色的转授权与撤销，是集中管理方式和分布式管理方式的有效结合。

本文所做的理论上的研究工作，主要包括以下六点。

（1）总结带时限的转授权树的特点，并得出了形式化的结论；

（2）不仅支持带时限的完全转授权，也支持转授角色中的部分权限；

（3）详细讨论了带时限的用户主动撤销授权的处理机制；

（4）转授权和撤销操作若是对有效时间的更改，归纳分析了对转授权树的结构产生影响之后的处理机制；

（5）扩展研究管理角色的转授权与撤销的处理机制，并重点讨论了管理角

色的转授权树与常规角色的转授权树之间建立联系的问题，即如何管理和监控常规角色的转授权与撤销操作的问题；

（6）定义重要的约束规则，包括静态职责分离中的角色冲突与权限冲突。

本文将扩展后的模型称为基于角色的带时限的转授权与撤销模型（Temporal Role-based Delegation and Revocation Model，TRDRM）。

最后，本文针对 TRDRM 中的核心功能进行了实现，本文所实现的 TRDRM 原型系统，是将 TRDRM 模型应用到科研机构协同工作平台项目里的一个探索，下一步工作则是把 TRDRM 原型系统整合到此项目中。

1.3.3 本文的结构安排

第一章是绪论部分，简单介绍访问控制的发展，本文所做的工作及意义。

第二章介绍与本文的研究相关的基础知识，包括时态相关的基础知识，访问控制的基础知识，重点介绍了转授权与撤销的基础概念。

第三章介绍了本文扩展研究的重点对象，RDM 模型和 TRDM 模型，并分析了这两个模型存在的不足，提出本文的研究思路。

第四章形式化地描述了本文扩展之后的转授权与撤销模型 TRDRM，详细分析了引入时间因素之后的转授权与撤销的各种情形对转授权树造成的影响如何处理的机制。同时使用了基于规则的策略描述语言定义了授权判定的规则以及重要的约束规则。

第五章在第四章的基础之上，把对常规角色的转授权与撤销机制的处理扩展到管理角色的转授权与撤销的方面。并且结合 ARBAC 的思想以及 SARBAC 中的管理范围的重要概念，在常规角色的转授权树与管理角色的转授权树之间建立了联系，以达到管理和监控的目的。

第六章则是对 TRDRM 模型的一个原型实现，针对 TRDRM 模型中的核心功能进行了实现，下一步工作是将其整合到本人参与的项目——科研机构协同工作平台中去。

第 2 章 相关基础知识介绍

本章介绍与本文的研究相关的基础知识。关于时态的基础知识来自文献[26]，访问控制的基础知识来自文献[1]，转授权与撤销相关的基本概念来自文献[12，13，16，27]。

2.1 关于时态的基础知识

2.1.1 时态信息

在现实世界中，时间无时不有，无处不在，客观世界中的事物都带有时间的属性。描述现实世界的带有时间属性的信息系统，称为时态信息系统。这些系统中的记录的信息是随时间变化的，这种随时间变化的信息称为时态信息（Temporal Information）。

时态信息需要用数据库进行记录，这些用于记录时态信息的数据就是时态数据（Temporal Data）。时态数据在计算机系统中一般保存在数据库系统中。

2.1.2 时间模型

在自然界中，时间是每时每刻都存在、连续发生且一去不复的，它在时间轴上是连续存在的。从计算机科学的角度来看，如何量化时间和确定某个时间点或者时间段，这关系到如何确定时间模型。

基于对时间轴结构的选择，时间模型可以划分成如下几种模型。

1．连续模型（Continuous Model）

连续模型把时间看作是同构于实数，每一个实数对应于一个时间点。这种模型能够最精确地为时间建模，但是由于现代计算机基于数字逻辑的工作方式，所以不可能无失真地记录时间。

2．步进模型（Stepwise Model）

步进模型把数据的状态看成是时间的函数。当时间点上的数据状态发生变化时才记录状态变化，否则保持不变。在这种模型下，时间序列上任一点上数据的

值对应于上一次数据改变时保持的状态，如果要查询当前数据的取值，则需要回溯。

3．离散模型（Discrete Model）

离散模型把时间和整数映射起来，且在相邻的两个时间点之间不存在另一个时间点。任一时间点有前驱和后继时间点。在实际应用中，该模型适用于记录那些在关键时间点上才有意义的数据。

4．恒定模型（Non Temporal Model）

有些数据是不随时间变化的，例如籍贯、出生地等。这些数据只有本身固有的属性。在一般情况下，建模时通常没有充分考虑值随时间变化的情况。

2.1.3 时间基本元素单位

时间元素（Time Element）就是指表示时间属性值的元素，时间元素在时态信息系统中有着基础的地位，它对于正确有效的表达记录的时间属性有着重要的意义。

1．时间点

基于点的时间将是时间离散化，事物或者事件的时间属性用时间点表示。用时间点的形式来表示时间元素，这和系统的时间量子及时间粒度的关系较大。基于点（Point-based）的时间元素，又称为时间点，或称时刻（Time Point）。这种描述方式把时间看成一个个离散的时间点，这些离散化的时间点的间隔大小适度时，就可以准确地描述现实世界事件发生及变化的状况。

2．时间区间

在基于区间（Interval-based）的时间元素中，时间的基本单位为时间段或者时间区间，即通过描述时间段的起始点和终止点来描述时间区间。时间区间是指一段时间，有固定的起止时间点。

3．时间跨度

时间跨度（Time Span）是指持续的一段时间，表示时间的长度。在数据库系统内，一般用一个整数表示时间跨度。与时间区间不同的是，时间跨度没有时间起点，也没有时间终点。

4．时间集合

时间集合（Time Set）也称时间域（Temporal Domain），时间域是一些时间区间（Interval）的有穷并集。

2.1.4 时态元素关系

时间的表示可以分成两大类，基于时间点的表示和基于时间区间的表示，由此可以把时态关系分成三大类，分别是时态区间之间、时态区间与时间点之间、时间点之间的时态关系。

基于时间点的时间处理方法是将时间看成是离散的，基于时间区间的处理方法是将时间看作是连续型的。

1．Allen 的时态区间关系

Allen 在 1983 年发表的论文《Maintaining Knowledge about Temporal Intervals》中指出十三种时态区间的关系，如表 2-1。表中的 t1 和 t2 表示两个时态区间。

表 2-1　Allen 的十三种时态区间的关系

时态区间	解释
Before(t1, t2)	t1 比 t2 早开始，同时 t1 与 t2 之间没有相交
After(t1, t2)	t1 比 t2 晚开始，同时 t1 与 t2 之间没有相交
During(t1, t2)	t1 比 t2 晚开始，且早结束，即在时间轴上 t1 的区间范围被包含在 t2 的区间范围内
Contains(t1, t2)	t1 比 t2 早开始，且晚结束，即在时间轴上 t1 的区间范围包括了 t2 的区间范围
Overlaps(t1, t2)	t1 比 t2 早开始，且两个区间在时间轴上有相交
Overlapped-by(t1, t2)	t1 比 t2 晚开始，且两个区间在时间轴上没有相交
Meets(t1, t2)	t1 比 t2 早开始，且 t1 和 t2 之间没有其他时态区间，即 t2 开始于 t1 的结束点
Met-by(t1, t2)	t1 比 t2 晚开始，且 t1 与 t2 之间没有其他时态区间，即 t1 开始于 t2 的结束点
Starts(t1, t2)	t1 和 t2 有共同的起始点，但 t1 比 t2 先结束
Started-by(t1, t2)	t1 和 t2 有共同的起始点，但 t2 比 t1 先结束
Finishes(t1, t2)	t1 和 t2 有共同的结束点，但 t1 比 t2 晚开始
Finished-by(t1, t2)	t1 和 t2 有共同的结束点，但 t2 比 t1 晚开始
Equals(t1, t2)	t1 和 t2 有共同的时间区间，即 t1 和 t2 在时间轴上重合

2．时态区间与时间点之间的时态关系

时间点可以看作延续时间为 0 的时间区间。关于时态区间与时间点的关系有 5 种。如图 2-1 所示，其中 t 表示时态区间，p 表示时间点。

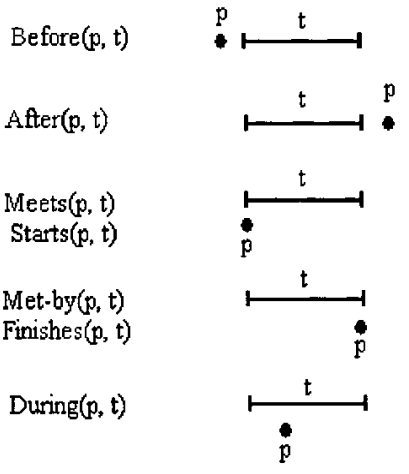


图 2-1 时间区间与时间点的时态关系

3．时间点之间的时态关系

时间点与点之间的时态关系相对而言比较简单，一共有三种关系，如图 2-2 所示。图中的 p、q 分别表示两个时间点。

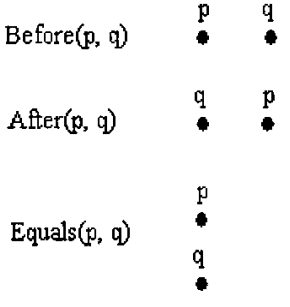


图 2-2 时间点之间的时态关系

2.2 访问控制的基础知识

访问控制系统中需要区分安全策略和机制。安全策略是组织或系统中决定访问权限的高层指导原则。安全机制是用来实现安全策略的底层软件和硬件方法。

2.2.1 自主访问控制（Discretionary Access Control, DAC）

自主访问控制是一种最普遍的访问控制安全策略，其最早出现在二十世纪七十年代初期的分时系统中，基本思想伴随着访问矩阵被提出。自主访问控制基于对主体的识别来限制对客体的访问，这种控制是自主的。自主的是指对其他主体具有授予某种访问权限权力的主体能够自主地(可以是间接地)将访问权限或访问权限的子集授予其他主体。自主访问控制存在明显的不足：资源管理比较分散；用户间的关系不能在系统中体现出来，不易管理；最严重的是不能对系统中信息流进行保护，信息容易泄漏。

2.2.2 强制访问控制（Mandatory Access Control, MAC）

由于自主访问控制不能抵御特洛伊木马的攻击，强制访问控制（Mandatory Access Control）应运而生。强制访问控制通过比较主体与客体的安全属性来决定是否允许主体访问客体，安全属性是由系统自动或由安全管理员分配给每个实体(主体和客体)的，它不能被任意更改。如果系统认为具有某一安全属性的主体不能访问具有一定安全属性的客体，那么任何人(包括该客体的主人)都无法使该主体访问该客体。即使存在特洛伊木马，强制访问控制也保证了信息可以在安全属性的格中按一个方向流动，这就解决了自主访问控制的问题。强制访问控制的不足主要表现在两方面：应用的领域比较窄；完整性方面控制不够。

2.2.3 基于角色的访问控制（Role-based Access Control, RBAC）

随着安全需求的不断发展和变化，自主访问控制和强制访问控制已经不能完全满足需求，研究者提出许多自主访问控制和强制访问控制的替代者，其中最引人瞩目的无疑是基于角色的访问控制（Role-based Access Control）。在基于角色的访问控制中，在用户（User）和权限（Permission）之间引入角色（Role）的概念，用户与一个或多个角色相联系。角色与一个或多个权限相联系，角色可以根据实际的工作需要生成或取消，而且用户可以根据自己的需要动态激活自己拥有的角色，避免了用户无意中危害系统安全。除此之外，角色之间、权限之间、角色和权限之间定义了一些关系，比如角色间的层次性关系，而且也可以按需要定义各种约束（Constraints），如定义出纳和会计两个角色为互斥角色。

与自主访问控制和强制访问控制相比，基于角色的访问控制具有显著优点。首先，它实际上是一种策略中立的访问控制技术。其次，基于角色的访问控制具有自管理的能力。此外，基于角色的访问控制还便于实施整个组织或单位的网络信息系统的安全策略。不过，基于角色的访问控制要复杂得多。

如图 2-3 所示是基于角色的访问控制模型，RBAC96 模型[4]。基本的术语和概念包括用户（Users）、角色（Roles）、权限（Permissions）、会话（Sessions）、约束（Constraints），两种指派关系即用户指派关系（User assignment，UA）、权限指派关系（Permission assignment，PA），以及角色的层次结构关系（Role Hierarchy，RH）。

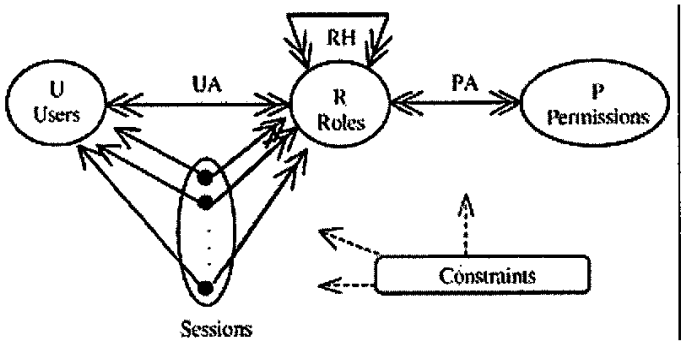


图 2-3　RBAC96 模型

2.3 转授权与撤销的基础知识

在分布式环境下，系统的管理工作异常繁重，完全依赖于某个或者某些管理者的集中管理方式，需要管理者参与系统中所有的授权行为，更加重了系统的管理负担[20]。

转授权（Delegation）的基本思想是用户将自己所具有的部分或者全部权限转授给其他用户，让接受授权的用户代表发出授权的用户执行某些任务。转授权技术允许将分布式环境下集中式管理工作分散实施，是提高分布式系统伸缩性的重要技术，而基于角色的转授权技术为在分布式系统中实现角色访问控制提供了一种有效的手段。文献[12，13，16，27]中对转授权和撤销的类型的归纳，本小节对其进行整理，以便于本文后续章节的讨论。

2.3.1 转授权的类型（Delegation）

2.3.1.1 从角色的角度来划分转授权的类型

转授权的类型从角色的角度来看，有两种。

1．具有系统管理权限的角色（administrative role）的转授

比如，新增和删除用户这样的操作权限。在文献[12，13，16，27]中均没有探索这种方式的转授，因为控制起来是比较复杂的，但是在实际生活中是存在这种需求的。例如，人力资源部门招聘了新的员工，就需要新增用户以及为该用户分配所属的部门，或者某位老员工离职则需要删除掉这个用户的相关信息。如果所有的管理工作均交给安全管理员来做，必然使得管理员的任务繁重，也不符合应用环境中的需求。

2．具有常规权限的角色（regular role）的转授

这种转授权的发生通常有三种情况 [13]，本文将一一举例说明。

（1）角色的备份（Back up of role）

例 2-1 在实际生活中，某个具备销售经理角色的用户要出差，就可能把销售经理的角色转授给其下属，也许是某个销售代表让其代为管理，直到经理出差返回再收回销售经理的角色。

（2）授权的逐级下放（Decentralization of authority）

例 2-2 具有总经理角色的用户，把某项任务交给其下属部门经理，而部门经理又把具体的任务交给他的下属，某个员工去完成，如此逐级下放转授权限。

（3）协同工作的需要（Collaboration of work）

例 2-3 某个庞大的工程项目需要工程部门内的两个项目小组一起负责，而不同的项目小组所拥有的资源是不相同的，为了协同的完成这个项目，就需要部门的同事之间共享某些资源，从而完成任务。

2.3.1.2 从转授权的方式划分转授权的类型

从转授权的方式来看，主要有几个相对应的概念。

1．永久转授权/暂时转授权（Permanent and Temporary Delegation）

如果转授用户（Delegating User）在转授出角色（Delegated Role）之后，再也收不回来该角色的权限，则是永久转授权，此种情况适用于被授用户（Delegated

User）完全替代转授用户的职能；如果转授的过程只是暂时的，则是暂时转授权。

例 2-4 一个教授拥有教授的角色，同时拥有某个学术组织的成员的角色，如果需要另一个人来接替该教授的学术组织成员的角色，则该教授在转授出该角色之后，就是永久的转授权。而如果是由于该教授出差暂时不能处理事务，则暂时地转授角色给被授用户，属于暂时转授权的情况。

2．单调转授权/非单调转授权（Monotonic and Non-monotonic Delegation）

如果转授用户在转授出自己拥有的角色之后，仍然享有该转授角色的权限，则是单调转授权。如果转授之后，失去了该转授角色的权限直到该转授关系失效，则是非单调转授权。

这两种情况的相同点是转授用户仍然拥有撤销转授操作的权力，这样的话，对于非单调转授权的情况来说，在撤销之后或者是转授关系的有效时间到期，则转授的用户可以收回转授出去的权限。

3．完全转授权/部分转授权（Total and Partial Delegation）

如果转授角色的用户将该角色的全部权限都转授给被授的用户，则称为完全转授权。如果转授的只是角色的部分权限，则称为部分转授权。

例 2-5 一个教授的角色拥有授课、实验研究、管理日常事务等权限，那么如果具备教授角色的某个教授，将授课的权限转授给助教、实验研究转授给研究员、管理日常事务转授给秘书，则属于部分转授权的情况。完全转授权则是将教授这个角色完整地转授给他人。

4．单步、多步与多重转授权（Single-step, Multi-step, Multiple Delegation）

如果被授用户还可以继续转授权，那么从初始的转授用户开始可以形成一条转授权路径，路径的长度不止一步，则是多步转授权，而单步转授权是多步转授权的特例，是路径长度为一的情况。多重转授权则是指转授用户在转授角色的时候，可以同时将该角色转授给多个用户。

2.3.2 撤销的类型（Revocation）

撤销的过程是与转授权的过程密不可分的一个重要过程。转授出去的权限如果不及时收回，不在一定时间内进行撤销，势必为访问控制带来安全隐患。

从撤销的方式来看，主要有三对相对应的概念。

1．级联撤销/非级联撤销（Cascading & Non-cascading Revocation）

在多步转授权的情况下，如果在转授权路径中的一个节点被撤销，则该节点的后代全部都将被撤销，属于级联撤销；如果仅仅撤销掉这个节点本身，它的后代不予以撤销，不受影响，则属于非级联撤销。

例 2-6　用户 Bob 将自己的某个角色转授给了用户 Alice，而 Alice 又将该角色转授给了另一用户 Charlie，如果要撤销 Alice 与这个被授角色的对应关系，也就是撤销掉转授权路径当中的这个节点，那么撤销操作执行之后，若 Charlie 仍然拥有此转授角色，则属于非级联撤销；若 Charlie 因为 Alice 被撤销也同时被撤销，则属于级联撤销的情况。

2. 授权依赖撤销/授权独立撤销（Grant-dependent & Grant-independent Revocation）

如果只有直接的转授用户能够撤销掉被授用户与被授角色之间的对应关系，则是授权依赖撤销；如果在转授权路径上的间接的转授用户均可以撤销，则是授权独立撤销。

例 2-7　仍然采用例 2-6 中的 Bob、Alice、Charlie 的例子。如果要撤销 Charlie 这个节点，若只有 Alice 能执行撤销操作，则属于授权依赖撤销；若 Alice 和 Bob 都可以执行对 Charlie 的撤销操作，则属于授权独立撤销的情况。

3．强撤销/弱撤销（Strong & Weak Revocation）[21]

如果一个用户与多个角色有对应关系，即该用户拥有多个角色，若因为撤销该用户和其中一个角色的对应关系，那么导致不仅是该用户与这个角色的关系被撤销掉，同时该用户所拥有的比被撤角色高级的角色对应关系也将被撤销掉，则属于强撤销；若该用户与其它角色的对应关系不会受到影响，仍然成立，则属于弱撤销。

例 2-8　公司里的某位员工，同时拥有项目开发部门员工的角色，软件开发工程师的角色，项目经理的角色，按实际情况来看，项目开发部门员工的角色、软件开发工程师角色、项目经理的角色，是从低级到高级的关系。那么如果要撤销该员工与软件开发工程师角色的对应关系，同时也撤销项目经理的角色，则属于强撤销的情况，因为该员工可能已经被调去部门内的其他职位；若仅仅是软件开发工程师的角色对应关系被撤销，则属于弱撤销的情况。

第 3 章 基于角色的转授权模型的研究

本章主要是对与本文相关的研究的一个总结与归纳，分析现有的基于角色的转授权模型存在的不足，从而提出本文的研究思路。

3.1 RDM（Role-based Delegation Model）模型

RDM2000[11]是在 RBAC96[4]和 RBDM0[10]的基础上扩展而成。RDM2000 支持角色的层次结构，在引入转授权关系之后同时也支持多步转授权。本小节部分重点介绍 RDM 模型，在本文的第四章部分将在 RDM 的基础上对其进行扩展研究。

3.1.1 基本元素和系统函数——来自 RBAC96 和 RBDM0：

RBAC96 的基本元素及关系如图 2-3 所示。它包括五种基本元素：用户（Users）、角色（Roles）、权限（Permissions）、会话（Sessions）、约束（Constraints）。两种关系：用户指派关系（User assignment，UA）、权限指派关系（Permission assignment，PA）。

从角色 role 的角度来看，用户可以被划分为两个集合：初始用户（Original Users），即被直接分配了角色 role 的用户的集合；被授用户（Delegated Users），即被转授了角色 role 的用户的集合。

一个用户可以是一个角色的初始用户，也可以是另一个角色的被授用户。一个用户还可能是同时是一个角色的初始用户和被授用户。

1．RBAC96 的概念

U，R，P，S 分别代表用户、角色、权限、会话的集合；

$UA \subseteq U \times R$ 是用户与角色之间的多对多关系；

$PA \subseteq P \times R$ 是角色和权限之间的多对多关系；

$RH \subseteq R \times R$ 是角色层次结构的是偏序关系；

函数 $Sessions : U \rightarrow 2^S$ 是从一个用户到会话集合的映射函数；

函数 $Roles: S \rightarrow 2^R$ 是从一个会话到一组角色的集合的映射函数；

函数 $Permissions: S \rightarrow 2^P$ 是从一个会话到一组来自 PA 的权限的集合的映射函数；

2. RBDM0 的概念

$UAO \subseteq U \times R$ 是初始用户与角色之间的多对多关系；

$UAD \subseteq U \times R$ 是被授用户与角色之间的多对多关系；

$UA = UAO \cup UAD$

$Users: R \rightarrow 2^U$ 是一个角色到一组用户的映射函数，$Users(r) = \{u \mid (u,r) \in UA\}$

$Users(r) = Users\_O(r) \cup Users\_D(r)$
$Users\_O(r) = \{u \mid (\exists r' \geq r)(u,r' \in UAO)\}$
$Users\_D(r) = \{u \mid (\exists r' \geq r)(u,r' \in UAD)\}$

3.1.2 基于角色的转授权（Role-based Delegation）

RDM 讨论的是支持角色层次结构的用户到用户的转授权关系（User-to-User）。为简化讨论，RDM 做了两个假设。

假设一、一个已经具备角色 r 的用户 u，不能再被授予同样的角色 r。

假设二、只讨论常规角色的转授，而不考虑管理角色的转授。

1. 在 RDM2000 中扩展的基本概念：

$DLGT \subseteq UA \times UA$ 是一对多的转授权关系，转授权关系也可表示为 $((u,r),(u',r')) \in DLGT$，代表具有角色 r 的转授用户 u 将角色 r' 授予被授用户 u'。

