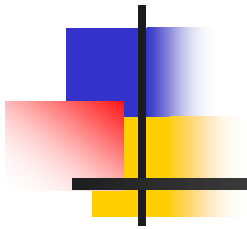
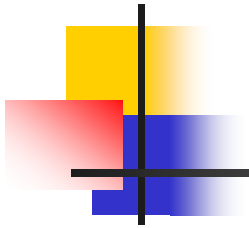


Ein Rollen- und Aufgabenbasiertes Sicherheitsmodell

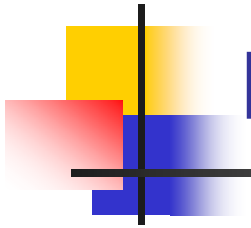
Humboldt-Universität zu Berlin
Andrey Bogdanov
14.01.2004



Inhalt

Begriffe(1).

- Subjekt=(*wer* etwas ausführt),
- Rolle=(*wie* | *in welchem Kontext* muss die Aufgabe erledigt werden),
- Aufgabe=(*was* das Subjekt erledigen möchte) $\leftrightarrow$Ziel;
  (Ziel, Rolle)$\leftrightarrow$Handlungsmuster={Handlungsschritt $_i$};
- Prozedur=(*womit* | *mit welchem Mittel* die Aufgabe erledigt wird)$\in$ Handlungsschritt,
- Objekt=(Daten, auf die die Prozeduren zugreifen) $\in$ Handlungsschritt,
- Handlungsmuster=(Ablauf einer Aufgabe in einer Rolle) $\leftrightarrow$(Aufgabe, Rolle),
- Handlungsschritt=(Prozedur, Objekt).



Begriffe(2).

Bsp.

- Subjekt=(eine natürliche oder juristische Person),
- Rolle=(Geldbörsenbesitzer),
- Aufgabe=(Bezahlen), als Geldbörsenbesitzer,
- Prozedur={Lesen, Schreiben, Löschen},
- Objekt=(Guthabenbestand der Geldbörse)...

Bemerkungen.

Rollen und Aufgaben können bezüglich eines Subjekts sein:

- autorisiert = das Subjekt kann die Rolle (oder die Aufgabe) prinzipiell auswählen,
- aktuell = das Subjekt hat die Rolle (oder die Aufgabe) schon gewählt.

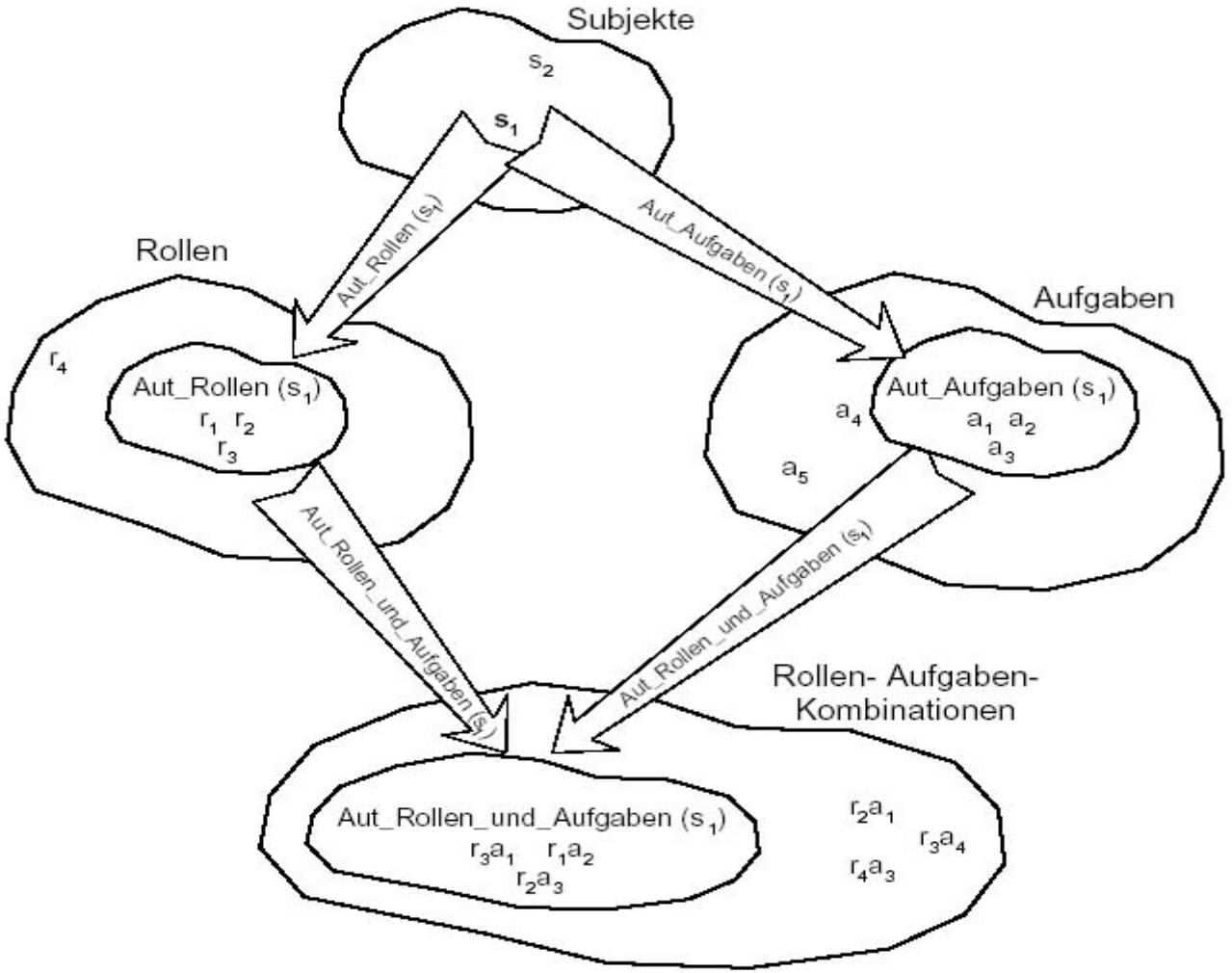
Unterschied von RBSM.

