임무분리와 역할 계층구조를 고려한 대칭 RBAC 모델

(Symmetric RBAC Model that Takes the Separation of Duties and Role Hierarchies into Consideration)

문 창 주 † 박 대 하 †† 박 성 진 ††† 백 두 권 ††††
(Chang Joo Moon) (Dae Ha Park) (Seong Jin Park) (Doo Kwon Baik)

요 약 RBAC은 기존의 DAC과 MAC보다 진보된 접근 통제 방법으로 받아들여진다. RBAC 모델의 권한-역할 부분은 사용자-역할 부분에 비해 상대적으로 연구가 부족하며 이를 극복하기 위한 대칭 RBAC 모델에 대한 연구도 시작 단계이다. 따라서 역할에 적합한 권한을 배정 하는데 많은 어려움이 있다.

본 논문에서는 기존 연구들에서 제시한 권한 배정 제약조건들을 보완한 대칭형 RBAC 모델을 제안한다. 제안한 대칭형 RBAC 모델은 임무분리와 역할의 계층구조를 고려한 권한 배정 제약조건을 제시함으로써 역할의 이해관계 충돌과 권한의 공유와 통합을 권한배정에 반영하고 있다. 또한, AND/OR 그래프를 통해 동적인 권한간의 선행관계를 규정하는 제약조건을 표현함으로써 권한들의 복잡한 선행관계를 효과적으로 제한할 수 있다. 제안한 대칭형 RBAC 모델의 권한 배정 제약조건들은 권한 배정시 지켜야하는 규칙들을 적절히 명세함으로써 권한 배정의 오류를 감소시킨다.

키워드 : RBAC, 역할, 권한부여, 제약조건, 권한

Abstract RBAC is accepted as a more advanced control method than existing DAC and MAC. Studies on the permission-role part of RBAC model are relatively insufficient compared with those on the user-role part, and researches on symmetric RBAC models to overcome this is also in an incipient stage. Therefore there is much difficulty in assigning permissions suitable for roles.

This paper proposes an symmetric RBAC model that supplements the constraints on permission assignment set forth by previous studies. The proposed symmetric RBAC model reflects the conflicts of interests between roles and the sharing and integration of permissions on the assignment of permissions by presenting the constraints on permission assignment that take the separation of duties and role hierarchies into consideration. In addition, by expressing constraints prescribing prerequisite relations between dynamic permissions through AND/OR graphs, it is possible to effectively limit the complicated prerequisite relations of permissions. The constraints on permission assignment for the proposed symmetric RBAC model reduce errors in permission assignment by properly detailing rules to observe at the time of permission assignment.

Key words : RBAC, Role, Authorization, Conatraint, Permission

1. 서 론

RBAC(Role-Based Access Control)의 개념은 다중 사용자와 다중 애플리케이션이 상호간에 운영되는 온라인 시스템에서 시작되었으며[1] 기존의 DAC(Discretionary Access Control)이나 MAC(Mandatory Access Control)보다 진보된 접근 제어 방법으로 받아들여지고 있다[2]. RBAC 모델에서는 각 권한(permission)이 역할(role)에 배정되고, 각각의 사용자(user) 역시 적합한 역할에 배정되는데 이러한 개념은 보안 관리자로 하여금 다수의 권한을 효율적으로 관리할 수 있게 한다. 역할에 권한을 배정하는 부분은 역할에 사용자를 배정하는 부분보다 일반적으로 느리게 변화하기 때

† 비 회 원 : 고려대학교 컴퓨터학과
mcj@swsys2.korea.ac.kr
†† 비 회 원 : 시큐리티테크놀로지스 정보보호기술연구소
dhpark@stitec.com
종신회원 : 한신대학교 정보시스템공학과 교수
sjpark@hanshin.ac.kr
종신회원 : 고려대학교 컴퓨터학과 교수
baikdk@korea.ac.kr
논문접수 : 2003년 2월 4일
심사완료 : 2003년 8월 22일

문에, 새로운 역할을 생성하거나 권한 배정을 변경하지 않고 대신에 기존 역할에 사용자를 추가로 배정하거나 삭제하는 것이 효과적이다[1,3]. 이러한 이유 때문에 기존의 RBAC 모델 관련 많은 연구들[1,4-9]은 주로 사용자와 역할의 관계 부분에 집중되었다.

한편, NIST(National Institute of Standards and Technology)의 RBAC 표준에 관한 연구[4]에서는 RBAC 모델들을 기능에 따라 기본형 RBAC(Flat RBAC), 계층형 RBAC(Hierarchical RBAC), 제한형 RBAC(Constrained RBAC), 대칭형 RBAC(Symmetric RBAC) 모델로 분류하였다. 이 중에서 대칭형 RBAC 모델은 권한-역할 관계를 다루는 기능이 추가되었으나 현재까지 구체적으로 실현되지는 못했다. 그 이유는 권한의 경우, RBAC이 적용되는 분야에 따라 각각 상이한 특성을 가짐으로 단일 대칭 RBAC 모델을 정의하기가 매우 어렵기 때문이다. 그럼에도 불구하고 권한 부여(authorization)의 핵심은 각 역할에 적합한 권한을 배정하는 부분[4]으로, 만약, 역할에 부적합한 권한들이 배정되었다면, 이 역할에 대한 사용자 배정은 아무런 의미가 없게 된다. 따라서, 각 역할에 적합한 권한을 배정할 수 있는 단일 대칭형 RBAC 모델이 반드시 필요하다.

대칭형 RBAC 모델의 공통된 문제는 역할의 계층 구조(role hierarchies)를 통하여 권한이 통합되는 과정에서 어떻게 역할에게 적합한 권한만을 배정하는가? 하는 것이다[4]. 일반적으로 상위 역할은 계층구조를 통하여 하위 역할의 권한들을 계승받게 되고 이 과정에서 역할에 부적합한 권한이 배정되면 역할은 자신의 권한을 벗어나는 행위를 하게 된다. 따라서, 역할의 계층 구조를 적절히 고려함으로써 부적절한 권한 계승을 방지할 수 있는 대칭형 RBAC 모델이 제시되어야 한다. 또한, 기존 RBAC 연구의 경우, 운영체제의 파일이나 데이타베이스의 테이블과 같은 정적인 객체들을 권한의 대상으로 고려하였다. 그러나 최근 활용이 급증하고 있는 소프트웨어 컴포넌트[10,11]와 같이 권한의 객체가 실행 가능한 코드인 경우, 이들 객체들은 동적이고 변화가 많은 특성을 갖기 때문에 기존 RBAC 모델로는 접근을 제어하기 어렵다. 즉, 컴포넌트처럼 다른 컴포넌트들과 상호작용 하고 필요에 따라 추가, 삭제되는 동적인 객체들에 대한 적절한 권한 배정을 위해서는 기존 대칭 RBAC 모델이 정적인 권한과 동적인 권한을 모두 수용할 수 있도록 확장될 필요가 있다.