$ODLGT \subseteq UAO \times UAD$ 是初始用户的转授权关系。

$DDLGT \subseteq UAD \times UAD$ 是被授用户的转授权关系。

$DLGT = ODLGT \cup DDLGT$

由此可以得出图 3-1 的转授权模型。而转授权路径和转授权树的例子本文将在第四章详细说明。

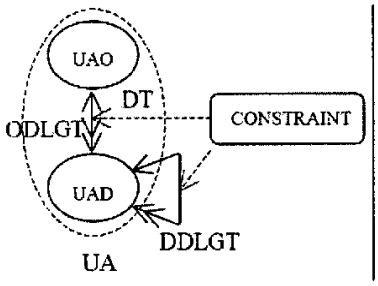


图 3-1　RDM 2000 转授权模型

2. 在多步转授权中的元素和函数：

N 是自然数的集合；

$DP \subseteq UA \times UA$ 是一个用户指派关系的有序序列，表示转授权路径。

$DT \subseteq UA \times UA$ 是一个用户指派关系的层次结构，表示转授权树。

$\Pr ior : UA \to UA$

$\Pr ior(ua) = \{ua' \mid ua \in UAD, (ua', ua) \in DLGT\}$，其中 ua=(u, r)

$\Pr ior(ua) = \{\phi \mid ua \in UAO\}$

$Path : UA \to DP$ 是从一个 UA 到一条转授权路径的映射函数，

$Path(ua_0) = \{ua_n \to ... \to ua_i \to ua_{i-1} \to ... \to ua_0 \mid ua_i = \Pr ior(ua_{i-1})\}$，

其中 $ua_0 = (u, r)$。

$Depth : UA \to N$ 是一个返回转授权路径的深度的函数。

3.1.3 基于规则的策略描述语言描述 RDM

RDM2000 定义了允许常规用户转授其角色的策略，同时也描述了如何撤销这些转授出去的角色的策略，使用基于规则的策略描述语言（rule-based policy specification language）可以使这些策略有效地执行。

语句（Clause），也称作规则（Rule），表达形式为：$H \leftarrow B$. 其中，H 代表规则头（rule head），B 代表规则体（rule body）。

若 B 为真则将触发 H 为真，这正好提供描述授权和执行授权的机制。因此，授权策略需要满足的条件可以置于规则体内，而授权本身则置于规则头的位置。

RDM 主要使用函数来说明如何去处理转授权和撤销策略。

3.1.3.1 函数定义

基于规则的语言的基本元素就是函数的集合。一个函数由函数名字、一组参数以及返回值组成。可以把函数分为三类函数：描述函数（specification functions）、实用函数（utility functions）、授权函数（authorization functions）。

描述函数描述RABC及RDM2000相关概念的信息，在RBAC96和RDM2000种定义了一组系统函数，这些系统函数映射到描述函数，见表3-1所示。

表3-1 RDM的描述函数

描述函数	函数定义及描述
active(u, r, s)	如果具备角色r的用户u在会话s中被激活，则函数返回值为真
conflicting(x, y)	如果x和y是冲突的，则函数返回值为真，其中x，y是用户或者角色
delegatable(u, r)	如果用户u具备转授角色r的权限，则函数返回值为真。若(u, r)为初始用户指派关系，则该函数永远返回真
depth(u, r)	$Depth: U \cup R \rightarrow N$，返回用户指派关系的转授权深度
duration(u, r)	$Duration: UA \rightarrow T$，返回被授用户指派关系的时间限制约束
expires(t)	如果时间限制t到期则返回值为真
junior(r, r')	≤，r是比r'低级的角色
path(u, r)	$Path: U \cup R \rightarrow DP$，返回被授用户指派关系(u, r)的转授权路径
prior(u, r)	$Prior: U \cup R \rightarrow U \times R$，返回用户指派关系在转授权路径中的前驱
permissions(s)	$Permissions: S \rightarrow 2^P$，返回会话s中所有被激活的权限
revoked_cascade(u, r)	如果在(u, r)所在的转授权路径中的任何一个用户指派关系被撤销，则返回值为真
revoke_strong(u,r,u',r')	如果(u', r')被(u, r)强撤销，则返回值为真
roles(s)	$Roles: S \rightarrow 2^R$，返回会话s中所有被激活的角色
senior(r, r')	≥，表示角色r是比r'高级的角色
sessions(u)	$Sessions: U \rightarrow S$，将一个用户映射到一组会话的函数
users(r)	$Users: R \rightarrow 2^U$，返回所有具备角色r的用户
users_o(r)	$Users\_O: R \rightarrow 2^U$，返回所有具备角色r的初始用户
users_d(r)	$Users\_D: R \rightarrow 2^U$，返回所有具备角色r的被授用户

实用函数则是一组通用的布尔函数，见表 3-2 所示。

表 3-2　RDM 的实用函数

实用函数	函数定义及描述
in(x, y)	如果 x 是 y 的成员，则返回值为真
equals(x, y)	如果 x=y，则返回值为真
lt(x, y)	如果 x<y，则返回值为真
not(x)	not(x)=!x，其中 x 是一个布尔表达式

授权函数定义了授权策略以及如何执行这些策略。还可把授权函数分成基本授权函数和推导的授权函数两类，见表 3-3 所示。其中的规则定义见 3.1.3.2 节。

表 3-3　RDM 的授权函数

基本的授权判定函数	函数定义及描述
can_delegate(r, cr, n)	见规则 3-1
can_revokeGD(r)	见规则 3-2
can_revokeGI(r)	见规则 3-3
推导的授权判定函数	**函数定义及描述**
allow(u, r, p, s)	由规则 3-4 推导出
der_can_revoke_auto_cascade(u, r)	由规则 3-8 推导出
der_can_revoke_auto_strong(u, r)	由规则 3-9 推导出
der_can_revoke_auto_expire(u, r, rvk_opt)	由规则 3-10 推导出
der_can_delegate(u, r, u', r', dlg_opt)	由规则 3-1 和 3-5 推导出
der_can_revokeGD(u, r, u', r',rvk_opt)	由规则 3-2 和 3-6 推导出
der_can_revokeGI(u, r, u', r', rvk_opt)	由规则 3-3 和 3-7 推导出
error(u, r, u', r')	由规则 3-11 推导出

3.1.3.2 授权判定规则定义

1．基本的授权判定规则

基本的授权判定规则都是$H \leftarrow .$的形式，其规则体的部分是空的，表示这些规则是永为真的。基本的授权判定规则是预先定义好的安全策略，以及在 RBAC 和 RDM2000 中说明的事实。

规则 3-1 用户-用户的转授权判定规则

$can\_delegate(r, cr, n) \leftarrow .$

其中r、cr、n分别表示角色、前置条件、最大转授权深度。该规则表示的意思是，在不超过最大转授权深度n的情况下，具备角色r或者是角色等级比r高

级的角色的用户，可以把角色r或者是比角色r低级的角色授予在当前状况下，具备的角色满足前置条件cr的用户。

规则 3-2 授权依赖撤销判定规则

$can\_revokeGD(r) \leftarrow .$

其中r是角色。该规则表达的意思是，具备被授角色r或者是角色等级比r低级的用户，只能够由其直接转授用户来撤销掉其角色r。

规则 3-3 非授权依赖撤销判定规则

$can\_revokeGI(r) \leftarrow .$

其中r是角色。该规则表达的是，具备被授角色r或者是角色等级比r低级的用户可以由在其转授权路径中的任何一个节点的用户来撤销掉角色r。

2．授权判定的推导规则

授权判定的推导规则体的部分描述的是推理的逻辑表达式（inference logic），规则的结果是真（授权被接受）或者假（授权被否决）两种结果。

规则 3-4 访问控制规则

$allow(u,r,p,s) \leftarrow active(u,r,s) \,\&\, in(p, permissions(s))$

其中，u、r、p、s分别表示用户、角色、权限、会话。该规则表示在会话s中激活的具备角色r的用户u，拥有权限p。

规则 3-5 用户-用户的转授权判定规则

$$
\begin{aligned}
&der\_can\_delegate(u,r,u',r',d\lg\_opt) \leftarrow \\
&active(u,r,s)\,\& \\
&delegatable(u',r')\,\& \\
&can\_delegate(r'',cr,n)\,\& \\
&senior(r,r'')\,\& \\
&in(u',cr)\,\& \\
&junior(r',r'')\,\& \\
&lt(depth(u,r),n).
\end{aligned}
$$

其中u和u'都是用户集合的元素，r、r'、r''都是角色集合的元素，cr是前置条件，s是会话；$d\lg_opt$是布尔值，如果其值为真，则$delegatable(u',r')$的值

为真，这个参数是使用了布尔控制的办法来控制转授权的繁殖。

该规则表达的意思是，在会话 s 中，具备角色r或者是比r高级的角色r''的用户u，可以在不超过最大转授权路径n的情况下，把角色r'（r'是比r''低级的角色）转授给当前具备的角色满足前置条件cr的用户。

3.1.3.3 撤销判定的推导规则：

规则 3-6 授权依赖撤销判定的推导规则

$$der\_can\_revokeGD(u,r,u',r',rvk\_opt) \leftarrow$$
$$active(u,r,s)\ \&$$
$$can\_revokeGD(r')\ \&$$
$$equals((u,r),prior(u',r')).$$

其中u和u'是用户集合的元素，$u \neq u'$；r和r'是角色集合的元素；rvk_opt表示的是撤销的机制，包括强级联授权依赖撤销（SCDR）、弱级联授权依赖撤销（WCDR）、强非级联授权依赖撤销（SNDR）、弱非级联授权依赖撤销（WNDR）。该规则表达的意思是，用户u'可以由被激活的具备角色r的转授用户u撤销掉角色r'，其中$(u,r) = prior(u',r')$。

规则 3-7 非授权依赖撤销判定的推导规则

$$der\_can\_revokeGI(u,r,u',r',rvk\_opt) \leftarrow$$
$$active(u,r,s)\ \&$$
$$can\_revokeGI(r')\ \&$$
$$in((u'',r''),path(u',r')).$$

其中，rvk_opt表示的是撤销机制，包括强级联（SCIR）、弱级联（WCIR）、强非级联（SNIR）、弱非级联（WNIR）。该规则表示的是，用户u'可以被任何转授权路径中的初始用户u撤销掉角色r'。

规则 3-8 自动级联撤销的判定规则

$$der\_can\_revoke\_auto\_cascade(u,r) \leftarrow$$
$$in((u',r'),path(u,r))\ \&$$
$$revoked\_cascade(u'',r'',u,r).$$

该规则表达的意思是，如果任何在转授权路径$path(u,r)$中的用户角色指派关系(u',r')被级联撤销，则用户u也被撤销掉角色r。

规则 3-9 自动强撤销的判定规则

$$der\_can\_revoke\_auto\_strong(u,r) \leftarrow$$
$$revoked\_strong(u'',r'',u,r') \&$$
$$senior(r,r') \&$$
$$der\_can\_revokeGI(u'',r'',u,r',WNIR).$$

该规则的意思是，如果用户角色指派关系 (u,r') 被具有角色 r'' 的用户 u'' 强撤销，则用户 u 被同一个用户 u'' 弱撤销掉任何比角色 r 高级的角色 r'。

规则 3-10 时间限制的自动触发式撤销的判定规则

$$der\_can\_revoke\_auto\_expire(u,r,rvk\_opt) \leftarrow expires(duration(u,r)).$$

该规则的意思是，如果加上时间限制的用户角色指派关系 (u,r) 到期，则 (u,r) 被撤销。

3.1.3.4 基于角色的约束的执行：

规则 3-11 完整性规则

$$error(u,r,u',r') \leftarrow B.$$

B 可以是很多表达形式。根据约束规则来定义，详细内容见[12]。

3.2 TRDM（Temporal Role-based Delegation Model）模型

文献[20]中提出了具有时限的转授权模型，本小节部分简单介绍下文献中的重要概念，以此总结其中的不足，从而提出本文的研究思路。

3.2.1 时限及具有时限的授权表示

在基于角色的转授权模型中，发起转授权动作的用户称为授权用户（delegating user）记作 u_{ing}，转授出去的角色成为被转授角色（delegated role），记作 r_{ed}，接受被转授角色的用户称为被授权用户（delegated user），记作 u_{ed}。

采用 $[t_b,t_e]$ 来表示时限 duration，$[t_b,t_e]$ 是一个时间段，t_b 是时间段 duration 的开始，t_e 是时间段 duration 的结束，其中 $t_b \in N, t_e \in N, t_b < t_e$。具有时限的转

授权的具体含义是u_{ing}在转授权的操作中，仅仅赋予u_{ed}在时间段 duration 中具有r_{ed}，即u_{ed}仅仅在时间段 duration 中可以行使r_{ed}所具有的权限，一旦当前时间 t>t_e，则系统自动撤销u_{ed}的r_{ed}。

TRDM 将 RBAC96 中有关授权管理的部分简化为 secoff 统一行使所有的授权管理工作。

定义具有时限的授权$t-auth=\{(u_{ing},r_{ing}),(u_{ed},r_{ed}),([t_b,t_e],t_d)\}$。

其中，$u_{ing}\in U,u_{ed}\in U,r_{ing}\in R,r_{ed}\in R,t_b\in N,t_e\in N\cup\infty,t_d\in N$

3.2.2 TRDM 模型的基本元素和系统功能函数

1．基本元素：P，R，U，S；UAO，UAD，UA；PA，RH。

2．函数：见表 3-4 所示。

表 3-4　TRDM 模型的函数

函数	定义	描述
$users-o(r,t)$	$R\rightarrow 2^U$	返回时刻t具有初始角色r的所有用户
$users-d(r,t)$	$R\rightarrow 2^U$	返回时刻t具有转授角色r的所有用户
$users(r,t)$	$R\rightarrow 2^U$	返回时刻t具有角色r的所有用户
$user(s_i,t)$	$S\rightarrow U$	返回时刻t会话s_i所属的用户
$role(s_i,t)$	$S\rightarrow 2^R$	返回时刻t会话s_i所具有的所有角色
$permissions(s_i,t)$	$S\rightarrow 2^P$	返回时刻t会话s_i所具有的所有权限
$role-u(u,t)$	$U\rightarrow 2^R$	返回时刻t用户u所具有的所有角色
$prior(ing_1,(u,r),time_1)$	$U\times R\rightarrow U\times R$	返回授权路径中在(u,r)之前的所有授权
$path(u,r)$	$U\times R\rightarrow((u_0,r_0),...,(u_i,r_i))$	返回授权(u,r)的授权路径

$depth(u,r)$	$U\times R\rightarrow N$	返回授权(u,r)的授权路径长度
$broths((u_{ing},r_{ing}),(u,r),time)$	$U\times R\rightarrow 2^{(U\times R)}$	返回由(u_{ing},r_{ing})转授的其他授权(u,r)
$width((u_{ing},r_{ing}),(u,r),time)$	$U\times R\rightarrow N$	返回同是由(u_{ing},r_{ing})转授的授权(u,r)的个数
$valid-d(u,r,t)$	$U\times R\times T\rightarrow[t_b,t_e]$	返回时刻t用户u具有r的有效时限

3.2.3 转授权判定和撤销判定

1．转授权的判定

$can\_delegate\subseteq R\times CR\times N\times Y\times TIME$；

$d\lg t=\{r,cr,n,y,([t_b,t_e],t_d)\}\in can\_delegate$

$\Leftrightarrow u_{ing}\in\{u\mid users(r'),r'\geq r\}$ I

$roles-u(u_{cd},t)\in cr$ I

$n(d\lg t)\leq n$ I

$y(d\lg t)\leq y$ I

$[t_b,t_e]\subseteq valid-d(u_{ing},r,t_d)$

其中，$R,CR,N,Y,TIME$分别是角色、前置条件、最大转授权深度、最大转授权宽度、时限的集合。$n(d\lg t)$表示转授权操作$d\lg t$中被转授角色r的再次转授次数，即转授权深度。$y(d\lg t)$表示当前操作中被转授角色被同一用户转授次数，即转授权宽度。

2．授权撤销的判定

$can\_auto\_revoke\in R\times TIME.$

$(r,t)\in can\_auto\_revoke\Leftrightarrow valid-d(u,r,t)=[t_b,t_e],t>t_e$。

基于时限的自动撤销授权方式是一种比较有用的撤销方式，可以让系统自动撤销到期的授权，从而减轻 secoff 的负担。TRDM 只定义了基于时限的自动撤销授权方式的判定规则，没有定义基于时限的用户主动撤销转授权的方式。

3.3 小结现有的基于角色的转授权模型的不足

通过 3.1 和 3.2 节的分析，可以看出其中存在的不足。

1. RDM 存在的主要不足

（1）只在转授权时带有有效时间的限制，到期自动撤销，没有把时间信息加入其他的基本元素；

（2）在转授权树中，如果是根节点被撤销，如何处理没有讨论；

（3）不支持多重转授权（在一次转授过程中同时转授给多个用户）；

（4）管理角色的转授权与撤销、管理角色如何去管理和监控常规角色的转授与撤销没有讨论。

2. TRDM 的不足

（1）TRDM 中为简化讨论，转授的初始用户只有管理员 secoff；

（2）不支持转授角色中的部分权限；

（3）只讨论了系统自动撤销授权的情况，没有详细讨论带时限的用户主动撤销的情况；

（4）没有讨论在转授权和撤销时，如果更改了授权的有效时间，对转授权树带来的影响如何处理。

3. 本文的研究思路

本文所做的工作将在后续章节中一一讨论，主要包括六点。

（1）总结带时限的转授权树的特点，并得出了形式化的结论；

（2）不仅支持带时限的完全转授权，也支持转授角色中的部分权限；

（3）详细讨论了带时限的用户主动撤销授权的处理机制；

（4）转授权和撤销操作若是对有效时间的更改，归纳分析了对转授权树的

结构产生影响之后的处理机制；

（5）扩展研究管理角色的转授权与撤销的处理机制，并重点讨论了管理角色的转授权树与常规角色的转授权树之间建立联系的问题，即如何管理和监控常规角色的转授权与撤销操作的问题；

（6）定义重要的约束规则，包括静态职责分离中的角色冲突与权限冲突。

本文的后续章节将本文所扩展的模型称作——基于角色的带时限的转授权与撤销模型（Temporal Role-based Delegation and Revocation Model, TRDRM），简称 TRDRM。

第 4 章 基于常规角色的带时限的转授权与撤销机制的探讨

本章主要探讨基于常规角色的带时限的转授权与撤销机制，首先需要对基本元素进行定义，而时间是最重要的因素，借用文献[26]的时间系统的定义，本文首先在此基础上扩展定义了时间点、时间区间、时间区间集合之间的包含关系。接着，对 TRDRM 的基本函数进行了定义，并且在详细分析了各种转授权与撤销的情形之后，使用基于规则的策略描述语言对转授权与撤销的判定规则进行了形式化描述。

4.1 基本元素定义

4.1.1 时间相关的基本定义

定义 4-1　时间系统[26]

令整个时间系统：$TS =< TP, \le_t >$，其中 $TP = \{p_1, p_2, ..., p_n\}$ 为时间点的有限集合。$\le_t$ 表示 TP 上的时序，$p_i \le_t p_{i+1}$ 表示 p_i 不会出现在 p_{i+1} 之后，即 p_i 发生在 p_{i+1} 之前或跟 p_{i+1} 同时发生。由此可以得到：

$\forall p_i \in TP, p_i \le_t p_i$；关系 $\le_t$ 是在 TP 上自反的；

$\forall p_i, p_j \in TP$，若 $p_i \le_t p_j$ 且 $p_j \le_t p_i$，则 $p_i = p_j$；关系 $\le_t$ 是在 TP 上反对称的；

$\forall p_i, p_j, p_k \in TP$，若 $p_i \le_t p_j$ 且 $p_j \le_t p_k$，则 $p_i \le_t p_k$；关系 $\le_t$ 是在 TP 上传递的；

因此，TS 是一个偏序。同时，对 $\forall p_i, p_j \in TP$，或者 $p_i \le_t p_j$，或者 $p_j \le_t p_i$，因此，TS 又是一个全序，具备了良好的代数结构。

定义 4-2　时间区间

时间区间是由两个时间点所构成的区间，即 $TI = \{(p_i, p_j) \mid p_i, p_j \in TP, p_i \le_t p_j\}$

定义 4-3　时间区间的集合

$TIS = 2^{TI}$，表示由时间区间所组成的集合。

定义 4-4　时间点属于时间区间的定义

对于$\forall p \in TP, t \in TI$，若$p$和$t$满足时间点与时间区间的五种关系[26]当中的以下三种关系的任意一种：

（1）$Meets(p,t), Starts(p,t)$，即p与t同时发生；

（2）$Met-by(p,t), Finishes(p,t)$，即$p$与$t$同时结束；

（3）$During(p,t)$，即p发生在t的时间区间范围内，在时间轴上p比t的起始时间晚，比t的结束时间早；

则称时间点p属于时间区间t，记为$p \in_p t$。

定义 4-5　两个时间区间之间的包含关系

对于$\forall t_1, t_2 \in TI$，若t_1与t_2满足 Allen 提出的十三种时态区间关系[26]的以下四种关系的任意一种：

（1）$During(t_1,t_2)$，即t_1比t_2晚开始，且早结束，在时间轴上t_1的区间范围被包含在t_2的区间范围内；

（2）$Starts(t_1,t_2)$，即t_1和t_2有共同的起始点，但t_1比t_2先结束；

（3）$Finishes(t_1,t_2)$，即t_1和t_2有共同的结束点，但t_1比t_2晚,开始；

（4）$Equals(t_1,t_2)$，即t_1和t_2有共同的时间区间，t_1和t_2在时间轴上重合；

则称时间区间t_1包含在t_2中，记为$t_1 \subseteq_t t_2$。

定义 4-6　时间区间与时间区间集合的属于关系

对于$\forall t \in TI, S_t \in TIS$，如果$\exists t' \in S_t$使得$t \subseteq_t t'$，则称时间区间$t$属于时间区间的集合$S_t$，记为$t \in_t S_t$。

定义 4-7　时间点与时间区间集合的属于关系

对于$\forall p \in TP, S_t \in TIS$，如果$\exists t \in TI$，使得$p \in_p t$且$t \in_t S_t$成立，则称时间

点 p 属于时间区间的集合 S_t，记为 $p \in_s S_t$。

定义 4-8　两个时间区间集合之间的包含关系

如果两个时间区间的集合 $S_t, S_t' \in TIS$，满足对于 $\forall t \in S_t$，均有 $t \in_s S_t'$，则称时间区间的集合 S_t 包含在 S_t' 中，记为 $S_t \subseteq_s S_t'$。

4.1.2 TRDRM 中具有有效时间的基本元素的定义

基本集合包括用户集 U、角色集 R、权限集 P、会话集 S、用户到角色的指派关系 UA、角色到权限的指派关系 PA。

考虑到用户在系统中对客体对象的访问是通过用户—角色—权限的层层映射关系来实现的，如果为用户到角色的映射关系 UA 添加有效时间的控制，也就为用户的访问控制添加了有效时间，这是一种动态的授权方式，用户只有在有效的时间内才具备有效的角色，才具备有效的权限。这不同于静态的授权方式，一旦赋予用户访问控制的权限，用户就一直占有直到这个用户被删除。在分布式环境中，显然需要的是动态的授权方式。