본 논문에서는 기존 RBAC 연구들의 권한 배정을 위한 제약조건들을 보완한 확장된 대칭형 RBAC 모델을 제안한다. 제안하는 대칭형 RBAC 모델은 기존 사용자-역할 관계에 적용했던 제약조건을 권한-역할 관계에도 대칭적으로 적용하며 적용 과정에서 역할의 계층구조, 임무분리(SOD: Separation Of Duty), 동적인 권한특성, 공유제한 등의 개념을 추가로 고려한다. 이러한 개념들이 추가된 4개의 권한 배정 제약조건들은 단순히 사용자 배정 제약조건을 권한 배정 제약조건으로 적용할 경우 발생할 수 있는 문제들을 해결하고 권한 배정의 오류를 줄일 수 있다.

논문의 나머지 구성은 다음과 같다. 2장에서는 기존 RBAC 모델들에 대해 분석하고 3장에서는 제시된 대칭형 RBAC 모델에 대해 기술한다. 4장에서는 사례 연구를 통하여 새롭게 정의하는 4개의 권한 배정 제약조건이 실세계에 적용되는 예를 보인다. 5장에서는 제안하는 대칭형 RBAC 모델과 기존 모델을 비교 평가하였으며, 마지막으로 6장에서는 본 논문에 대한 결론과 향후 과제에 대해서 언급한다.

2. RBAC 모델

그림 1은 RBAC96[1] 모델에 대한 개념적인 다이어그램이다. 이 모델은 사용자, 역할, 권한, 세션 등 4가지 구성요소를 소유한다. 사용자(U)는 인간의 행위나 자율적인 에이전트 등을 나타내고 역할(R)은 구성원들의 직책이나 책임 등을 고려한 일의 기능이나 직책을 나타낸다. 권한(P)은 하나 이상의 객체들에 대한 접근의 승인이나 특정 행위를 할 수 있는 특권을 나타낸다. 세션(S)은 한 사용자에게 가능한 여러 역할을 나타낸다. 특정 세션(session) 안에 있는 사용자는 자신이 속한 역할들의 일부분을 사용할 수 있다.

NIST는 다음과 같이 RBAC 모델을 분류하였다[4]. 첫 번째, 기본형 RBAC에서 사용자는 역할에 배정되고, 권한도 역할에 배정된다. 사용자는 특정 세션 안에서 사용 가능한 모든 역할에 배정된 권한의 합집합을 소유한다. 사용자-역할, 권한-역할 배정은 다대다(many-to-many) 관계이고 사용자는 동시에 여러 세션을 소유할 수 있다.

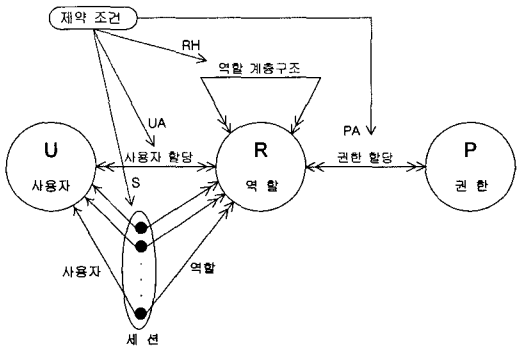

그림 1 RBAC 모델

두 번째, 계층형 RBAC은 기본형 RBAC에 역할간의 계층구조 지원을 추가한 형태이다. 계층구조에서 상위에 있는 역할은 하위에 있는 역할의 권한들을 소유한다. 일반적인 계층형 RBAC은 역할의 계층구조에 임의의 부분 순서를 지원하고 제한된 계층형 RBAC은 트리나 역트리 형태의 계층구조만을 지원한다.

세 번째, 제한형 RBAC은 계층형 RBAC에 임무분리의 지원을 추가한 형태이다. 임무 분리는 사용자간의 이해관계 충돌을 막아 사용자가 직위나 직책을 벗어나는 행위를 할 수 없게 하는 것으로, 다시 정적인 임무분리(SSD: Static Separation of Duty)와 동적인 임무분리(DSD: Dynamic Separation of Duty)로 분류된다. 정적인 임무분리는 동일 사용자가 정적인 임무분리가 선언된 두 개 이상의 역할에 배정되지 못하게 하는 것이며, 동적인 임무분리는 동일 사용자가 동적인 임무분리가 선언된 두 개 이상의 역할에 배정될 수 있는 반면 동일 사용자가 동시에 이 역할들을 사용할 수 없게 한다.

네 번째, 대칭형 RBAC은 제한형 RBAC에 사용자 배정 제약조건과 유사한 권한 배정 제약조건을 추가한 형태이다. 적절한 권한-역할 배정은 사용자-역할 배정에 못지않게 권한부여 관리구조에서 핵심적인 요소이다. 대칭형 RBAC을 운영할 때 과거의 권한-역할 배정은 새로운 상황 변화에 따라 적절하지 않을 수 있으므로 상황변화에 능동적으로 대처할 수 있어야 하며, 관리상의 다양한 상황들이 고려되어야 한다.

3. 보완된 대칭형 RBAC 모델

본 논문에서는 기존 연구에서의 권한 배정 제약조건들의 문제점을 지적하고, 이를 보완한 새로운 제약조건을 갖는 대칭형 RBAC 모델을 제시한다. 그림 2는 새로운 권한 배정 제약조건을 정의하기 위해 필요한 RBAC96 모델의 기본 요소들과 함수이다[6].

3.1 분리 권한(DP : Disjoint Permission) 제약조건

RBAC96 모델은 정적인 임무분리가 선언된 역할의 권한 배정을 위해서 사용자 배정과 동일한 제약조건을 적용한다. 즉, 동일한 권한은 정적인 임무분리가 선언된 두 개 이상의 역할에 배정되지 못한다는 것[1]이다. 그러나, 이러한 제약조건은 개념적으로는 이해되지만 정확하지 못하다. 권한은 사용자와는 다르게 역할의 계층 구조를 통하여 상속이 가능하므로 만일 두 역할이 임무분리가 선언되고 동일한 하위 역할을 가지고 있다면 계승된 권한은 RBAC96의 제약조건을 위반하게 된다. 또한, 두 역할은 계승된 권한보다는 새롭게 추가된 일부 권한들에 의해서 상호배제적인 특성을 가질 수 있다. 따라서, 이러한 제약 조건이 역할의 계층구조상에서 유효하려면 제약조건이 적용되는 권한들의 범위를 명확히 지정해 주어야 한다.

R = 모든 역할들의 집합, $\{r_1, \ldots, r_n\}$
U = 모든 사용자들의 집합, $\{u_1, \ldots, u_m\}$
P = 모든 권한들의 집합, $\{p_1, \ldots, p_o\}$
UA ⊆ U × R, 다대다 관계인 R 에 대한 U의 배정 관계.
PA ⊆ P × R, 다대다 관계인 R 에 대한 P의 배정 관계.
RH ⊆ R × R, 역할의 계층구조 또는 역할의 지배관계에서 R에 대한 부분순서.
users : $R \rightarrow 2^U$, 각각의 역할 r_a 에 사용자들의 집합을 사상하는 함수. $(1 \le a \le n)$
$users(r_a) = \{u \in U \mid (u, r_a) \in UA\}$,
역할 r_a 에 배정된 사용자들을 반환하는 함수.
perms : $R \rightarrow 2^P$, 각각의 역할 r_a 에 권한들의 집합을 사상하는 함수.
$perms(r_a) = \{p \in P \mid (p, r_a) \in PA\}$,
역할 r_a 에 배정된 권한들을 반환하는 함수.
roles : $P \rightarrow 2^R$, 각각의 권한 p_b와 역할들의 집합을 사상하는 함수. $(1 \le b \le o)$
$roles(p_b) = \{r \in R \mid (p_b, r) \in PA\}$
권한 p_b가 배정된 역할들을 반환하는 함수.
juniors : $R \rightarrow 2^R$, 각각의 역할 r_a에 r_a의 하위 역할들의 집합을 사상하는 함수.
$juniors(r_a) = \{r \in R \mid (r_a, r) \in RH\}$
역할 r_a의 하위 역할들을 반환하는 함수.
seniors : $R \rightarrow 2^R$, 각각의 역할 r_a에 r_a의 상위 역할들의 집합을 사상하는 함수.
$seniors(r_a) = \{r \subset R \mid (r_a, r) \subset RH\}$
역할 r_a의 상위 역할들을 반환하는 함수

그림 2 RBAC96 모델의 기본요소와 함수

그림 3은 DP 제약조건의 기본요소이다. 정적인 임무분리[12,13]는 DP 제약조건을 이해하는데 필수적이며 다음과 같이 먼저 정의한다.

ssd = 정적인 임무분리가 선언된 역할들의 집합,
$\{sr_1, \ldots, sr_q\}$, $ssd \subseteq R$
SSD = ssd의 집합, $\{ssd_1, \ldots, ssd_p\}$
dp = DP 제약조건이 적용되는 권한의 집합, $dp \subseteq P$
DP = dp의 집합, $\{dp_1, \ldots, dp_p\}$
dp_c, ssd_c는 쌍을 이룸$(1 \le c \le p)$

그림 3 DP 제약조건의 기본요소

정의 1 (SSD 제약조건). 모든 사용자는 ssd에 속한 두 개 이상의 역할에 배정될 수 없다.

정의 1에 대한 정형명세는 다음과 같다

$$\forall ssd \left(\bigcap_{d=1}^{q} users(sr_d) = \emptyset \right)$$

보조정리 1. ssd에 속한 역할들에는 동일한 상위역할이 존재하지 않는다.

보조정리 1 에 대한 정형명세는 다음과 같다.

$$\forall ssd \left(\bigcap_{d=1}^{q} seniors(sr_d) = \emptyset \right)$$

정의 2 (DP 제약조건). dp에 속한 권한은 ssd에 속한 두 개 이상의 역할에 배정될 수 없다.

정의 2에 대한 정형명세는 다음과 같다.

$$\forall ssd\ (\bigcap_{d=1}^{q} perms(sr_d) \cap dp) = \varnothing)$$

예제 1. $ssd_c=\{r_1, r_2, r_3\}$, $dp_c=\{p_3, p_4, p_5\}$이라면 권한 p_3, p_4, p_5는 역할 r_1, r_2, r_3에 두 번 이상 배정될 수 없으므로 $perms(r_1)=\{p_1, p_3\}$, $perms(r_2)=\{p_2, p_4\}$, $perms(r_3)=\{p_1, p_2, p_5\}$와 같은 배정이 가능하다.

본 논문에서는 제약조건 정형 명세를 위하여 집합 첨수족(Indexed Family of Sets)[14]을 사용하였다. 집합의 모임을 집합족 이라 하고 첨수를 붙이는 대상이 집합이 될 경우, 이 집합 등이 모여서 만드는 집합족을 집합 첨수족이라 한다. SSD와 DP은 권한의 집합들을 원소로 가지고 있어 첨수 집합족으로 정형 명세하는 것이 효율적이다.

DP 제약조건은 사용자 배정에 적용되는 임무분리의 개념을 모든 권한에 대해서 적용하는 것이 아니라 DP 제약조건에서 명시한 권한에 대해서만 적용한다. ssd_c에 속한 역할들에 대해서 DP 제약조건을 적용할 때 ssd_c에 속하지 않은 역할에 배정된 권한은 고려할 필요가 없으므로 dp_c와 ssd_c는 쌍을 이루게 했다. ssd_c의 원소인 역할에 dp_c의 원소인 권한이 배정 되는 경우만 DP 제약조건이 적용된다.

보조정리 2. jr은 ssd에 속한 역할들에 대한 하위역할의 합집합이다.

보조정리 2 에 대한 정형명세는 다음과 같다

$$jr = (\bigcup_{d=1}^{q} juniors(sr_d)$$

정리 1 (DP 제약조건이 적용되는 권한의 범위). ssd에 속한 역할들에 대한 권한의 합집합을 A라하고 jr에 속한 역할들에 대한 권한의 합집합을 B라 하면 dp는 A-B의 부분집합이다.

정리 1에 대한 정형 명세는 다음과 같다

$$jr = \{jr\_r_1, \ldots, jr\_r_s\},\ jr \subseteq R$$

$$dp \subseteq (\bigcup_{e=1}^{q} perms(sr_e) - \bigcup_{f=1}^{s} perms(jr\_r_f))$$

증명 : ssd의 원소인 역할 sr에 배정된 권한 중에서 계승된 권한, $\bigcup_{f=1}^{s} perms(jr_r_f)$은 sr이 상호배제적인 특성을 가지는데 영향을 주지 않고, DP 제약조건을 위반하므로 적용범위에서 제외되어야 한다.