因此本文不给出具有有效时间的 U、R、P、S 集合元素的显式定义，形式上与 RBAC[4]里的基本定义相同，只是默认 U、R、P、S 中的元素的有效时间是 forever[26]。

定义 4-9　用户 U、角色 R、权限 P、会话 S（来自 RBAC96）

用户（Users）U：$U = \{u_1, u_2, ...u_n\}$

角色（Roles）R：$R = \{r_1, r_2, ...r_n\}$

权限（Permissions）P：$P = 2^{OPS \times OBS}$，其中 $OPS = \{op_1, op_2, ..., op_n\}$，表示所有操作（Operation）的集合，比如查看、增加、删除、修改等等。$OBS = \{ob_1, ob_2, ...ob_n\}$，表示所有客体对象的集合，比如数据库的一张表、一张网页中的一个按钮等等。

会话（Sessions）S：$S = \{s_1, s_2, ...s_n\}$，表示所有会话的集合，会话对应于一个用户和一组激活的角色，表示用户进行角色激活的过程。一个用户可以进行多次会话，在每次会话中激活不同的角色，这样用户也将具有不同的访问权限。用

户必须通过会话才能激活角色。

定义 4-10　具有有效时间的用户角色指派关系定义

$UAT \subseteq U \times R \times TIS, UAT = \{(u,r,S_t) \mid (u,r) \in UA, S_t \in TIS\}$

$UAOT \subseteq U \times R \times TIS, UADT \subseteq U \times R \times TIS, UAT = UAOT \cup UADT$

在实际应用中，角色与权限之间的对应关系相对来说是固定的，为了简化讨论，这里不显式定义具有有效时间的角色到权限的指派关系，仍然认为其有效时间为 forever。

定义 4-11　角色到权限的指派关系

$PA \subseteq P \times R, PA = \{(perm,r) \mid perm \in P, r \in R\}$

定义 4-12　角色的层次结构（Role Hierarchy）

$RH \subseteq R \times R$

4.1.3 TRDRM 中具有有效时间的转授权相关的定义

定义 4-13　具有有效时间的转授权关系的定义

$DLGTT \subseteq UAT \times UAT$，即 $((u,r,S_t),(u',r',S_t')) \in DLGTT$ 表示在有效时间 S_t 内具有角色 r 的用户 u，把角色 r' 转授给用户 u'，并且在 S_t' 的范围内有效。其中，角色 r' 是等同于 r 的角色或者是比 r 低级的角色，并且有效时间集合 S_t' 必须是包含在中 S_t 的。

定义 4-14　多步转授权中具有有效时间的基本定义

具有有效时间的转授权路径：$DPT \subseteq UAT \times UAT$

具有有效时间的转授权树：$DTT \subseteq UAT \times UAT$

从定义中可以看出，DTT 表示树的方法是用树中的边的关系来表示转授权树。那么树中的节点 $Node$ 必须满足：

$DTT \subseteq Node \times Node \subseteq DLGTT \subseteq UAT \times UAT$。

结论 4-1　UAOT 集合就是所有存在的转授权树的根节点的集合，即对于每一个属于 UAOT 集合的元素而言，必然存在一棵以这个元素为根节点的转授权

树。而 UADT 中的某个元素的转授权路径则是从这个节点到根节点的一个寻找前驱的过程。

4.2 基本函数定义

本小节给出多步转授权中的基本函数定义。

定义 4-15　具有有效时间的前驱函数、转授权路径函数、转授权深度函数

求前驱函数：

$$PriorT : UAT \rightarrow UAT$$

$$PriorT(uat) = \{uat' \mid uat \in UADT, (uat', uat) \in DLGTT\}, uat = (u, r, S_t)$$
$$PriorT(uat) = \{\phi \mid uat \in UAOT\}$$

求转授权路径函数：

$$PathT(uat_0) = \{uat_n \rightarrow \ldots \rightarrow uat_i \rightarrow uat_{i-1} \rightarrow \ldots \rightarrow uat_0 \mid uat_i = PriorT(uat_{i-1})\}, uat_0 = (u, r, S_t)$$

求转授路径中的转授权深度函数：

$$DepthT : UAT \rightarrow N$$

从以上的分析，可以归纳出每一颗转授权树所具有的特性。

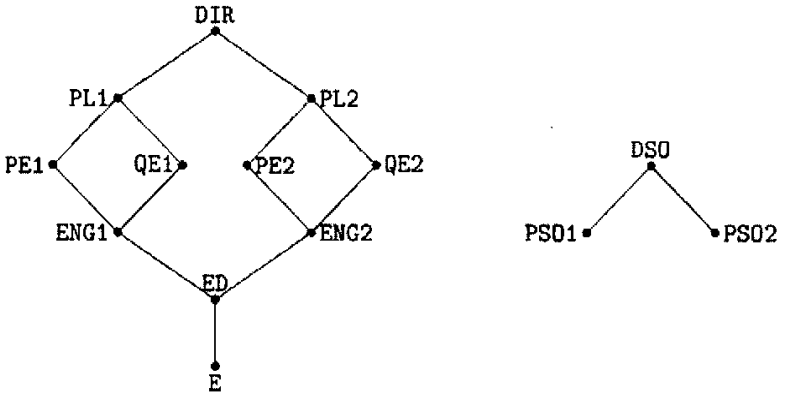
对于转授权树中的任意节点，则该节点的有效时间区间集合必然包含在其转授权路径上的所有节点的有效时间区间集合中。利用定义 4-8 的两个时间区间集合的包含关系可以对此结论进行形式化描述。

结论 4-2　对于 $\forall uat_0(u, r, St_0) \in DTT$，若有 $\forall_{0 \le i \le n} uat_i(u, r, St_i) \in PathT(uat_0)$ 成立，则有 $St_0 \subseteq_s St_i$ 成立。

4.3 基于常规角色的转授权与撤销的处理机制探讨

4.3.1 本文使用的角色层次结构（Role Hierarchy）的例子

为了更好地对比分析，此处仍然沿用 RBAC96 中的角色层次结构的例子，如图 4-1 所示。图 4-1 中各名称的含义如表 4-1 所示。



（a）常规角色的层次结构　　（b）管理角色的层次结构

图 4-1 RBAC96 的角色层次结构的例子

表 4-1 图 4-1 中各节点名字的含义

DIR	Director
PL	Project Leader
PE	Production Engineer
QE	Quality Engineer
ENG	Engineer
ED	Engineering Department
E	Employee
DSO	Department Security Officer
PSO	Project Security Officer

从图 4-1 和表 4-1 中可以看出，最初级的角色是雇员（Employee），然后只给出了一个部门——工程部（Engineering Department）。工程部按工程的不同分成了两个不同的团队，即 ENG1 和 ENG2，每个团队里又包含了产品工程师（PE）、质量工程师（QE），以及 PE 和 QE 的上级——项目组长（PL）这三种角色，而 PL1 和 PL2 的上级是总监（DIR）。这里为了讨论问题的方便，在工程部中只列举出了两个不同的项目团队，很显然，不同的项目小组，他们所拥有的访问权限，所能访问的资源是不同的。

4.3.2 加入有效时间控制之后的转授权与撤销的主要情形

本小节对加入有效时间控制之后的转授权与撤销的主要情形进行了归纳。

1．转授权的情形

（1）在有效时间范围内，基于角色的完全授权。即转授的权限都是以整体角色为单位，转授的权限是某个角色的全部权限。

（2）在有效时间范围内，基于角色的部分授权。即转授的权限仅仅是某个角色的部分权限，但是通过将部分权限授予给一个临时的角色来转授给用户，转授权的单位仍是角色，可参考文献[13]的处理方式。

（3）对于已有的转授权关系$((u,r,S_t),(u',r',S_t'))\in DLGTT$，增加$(u',r',S_t')$的有效时间。即对$S_t'$作更新操作。

（4）(u,r,S_t)对(u',r',S_t')执行增加有效时间的操作，但(u',r',S_t')不是由于(u,r,S_t)的转授操作而创建，即在此操作之前$((u,r,S_t),(u',r',S_t'))\notin DLGTT$。

2．撤销的情形

（1）基于角色的完全撤销，即撤销掉整个被授的角色；

（2）基于角色的部分撤销，即只撤销掉被授角色中的部分权限；

（3）对于已有的转授权关系，减少授权的有效时间的范围。

4.3.3 各种转授权和撤销的情形对转授权树的结构带来的影响

本小节将举例说明针对转授权和撤销的各种情况的处理机制。在此之前，需要说明几个前提。

（1）为了简化讨论，本文暂时不考虑如何去选择时间粒度的问题。本文认为选择时间粒度的问题，属于系统实现细节的范畴，需要根据实际需求选择。时间粒度的详细讨论见文献[26]。

（2）为了能突出实质问题，本小节的所有例子都属于有效时间区间的集合里只包含一个有效时间区间的情况，对于含有多个有效时间区间的集合的讨论，以此类推。

（3）本文仅在本小节为了举例的方便，使用$Natural =< N,\leq >$来表示有效时间。本文认为这是合理的，因为本文所定义的时间系统$TS =< TP,\leq_t >$是将时间看

作是离散的，同时也是一个全序关系，与自然数集合 $Natural =< N, \leq >$ 是同构的。

表 4-2 中给出的用户和角色的对应关系的例子，但在后面的简化讨论中，本文只选取表中的有效时间集合的第一个元素作为其有效时间区间。

表 4-2 用户－角色－有效时间的示例

用户名	角色名	有效时间
Mike	DIR	{[1,10], [20, 30], ……}
John	PL2	{[1, 20], [40, 50],……}
Betty	QE1	{[1,30], [60, 70], ……}
Tom	PE2	{[1, 5], [10, 25], ……}
Bob	ENG1	{[2,10], [45, 90], ……}
Cathy	ED	{[1,30], [35, 55], ……}

4.3.3.1 转授权的情况

1．在有效时间范围内，基于完整角色的转授权

$((Mike, DIR, [1,10]), (John, DIR, [2,9])) \in DLGTT$

$((Mike, DIR, [1,10]), (Betty, PL1, [2,7])) \in DLGTT$

$((Mike, DIR, [1,10]), (Betty, DIR, [5,10])) \in DLGTT$

$((Betty, PL1, [2,7]), (Cathy, QE1, [3,4])) \in DLGTT$

$((Betty, PL1, [2,7]), (Bob, PE1, [2,5])) \in DLGTT$

$((Betty, DIR, [5,10]), (Tom, PE2, [6,8])) \in DLGTT$

2．在有效时间范围内，基于角色的部分转授权

$((John, DIR, [2,9]), (Tom, \{PL2, p\_range\}, [2,9])) \in DLGTT$

这个部分转授权的例子，为了清楚的显示 *Tom* 被授的角色的权限，用了 $\{PL2, p\_range\}$ 的表达方式，实际在处理的时候，是将 $\{PL2, p\_range\}$ 的权限赋予了一个临时的角色 T_R，再将 T_R 和用户 *Tom* 建立对应关系即可。被授予角色中的部分权限的用户不能再将这个临时的角色 T_R 转授给他人，这是出于安全性的考虑。可参考文献[13]的处理方式。

通过上述转授权的操作，就形成了图 4-2 中的转授权树。

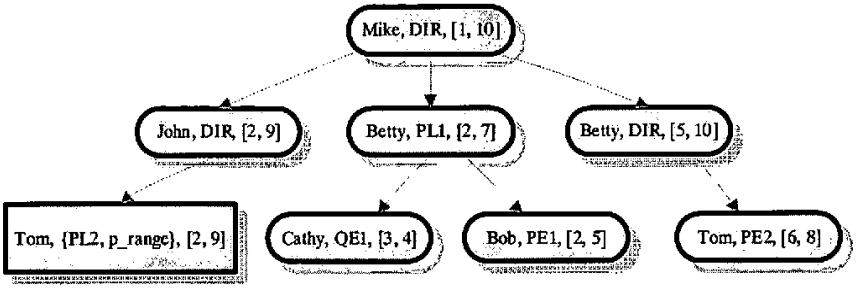


图 4-2　转授权树的例子

3．对于已存在的转授权关系，增加授权关系的有效时间

（1）情形一、不会对转授权树的结构造成影响

例 4-1　已有的转授权关系 $((Betty, DIR, [5,10]), (Tom, PE2, [6,8])) \in DLGTT$，$(Betty, DIR, [5,10])$ 在新增 $((Betty, DIR, [5,10]), (Tom, PE2, [8,9])) \in DLGTT$ 之后，则需要对有效时间的区间进行合并。

合并之后的关系为 $((Betty, DIR, [5,10]), (Tom, PE2, [6,9])) \in DLGTT$，转授权树的结构不会受到影响，只是需要将树中原有的结点 $(Tom, PE2, [6,8])$ 修改为 $(Tom, PE2, [6,9])$。

（2）情形二、会对转授权树的结构造成影响

例 4-2　如果 $((Mike, DIR, [1,10]), (Cathy, QE1, [3,8])) \in DLGTT$ 成立，那么原有的树的结构就要发生变化，这是由于 $(Cathy, QE1, [3,4])$ 被修改了有效时间导致。

因为原有的关系 $((Betty, PL1, [2,7]), (Cathy, QE1, [3,4])) \in DLGTT$，在发生 $(Cathy, QE1, [3,4]) \rightarrow (Cathy, QE1, [3,8])$ 的变化之后，已经超过了 $(Betty, PL1, [2,7])$ 的有效时间。则新的转授权树如图 4-3 所示。

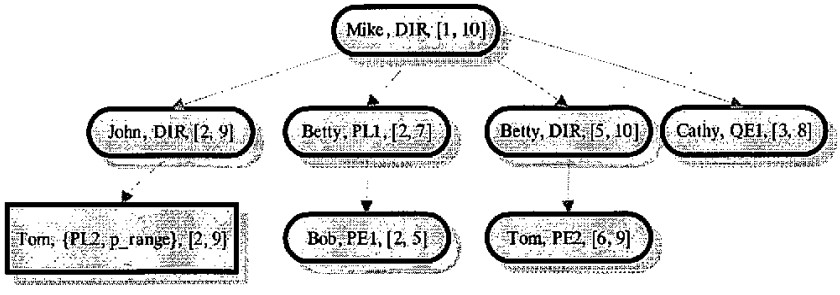


图 4-3 增加有效时间引起树的变化

4．不是由于转授用户的转授操作而形成的用户角色对应关系，增加其有效时间

例 4-3　在表 4-2 中，如果拥有角色 DIR 的 Mike 要增加 Tom 拥有 PE2 角色的有效时间，但是 Tom 的 PE2 角色并不是其转授而来，则属于此种情况，如图 4-4 所示。

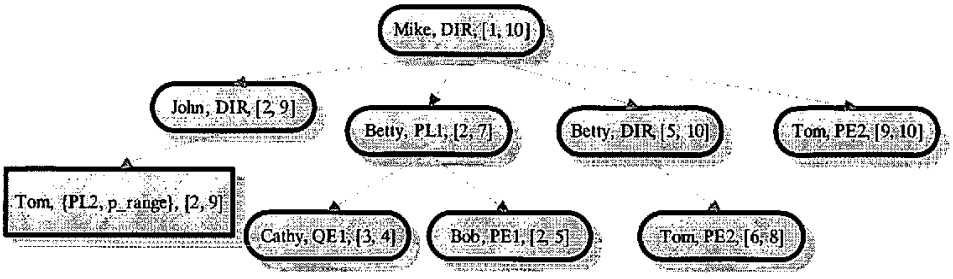


图 4-4 增加初始的用户角色对应关系的有效时间

4.3.3.2 撤销的情况

对转授权关系的撤销，可以分成两大类别：系统自动撤销、用户主动撤销。

1．有效时间到期，系统自动撤销的情况

此种情况比较简单，因为转授权树中存在的节点满足结论 4-2，则如果是自动撤销的情况，那么一定是从叶子节点开始，由于时间到期而逐层删除节点，且只会删除叶子节点。不需要对树的结构进行额外的操作来调整。

2．用户主动撤销的情况

要对转授权树中的某个节点进行撤销的操作，先要讨论清楚究竟谁有权利去撤销它的问题。

从这个角度来说，撤销可分为两类，一类是授权依赖撤销，一类是非授权依赖撤销（授权独立撤销）。在本文 2.3.2 节部分已经进行了介绍。

例 4-4　如果只有 $(Betty, DIR, [5,10])$ 能对 $(Tom, PE2, [6,9])$ 进行撤销操作，因为 $(Tom, PE2, [6,9])$ 是由于 $(Betty, DIR, [5,10])$ 直接转授而来，则属于授权依赖撤销的情况。

如果在 $(Tom, PE2, [6,9])$ 的转授权路径上的节点 $(Mike, DIR, [1,10])$ 也能对 $(Tom, PE2, [6,9])$ 进行撤销操作，则属于非授权依赖撤销。

下面讨论由于用户主动撤销机制的不同，对转授权树造成的影响。

（1）在有效时间范围内，基于完整角色的撤销

这种情况分为强级联撤销、弱级联撤销、强非级联撤销、弱非级联撤销。本文将一一举例说明。

例 4-5　(*Mike*, *DIR*, [1,10]) 想要对 (*Betty*, *PL*1, [2,7]) 进行撤销操作，则四种不同的撤销机制对树的影响是不同的，仍然以图 4-2 中的树为例。

a. 强级联撤销

如图 4-5 所示，若属于强级联撤销的情况，则是不仅 (*Betty*, *PL*1, [2,7]) 这个节点及其子树要被撤销掉，同时与 Betty 这个用户对应的显式的高级角色的节点也将被撤销掉，即图中的 (*Betty*, *DIR*, [5,10]) 这个节点及其子树也将被撤销。

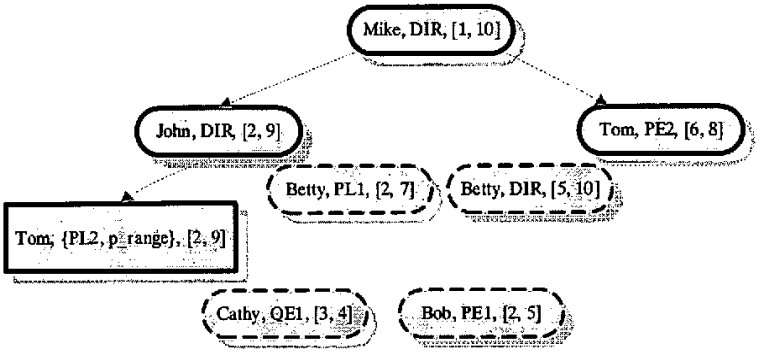


图 4-5 强级联撤销

b. 弱级联撤销

如图 4-6 所示，若是属于弱级联撤销的情况，则只需要撤销掉 (*Betty*, *PL*1, [2,7]) 这个节点及其子树。

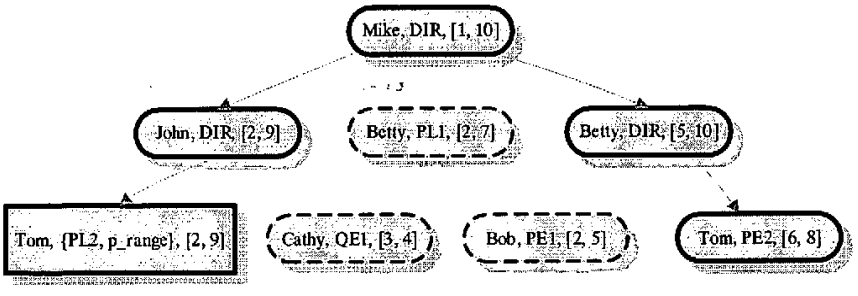


图 4-6 弱级联撤销

c. 强非级联撤销

如图 4-7 所示，若是属于强非级联撤销的情况，则需要撤销掉 $(Betty, PL1, [2,7])$ 这个节点，但是其子树保留，子树的节点由发出撤销动作的节点接管，同时，Betty 这个用户对应的显式的高级角色的节点被撤销，即图中的 $(Betty, DIR, [5,10])$ 节点被撤销，但其子树仍然保留，由发出撤销动作的节点接管。

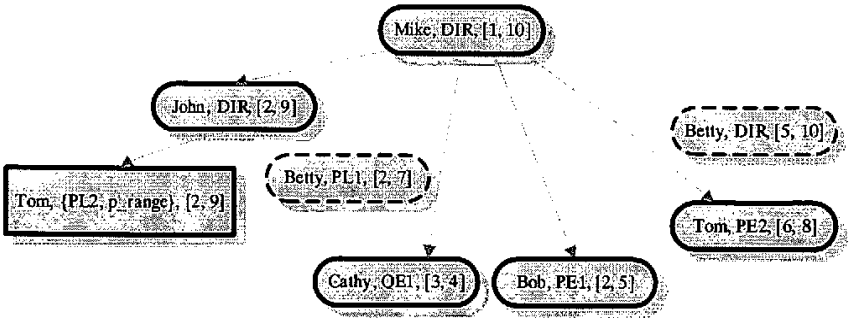


图 4-7 强非级联撤销

d. 弱非级联撤销

如图 4-8 所示，若属于弱非级联撤销的情况，则只需要撤销掉 $(Betty, PL1, [2,7])$ 这个节点即可。

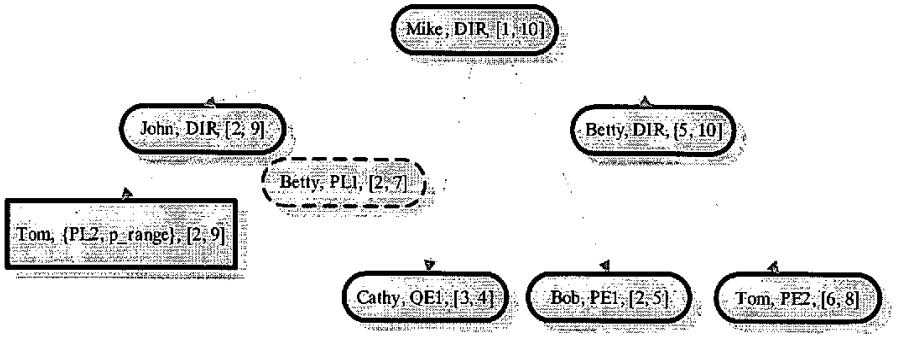


图 4-8 弱非级联撤销

（2）有效时间范围内，撤销已转授角色中的部分权限

例 4-6　仍然是 $(Mike, DIR, [1,10])$ 对 $(Betty, PL1, [2,7])$ 进行撤销操作。

但是只撤销 Betty 对应的角色 PL1 中的一部分权限，暂且记作 $\overline{P}$，则

$\overline{P} \subseteq permissionsR(PL1)$，而 Betty 仍然具备角色 PL1 的未被撤销的那一部分权限，暂且记为 P，则有 $P \subseteq permissionsR(PL1)$。

那么 $P \cap \overline{P} = \phi \wedge P \cup \overline{P} = permissionsR(PL1)$ 成立。

本文认为此种情况在实际应用中并不常见，这里给出一种可行的解决方案。

首先，撤销掉 $(Betty, PL1, [2,7])$ 这个节点，而其孩子 $(Bob, PE1, [2,5])$ 和 $(Cathy, QE1, [3,4])$ 由 $(Mike, DIR, [1,10])$ 接管，然后再由 $(Mike, DIR, [1,10])$ 使用前面讨论的部分授权方法，将 $\overline{P}$ 通过一个临时角色授予给 Betty，而 Betty 由于只具有 PL1 角色的部分权限，不能再转授给他人。如图 4-9 所示。