예제 2. 그림 4에서 역할 계산원, 매장관리 사원, 창고관리 사원이 정적인 임무분리가 선언된 이유는 계승된 권한 p_1, p_2, p_3 때문이 아니라 추가된 권한 p_4, p_5, p_6 때문이다. 즉, 특정 역할에 권한 p_4, p_5, p_6을 모두 배정하면 그 역할에 배정된 사용자는 부당하게 물품을 판매할 수 있다. 따라서, 권한 p_4, p_5, p_6를 역할 계산원, 매장관리 사원, 창고관리 사원에 배정할 때 DP 제약조건이 적용되어야 한다.

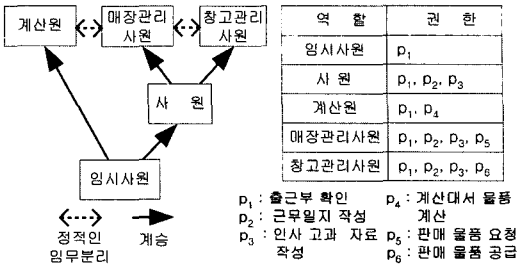

그림 4 DP 제약조건 적용범위

예제 2에 정리 1의 내용을 적용하면 ssd_c={계산원, 매장관리 사원, 창고관리 사원}, A=perms(계산원)∪perms(매장관리 사원)∪perms(창고관리 사원) 즉 $\{p_1, p_2, p_3, p_4, p_5, p_6\}$, B=perms(임시사원)∪perms(사원) 즉 $\{p_1, p_2, p_3\}$가 된다. 따라서 $dp_c \in \{\{p_4\}, \{p_5\}, \{p_6\}, \{p_4, p_5\}, \{p_5, p_6\}, \{p_4, p_6\}, \{p_4, p_5, p_6\}\}$이어야 한다. dp_c가 공집합인 경우는 모든 권한배정이 DP제약조건을 만족하므로 ssd_c와 dp_c사이에 DP 제약조건이 작용하지 않는다.

3.2 충돌 권한(CP : Conflicting Permissions) 제약조건

충돌 권한들은 동일 역할에 배정될 수 없다[6,15]. 이 제약조건은 권한간의 이해관계 충돌에 대해서 기술한다. 예를 들어, 권한 p와 q가 충돌 권한이라면 역할 A에 둘 중에 하나만 배정될 수 있다. 궁극적으로는 사용자는 둘 이상의 충돌 권한을 가질 수 없다. 이 제약조건은 상당히 설득력이 있지만 임무분리와 권한 공유와의 관계를 언급하지 않고 있다. 권한공유란 두개 이상의 역할들에 동일한 권한이 배정된 경우를 말한다. 따라서 상위 역할과 하위 역할은 하위역할의 모든 권한을 공유한다.

그림 5는 CP 제약조건의 기본요소이다.

cp = 이해관계의 충돌을 일으키는 권한의 집합,
{cp_p1, ... , cp_pn}, cp ⊆ P
CP= cp들의 집합, {cp1, ... ,cpt},
RC = R × CP, {(r, cp) ∈ RC | r ∈ R ∧ cp ∈ CP}

그림 5 CP 제약조건의 기본요소

정의 3 (CP 제약조건). cp에 속한 권한은 동일한 역할에 둘 이상 배정될 수 없다.

정의 3을 정형명세로 나타내면 다음과 같다.

$$\forall (r, cp)\ (counts(perms(r) \cap cp) \leq 1)$$

함수 counts(S)은 임의의 집합 S의 원소의 개수를 반환하는 함수이다.

예제 3. CP = {{p_1, p_2}, {p_1, p_3}}, p_1 = 물품구매, p_2 = 청구서 처리, p_3 = 입출금이면 perms(r_1)={p_1, p_2, p_4} 또는 perms(r_2)={p_1, p_3, p_4}와 같은 배정은 허락되지 않는다. 역할 r_1 또는 r_2와 같은 역할이 존재한다면, 이들 두 역할에 배정된 사용자는 부당하게 물품을 구입할 수 있다.

정의 3과 같이 권한의 배정을 금지하는 것만으로는 부족하다. 권한 p_1과 p_2가 각기 다른 역할들에 배정되어도 이 역할들에 모두 배정된 사용자는 결국 충돌권한 p_1과 p_2를 모두 소유하게 되어 보안상 문제가 발생한다. 그리고, 이 역할들이 동일한 상위역할을 가지고 있어 p_1과 p_2가 계승을 통해 공유된다면 역시 CP 제약 조건을 위배한다. 따라서, 아래와 같은 정리가 추가적으로 필요하다.

정리 2 (충돌권한과 정적인 임무분리와의 관계). $(p \in cp) \wedge (p \in perms(r))$이 참이고 $(p' \in cp) \wedge (p' \in perms(r'))$이 참이면 $\{r, r'\} \subseteq ssd$ 이어야 한다.

정리 2에 의해서 두 개 이상의 서로 다른 충돌권한이 각각 배정된 역할들은 정적인 임무분리가 정의되어야 한다.

예제 4. CP={{p_1, p_2}, {p_1, p_3}}, SSD={{r_1, r_2, r_3}, {r_6, r_7}}, p_1 = 물품구매, p_2 = 청구서 처리, p_3 = 입출금, r_1 = 계좌관리 사원, r_2 = 구매관리 사원이면 perms(r_1)={p_2, p_3}, perms(r_2)={p_1}의 배정이 가능하다. 이때 {r_1, r_2}는 SSD의 원소 중에 하나인 {r_1, r_2, r_3}의 부분집합이어야 한다.

예제 4에서 r_1과 r_2사이에 정적인 임무분리가 선언되었다면, 정의 1에 의해서 두 역할에 모두 배정된 사용자는 존재하지 않으므로 p_1, p_2 권한을 모두 소유한 사용자는 존재하지 않는다. 또한, 보조정리 1에 의해서 r_1과 r_2에 대해서 공통된 상위역할을 소유할 수 없으므로 계승에 의해서 충돌 권한들이 동일역할에 배정되는 것을 막을 수 있다. 따라서, CP 제약조건은 두 권한 사이의 이해관계의 충돌과 임무분리를 동시에 고려하여 충돌권한에 의해서 발생할 수 있는 권한 배정의 오류를 줄인다.

CP 제약조건과 DP 제약조건은 상호보완 되어 사용되어야 한다. DP 제약조건은 권한의 배정을 통제할 수 있는 강력한 수단을 제공해준다. 그러나, DP 제약조건을 준수해서 배정된 권한간에도 충돌권한들이 존재할 수 있다. 즉, DP 제약조건은 CP 제약조건이 해결한 문제를 해결하지 못한다. CP 제약조건은 권한간의 이해관계 충돌을 해결할 수 있다. 그러나, 임의의 cp에 속한 한 권한이 정적인 임무분리가 선언된 여러 역할에 할당되는 것은 막을 수는 없다. 즉, CP 제약조건은 DP 제약조건이 해결한 문제를 해결하지 못한다. 따라서, 이들은 함께 사용되면 좀더 효과적인 권한 배정을 제한할 수 있다.

3.3 선행 권한 (PP : Prerequisite Permission) 제약조건

RBAC96 모델의 선행권한 제약조건에 의해 권한 p가 임의의 역할에 배정되려면 권한 q가 이미 그 역할에 배정되어 있어야 한다[1]. 이때 권한 q는 권한 p의 선행권한이다. 그러나, 이 권한 배정 제약조건 너무 단순하며 이러한 제약조건만으로는 동적인 권한간의 복잡한 의존관계를 반영하기 어렵다. 따라서, 이 제약조건은 하나 이상의 선행관계가 연속적으로 존재하는 경우 효과적인 권한의 배정 제약조건으로 작용하기 힘들다.

그림 6은 PP 제약조건의 기본요소이다. PP 제약조건은 [권한, 관계, 선행권한집합] 형태의 구조체로 명세된다. tp는 선행권한을 필요로 하는 권한이고 ao은 'AND' 또는 'OR' 중에 하나의 값을 가진다. pps는 tp에 대한 선행권한의 집합이다.

```
tp = 선행 권한을 필요로 하는 권한, tp ∈ P
pps = 선행권한의 집합, {pps_p1, . . . . ,pps_pw}, pps ⊆ P
pp = 권한(tp), 관계(ao), 선행권한 집합(pps) 으로 구성된 구조체,
     [tp, ao, pps], ao = 'AND' ∨ 'OR'
PP = pp의 집합, {pp1, . . . , ppv}
```

그림 6 PP 제약조건의 기본요소

정의 4 (PP 제약조건).

(i) ao가 'AND'이면 tp가 임의의 역할에 배정되기 위해서는 역할에 이미 pps의 모든 권한이 배정되어야 한다.

(ii) ao가 'OR'이면 tp가 임의의 역할에 배정되기 위해서는 역할에 이미 pps의 권한 중에 하나 이상의 권한이 배정되어야 한다.

정의 4를 정형명세로 나타내면 다음과 같다.

(i) $\forall r \forall pp(((tp \in perms(r)) \wedge (ao='AND'))$
$\rightarrow pps \subseteq perms(r))$

(ii) $\forall r \forall pp(((tp \in perms(r)) \wedge (ao='OR'))$
$\rightarrow pps\_p_i \in perms(r)) \ (1 \leq i \leq w)$

PP 제약조건은 그림 7과 같이 AND/OR 그래프[16,17]로 나타낼 수 있다. 그래프상에서 노드는 권한이고 화살표는 선행관계를 나타낸다. 화살표 방향에 있는 권한이 선행 권한이다. 두 화살표 사이에 연결하는 원호가 존재하면, 이 두 권한은 AND관계이고 원 호가 없으면 OR관계이다. 권한이 선행 권한을 가지고 있으면, 선행 권한 레벨의 모든 노드는 AND관계 또는 OR관계가 되어야 한다. 그림 7의 (a)와 같이 AND/OR 그래프의 동일 수준에 AND관계와 OR관계가 같이 있을 때는 (b)와 같이 변환되어야 한다. 이러한 변환은 노드를 순회하는 알고리즘을 용이하게 한다. vp는 가상 권한으로 실제 존재하는 권한은 아니지만, 그래프의 변환을 위하여

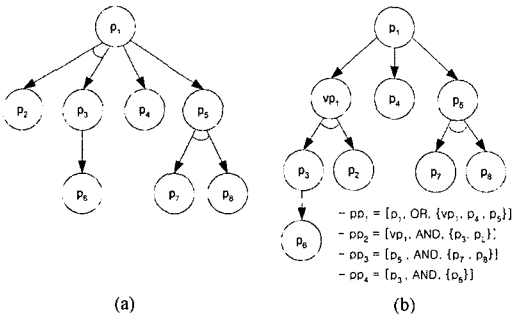

그림 7 선행권한에 대한 AND/OR 그래프

사용되며 반드시 서브 노드를 가진다. 가상권한을 RBAC 서버에 적용할 때는 자신의 선행권한으로 치환해야 한다. 그래프상에서 가상 권한은 다른 가상 권한의 선행권한이 될 수 없다.

컴포넌트의 서비스와 같이 상호작용 하는 동적인 권한은 서비스 제공을 위해서 다른 서비스가 필요하거나 자신이 다른 서비스 제공을 위해 필요한 서비스가 된다. 따라서 업무 로직이 수행되는 도중에 한 서비스(caller)가 호출하게 되는 다른 서비스(callee)가 선행 서비스가 된다. 역할은 처음 접근한 서비스에 대한 권한과 그 서비스의 선행 서비스에 대한 권한을 모두 소유해야만 역할에 속한 사용자들은 정상적인 서비스를 받을 수 있다. 컴포넌트 환경에서는 동일한 서비스를 제공하는 벤더(vendor)가 다른 여러 컴포넌트가 존재한다. 이러한 서비스를 선행 권한으로 하는 권한은 OR 관계에 의해서 표현될 수 있다.

예제 5. 그림 7의 (b)에서 p_7과 p_8은 p_5에 대해서 'AND'관계의 선행 권한이므로 어떤 역할에 p_5가 배정되려면 역할은 이미 p_7과 p_8이 배정되어 있어야 한다. 그리고 p_4, p_5, vp_1는 p_1에 대해서 'OR'관계의 선행 권한이므로 어떤 역할에 p_1가 배정되려면 역할은 이미 p_4, p_5, vp_1 중에서 하나 이상의 권한이 배정되어 있어야 한다.

3.4 단일 역할에만 권한 배정(PASR : Permission Assigned to Single Role) 제약조건

PASR 권한 배정 제약조건은 역할과 권한이 강한 연관 관계를 가지는 경우 권한의 공유를 제한하는데 유용하다. 그림 8은 PASR 제약조건의 기본요소이다. 이 제약조건은 [역할, 권한 집합]의 구조체로 명세된다.