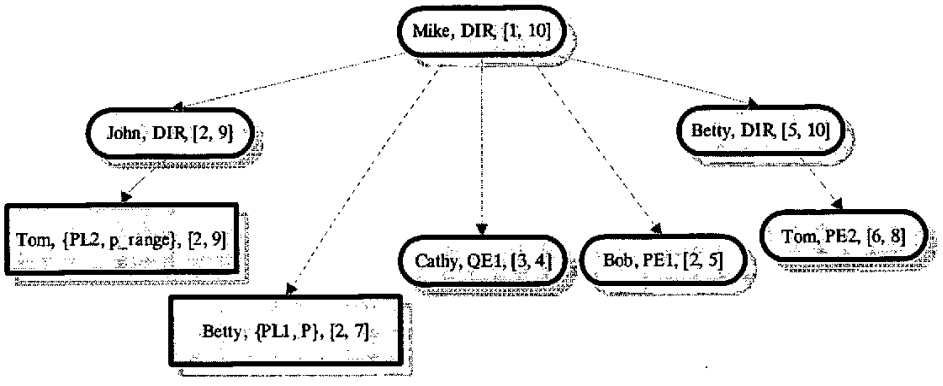


图 4-9 撤销已转授的完整角色中的部分权限

（3）撤销临时角色与用户的指派关系（即对转授的部分权限的撤销）

此种情况比较简单，因为不允许转授给临时角色的权限再次被转授，所以撤销的节点均是对叶子节点的操作，不会有额外的操作来调整树的结构。

（4）对于已存在的转授权关系，减少其有效时间

a. 情形一、不会对树的结构造成影响

例 4-7　$(Mike, DIR, [1,10])$ 对 $(Betty, PL1, [2,7])$ 进行撤销操作，修改之后的转授权关系为 $((Mike, DIR, [1,10]), (Betty, PL1, [2,5])) \in DLGTT$，则不会改变树的结构，只需要更新原有的节点 $(Betty, PL1, [2,7])$ 为 $(Betty, PL1, [2,5])$ 即可。

b. 情形二、会对树的结构造成影响

例 4-8　如果 $(Mike, DIR, [1,10])$ 对 $(Betty, PL1, [2,7])$ 进行撤销操作，修改之后的转授权关系为 $((Mike, DIR, [1,10]), (Betty, PL1, [3,4])) \in DLGTT$，则转授权树将会发生结构调整。因为 $(Bob, PE1, [2,5])$ 的有效时间已经不在 $(Betty, PL1, [3,4])$ 的范围内。如图 4-10 所示。

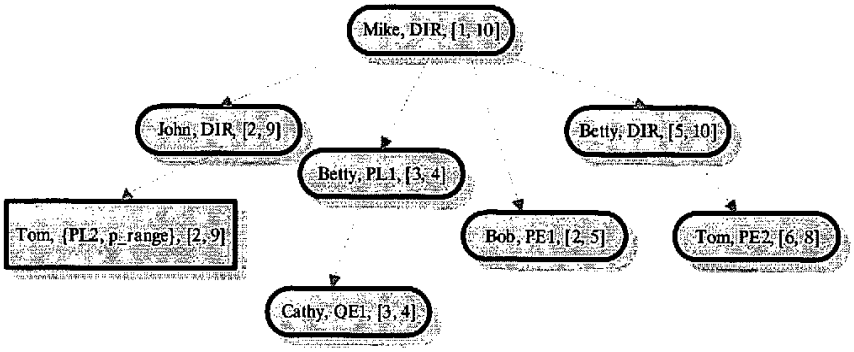


图 4-10　撤销掉部分有效时间的变化一

修改之后的转授权关系，若是 $((Mike, DIR, [1,10]), (Betty, PL1, [2,3])) \in DLGTT$，则不仅是 $(Bob, PE1, [2,5])$ 的有效时间超过了 $(Betty, PL1, [2,3])$ 的控制范围，$(Cathy, QE1, [3,4])$ 同样也是。则转授权树变为图 4-11。

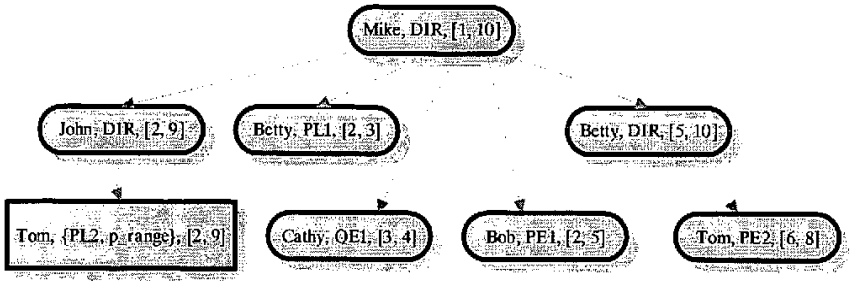


图 4-11 撤销掉部分有效时间的变化二

这样就会因为 $(Betty, PL1, [2,7])$ 被撤销的区间的不同而导致树的结构出现不同的变化。

因此，最好有一种统一的方法来处理撤销掉某个节点一部分有效时间的情

况，所以不再逐个判断其孩子的有效时间是否在其控制范围之内，而是一旦有这样的操作存在，就秉承发出撤销动作的节点需要对被执行了撤销操作的节点负责的原则，将它的孩子作为自己的孩子来管理。那么，前一个例子中，修改后的关系是$((Mike, DIR, [1,10]), (Betty, PL1, [3,4])) \in DLGTT$，调整之后的转授权树就应该是图 4-12 所示。

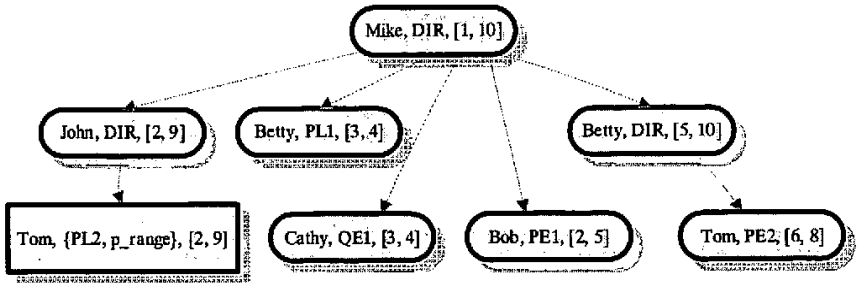


图 4-12 撤销掉部分有效时间的统一处理方式

4.4 定义转授权与撤销的判定规则

4.4.1 基础定义

本文仍然沿用[21]ARBAC97 的前置条件的概念来进行转授权的判定。

对 RDM2000 中的基本规则 can_delegate 进行了扩展。由于本文考虑的时态扩展主要是对于用户和角色的指派关系而言，所以这里的判定规则仍然是基于角色之间的基本的判定规则，只是添加了转授权深度和宽度的限制。

定义 4-16　前置条件（prerequisite condition, CR）

CR 是一个布尔表达式，采用“&”作为“与”操作符，“|”作为“或”操作符，来连接以 x 和$\bar{x}$形式表达的常规角色。例如，$CR = r_1 \,\&\, r_2 \,|\, \overline{r_3}$。

定义 4-17　下面的关系判定用户到用户的转授权：

$can\_delegate \subseteq R \times CR \times D \times W$，其中$R$、$CR$、$D$、$W$分别是角色的集合、前置条件的集合、最大的转授权路径长度以及转授所允许的宽度。

即$(r, cr, d, w) \in can_delegate$的意思是在不超过最大的转授权路径长度$d$以

及转授宽度 w 的情况下，具备角色 r 的用户（或者是比角色 r 高级的角色）能够将角色 r（或者是比 r 低级的角色）转授给任何一个满足前置条件 cr 的用户。

定义 4-18 下面的关系对转授权的撤销进行判定，分别是授权依赖撤销、非授权依赖撤销的判定（来自 RDM）：

（1）授权依赖撤销判定 $can\_revokeGD \subseteq R$

（2）非授权依赖撤销判定 $can\_revokeGI \subseteq R$

4.4.2 TRDRM 中的函数定义

4.4.2.1 描述函数（Specification Functions）定义

1. 返回时刻 p_t 具有角色 r 的所有用户的集合 $usersT:(r:R,p_t:TP)\rightarrow 2^U$

$$usersT(r,p_t)=\{u\in U \mid (u,r,S_t)\in UAT \wedge p_t \in_s S_t\}$$

2. 返回时刻 p_t 用户 u 所拥有的全部角色的集合 $rolesT:(u:U,p_t:TP)\rightarrow 2^R$

$$rolesT(u,p_t)=\{r\in R \mid (u,r,S_t)\in UAT \wedge p_t \in_s S_t\}$$

3. 返回用户 u 具有角色 r 的有效时间集合 $valid-TIS:(u:U,r:R)\rightarrow TIS$

$$valid-TIS(u,r)=\{S_t\in TIS \mid (u,r,S_t)\in UAT\}$$

4. 返回角色 r 所拥有的全部权限的集合 $permissionsR:(r:R)\rightarrow 2^P$

$$permissionsR(r)=\{perm\in P \mid (perm,r)\in PA\}$$

5. 返回会话 s 中所有被激活的用户的集合 $active\_users:(s:S)\rightarrow 2^U$

6. 返回会话 s 中所有被激活的角色的集合 $active\_roles:(s:S)\rightarrow 2^R$

7. 返回会话 s 中所有被激活的权限的集合 $active\_perm:(s:S)\rightarrow 2^P$

8. 返回转授权路径中某节点的直接前驱（父亲节点）的函数 $priorT:(uat:UAT)\rightarrow UAT$

$$priorT(uat)=\{uat' \mid uat'\in UAT \wedge uat\in UADT \wedge (uat',uat)\in DLGTT\}$$
$$priorT(uat)=\{\phi \mid uat\in UAOT\}$$

9. 返回转授权树中某节点的转授权路径的函数

$pathT:(uat_0:UAT)\rightarrow UAT\times UAT\times...\times UAT$

$pathT(uat_0)=\{(uat_n,uat_{n-1},..uat_i,..uat_0)\,|\,uat_i=priorT(uat_{i-1})\}$

10. 返回转授权树中具有角色为 r 的节点的父亲转授的其他的角色同是 r 的授权个数，$width:(uat:UAT)\rightarrow N$

11. 返回转授权路径中某个节点的路径长度 $depth:(uat:UAT)\rightarrow N$

12. $active:(u:U,r:R,s:S)\rightarrow\{true,false\}$，判断用户 u 在会话 s 中是否激活了角色 r。若用户 u 在会话 s 中激活了角色 r，则函数返回 *true*，否则为 *false*。

13. 判断角色 r 与角色 r' 的等级关系。$senior(r:R,r':R)\rightarrow\{true,false\}$，$junior(r:R,r':R)\rightarrow\{true,false\}$。若 r 比 r' 高级，则 *senior* 函数返回 *true*，*junior* 函数返回 *false*；反之，则 *senior* 函数返回 *false*，*junior* 函数返回 *true*。

14. $further\_delegatable:(u:U,r:R,d\lg\_opt:\{true,false\})\rightarrow\{true,false\}$，判断用户 u 被转授的角色 r 是否允许下一步转授权。若 $d\lg\_opt$ 为 *true* 则函数返回 *true*，反之，若 $d\lg\_opt$ 为 *false*，则函数返回 *false*。

4.4.2.2 实用函数（Utility Functions） 定义

需要说明的是，在表 4-2 中的函数 $has\_relation(x,y)$，若 x 是用户集合的元素，即 $x\in U$，则对应 y 必定是角色 $y\in R$；若 x 是角色集合中的元素，即 $x\in R$，则对应的 y 必定是权限 $y\in P$。前者记录的是用户与角色的映射关系，后者记录的是角色与权限的映射关系。以此类推。

函数 $conflicting(x,y)$，x 和 y 必定是同一个集合中的元素。即或者 $x,y\in R$，或者 $x,y\in P$，前者表示 x 和 y 是冲突的角色，后者表示 x 和 y 是冲突的权限，以此类推。

表 4-2 实用函数定义

实用函数	定义描述
lt(x, y)	若 x < y, 则函数返回 true; 反之函数返回 false。
in(x, y)	若 x 在 y 的范围之内，则函数返回 true；反之为 false。
equals(x, y)	若 x 与 y 相同，即指向同一个对象，则函数返回 true；反之为 false。
has_relation(x, y)	若 x 与 y 存在对应的映射关系，则函数返回 true；反之为 false。
conflicting(x, y)	若 x 与 y 是冲突的，则函数返回 true；反之为 false。

4.4.2.3 授权判定函数（Authorization Functions）定义

1．基本的授权判定函数定义

$can\_delegate:(r:R,cr:R,d:N,w:N)\rightarrow\{true,false\}$

$can\_revokeGD:(r:R)\rightarrow\{true,false\}$

$can\_revokeGI:(r:R)\rightarrow\{true,false\}$

2．推导的授权判定函数定义

在此就不一一说明，见表 4-3 所示。所有的规则定义详细见本文 4.4.3 节部分。

表 4-3 TRDRM 中的授权判定函数

基本的授权判定函数	描述
$can\_delegate\ (r,cr,d,w)$	见规则 4-1
$can\_revokeGD\ \ (r)$	见规则 4-2
$can\_revokeGI\ \ (r)$	见规则 4-3
推导的授权判定函数	**描述**
$allow\ (u,r,p,s)$	由规则 4-4 推导出
$total\_can\_delegate\ (u,r,u',r',S_t',d\lg\_opt)$	由规则 4-1 和规则 4-5 推导出
$partial\_can\_delegate\ (u,r,u',r',p\_range,S_t')$	由规则 4-1 和规则 4-6 推导出
$total\_can\_revokeGD(u,r,u',r',rvk\_opt)$	由规则 4-2 和规则 4-7 推导出
$partial\_can\_revokeGD(u,r,u',rd',p\_range)$	由规则 4-4 和规则 4-8 推导出
$total\_can\_revokeGI(u,r,u',r',rvk\_opt)$	由规则 4-3 和规则 4-9 推导出
$can\_revoke\_auto\_\mathrm{expire}(u,r)$	由规则 4-10 推导出

$update\_total\_can\_delegate(u,r,u',r',S_{t\_NEW})$	由规则 4-11 推导出
完整性规则判定函数	**描述**
$error(u,r,u',r')$	由规则 4-12 推导出
$error(r,p)$	由规则 4-13 推导出

4.4.3 基于规则的策略描述语言定义 TRDRM 中的判定规则

1．基本的授权规则

基本的授权规则都是$H \leftarrow .$的形式，其规则体的部分是空的，表示这些规则是永为真的。基本的授权规则是预先定义好的安全策略，以及说明的事实。

注意，与 RDM 中的规则含义相同的基本规则，本小节不做详细说明，见本文 3.1.3.2 小节。

规则 4-1 用户-用户的转授权判定规则（对规则 3-1 的扩展）

$can\_delegate(r,cr,d,w) \leftarrow .$

其中r、cr、d、w分别表示角色、前置条件、最大转授权深度以及转授权宽度。该规则表示的意思是，在不超过最大转授权深度d以及宽度w的情况下，具备角色r或者是角色等级比r高级的角色的用户，可以把角色r或者是比角色r低级的角色赋予在当前状况下，具备的角色满足前置条件cr的用户。

规则 4-2 授权依赖撤销的判定规则（来自 RDM，同规则 3-2）

$can\_revokeGD(r) \leftarrow .$

规则 4-3 非授权依赖撤销的判定规则（来自 RDM，同规则 3-3）

$can\_revokeGI(r) \leftarrow .$

2．推导的授权规则

规则 4-4 访问控制规则（来自 RDM，同规则 3-4）

$allow(u,r,p,s) \leftarrow active(u,r,s)\ \&\ in(p,permissions(s))$

规则 4-5 用户-用户的完全转授权判定规则（**TRDRM** 对规则 **3-5** 带时限的扩展）

$total\_can\_delegate(u,r,u',r',S_t',d\lg\_opt) \leftarrow$
$active(u,r,s)\ \&$
$further\_delegatable(u',r',d\lg\_opt)\ \&$
$senior(r,r'')\ \&$
$junior(r',r'')\ \&$
$can\_delegate(r'',cr,d,w)\ \&$
$has\_relation(u',cr)\ \&$
$lt(depth(u,r,valid\_TIS(u,r)),d)\ \&$
$lt(width(u,r,valid\_TIS(u,r)),w)\ \&$
$in(S_t',valid\_TIS(u,r))$.

其中，u、r、u'、r'、S_t'分别表示转授用户、转授角色、被授用户、被授角色、u'与r'对应关系的有效时间集合。$d\lg\_opt$ 则表示是否允许下一步转授权。若 $d\lg\_opt$ 为 $true$，则 $further\_delegatable(u',r',d\lg\_opt)$ 为 $true$，否则为 $false$。

该规则表达的意思是，在会话 s 中激活角色 r 的用户 u 能将角色 r' 授予给用户 u'，有效时间集合为 S_t'，并指明是否允许被授用户进行下一步转授权，但必须满足以下所有的条件： r 比 r' 高级；不超过最大转授权深度和宽度； u' 必须满足前置条件 cr； S_t' 必须在 u 与 r 对应关系的有效时间集合的范围之内。

注意，由定义 4-7 的时间点属于时间区间集合的定义，本文认为，如果对于 $\forall p_t \in_s S_t'$，$\exists conflicting(rolesT(u',p_t),r')$ 为 $true$，则转授权操作将返回失败。

规则 4-6 用户-用户的部分转授权判定规则（**TRDRM** 的扩展）

$partial\_can\_delegate(u,r,u',r',p\_range,S_t') \leftarrow$
$active(u,r,s)\ \&$
$has\_relation(r',p\_range)\ \&$
$further\_delegatable(u',r',false)\ \&$
$senior(r,r'')\ \&$
$junior(r',r'')\ \&$
$can\_delegate(r'',cr,d,w)\ \&$
$lt(depth(u,r,valid\_TIS(u,r)),d)\ \&$
$lt(width(u,r,valid\_TIS(u,r)),w)\ \&$
$in(S_t',valid\_TIS(u,r))$.

其中u、r仍然分别表示转授用户、转授角色，u'表示被授用户，r'表示被授权限范围的源角色，p_range是被授的部分权限范围。同规则 4-5 所表达的含义基本相似，不同之处是转授出去的是角色r'的一部分权限，所以r'与p_range必须要有对应关系。该规则成立必须满足的其他条件同规则 4-5。

规则 4-7 完全授权依赖撤销的判定规则（TRDRM 对规则 3-6 的带时限的扩展）

$$total\_can\_revokeGD(u,r,u',r',rvk\_opt) \leftarrow$$
$$active(u,r,s)\ \&$$
$$can\_revokeGD(r')\ \&$$
$$equals((u,r,valid\_TIS(u,r)),priorT(u',r',valid\_TIS(u',r'))).$$

其中，u表示转授用户，r表示转授角色，u'表示被授用户，r'表示被授角色，并且u'的角色r'必须是由具有角色r的用户u转授而来，才可以进行撤销操作。rvk_opt指明了此次撤销操作的撤销机制，包括强级联撤销、弱级联撤销、强非级联撤销、弱非级联撤销。

规则 4-8 部分授权依赖撤销的判定规则（TRDRM 的扩展）

$$partial\_can\_revokeGD(u,r,u',rd',p\_range) \leftarrow$$
$$active(u,r,s)\ \&$$
$$has\_relation(rd',p\_range)\ \&$$
$$equals((u,r,valid\_TIS(u,r)),priorT(u',r',valid\_TIS(u',r'))).$$

其中，u表示转授用户，r表示转授角色，u'表示被授用户，rd'表示由于部分转授权操作而产生的临时转授角色，p_range是被撤销的权限范围。rd'与p_range必须存在对应关系。而对部分转授权的撤销，只能是授权依赖撤销，即直接转授的一方才能对部分转授权的关系进行撤销。本文认为在实际应用中，部分转授权的存在是由于协同工作的需要，所以这是出于安全性的考虑。

规则 4-9 完全非授权依赖撤销的判定规则（TRDRM 对规则 3-7 的带时限的扩展）

$$total\_can\_revokeGI(u,r,u',r',rvk\_opt) \leftarrow$$
$$active(u,r,s)\ \&$$
$$can\_revokeGI(r')\ \&$$
$$in((u'',r'',valid\_TIS(u'',r'')),pathT(u',r',valid\_TIS(u',r'))).$$

其中，u表示转授用户，r表示转授角色，u'表示被授用户，r'表示被授角色，并且u'的角色r'必须是由具有角色r的用户u直接或者间接转授而来，才可

以进行撤销操作。*rvk_opt* 指明了此次撤销操作的撤销机制，包括强级联撤销、弱级联撤销、强非级联撤销、弱非级联撤销。

规则 4-10 时间到期的自动触发式撤销规则（TRDRM 对规则 3-10 的带时限扩展）

$$can\_revoke\_auto\_expire(u,r) \leftarrow expires(valid\_TIS(u,r)).$$

该规则表达的意思是，当 u 与 r 对应关系的有效时间到期，则系统会自动撤销 u 和 r 的对应关系。

规则 4-11 更新已有转授权关系的有效时间的判定规则（TRDRM 的扩展）

$$update\_total\_can\_delegate(u,r,u',r',S_{t\_NEW}) \leftarrow$$
$$active(u,r,s)\,\&$$
$$equals((u,r,valid\_TIS(u,r)),priorT(u',r',valid\_TIS(u',r')))\,\&$$
$$in(S_{t\_NEW},valid\_TIS(u,r)).$$

注意，对已有的转授权关系增加有效时间，或者是撤销有效时间的操作，本文都归为更新有效时间的操作。而规则 4-11 的目的只是判定能否更新，而对于此更新操作执行之后，会引发的不满足结论 4-2 的问题，应该如何去调整转授权树的结构，本文 4.3.3 节的部分已经予以了讨论。

3．完整性规则

本文此处只定义最基本最重要的静态职责分离规则，其他的约束规则定义不在本文的讨论范围之内。

规则 4-12　角色冲突

$$error(u,r,u',r') \leftarrow has\_relation(u',r'')\,\&\,conflicting(r',r'')$$

该规则表示的意思是，如果具有角色 r 的用户 u 想要把角色 r' 转授给用户 u'，若用户 u' 本身具有的角色 r'' 与角色 r' 是冲突的角色，那么转授权操作失败。

规则 4-13 权限冲突

$$error(r,p) \leftarrow has\_relation(r,p')\,\&\,conflicting(p,p')$$

该规则表达的意思是，如果要为角色 r 分配权限 p，若角色 r 本身拥有的权限 p' 与权限 p 是冲突的权限，那么此次为角色分配权限的操作失败。

第5章 利用RBAC原理管理RBAC的转授权与撤销

本章首先介绍基于角色的访问控制管理模型的研究现状，然后扩展第四章的基本定义和函数从而支持管理角色的转授权与撤销，并定义了利用 RBAC 的原理管理 RBAC 的转授权与撤销的有效规则。

5.1 基于角色的访问控制管理模型的研究

虽然关于基于角色的访问控制模型 RBAC 的研究相当多，但是利用 RBAC 的原理去管理 RBAC 系统的研究却为数不多。最早 RBAC96[4]引入了管理角色的概念，到后来的 ARBAC97[21]、ARBAC02[22]，以及 SARBAC 模型的不断补充完善[23，24，25]，这几个重要的模型为本文的研究提供了基础。

5.1.1 基于角色的访问控制管理模型的发展

早在 RBAC96 模型中，就提出了 RBAC 系统如何管理的问题。一般都假设 RBAC 系统是在为数不多系统管理员的直接控制下进行管理。但是在大型的应用系统中，存在的角色总数可能成百上千，仅仅依靠少数的系统管理员的集中式管理方式必然增加管理的开销，也带来安全隐患。RBAC 的初衷是在引入角色这一中介之后简化了权限的管理，那么实际上也可利用 RBAC 本身来管理 RBAC。在 RBAC96 模型中提出的基于角色的访问控制管理模型如图 5-1 所示。

从图 5-1 中可以看出，图的下半部分与上半部分十分相似，只是增加了管理角色和管理权限的概念。在 RBAC 中的管理授权就是指对用户角色的指派，角色权限的指派，以及对角色层次结构关系的修改。管理模型的关键问题就是要为管理角色定义好可管理的范围。

于是，SandHu 等人又提出了 ARBAC97 模型，对基于角色的访问控制模型进行了更深一步的讨论。ARBAC97 模型由三个组件构成，分别是 URA97 子模型控制用户到角色的指派，PRA97 子模型控制角色到权限的指派，RRA97 子模型控制角色层次结构中角色与角色之间的等级关系。

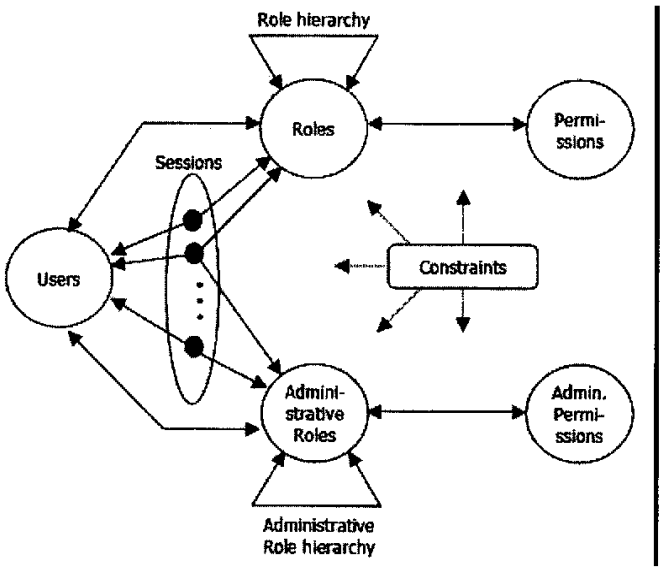


图 5-1 RBAC96 模型中引入管理角色的概念

ARBAC02 模型对 ARBAC97 模型的扩展主要是利用组织结构单元（organization structure）的概念构建用户池（user pool）和权限池（permission pool），用户池和权限池是独立于角色的层次结构关系的。同时采用由下至上（bottom-up）的方式对角色权限对应关系进行管理，而不是 ARBAC97 中的由上至下（top-down）的管理方式。详细内容见文献[22]的对比分析。

而文献[24]认为，ARBAC97 不是一个完备（complete）的模型，同时也缺乏灵活性和通用性。Jason Crampton 提出的 SARBAC 模型系列，文献[23，24，25]的不断探索，利用偏序关系的理论，对管理角色的管理范围做出了重要定义，具备极大的灵活性和通用性。

本文重点介绍 SARBAC 中的管理范围的定义，引入到转授权与撤销中来，也是对利用 RBAC 管理 RBAC 的原理的进一步扩展。

5.1.2 SARBAC 中的重要概念

5.1.2.1 基本概念定义

本小节部分介绍了文献[24，25]中的偏序关系、盖住关系、Hasse 图、反链、极大元、极小元等等概念。

已知一个偏序集<X,≤>，如果 $y < x$ 并且对于所有的 $z \in X, y \le z < x$ 推出

$z = y$，则称 x 盖住 y（x covers y），记作 $y \pi x$。图 $< X, \pi >$ 称为 X 的哈希图（Hasse diagram）。在本文中，角色的集合 R 就是一个偏序关系，并且角色树的层次结构可以用 R 的 Hasse 图来表示。

已知 $x, y \in X$，如果 $x \not\leq y$ 并且 $y \not\leq x$，则记作 x ‖ y。

已知 $Y \subseteq X$，如果对于所有的 $x, y \in Y$，有 x = y 或者 x ‖ y 成立，则称 Y 是反链，并且将 X 中的反链的集合记作 A(X)，X 的宽度就是在 X 中的最长的反链所包含的元素个数。

对于 $y \in Y$，如果对于所有的 $z \in Y, z \geq y$ 推出 $z = y$，则称 y 是集合 Y 中的极大元。同理，如果 $y \in Y$，对于所有的 $z \in Y, z \leq y$ 推出 $z = y$，则称 y 是集合 Y 中的极小元。我们把集合 Y 中的极大元的集合记作 $\overline{Y}$，极小元的集合记作 $\underline{Y}$。那么对于所有的 $Y \subseteq X$，$\overline{Y}$ 和 $\underline{Y}$ 是反链。

对于 $x \in X$，x 的下阴影（lower shadow）记作△x，△x=$\{y \in X : y \pi x\}$，x 的上阴影（upper shadow）记作▽x，▽x=$\{y \in X : x \pi y\}$。

已知 $y \in X, Y \subseteq X$，定义 $\downarrow y = \{x \in X : x \leq y\}, \uparrow y = \{x \in X : x \geq y\}$，

$\downarrow Y = \bigcup_{y \in Y} \downarrow y$，$\uparrow Y = \bigcup_{y \in Y} \uparrow y$。

定义集合 X 中以 x 和 y 为端点的封闭区间（closed range），记作 $[x, y] = \{z \in X : x \leq z \leq y\}$，开区间（open range）记作 $(x, y) = \{z \in X : x < z < y\}$。

例 5-1　利用图 4-1 中的角色层次结构图来举例说明上述定义的含义。

a. {PE1}，{PE1，QE1}，{PE1，QE1，PE2}，{PE1，QE1，PE2，QE2}都是角色树中的反链。例如，$\{PE1, QE1, PE2, QE2\} \in A(R)$。

b. 角色树的宽度是 4。

c. △ENG1={ED}，▽ENG1={PE1，QE1}

d. $\underline{\{ENG1, PE1, QE1, PL1\}} = \{ENG1\}, \overline{\{ENG1, PE1, QE1, PL1\}} = \{PL1\}$

e. $\downarrow ENG1 = \{ENG1, ED, E\}, \uparrow ENG1 = \{ENG1, PE1, QE1, PL1, DIR\}$

f. $[ENG1, PL1] = \{ENG1, PE1, QE1, PL1\}, (ENG1, PL1) = \{PE1, QE1\}$

$[ENG1, PL1) = \{ENG1, PE1, QE1\}, (ENG1, PL1] = \{PE1, QE1, PL1\}$

5.1.2.2 管理范围的定义

SARBAC 模型是围绕着管理范围（administrative scope）的概念建立的。对于每一个角色$r \in R$都有一个管理范围，管理范围定义了能够被角色r修改的一组角色的集合。管理范围的确定取决于角色本身的结构关系。也就是说，如果角色r'在角色r的管理范围内，那么对r'的任何操作都只会被r或者是比r高级的角色观察到。这样就可以避免因为角色层次结构的更改而引起对r'的某些修改所产生的意料不到的副作用。

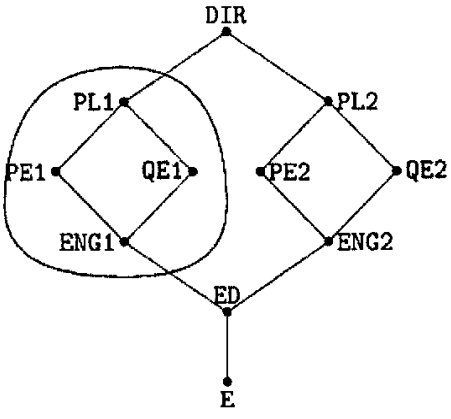
定义 5-1 [25] 一个角色r的管理范围，记作$\sigma(r)$，定义为$\sigma(r) = \{s \in \downarrow r : \uparrow s \subseteq_{\beta} r\}$，其中$\beta\ r = \downarrow r \cup \uparrow r$。一个角色的严格管理范围定义为$\sigma(r) \setminus \{r\}$，记作$\hat{\sigma}(r)$。则对于$A \subseteq R, \sigma(A) = \{r \in \downarrow A : \uparrow r \subseteq_{\beta} A\}, \hat{\sigma}(A) = \sigma(A) \setminus A$。

例5-2　求$\sigma(PL1)$。

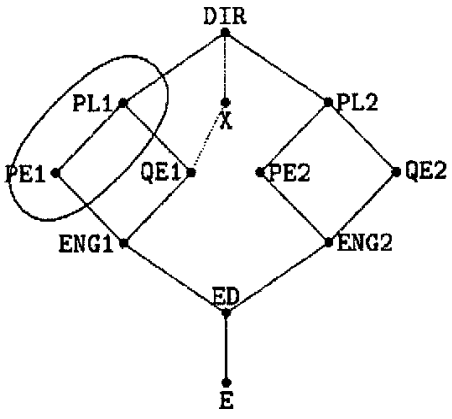
因为$\downarrow PL1 = \{PL1, PE1, QE1, ENG1, ED, E\}, \uparrow PL1 = \{DIR\}$，那么对于 ENG1 来说，$ENG1 \in \downarrow PL1, \uparrow ENG1 = \{ENG1, PE1, QE1, PL1, DIR\} \subseteq_{\beta} PL1$，则$ENG1 \in \sigma(PL1)$。而$ED \notin \sigma(PL1)$，虽然$ED \in \downarrow PL1$但是$\uparrow ED \not\subseteq_{\beta} PL1$，因为对于$\uparrow ED$中的元素 ENG2，有$ENG2 \notin_{\beta} PL1$。于是我们可以推导出$\sigma(PL1) = \{ENG1, PE1, QE1, PL1\}$。

任何访问控制系统都需要动态性，并且能够可能控制系统的变化。文献[21]中的例子足以说明管理范围的动态特性。如图 5-2 所示。

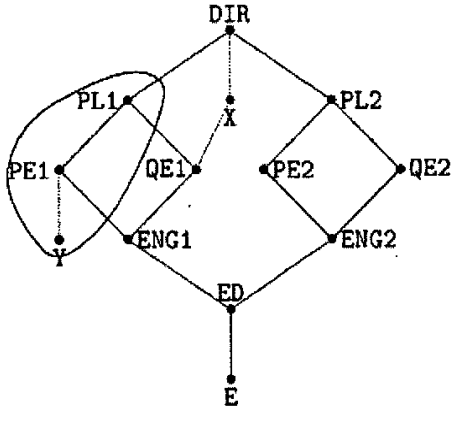
而本文也将利用定义 5-1 的管理范围定义引入到转授权与撤销模型当中。由于第四章已经讨论了常规角色的不同的转授权与撤销的处理机制，那么对管理角色而言，这些机制同样适用，只是操作的角色层次结构从常规角色 RR（Regular Role）变为管理角色 AR（Administrative Role），并且在为角色分配权限的过程里，分配的权限是新增用户、删除用户、新增 UA、删除 UA、新增 PA，删除 PA 等等管理权限，而不是普通的访问权限。常规角色的层次结构以及管理角色的层次结构示例图，在图 4-1 中的（a）和（b）已经清楚地给出了示例。



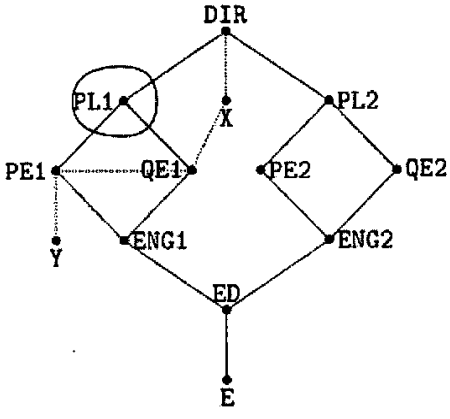
(a) 初始的角色结构图



(b) AddRole(DIR, X, {QE1},{DIR})



(c) AddRole(PSO1, Y,∅,{PE1})



(d) AddEdge(DSO, PE1, QE1)

图 5-2　管理范围的动态特性（图中标出的区域是 PL1 的管理范围）

5.2 对基本定义及基本函数的扩展

5.2.1 基本定义的扩展

为了要支持管理角色的转授权与撤销，也就是利用图 4-1（b）中的管理角色的层次结构图，在用户之间进行转授权与撤销。

首先当然是要对原有的 TRDRM 模型中的基本定义进行扩展和补充。暂且把本文第四章部分描述的模型称作$TRDRM_0$模型，显然$TRDRM_0$只支持常规角色。

而表 5-1 将本文的$TRDRM_0$中的定义 4-9 至定义 4-14 的所有定义进行了扩展

和补充说明。本文把扩展之后的管理模型称作$TRDRM_1$模型。

表 5-1 $TRDRM_1$的扩展定义

$TRDRM_0$原始定义	扩展	$TRDRM_1$扩展后的定义
用户 U	不变	用户 U 的定义不变
角色 R	原有的 R 记作 RR，表示常规角色，AR 表示管理角色	$RR \cap AR = \phi, R = RR \cup AR$
权限 P	原有的 P 记作 RP，表示常规权限，AP 表示管理权限	$RP \cap AP = \phi, P = RP \cup AP$
会话 S	不变	会话 S 的定义不变
用户角色关系 UAT	原有的 UAT 记作 UART，表示用户与常规角色的指派关系。UAAT 表示指派的是管理角色	$UAT = UART \cup UAAT$, $UART \cap UAAT = \phi$, $UAAT = UAAOT \cup UAADT$, $UAAT \subseteq U \times AR \times TIS$
角色权限关系 PA	原有的 PA 记作 RPA，表示角色指派的是常规权限。APA 表示角色指派的是管理权限。	$PA = RPA \cup APA$, $RPA \subseteq RP \times RR$, $APA \subseteq AP \times AR$, $RPA \cap APA = \phi$
角色层次结构 RH	原有的 RH 记作 RRH，表示常规角色的层次结构。ARH 表示管理角色的层次结构。	$RH = RRH \cup ARH$, $RRH \subseteq RR \times RR$, $ARH \subseteq AR \times AR$, $RRH \cap ARH = \phi$
转授权关系 DLGTT	原有的 DLGTT 记作 RDLGTT，表示常规角色的转授权关系。ADLGTT 表示管理角色的转授权关系。	$DLGTT = RDLGTT \cup ADLGTT$, $RDLGTT \subseteq UART \times UART$, $ADLGTT \subseteq UAAT \times UAAT$, $RDLGTT \cap ADLGTT = \phi$
转授权路径 DPT	原有的 DPT 记作 RDPT，表示常规角色的转授权路径。ADPT 表示管理角色的转授权路径。	$DPT = RDPT \cup ADPT$, $RDPT \subseteq UART \times UART$, $ADPT \subseteq UAAT \times UAAT$, $RDPT \cap ADPT = \phi$
转授权树 DTT	原有的 DTT 记作 RDTT，表示常规角色的转授权树。ADTT 表示管理角色的转授权树。	$DTT = RDTT \cup ADTT$, $RDTT \subseteq UART \times UART$, $ADTT \subseteq UAAT \times UAAT$, $RDTT \cap ADTT = \phi$

5.2.2 基本函数的扩展

支持常规角色的多步转授权函数，同定义 4-15。本小节定义支持管理角色的多步转授权函数。

1．求前驱函数：

$$PriorAT: UAAT \rightarrow UAAT$$

$$PriorAT(uaat) = \{uaat' \mid uaat \in UAADT, (uaat', uaat) \in ADLGTT\}, uaat = (u, ar, S_t)$$
$$PriorAT(uaat) = \{\phi \mid uaat \in UAAOT\}$$

2．求转授权路径函数：

$$PathAT(uaat_0) =$$
$$\{uaat_n \rightarrow ... \rightarrow uaat_i \rightarrow uaat_{i-1} \rightarrow ... \rightarrow uaat_0 \mid uaat_i = PriorAT(uaat_{i-1})\}, uaat_0 = (u, ar, S_t)$$

3. 求转授路径中的转授权深度函数：

$$DepthAT: UAAT \rightarrow N$$

同理，每一颗管理角色的转授权树所具有的特性，与结论 4-2 相似。在此不再详细描述。

5.3 支持管理角色的转授权与撤销的处理机制

5.3.1 支持管理角色的转授权与撤销必须解决的问题

支持管理角色的转授权与撤销，总的来看由三类相互联系的操作构成。

（1）常规角色的转授权与撤销，转授权与撤销的角色范围依赖于图 4-1（a）中所示的常规角色的层次结构图；

（2）管理角色的转授权与撤销，转授权与撤销的角色范围依赖于图 4-1（b）中所示的管理角色的层次结构图；

（3）如何将常规角色的转授权树与管理角色的转授权树建立联系，从而使具备管理角色的用户可以实时的监控常规角色的转授权与撤销。

本文在第四章已经详细讨论了常规角色的转授权与撤销，而 5.2 节则扩展原有的定义以支持管理角色的转授权与撤销。因为管理角色的转授权与撤销同常规角色的转授权与撤销机制可以类比，在此不再详细探讨。

所以，本文将重点讨论如何在常规角色的转授权树与管理角色的转授权树建

立联系的问题，也就是第三类问题。

5.3.2 常规角色的转授权树与管理角色的转授权树建立联系的问题

为了要解决这个问题，首先当然是需要确定管理角色的管理范围；其次是确定能够被授予管理角色的用户所必须满足的前提条件。

5.3.2.1 确定管理角色的管理范围

利用定义 5-1，可以借助常规角色层次结构来动态地确定角色的管理范围。那么就只需要根据需要，为管理角色进行管理授权，分配对应的管理范围即可。

定义 5-2 管理域（administrative domain）[25]

如果$D \subseteq R$，且存在$r \in R$，使得$D = \sigma(r)$，那么称D_R是R中的管理域的集合。通常会根据上下文省去下标R。

那么可以得出，图 4-1（a）中的角色层次结构图中对应角色的管理域。如图 5-3 所示。

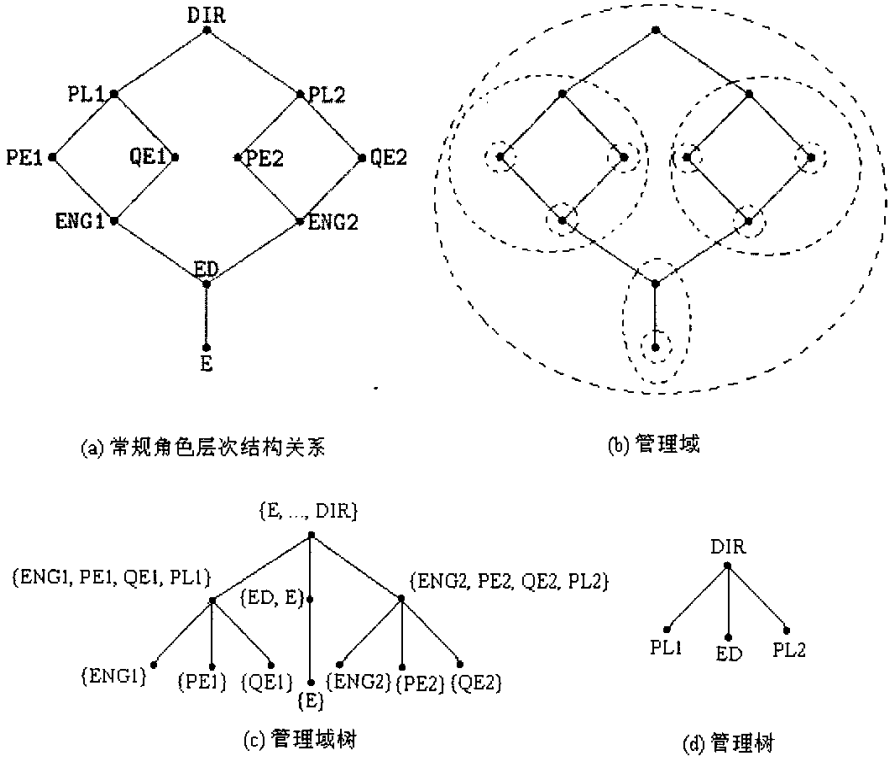


图 5-3 管理域的概念

实际上，根据定义 5-2，可以得出结论，即两个管理域之间的关系，只存在两种，嵌套和互斥。由此得到定理 5-1。

定理 5-1 两个管理域之间的关系

如果 $a,b\in R$，那么 $\sigma(a)\cap\sigma(b)=\begin{cases}\sigma(a) & if \quad a\in\sigma(b)\\ \sigma(b) & if \quad b\in\sigma(a)\\ \phi & if \quad otherwise\end{cases}$

定理 5-1 的证明参阅文献[25]。此处不再证明。

文献[24]中定义了管理授权的概念。此处用本文定义的基本集合进行描述。

定义 5-3 管理授权

$can\_administer\subseteq AR\times RR$，其中 AR 表示管理角色的集合，RR 表示常规角色的集合。如果 $(ar,rr)\in can\_administer$，则表示管理角色 ar 的管理域等同于 $\sigma(rr)$。

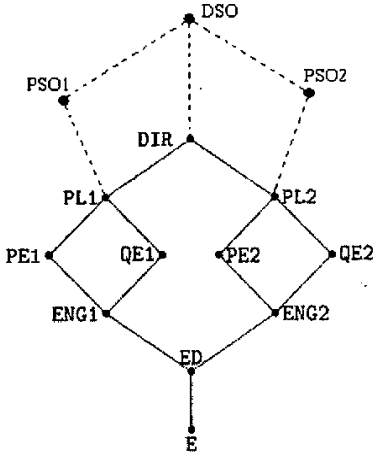
例 5-3　$(PSO1,PL1)\in can\_administer$，表示 PSO1 被授予的管理域为 $\sigma(PL1)=\{ENG1,PE1,QE1,PL1\}$。

那么按照此方式，就可以为每一个管理角色分配管理域了，并且管理域是根据角色结构的动态变化而动态更新的。图 5-2 中已经做出了解释。

此处给出一个为管理角色分配管理域的图例，如图 5-4 所示。

can_administer	
PSO1	PL1
PSO2	PL2
DSO	DIR

(a) can_administer管理授权关系



(b) 管理授权之后对应扩展的角色结构图

图 5-4 管理授权关系以及对应的扩展的角色结构图

5.3.2.2 确定能够被授予管理角色的用户

显然，要给具备常规角色的用户授予管理角色，为了防止管理权限的泄漏和滥用，必须满足一定的约束条件。出于对安全性的考虑，本文认为，被授予的管理角色的管理域必须包含在被授用户的所有角色中的任何一个角色的管理范围内。因此，根据本文 5.1.2.1 小节的基本定义，本文给出一个有效的管理转授权的定义。