정의 5 (PASR 제약조건). pasr에서 pasr_ps에 포함된 권한은 pasr_r과 seniors(pasr_r)에 속한 역할에만 배정된다.

정의 5를 정형 명세로 나타내면 다음과 같다.

$$\forall pasr\ ((pasr\_ps \subseteq perms(pasr\_r)) \wedge (pasr\_ps \cap \bigcup_{k=1}^{y} perms(cop\_r_k) = \varnothing))$$

pasr = 역할(pasr_r), 권한집합(pasr_ps)으로 구성된 구조체,
[pasr_r, pasr_ps], pasr_r ∈ R , pasr_ps ⊆ P
PASR = pasr의 집합, {pasr1, ... , pasrx}
cop = 전체역할에서 pasr_r역할과 pasr_r의 상위역할을 제외한 집합,
R-({pasr_r} ∪ seniors(pasr_r)), {cop_r1, . . . , cop_ry}

그림 8 PASR 제약조건의 기본요소

예제 6 은행에서 대출심사 권한은 대출사원 역할과 그 상위역할에만 배정되어야 한다. 그리고, 컴포넌트 기반 시스템에서 관리자만이 처리해야 하는 업무는 관리자 역할을 제외한 다른 역할에는 배정되지 않아야 한다.

PASR 제약 조건은 예제 6에서처럼 업무 특성상 또는 보안 관리상의 이유로 특정 권한을 한 역할에만 배정하는 경우 사용된다.

4. 적용 사례

이 장에서는 그림 9의 역할들에 배정된 권한들을 예로 하여 3장에서 제시된 권한 배정 제약조건들이 효과적으로 사용될 수 있음을 보인다. 그림 9는 사례연구에서 사용되는 역할의 임무분리와 계층구조를 나타낸다.

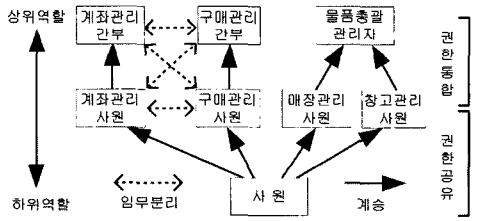

그림 9 역할의 구조

그림 10의 표는 그림 9의 역할들에 대한 권한 배정을 나타낸다. 표 하단에는 각 권한배정에 적용된 제약조건을 나타낸다. 표 우측으로 갈수록 제약조건이 추가된다.

1) DP 제약조건

(a)에서 정적인 임무분리가 선언된 역할들은 p_1, p_2, p_3, p_6, p_7 권한을 공유하고 있다. 권한 p_1은 사원 역할로부터 계승된 권한이므로 공유되어도 문제가 없다. 권한 p_2(입금, 출금 요청)는 모든 사원들이 소유할 필요는 없지만, 임무분리가 선언된 역할에 공유되어도 역할의 이해관계 충돌은 일어나지 않는다. 역할의 이해관계 충돌은 역할이 자신에게 부적합한 권한을 배정 받음으로 다른 역할과 충돌이 일어나는 것을 말한다. 예를 들어, 구매관리사원 역할이 회사 계좌에 대한 입출금을 할 수 있는 권한을 부여받게 되면 구매관리사원 역할과 계좌관리사원 역할은 충돌을 일으킨다. 왜냐하면, 두 역할에 임무분리 선언의 근거가 되는 계좌관리사원 역할의 고

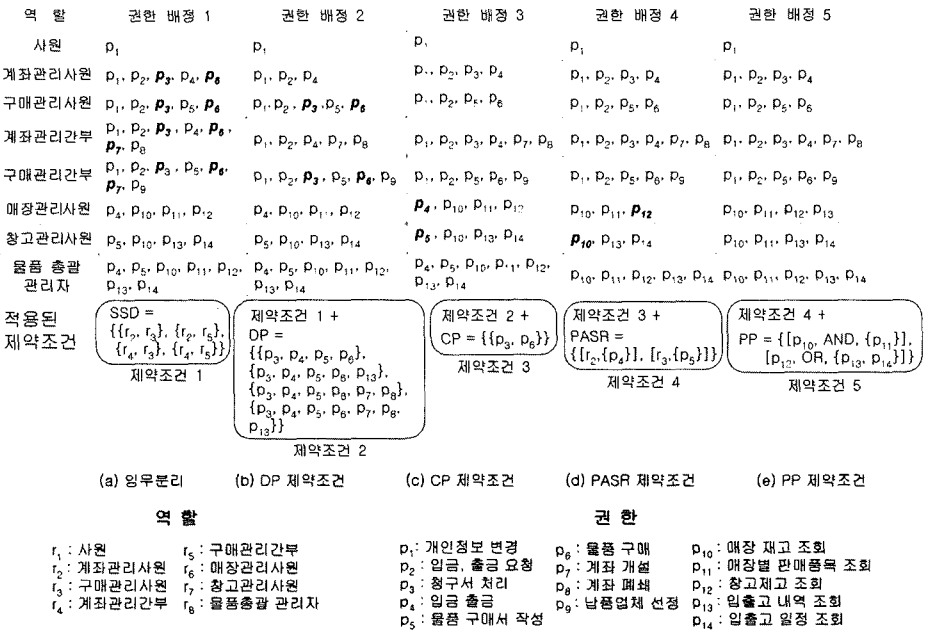

그림 10 권한 배정 제약조건 적용의 예

유 업무를 구매관리사원 역할에게 침해당하기 때문이다.

굵은 이탤릭으로 표시된 p_3, p_6, p_7 권한은 임무분리가 선언된 역할들 사이에서 공유되는 경우, 역할의 이해관계 충돌을 가져온다. 즉, 청구서 처리를 구매관리사원이 처리하거나 물품구매를 계좌관리사원이 처리해서는 안된다. (b)는 (a)의 이러한 권한 배정 오류를 해결하기 위해서 정의 2와 정리 1에 근거한 DP 제약조건을 추가하였다. DP의 원소에 포함된 권한은 쌍을 이루는 SSD의 원소에 포함된 역할들에 두 번 이상 배정되지 않는다. DP 제약조건에 의해서 {계좌관리사원, 구매권리사원}의 역할들에는 {p_3, p_4, p_5, p_6}의 권한이 두 번 이상 배정되면 안된다. DP 제약조건은 정적인 임무분할이 선언된 역할사이에 권한 배정을 제한하는 효과적인 방법으로 사용된다.