定义 5-4 有效的管理转授权

$valid\_ad\min\_delegation \subseteq UAAT \times UART$

如果 $vad\lg t = \{(u,ar,S_t),(u',rr,S_t')\} \in valid\_ad\min\_delegation$，那么则有对于每一个 $p_t \in_s S_t'$，存在 $(ar,r') \in can\_ad\,minister$，满足 $\sigma(r') \subseteq \sigma(rr)$。

例 5-4　以图 5-4 中为每个管理角色分配的管理域为例。

图中管理角色 $PSO1$ 对应管理域为 $D = \sigma(PL1) = \{ENG1, PE1, QE1, PL1\}$，那么具有角色 DIR 或者角色 $PL1$ 的用户都满足定义 5-4，可以进行管理转授权，因为 $D = \sigma(PL1) \subseteq \sigma(DIR)$。再如，如果是具有角色 $PE1$ 的用户则不满足定义 5-4，不能被授予 $PSO1$ 的管理角色，因为 $D = \sigma(PL1) \not\subset \sigma(PE1) = \{PE1\}$。

5.3.2.3 有效的管理转授权的判定

定义 5-4 实际上只是定义了有效的管理转授权必须满足的前提条件，至于管理转授权的操作是否真正的有效，还需要定义规则来进行判定。

如果对本文 4.4.2.1 小节中的描述函数进行扩展补充，并且仍然使用基于规则的策略描述语言，定义基本的管理授权判定规则，则可以描述有效的管理转授权的判定规则。那么，就按照这三个步骤来扩展。

1．补充描述函数——返回角色的管理域的函数 $domain:(r:RR) \rightarrow 2^{RR}$

$domain(r) = \{s \in RR \mid s \in \sigma(r)\}$

2. 补充定义基本的管理授权判定规则 $can\_ad\,minister$

规则 5-1 基本的管理授权判定规则

$$can\_ad\,minister(ar,rr) \leftarrow .$$

该规则表达的意思是，具备管理角色 ar 的用户，所管辖的范围为具有 $\sigma(rr)$ 中的角色的所有用户。

3. 补充定义有效的管理转授权的判定规则

规则 5-2 有效的管理转授权判定规则

$$\begin{aligned}&valid\_ad\,\min\_delegate(u,ar,u',rr,S_{t\_AD}) \leftarrow \\ &active(u,ar,s)\,\& \\ &can\_ad\,minister(ar,rr')\,\& \\ &in(domain(rr'),domain(rr))\,\& \\ &in(S_{t\_AD},valid\_TIS(u',rr))\,\& \\ &in(S_{t\_AD},valid\_TIS(u,ar)).\end{aligned}$$

该规则表达的意思是，在会话 s 中激活管理角色 ar 的用户 u，能够将管理角色 ar 转授给具有常规角色 rr 的用户 u'，必须满足的条件是管理角色被分配的管理范围必须包含在角色 rr 的管理域内，同时有效时间的约束也必须满足。其中 S_{t_AD} 表示被授予的管理角色的有效时间区间的集合，S_{t_AD} 必须在转授管理角色一方的有效时间范围内，同时也必须在被授管理角色一方的有效时间范围内。

为了更好地说明规则 5-2，用图表的方式来举例说明该规则。首先，如表 5-2 所示，是管理角色的转授权关系的对应表。

表 5-2 管理角色的转授权关系示例

转授用户	转授角色	有效时间	被授用户	被授角色	有效时间
Jeff	DSO	{[1, 100],...}	Jacky	PSO1	{[1, 80],...}
Jeff	DSO	{[1, 100],...}	Rose	PSO2	{[1, 90],...}

按照规则 5-2 中有效的管理授权的判定定义，仍然以图 4-2 中的常规角色的转授权树为例，若实施管理授权的操作，则示例图如图 5-5 所示。

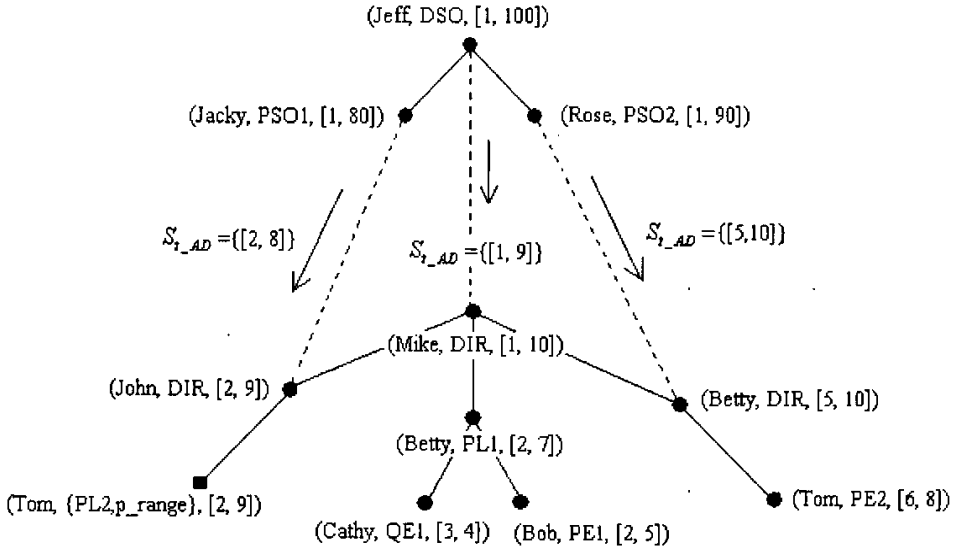


图 5-5 对常规角色进行管理角色授权的示例图

注意，在图 5-5 中，实际存在三个有效的管理授权关系满足规则 5-2。

第一个关系是，$(Jeff, DSO, [1,100])$将管理角色 DSO 授予给了常规角色转授权树中的节点$(Mike, DIR, [1,10])$，有效时间集合是$S_{t_AD} = \{[1,9]\}$；

第二个关系是，$(Jacky, PSO1, [1,80])$将管理角色$PSO1$授予给了常规角色转授权树中的节点$(John, DIR, [2,9])$，有效时间集合是$S_{t_AD} = \{[2,8]\}$；

第三个关系是，$(Rose, PSO2, [1,90])$将管理角色$PSO2$授予给了常规角色转授权树中的节点$(Betty, DIR, [5,10])$，有效时间集合是$S_{t_AD} = \{[5,10]\}$。

第 6 章 TRDRM 模型的原型实现

6.1 项目背景

本文的研究工作始于本人参与开发的项目——科研机构协同工作平台项目。

其目的在于研究与开发基于中小型科研机构管理的解决方案，实现科研机构里的各项工作，比如说，教学、实验室管理、论文管理等活动的电子信息化。

主要成果将是极大的提高中小型科研机构的管理水平，提高效率。解决某些科研机构地点分散，人员众多，教务繁忙，各种工作不能协调开展的问题。

目前，此项目的访问控制模块的设计与开发是以 RBAC96 模型为指导的，是一种静态地访问控制方式。不支持转授权与撤销，也没有引入时间的因素进行有效的动态控制，势必不能满足协同工作的需求。

因此本文所实现的原型系统，是将 TRDRM 模型应用到科研机构协同工作平台项目里的一个探索，下一步工作则是把 TRDRM 原型系统整合到此项目中。

6.1.1 系统功能简介

在科研机构协同工作平台项目里，主要分成三大功能平台，即信息管理平台、通讯沟通平台、办公管理平台。三大模块互为补充，灵活交叉，并充分考虑到了用户的高度自定义功能和企业用户的通用需求实现之间的平衡。系统采用 J2EE 技术体系构架，灵活、安全、高效、可扩展，为机构的日常办公管理提供了一个高效、便捷而且规范的系统。

该项目目前已经基本完成办公管理平台中的协同办公模块的开发工作。具体分成了系统维护、个人秘书、中心概况、教学管理、实验室管理、科学研究、人才培养、信息中心等子模块。正在进一步开发之中。

6.1.2 用户使用过程

用户使用本系统的流程如图 6-1 所示。而本文原型实现部分的主要工作，就是在用户选择进入转授权功能模块、撤销授权功能模块之后，针对该用户发出的转授权与撤销操作的请求的处理机制的实现。

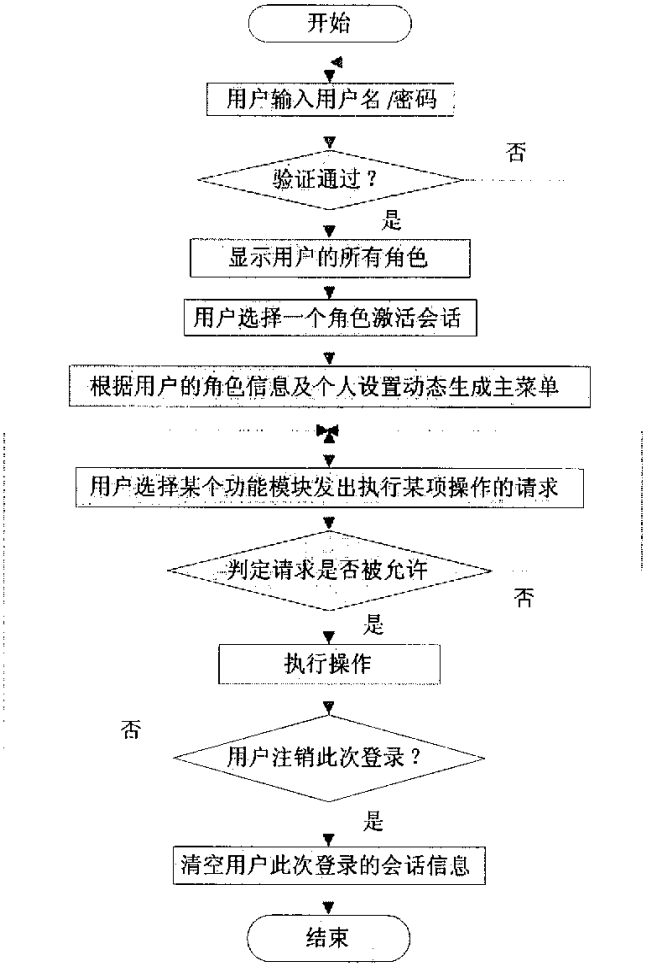


图 6-1 用户使用过程示意图

6.2 TRDRM 原型系统的体系结构设计

6.2.1 实现架构图

由于本文只是对 TRDRM 模型应用到科研机构协同工作平台的一个探索，所以在设计实现原型系统时，考虑采用最精简的架构去实现相同的功能，如图 6-2 所示。

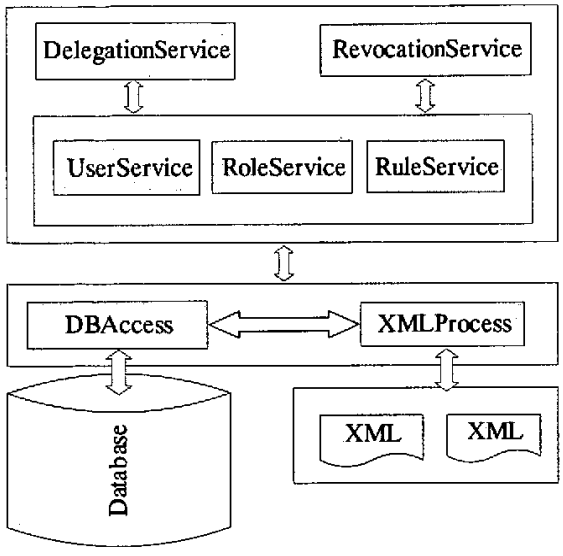


图 6-2 TRDRM 原型系统架构设计

TRDRM 原型系统的架构遵循的是三层模型的体系结构。表示层（Presentation）负责提供用户界面，业务逻辑层（Business Logic）负责实现业务逻辑，数据持久层（Data Persistence）负责业务逻辑层中所有数据的持久的存储。

1．数据存储

数据存储采用数据库的表和 XML 文件两种存储方式。用户的个人信息、角色权限信息、转授权与撤销判定的规则、职责分离规则等以数据库表的形式存储在数据库里。而角色的层次结构信息、转授权树的信息就以 XML 文件的形式存储。在本文 6.3 节部分将详细说明。

2．数据持久层

数据持久层主要包含两个部分。一个是 DBAccess，是对数据库的表的访问和操作，另一个是 XMLProcess，是对以 XML 文件存储的数据进行读取、修改、查询等操作。

3．业务逻辑层

业务逻辑层的基本模块包括 UserService、RoleService、RuleService 等等，主

要是为 Delegation Service 和 Revocation Service 提供基本的服务。

UserService（用户服务）涉及到用户信息的查看、新增、删除、修改，为用户分配角色以及撤销用户角色指派关系等业务逻辑的处理。RoleService（角色服务）则是对角色信息的查看、新增、删除、修改的操作，为角色分配权限以及撤销角色权限指派关系等业务逻辑的处理。RuleService（规则服务）是对转授权判定规则、撤销授权判定规则、职责分离规则、其他的约束规则等操作的业务逻辑的处理。其他的一些基本服务此处不再赘述。

而本文的核心部分 DelegationService（转授权服务）和 RevocationService（撤销授权服务）则需要调用上述的基本服务来完成业务逻辑的处理过程。

4．表示层

图 6-2 中并没有标识出表示层，表示层负责提供用户界面。一般使用 Web 页面的形式提供给用户。此处不再详细介绍，本文 6.5 节有运行界面的演示。

6.2.2 主要流程图

各种转授权与撤销情况的处理机制，已经在本文 4.3 节部分举例说明。本节则对具体的处理流程以图的形式表达出来，为 TRDRM 原型系统的实现做好前期工作。图 6-3 是转授权处理机制的总图，图 6-4 是撤销处理机制的总图，流程图里的分支情况的处理，在第四章部分已经详细讨论，此处不再说明。

6.3 数据存储的详细设计

在 6.2.1 节中说明了 TRDRM 原型系统的数据存储采用数据库的表和 XML 文件两种存储方式。用户的个人信息、角色权限信息、转授权与撤销判定的规则、职责分离规则等以数据库表的形式存储在数据库里。而角色的层次结构信息、转授权树的信息就以 XML 文件的形式存储。

原因是角色之间的层次结构信息本身就是结构性的数据，每个数据元素之间都存在着联系，为了方便地表示角色之间的关系，同时出于对角色结构进行访问和操作需要对其进行遍历操作的考虑，故将其用 XML 文件的形式存储。

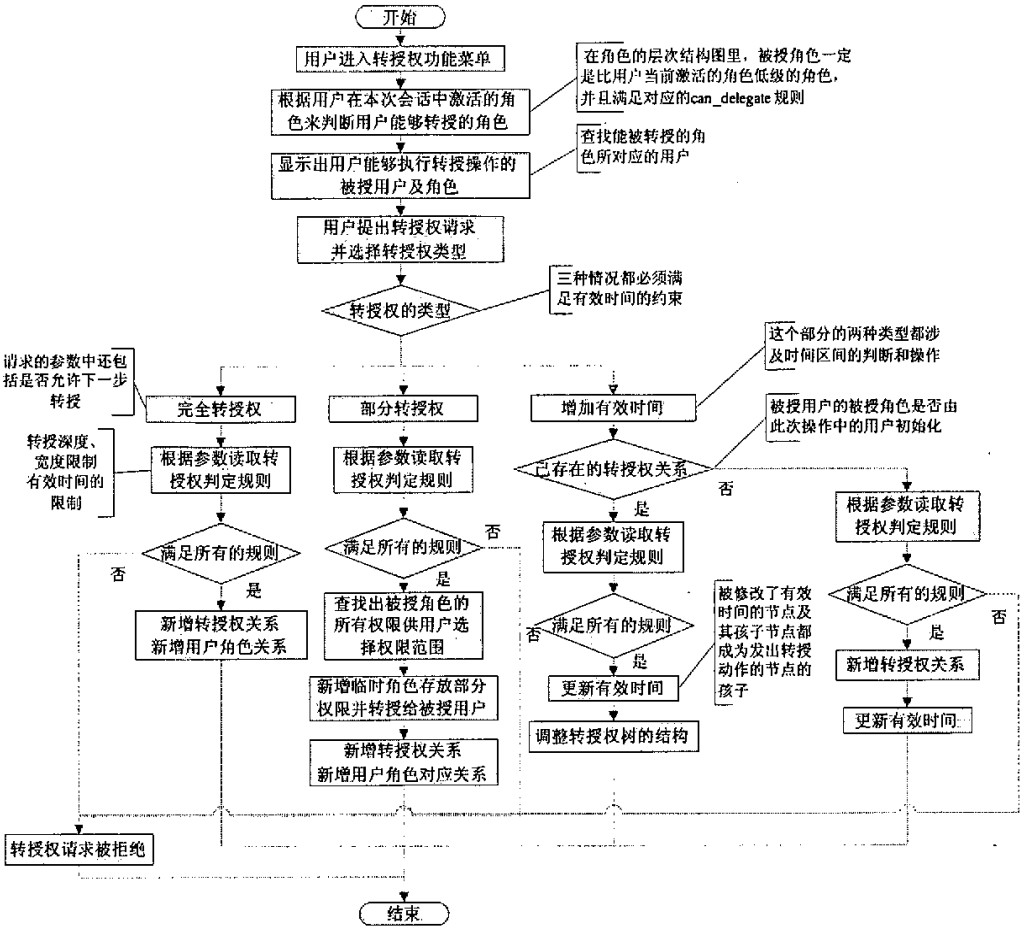


图 6-3 转授权处理机制的总图

同时，由于用户的转授权和撤销的操作都会对转授权树造成影响，须要调整树的结构。而如果将转授权树的信息存储在数据库的表里，存在着诸多不便，所以转授权树的相关信息，本文也考虑以 XML 文件的形式来存储。

用 XML 文件存储信息必然要考虑到数据的保密性的问题。由于本文只是将角色的层次结构和转授权树的关系以 XML 文件存储，其他的相关的机密信息均存储在数据库的表里，所以系统的安全性可以得到保证。

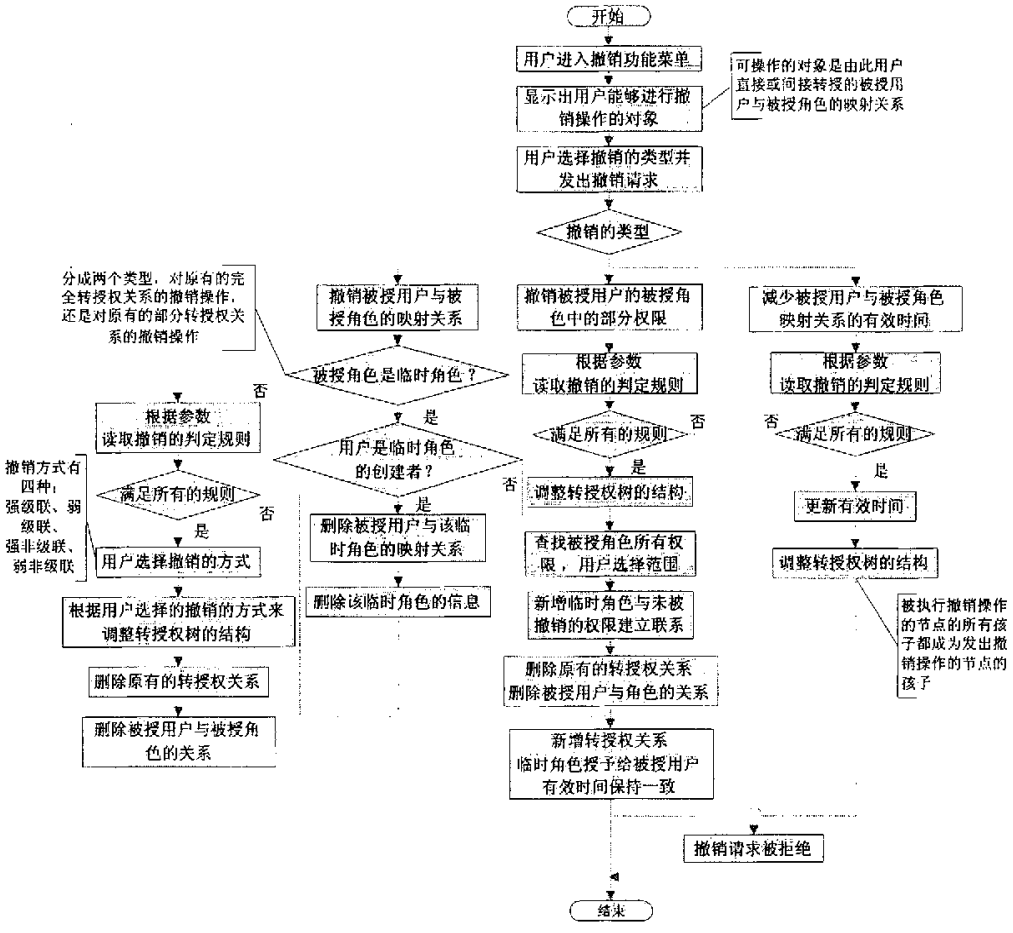


图 6-4 撤销处理机制的总图

6.3.1 数据库的表的设计

1．用户表

USER(USERID, USERNAME, PASSWORD, DEPT, SALARY)，记录用户的基本信息，可根据需求添加需要记录的字段。

2．角色表

ROLE(ROLEID, ROLETYPE, ROLENAME)，记录三种角色类型的基本信息，包括常规角色、管理角色、以及因部分转授权动作产生的临时角色，ROLEID 唯一标识角色，同时可用 ROLETYPE 字段来区分不同的类型。

3. 临时角色表

TEMPROLE(TEMPROLEID, PARENTROLE, CREATEUSER)，记录因某个用户的部分转授权而产生的临时角色。临时角色 TEMPROLEID 所对应的部分权限源于哪一个角色记录在 PARENTROLE 字段，CREATEUSER 字段记录此临时角色的创建者用户的 ID。而临时角色的详细信息仍然记录在角色表里。

4. 权限表

PERMISSION(PERMISSIONID, PERMISSIONNAME)，记录所有的权限信息，包括常规角色的权限，以及管理角色的权限。

5. 角色权限对应关系表

RPRELATION(ROLEID, PERMISSIONID, DESCRIBE)，记录所有的角色被赋予的权限，DESCRIBE 字段对相关信息进行描述，此表记录的只是角色和权限的 ID 信息，详细信息需要去查找对应的表。

6. 用户角色对应关系表

URRELATION(USERID, ROLEID, VTSTART, VTEND)，记录每一个用户拥有的每一个角色的有效时间。

7. 转授权判定规则表

DLGTRULES(DLGTROLEID, PREROLE, DEPTHRSTR, WIDTHRSTR)，记录转授权判定规则的相关信息。

每条记录表示的含义是拥有角色 ID 为 DLGTROLEID 的用户能够将其转授给角色满足条件 PREROLE 的另一个用户，DEPTHRSTR 字段记录转授权深度的限制，WIDTHRSTR 记录转授宽度的限制。

8. 撤销判定规则表

RVCRULES(ROLEID, OPTION, DESCRIBE)，记录撤销判定规则的相关信息。字段 OPTION 记录该角色是否能被授权依赖撤销或者是非授权依赖撤销。DESCRIBE 字段记录描述信息。

9. 职责分离规则

SODRULES(CONFLICTING, CONFLICTED, TYPE, DESCRIBE)，记录职责分离规则。CONFLICTING 和 CONFLICTED 两个字段定义的是互相冲突的角色或者权限，由字段 TYPE 标识出。

6.3.2 XML 文件存储数据的设计

1. 角色层次结构的 XML 表

图 6-5 是角色层次结构图的 XML 文件存储示例。

```
<?xml version="1.0" encoding="GB2312"?>
<RolesH>
  <role roleID="0000000014">
    <role roleID="0000000010">
        <role roleID="0000000008">
            <role roleID="0000000004">
                <role roleID="0000000002">
                    <role roleID="0000000001"/>
                </role>
            </role>
        </role>
        <role roleID="0000000009">
  ... ...
  ... ...

        </role>
    </role>
 ... ...
 ... ...
    <role roleID="0000000018">
      <role roleID="0000000016"/>
      <role roleID="0000000017"/>
    </role>
</RolesH>
```

图 6-5 角色层次结构的 XML 存储示例

2. 转授权树的 XML 表

转授权或者撤销的操作，将会对存储转授权树的 XML 文件产生影响。图 6-6 中给出了只包含一棵转授权树的简单示例。在 DLGT 这个 XML 文件中，存在着多棵转授权树，并且每一棵树都会由于用户相应的操作产生出多个子树。

```
<?xml version="1.0" encoding="GB2312" ?>
- <DLGT>
  - <DLGTTree roleID="0000000014" userID="0000001000" vtEnd="2007-12-31" vtStart="2002-01-01">
      <child roleID="0000000010" userID="0000001009" vtEnd="2006-06-08" vtStart="2004-10-24" />
      <child roleID="1000000003" userID="0000001002" vtEnd="2004-06-22" vtStart="2003-11-13" />
      <child roleID="0000000011" userID="0000001003" vtEnd="2006-04-28" vtStart="2005-08-09" />
    - <child roleID="0000000013" userID="0000001011" vtEnd="2006-11-11" vtStart="2006-09-01">
        <child roleID="1000000002" userID="0000001005" vtEnd="2006-10-11" vtStart="2006-10-01" />
      </child>
    </DLGTTree>
  </DLGT>
```

图 6-6 转授权树的 XML 存储示例

6.4 实现细节

6.4.1 包图

根据图 6-2 中的 TRDRM 原型的架构设计，具体的项目实现主要分成了四个包，数据包（data）、数据持久层的数据持久包（dp） 、业务逻辑层的服务包（services）、以及通用的实用包（utility）。包之间的关系如图 6-7 所示。

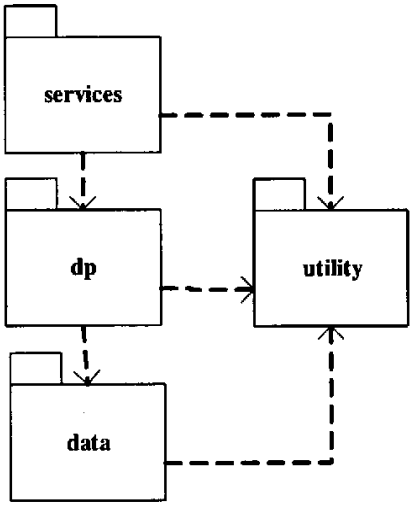


图 6-7　TRDRM 的包图

数据包里包含了三个子包。model 子包内的类与数据库的表一一对应，是对数据库表的字段的描述。sysxml 包是存放系统 xml 文件的路径，主要是指角色层次结构 xml 文件、转授权树 xml 文件。userxml 包是存放用户相关 xml 文件的路径，主要是指用户在登录系统并激活会话之后，会将用户的个人信息以及此次会话相关的权限信息一次提取出来，存放在 xml 文件里。

数据持久包包含两个子包。dbaccess 子包的类仍然与数据库的表一一对应，每一个类都是对数据库的访问和操作的封装。xmlprocess 子包的类处理对 xml 文件的读取、修改、查询等操作。

服务包中的类处理业务逻辑，与图 6-2 中的业务逻辑层的服务一一对应之外，还增加了与之相关的一些必备的基础服务。

通用实用包中的类则是对系统需要的一些通用的函数的定义。

6.4.2 类图

1．data.model 子包内的类

如图 6-8 所示。每个类的私有属性一一对应数据库的字段，通过公有的 get 和 set 方法提供给上层的类调用而进行操作。

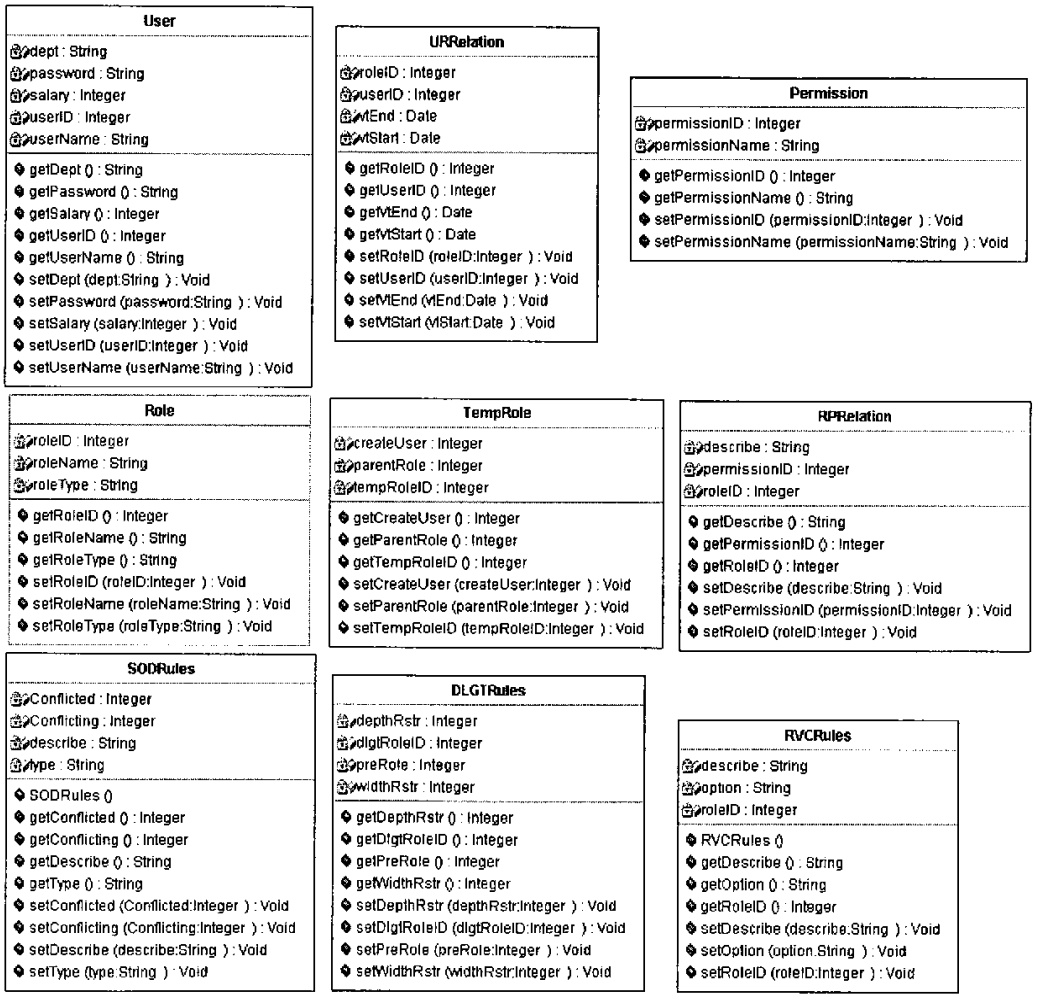


图 6-8　data.model 子包内的类

2．dp.dbaccess 子包内的类

如图 6-9 所示，类 ConnectionManager 负责连接数据库，而此包内的其他的类，以 DB 开头命名，表示通过调用 ConnectionManager 提供的方法，实现对数

据库表的访问和操作，它们都是对数据库表的访问的封装，主要包括新增、删除、查询等操作，不进行业务逻辑的处理。

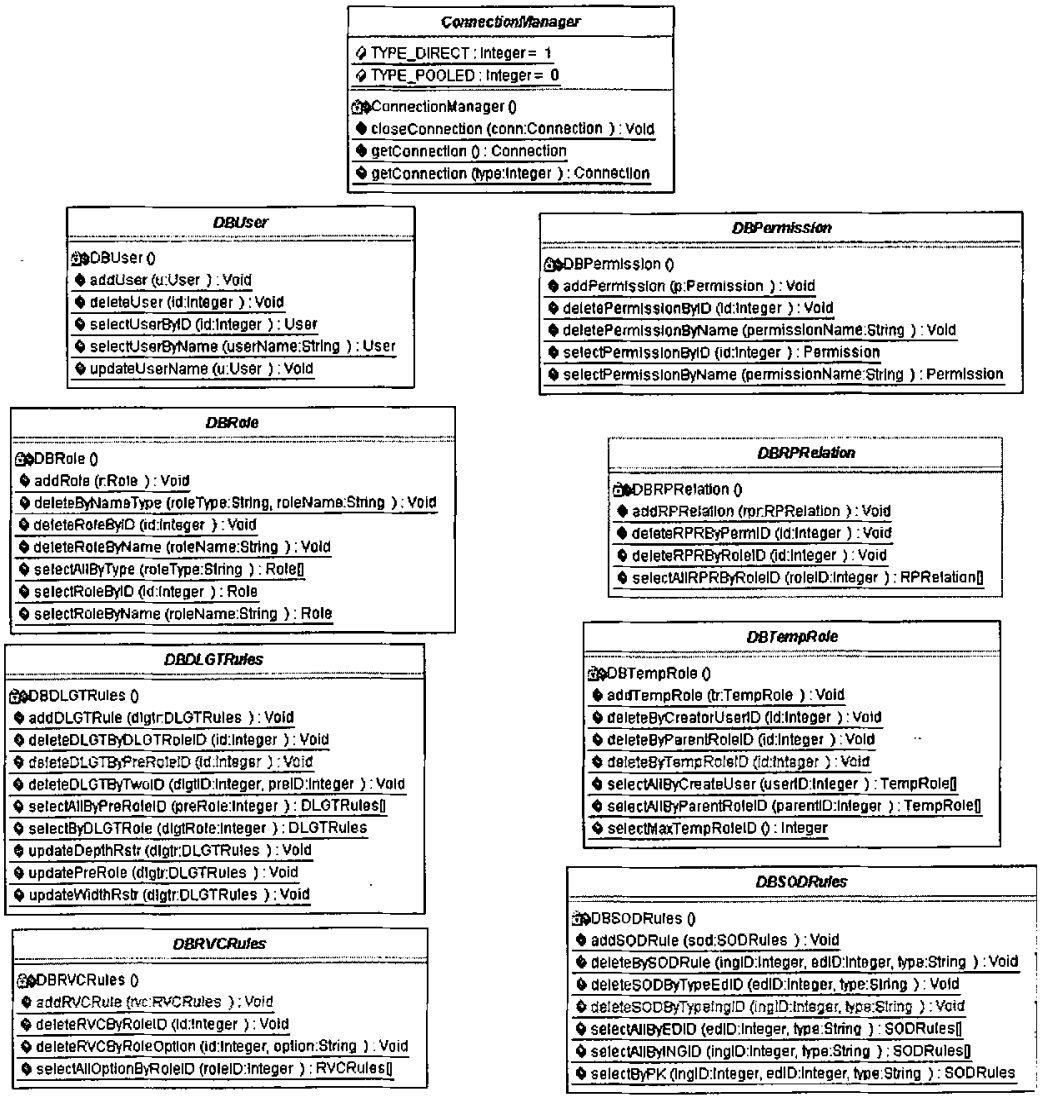


图 6-9 dp.dbaccess 子包内的类

3．dp.xmlprocess 子包内的类

dp.xmlprocess 子包内的类，是对 XML 文件的读取和存储操作。图 6-10 所示的类 XMLDLGT 是对转授权树的 XML 文件 DLGT.xml 的创建、读取、添加删除节点、输出存储等操作。

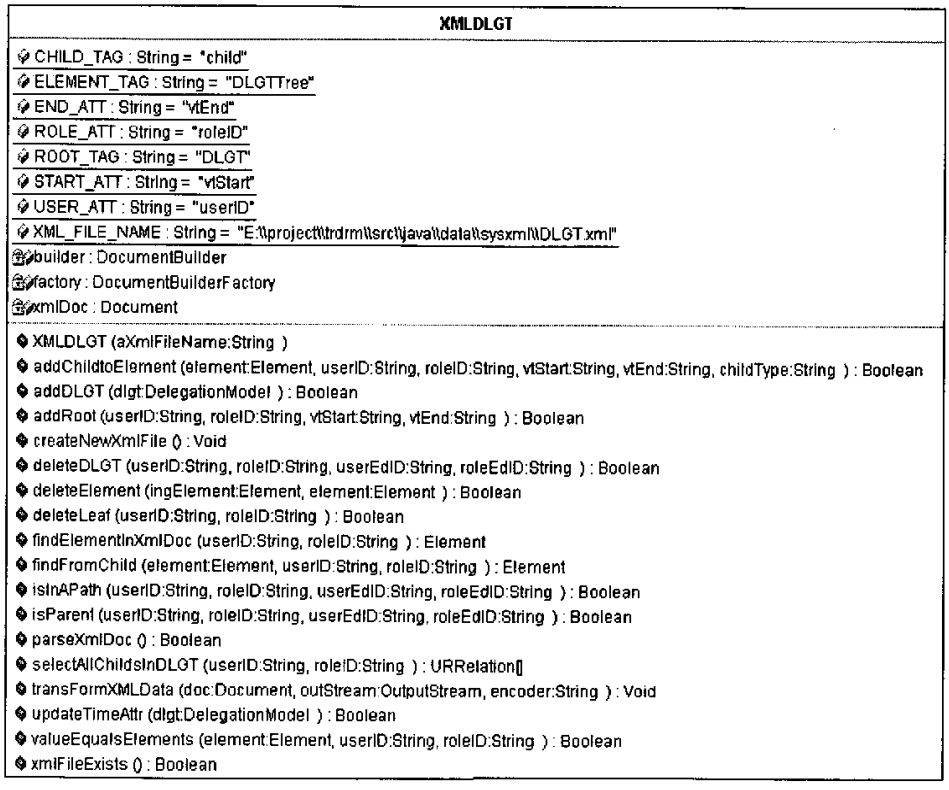


图 6-10 类 XMLDLGT

而图 6-11 与图 6-12 则是对角色层次结构的 XML 文件 roleH.xml 的读取、创建、输出、存储等操作。分为 XMLFile 类以及 XMLFind 类。

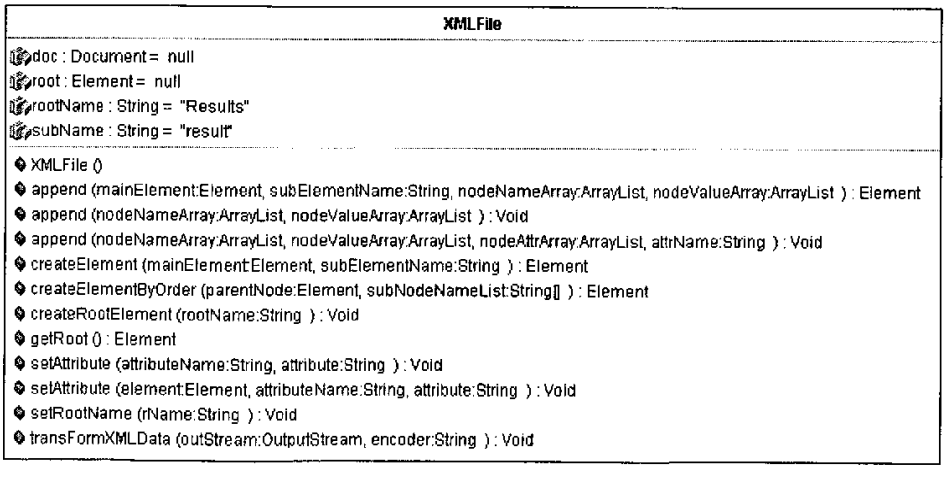


图 6-11　类 XMLFile

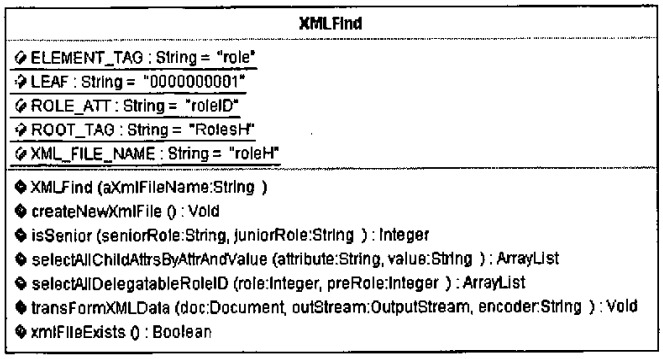


图 6-12　类 XMLFind

4．services 包内的类

如图 6-13 所示，services 包内的类处理每个表对应的业务逻辑。

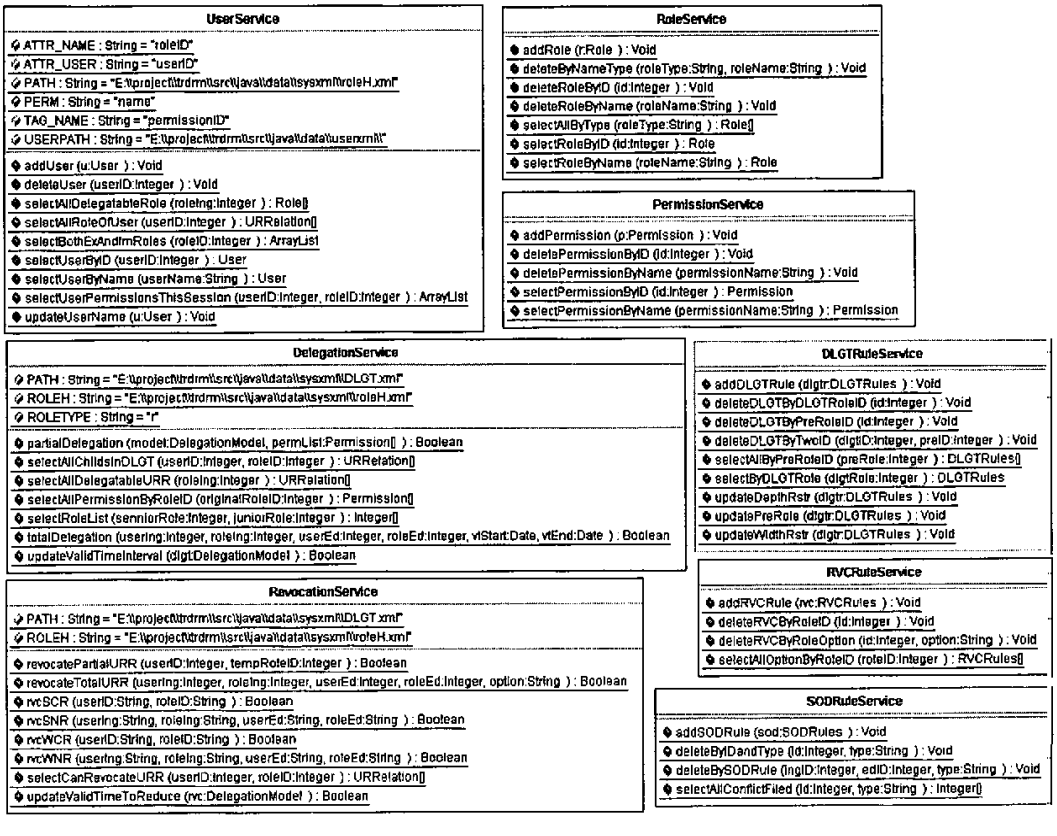


图 6-13　services 包内的类

类 UserService、类 DelegationService、类 RevocationService 是完成转授权与

撤销操作的逻辑处理的主类，而 services 包内的其他的服务类、dp.xmlprocess 包内的类都需要为这三个类提供调用的方法，以实现对数据的访问和操作。

6.5 TRDRM 原型系统的运行界面

TRDRM 原型系统，首先需要用户进行登录验证等过程，然后用户选择多个角色中的一个进行本次会话。同时在服务器端生成以用户名命名的 XML 文件记录该用户此次会话的所有角色权限信息。在用户进入主菜单之后，则可进行相应的功能模块的选择而进行下一步操作。

由于本文的 TRDRM 系统还只是一个原型，所以用户的具体的权限如何在功能模块里进行控制，本文作了简化处理。而在科研机构协同工作平台的项目里，则对权限的具体的控制有详细的实现。

6.5.1 用户登录

如图 6-14 所示，用户首先输入用户名和密码进行登录。验证通过则显示出用户当前的有效的角色列表，而无效的角色是指，用户的某个角色已经过期或者是被撤销掉。

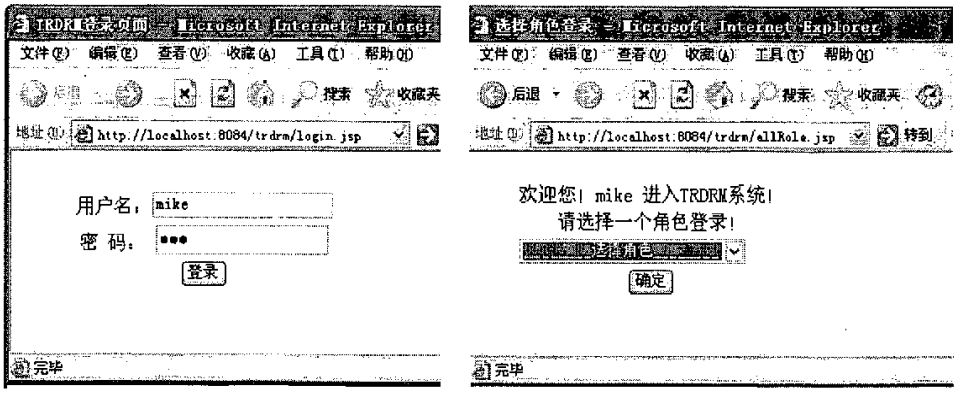


图 6-14 用户选择一个有效角色登录

登录成功，则返回用户此次登录的所有权限列表，并在服务器端生成相应的用户 XML 文件。当然此处本文做了简化处理，在实际运用当中，权限的控制应

更为具体复杂。如图 6-15 所示，用户接着进行下一步操作的选择。

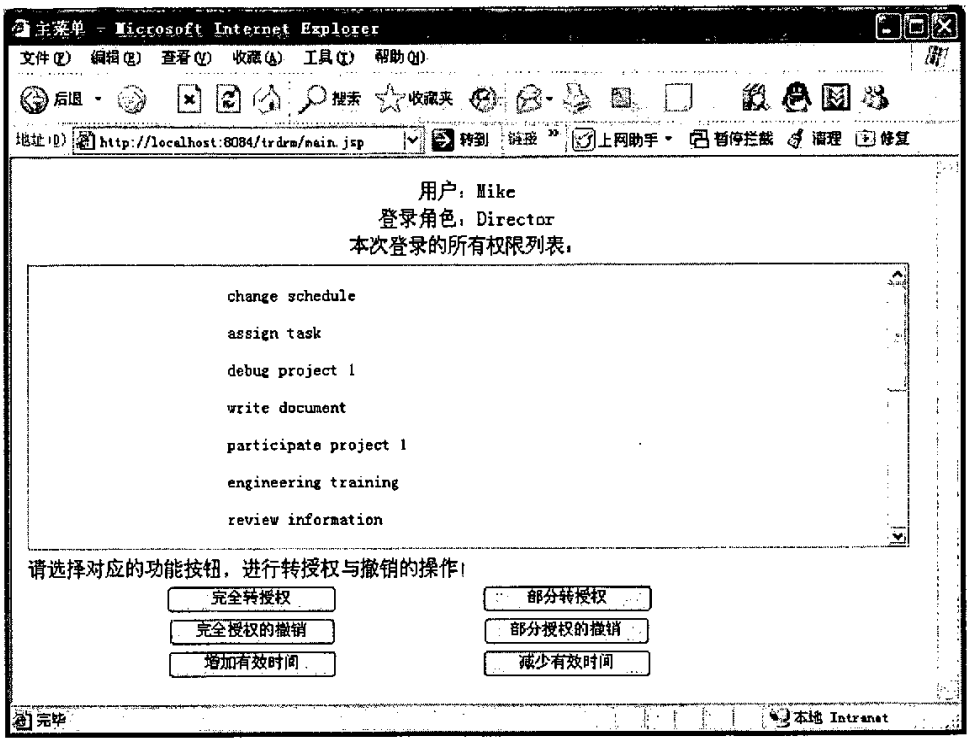


图 6-15 用户登录成功后的主菜单

6.5.2 授权与撤销的各种情况

在选择了图 6-15 中的功能按钮之后，则 TRDRM 将对各种不同的情况进行相应的处理。

1．完全转授权操作

图 6-16 是用户在选择完全转授权操作之后，返回给用户可以进行转授权的对象的列表，图 6-17 则是选择了其中一个对象之后，需要编辑相关的信息，比如选择被授的角色、有效时间的设置。注意，有效时间的设置一定是在转授一方的有效时间区间内，见本文的结论 4-2。如果时间参数设置不对，则此次转授权操作不会成功。