2) CP 제약조건

(b)의 굵은 이탤릭으로 표시된 p_3 권한과 p_6 권한은 여전히 문제가 있다. 청구서를 처리하는 권한(p_3)과 물품을 구매하는 권한(p_6)을 한 사람이 모두 가지고 있으면 거짓 구매가 일어날 수가 있다. (c)는 (b)의 권한 배정 오류를 해결하기 위해서 정의 3과 정리 2에 근거한 CP 제약조건을 추가하였다. (c)는 CP 제약조건에 의해서 모든 역할에는 p_3권한과 p_6권한이 같이 배정되지 않았다. 그리고, p_3권한과 p_6권한이 각각 배정된 역할들은 모두 정적인 임무분리가 선언되어 있다. 따라서, 두 권한을 모두 소유한 사용자와 역할은 존재하지 않는다.

3) PASR 제약조건

DP는 임무분리가 선언된 역할들 사이에서 권한 배정을 제한할 수 있다. 그러나, 임무분리가 선언된 역할과 그렇지 않은 역할 사이의 권한 공유를 제한하지는 못한다. 따라서, (c)의 굵은 이탤릭으로 표시된 p_4, p_5권한과 같이 DP에 속한 권한들은 정적인 임무분리가 선언되지 않은 매장관리 사원과 창고관리 사원 역할에 배정될 수 있다. 그러나, 업무특성상 이러한 권한의 공유를 금지해야 한다면 PASR 제약조건을 사용해야 한다. (d)는 정의 5를 적용하여 PASR 제약조건을 (c)에 추가한 예이다. 업무방침에 따라 회사공금에 대한 입출금(p_4)은 계좌 관리사원만 할 수 있도록 제한하였고, 물품 구매서 작성(p_5)은 구매관리 사원만 할 수 있도록 제한하였다.

4) PP 권한 배정 제약조건

(e)는 (d)에 PP 제약조건을 추가한 예이다. 회사의 업무 규칙에 의해서 매장관리 사원은 창고재고 조회(p_{12}) 권한을 배정 받기 위해서는 입출고 내역 조회(p_{13})권한이나 입출고 일정 조회(p_{14})권한을 소유하고 있어야 하고 매장관리 사원이 매장 재고 조회(p_{10})권한을 배정 받기 위해서는 매장별 판매품목 조회(p_{11})권한이 배정되어 있어야 한다면 (d)의 굵은 이탤릭으로 표시된 권한 배정은 오류가 있다. 이 경우 (e)와 같이 PP 제약조건의 AND/OR 관계를 이용하여 권한의 선행 관계를 명시해 주어야 한다.

유능한 보안 관리자가 정확한 권한 배정을 할 수 있

다면 본 논문에서 제시한 권한 배정 제약조건의 필요성이 축소될 수도 있다. 그러나 권한 배정 제약조건 없이 보안 관리자 판단에만 의존한 권한 배정은 상당히 위험하다. 보안 관리자의 실수도 있을 수 있고, 한 명 이상의 보안 관리자가 존재하거나, 많은 역할과 권한이 존재하는 경우 이 모든 것을 완벽하게 관리하는 것은 매우 어려운 일이다. 따라서, 본 논문에서 제시하는 제약 조건들로 권한 배정의 주요한 원칙을 정의함으로써 보안 관리자의 실수를 최소화하고, 잘못된 권한 배정의 오류를 사전에 방지해야 한다.

5. 비교 평가

표 1은 제안하는 권한 배정 제약조건과 기존의 권한 배정 제약조건을 비교하였다.

DP 제약조건에 대해서 기존의 모델에서는 임무분리를 기반으로 하는 권한 배정 기본적인 개념은 언급하였지만 역할의 계층구조에 의해서 발생하는 권한공유와 공유되는 권한을 고려한 제약조건의 적용범위에 대해서는 언급하지 못했다. 제시된 모델에서는 정리 1에서 제약조건이 적용되는 권한의 범위 지정함으로 공유되는 권한과 제한받는 권한을 구분하였다. 이러한 구분은 역할의 계층구조상에서 권한의 공유를 허용하면서도 배정을 제한해야하는 권한은 제한함으로 역할의 계층구조상에서 유효한 권한 제한을 할 수 있게 한다.

PP 제약조건에 대해서 기존모델에서는 선행권한의 개념은 언급하였지만 AND/OR 선행관계, 연속적인 선행관계의 표현 등에 대해서는 언급하지 못했다. 제시된 모델에서는 권한에 대한 하나 이상의 선행 관계에서 AND/OR의 개념을 도입함으로 선행권한의 개념을 발전시켰다. 그리고, 선행관계를 표현하는데 사용된 AND/OR 그래프는 동적인 권한들 사이의 복잡한 선행관계를 정확하게 표현함으로 선행관계를 효과적으로 관리할 수 있게 해준다.

표 1 권한 배정 제약조건의 비교

제약조건		RBAC96	RCL2000	제안 모델
DP	기본개념	○	-	○
	역할의 계층구조	×	-	○
	제약조건 적용 범위 설정	×	-	○
PP	기본개념	○	-	○
	AND/OR 선행관계	×	-	○
	연속적인 선행관계의 표현방법	×	-	○
CP	기본개념	-	○	○
	임무분리	-	×	○
	권한통합	-	×	○
PASR	기본개념	-	-	○
	역할의 계층구조	-	-	○

CP 제약조건에 대해서 기존모델은 권한충돌의 개념을 언급했지만 충돌권한과 임무분리, 권한통합 등의 개념은 고려하지 못했다. 정리2에서 제안 모델에서는 권한충돌과 임무분리와의 관계를 정의하여 권한 통합이나 사용자 배정으로 인한 권한 배정의 오류를 사전에 방지할 수 있게 하였다.

PASR 제약조건은 기존의 모델에서는 언급되지 않았다. PASR 제약조건은 업무 특성상 또는 보안 관리상의 이유로 주요 권한의 공유를 통제함으로 권한 배정의 오류를 사전에 막는다.

본 논문에서 제안된 제약조건은 기존의 제약조건에 비해서 복잡해졌음으로 직관적으로 이해 하기가 힘들다. 따라서 보안관리자가 도구의 지원없이 제약조건을 작성, 검증하기 어렵다. 도구는 새롭게 작성된 제약조건과 기존의 제약조건 사이의 무결성을 검증해야 하며, 또한 제약조건과 권한 배정 사이의 무결성도 검증 해야 한다.