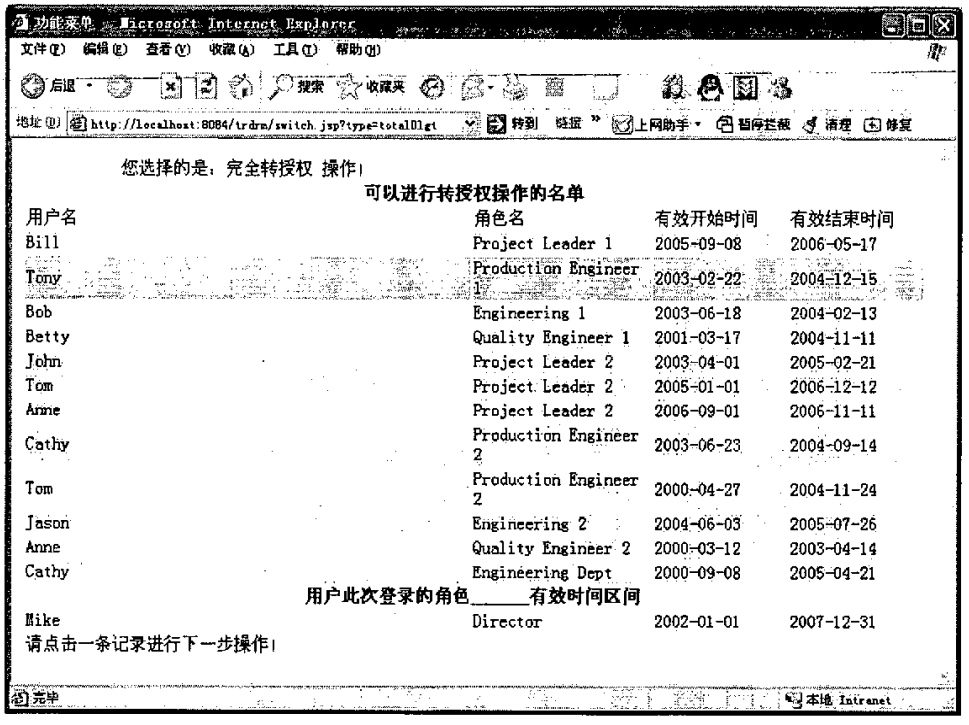


图 6-16　用户选择完全转授权操作的对象

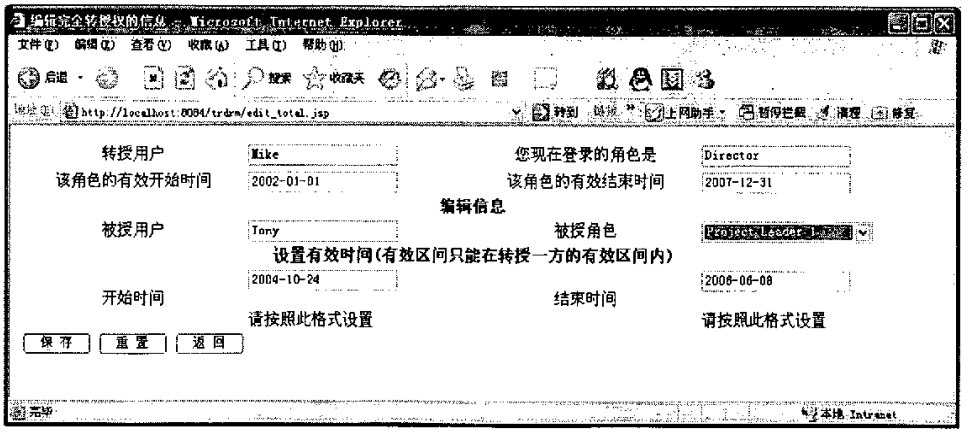


图 6-17 编辑完全转授权操作的信息

2．部分转授权操作

图 6-18 则是用户在主菜单选择了部分转授权的操作。当用户在被授角色的下拉框里选择了转授哪一个角色的一部分权限时，会在下方的多选框显示出该角色的所有权限，用户则可选择其中的一部分权限授予给用户。在设置好有效时间之后，则可以保存此次操作的信息了。

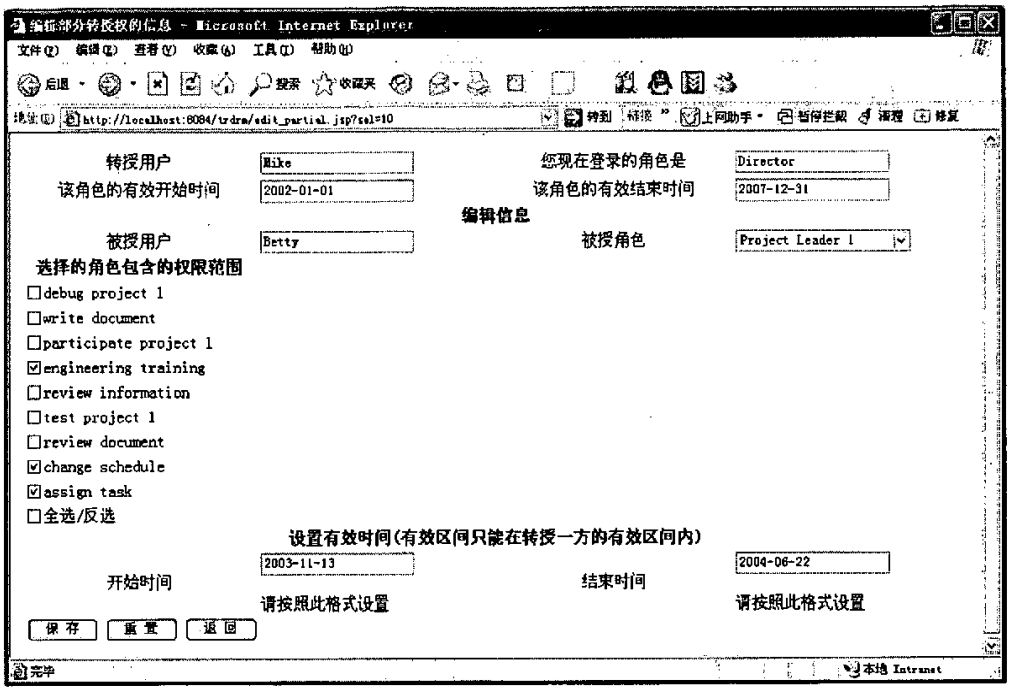


图 6-18 部分转授权操作

3. 更新有效时间

对于已有的直接或间接的转授权关系，也就是说在转授权树中的一个节点与它的子树，这个节点是可以对其子树中的节点的有效时间进行更新的。增加有效时间和减少有效时间的处理，可以归并为同一种情况，本文 4.3 节部分已经详细讨论。图 6-19 和图 6-20 表示了更新有效时间的操作。

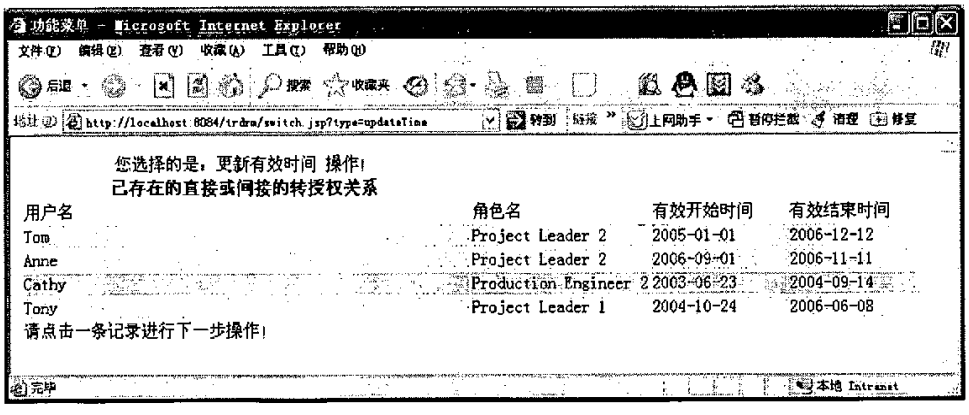


图 6-19　用户选择增加或减少有效时间的操作

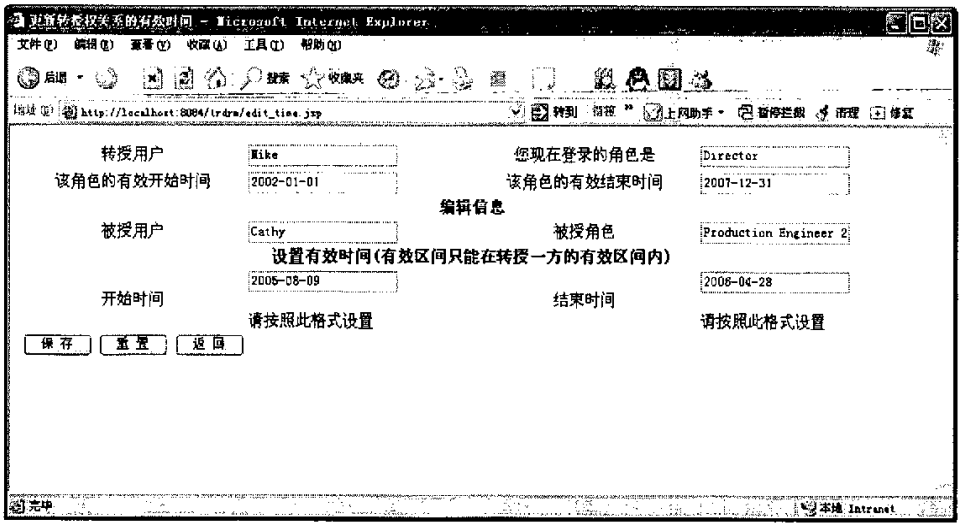


图 6-20 更新被授用户和被授角色对应关系的有效时间

4．对完全转授权关系的撤销

图 6-21 表示了撤销已有的完全转授权关系的操作。用户需要选择一个已有的转授权关系，以及撤销的机制。撤销机制包括强级联撤销（SCR）、强非级联撤销（SNR）、弱级联撤销（WCR）、弱非级联撤销（WNR），详细讨论见本文 4.3 节部分。

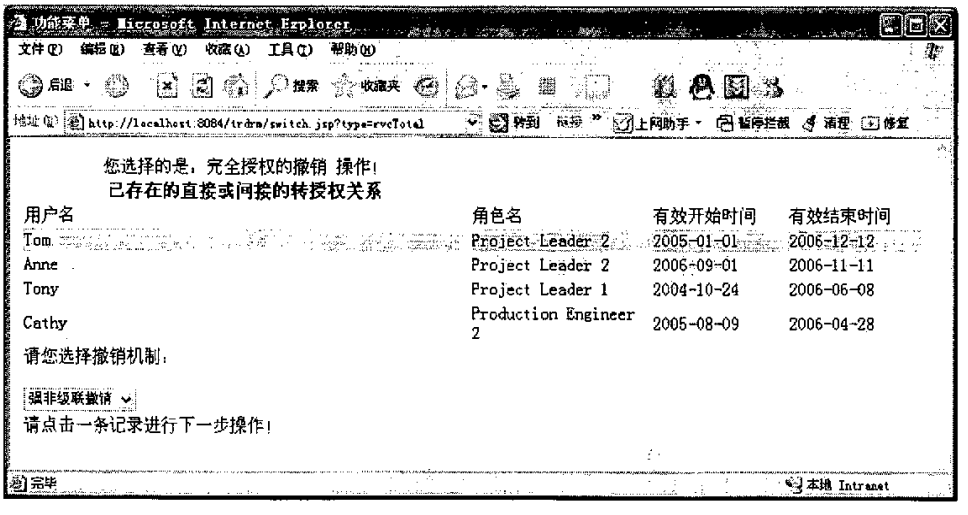


图 6-21 已有的完全转授权关系的撤销

综上所述，本小节部分已经显示了 TRDRM 原型系统的核心功能，其它的转授权与撤销情况，此处就不再一一举例说明。

第7章 总结与展望

当前基于角色的访问控制系统完全依赖于管理者的集中管理方式，不能够满足分布式环境下的系统管理的需求，基于角色的转授权模型更适于分布式环境的授权管理。同时，时间是自然界无所不在的客观属性，所有的信息都具有相应的时间属性，通过在转授权模型中引入时间的因素，能够更好地控制用户之间的转授权与撤销的过程。

目前已有的转授权模型的研究都仅限于基于常规角色的转授权与撤销。RDM系列模型（Role-based Delegation Model）只有转授权的有效期到期将其自动撤销的概念，没有将时间因素引入到模型里，而 TRDM 模型（Temporal Role-based Delegation Model）是带时限的转授权模型，但是不支持角色的部分转授权，也没有详细讨论带时限的用户主动撤销转授权的机制。

因此，本文对 RDM 和 TRDM 中的基于常规角色的转授权与撤销机制进行扩展，扩展后的模型称为基于角色的带时限的转授权与撤销模型（Temporal Role-based Delegation and Revocation Model，TRDRM）。同时将时间因素、ARBAC（Administrative RBAC）的思想、SARBAC（Scoped ARBAC）的管理范围的定义引入到 TRDRM 中，从而在支持常规角色的转授权与撤销的同时，也支持管理角色的转授权与撤销，是集中管理方式和分布式管理方式的有效结合。

本文所做的理论上的研究工作，主要从六个方面来开展。

（1）总结带时限的转授权树的特点，并得出了形式化的结论；

（2）不仅支持带时限的完全转授权，也支持转授角色中的部分权限；

（3）详细讨论了带时限的用户主动撤销授权的处理机制；

（4）转授权和撤销操作若是对有效时间的更改，归纳分析了对转授权树的结构产生影响之后的处理机制；

（5）扩展研究管理角色的转授权与撤销的处理机制，并重点讨论了管理角色的转授权树与常规角色的转授权树之间建立联系的问题，即如何管理和监控常规角色的转授权与撤销操作的问题；

（6）定义重要的约束规则，包括静态职责分离中的角色冲突与权限冲突。

最后，本文针对TRDRM中的核心功能进行了实现，本文所实现的TRDRM

原型系统，是将TRDRM模型应用到科研机构协同工作平台项目里的一个探索。

本文所扩展的 TRDRM 模型仍然存在在不完善的地方，可作为下一步工作继续深入研究。

（1）本文默认不允许对转授权树的根节点进行撤销操作，但在实际应用中是存在这种需求的；

（2）需要根据 TRDRM 模型中的时间特性，完善与时间相关的约束规则的定义；

（3）本文简化了对时间信息的处理，没有详细讨论重叠的时间区间之间的合并和拆分问题；

（4）本文实现的 TRDRM 原型系统只实现了基于常规角色的转授权与撤销的功能，管理角色的转授权与撤销还停留在理论研究基础上，下一步工作则需要完善 TRDRM 原型系统，并整合到科研机构协同工作平台项目中。

参考文献

[1] 赵亮，茅兵，谢立. 访问控制研究综述. 计算机工程，2004，30(2)：1-3

[2] 许峰，赖海光，黄皓，谢立. 面向服务的角色访问控制技术研究. 计算机学报，2005，28(4)：686-693

[3] David Ferraiolo, Richard Kuhn. Role-based Access Control. In Proceedings of the 15th National Computer Security Conference, 1992：554-563

[4] Ravi S. Sandhu, Edward J. Coyne, Hal L. Feinstein, Charles E. Youman. Role-based Access Control Models. IEEE Computer, 1996, 29(2)：38-47

[5] 董光宇，卿斯汉，刘克龙. 带时间特性的角色授权约束. 软件学报，2002，13(8)：1521-1527

[6] 黄建，卿斯汉，温红子. 带时间特性的角色访问控制. 软件学报，2003，14(11)：1944-1954

[7] Elisa Bertino, Piero A Bonatti, Elena Ferrari. TRBAC: A Temporal Role-based Access Control Model. ACM Transaction on Information and System Security, 2001, 4(3)：191-223

[8] James BD Joshi, Elisa Bertino, Arif Ghafoor. Temporal Hierarchy and Inheritance Semantics for GTRBAC. In Proceedings of the 7th ACM Symposium on Access Control Models and Technologies, 2002：74-83

[9] James BD Joshi, Elisa Bertino, Basit Shafiq, Arif Ghafoor. Dependencies and Separation of Duty Constraints in GTRBAC. In Proceedings of the 8th ACM Symposium on Access Control Models and Technologies, 2003：51-64

[10] Ezedin Barka , Ravi Sandhu. A Role-based Delegation Model and Some Extensions. In the 23rd National Information Systems Security Conference, 2000：101-114

[11] LongHua Zhang, Gail-Joon Ahn, Bei-Tseng Chu. A Rule-based Framework for Role-based Delegation. In Proceedings of the 6th ACM Symposium on Access Control Models and Technologies, 2001：153-162

[12] LongHua Zhang, Gail-Joon Ahn, Bei-Tseng Chu. A Rule-based Framework for Role-based Delegation and Revocation. ACM Transaction on Information and System Security, 6(3)：404-441

[13] XinWen Zhang, Sejong Oh, Ravi S.Sandhu. PBDM: A Flexible Delegation Model in RBAC. In Proceedings of the 8th ACM Symposium on Access Control Models and Technologies, 2003：149-157

[14] Olav Bandmann, Mads Dam, Babak S. Firozabadi. Constrained Delegation. In Proceedings of the 23rd Annual IEEE Symposium on Security and Privacy, 2002：131-143

[15] 徐震，李斓，冯登国. 基于角色的受限委托模型. 软件学报，2005，16(5)：970-978

[16] Shihyu Chou, Eric Jui-Lin Lu, Yi-Hui Chen. X-RDR: A Role-based Delegation Processor for Web-based Information System. ACM SIGOPS Operating Systems Review, 2005, 39(1)：4-21

[17] 李成锴，詹永照，茅兵，谢立. 基于角色的CSCW系统访问控制模型. 软件学报，2000, 11(7)：931-937

[18] William Tolone, Gail-Joon Ahn, Tanusree Pai, Seng-Phil Hong. Access Control in Collaborative Systems. ACM Computing Surveys, 2005, 37(1) ： 29-41

[19] 陈伟鹤，殷新春，茅兵，谢立. 基于任务和角色的双重Web 访问控制模型. 计算机研究与发展，2004，41(9)：1466-1473

[20] 孙波，赵庆松，孙玉芳. TRDM——具有时限的基于角色的转授权模型. 计算机研究与发展，2004，41(7)：1104-1109

[21] Ravi Sandhu, Venkata Bhamidipati, Qamar Munawer. The ARBAC97 Model for Role-Based Administration of Roles. ACM Transactions on Information and System Security, 1999, 2(1)：105-135

[22] Sejong Oh, Ravi S.Sandhu. A Model for Role Administration Using Organization Structure. In Proceedings of the 7th ACM Symposium on Access Control Models and Technologies, 2002：155-162

[23] Jason Crampton, George Loizou. Administrative Scope and Role Hierarchy Operations. In Proceedings of the 7th ACM Symposium on Access Control Models and Technologies，2002：145-154

[24] Jason Crampton, George Loizou. Administrative Scope: A Foundation for Role-Based Administrative Models. ACM Transactions on Information and System Security, 2003, 6(2)：201-231

[25] Jason Crampton. Understanding and Developing Role-Based Administrative Models. In Proceedings of the 12th ACM Conference on Computer and Communications Security CCS '05, 2005：158-167

[26] 汤庸. 时态数据库导论. 北京：北京大学出版社，2004：13-29

[27] Ezedin Barka, Ravi Sandhu. Framework for Role-based Delegation Models. In Proceedings of 16th Annual Computer Security Application Conference, 2000：168-176

致 谢

首先衷心感谢我的导师汤庸教授。在我就读中山大学的两年间，对我的悉心指导以及对我所从事的科研机构协同工作平台项目的信任和支持，使得本人能够顺利完成毕业论文和项目开发的前期工作。他勇于创新的精神、严谨的治学态度、孜孜不倦的工作热情以及宽厚待人的态度一直是我学习的榜样。

特别要感谢叶小平老师对毕业论文的悉心指导、建议以及督促，在毕业论文研究的过程中遇到困难的时候，叶老师帮助我分析原因，理清思路，才使得我的毕业论文能够进一步研究下去。

感谢中山大学数据库与协同软件实验室的所有成员。感谢冀高峰师兄，在我确定毕业论文的研究方向时，给与我建设性的建议和帮助。

感谢杨坤同学给我的毕业论文提了许多中肯的意见，同时在很大程度上帮助我解决了在论文实现的代码编写过程中所遇到的困难。感谢邓楚燕、程明、蔡苗苗等同学在两年的研究生生涯中，我们互相交流、互相帮助、互相鼓励，一起成长。这一切都使我获益良多。感谢在研究生期间认识的朋友肖裕，是你们让我度过了一个充实快乐又多姿多彩的研究生生涯。

最后，还要感谢我的家人在我求学期间，在生活上对我无微不至的关心以及在我遭遇挫折、倍感失落的时候的默默支持，他们是我积极生活的最大动力。

原创性声明

本人郑重声明：所呈交的学位论文，是本人在导师的指导下，独立进行研究工作所取得的成果。除文中已经注明引用的内容外，本论文不包含任何其他个人或集体已经发表或撰写过的作品成果。对本文的研究做出重要贡献的个人和集体，均已在文中以明确的方式标明。本人完全意识到本声明的法律结果由本人承担。

学位论文作者签名：杨虹铁

2006年 6 月 6 日