6. 결 론

본 논문에서 제시한 대칭 RBAC 모델의 권한 배정 제약조건들은 기존 연구에서 제시한 권한 배정 제약조건들을 보완하여 NIST의 대칭 RBAC 모델을 구체화한다. 제안한 모델에서는 임무분리와 역할의 계층구조를 고려한 확장된 권한 배정 제약조건을 제시하고있다. 이 제약조건은 역할간의 이해관계 충돌과 권한의 공유와 통합을 권한배정에 반영함으로써 적절한 권한배정을 가능하게 한다. 또한, 제안한 모델은 AND/OR 그래프를 이용하여 권한간의 선행관계를 규정하는 제약조건을 표현하였으며 제약조건은 권한의 선행관계에 'AND'와 'OR'의 개념을 추가하여 복잡한 선행관계를 효과적으로 제한할 수 있게 한다. 제시된 대칭 RBAC 모델의 권한 배정 제약조건들은 권한 배정에서 지켜야 하는 규칙을 명세하여 권한배정의 오류를 감소하는 데 공헌한다.

향후 연구로는 권한부여 정책과 제약조건 정책간의 충돌 그리고 제약조건 정책간의 충돌에 대한 연구가 필요하다.

참 고 문 헌

[1] Ravi S. Sandhu, Edward J. Coynek, Hal L. Feinsteink, Charles E. Youmank, Role-Based Access Control Models, IEEE Computer, Volume 29, Number 2, February 1996, pages 38-47.

[2] Sylvia Osborn, Ravi Sandhu, Qamar Munawer, Configuring Role-Based Access Control to Enforce Mandatory and Discretionary Access Control Policies, ACM Transactions on Information and System Security, Vol. 3, No. 2, May 2000, Pages 85 - 106.
[3] David F. Ferraiolo, Dennis M. Gilbert, and Nickilyn Lynch. An examination of federal and commercial access control policy needs. In NIST-NCSC National Computer Security Conference, pages 107-116, Baltimore, MD, September 20-23 1993.
[4] Ravi Sandhu, David Ferraiolo, Richard Kuhn, The NIST Model for Role-Base Access Control: Toward A Unified Standard, Proceedings, 5th ACM Workshop on Role Based Access Control, July 26-27, 2000.
[5] David F. Ferraiolo, Ravi Sandhu, Serban Gavrila, Proposed NIST Standard for Role-Based, Access Control, ACM Transactions on Information and System Security, Vol. 4, No. 3, August 2001, pages 224-274.
[6] Gail-Joon Ahn, Ravi Sandhu, Role-Based Authorization Constraints Specification, ACM Transactions on Information and System Security, Vol. 3, No. 4, November 2000, pages 207-226.
[7] Joon S. Park, Ravi Sandhu, Role-Based Access Control on the Web, ACM Transactions on Information and System Security, Vol. 4, No. 1, February 2001, pages 37-71.
[8] Elisa Bertino, Elena Ferrari, Vijay Atluri, The specification and enforcement of authorization constraints in workflow management systems, ACM Transactions on Information and System Security, Vol. 2, No. 1, February 1999, pages 65-104.
[9] David F. Ferraiolo, John F. Barkley, and D. Richard Kuhn, A Role-Based Access Control Model and Reference Implementation Within a Corporate Intranet, ACM Transactions on Information and System Security, Vol. 2, No. 1, February 1999, pages 34-64.
[10] Collin Atkinson et al., Component-based Product Line Engineering with UML, ADDISON WESLEY, 2002, page 67-69.
[11] Sun Microsystems, Enterprise JavaBeansTM Specification Version 2.1, ch4 Enterprise Beans as components.
[12] D.R. Kuhn, Mutual Exclusion of Roles as a Means of Implementing Separation of Duty in Role-Based Access Control Systems, Second ACM Workshop on Role-Based Access Control, 1997.
[13] Gligor, V.D., S.I. Gavrila, and D. Ferraiolo. On the formal definition of separation-of-duty policies and their composition. IEEE Symposium on Security and Privacy, May 1998, Oakland, California.
[14] You-Feng Lin, Shwu-Yeng T. Lin, Set Theory : An Intuitive Approach, kyung moon, 1999, ch2.
[15] Gail-Joon Ahn, Michael. E. Shin, Role-based Authorization Constraints Specification Using Object Constraint Language, In Proceedings of 6th IEEE International Workshop on Enterprise Security (WETICE 2001), MIT, MA, June 20-22, 2001.
[16] A. Mahanti, A. Bagchi, AND/OR Graph Heuristic Search Methods, Journal of the Association for Computing Machinery, Vol. 32, No. 1, January 1985, page 28-51.
[17] George F Luger, Artificial Intelligence, Addison Wesley, 2002, page 109-121.

문 창 주

1997년 고려대학교 컴퓨터학과 학사 1999년 고려대학교 일반대학원 컴퓨터학과 석사. 2000년~현재 고려대학교 일반대학원 컴퓨터학과 박사과정. 관심분야는 컴포넌트, 인증, 승인, 보안공학, RBAC

박 대 하

1992년 고려대학교 컴퓨터학과 학사 1994년 고려대학교 일반대학원 컴퓨터학과 석사. 1996년 고려대학교 일반대학원 컴퓨터학과 박사과정 수료. 1999년~현재 (주)시큐리티테크놀로지스 책임연구원 관심분야는 XML 보안, 보안 프로토콜, 이동코드 보안, 임베디드 시스템 보안

박 성 진

1991년 고려대학교 전산학(학사). 1993년 고려대학교 일반대학원 전산학(석사) 1998년 고려대학교 일반대학원 전산학(박사). 1998년~2000년 한국전자통신연구원 선임연구원. 2000년~현재 한신대학교 정보시스템공학과 조교수. 관심분야는 Database, XML Respository, OLAP, Mining

백 두 권

1974년 고려대학교 수학과(학사). 1977년 고려대학교 대학원 산업공학과(석사) 1983년 Wayne State Univ. 전산학과(석사). 1985년 Wayne State Univ. 전산학과(박사). 1986년~현재 고려대학교 컴퓨터학과 교수. 1989년~현재 한국 정보과학회 평의원/이사. 1991년~현재 한국 시뮬레이션 학회 이사/부회장. 1991년~현재 ISO/IEC JTC1/SC32 국내위원회 위원장. 1999년~현재 정보통신진흥협회 데이타 기술위원회 의장. 2002년~현재 고려대학교 정보통신대학 학장. 관심분야는 데이타베이스, 소프트웨어 공학, 데이타 공학, 컴포넌트 기반 시스템, 메타데이타 레지스트리, 정보 통합