- Im RBSM ist die Rolle nur eine Menge von Transaktionen, die auf Objekte zugreifen. Die Zuordnung von Rollen zu Transaktionen ist statisch.

Formale Definition. Zustandsvariablen und Funktionen(1).

- Zustandsvariablen:
  - Subjekte,
  - Aufgaben,
  - Rollen,
  - Prozeduren,
  - Objekte.
- Jede Variable wird durch eine Menge von Werten dargestellt. Der Zugriff auf diese Variablen erfolgt über spezielle Funktionen. Die Funktionen operieren auf die Mengen unter bestimmten Bedingungen und geben ihre Teilmengen aus.

Formale Definition. Zustandsvariablen und Funktionen(2).



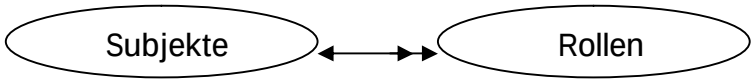
Formale Definition. Zustandsvariablen und Funktionen(3).

- *AutRollen(s_i)*={die Menge der für das Subjekt s_i autorisierten Rollen}$\subseteq$ R
- *AutAufgaben(s_i)*={die Menge der für das Subjekt s_i autorisierten Aufgaben} $\subseteq$ A
- *AutRollenUndAufgaben(s_i)*={die Menge der für das Subjekt s_i autorisierten Rollen-Aufgaben-Kombinationen}$\subseteq$ R$\times$A

Formale Definition. Zustandsvariablen und Funktionen(4).

- *Subjekte*={s_1, s_2, …, s_n}
- *Rollen*={r_1, r_2,…,r_m}

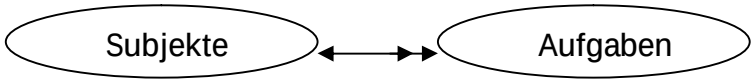
$$\bigcup_{i=1...m} AutRollen(s_i) \subseteq Rollen$$



Eins-zu-N-Beziehung

- AktRollen(s_i) ⊆AutRollen(s_i)⊆Rollen
- *Aufgaben*={a_1, a_2,…,a_k}

$$\bigcup_{i=1...k} AutAufgaben(s_i) \subseteq Aufgaben$$



Eins-zu-N-Beziehung

- AktAufgaben(s_i)⊆AutAufgaben(s_i)⊆Aufgaben

Formale Definition. Zustandsvariablen und Funktionen(5).

- *Rollen-Aufgaben-Kombinationen=Rollen´Aufgaben*
- Um nach Wahl einer Rolle eine Aufgabe zu bestimmen (oder umgekehrt), muss diese Kombination autorisiert sein.
- *AutRollenUndAufgaben(s_i) Í AutRollen(s_i)´ AutAufgaben(s_i) Í Rollen´Aufgaben*
- Diese Beziehungen können durch eine Matrix dargestellt werden.
- Der Eintrag an der Stelle r_i, a_j bedeutet, dass die Aufgabe a_i für die Rolle r_i autorisiert ist.
- Die Autorisierung ist positiv definiert.
- Eine solche Matrix existiert für jedes Subjekt.

AutRollen(s_i) / AutAufgaben(s_i)	(r_1)	(r_2)	(r_3)	(r_m)
(a_1)	✓	✓	✓	✓
(a_2)	✓			
(a_3)		✓		✓
(a_4)	✓			✓
(a_k)		✓	✓	

Formale Definition. Zustandsvariablen und Funktionen(6).

- *AktRollenUndAufgaben(s_i) Í*
  AktRollen(s_i)´AktAufgaben(s_i) Í
  Rollen´Aufgaben

Formale Definition. Autorisierung. Statische Trennung von Pflichten.(1)

- Ein Subjekt kann mehrere Rollen haben, falls sich diese Rollen nicht gegenseitig ausschließen.
  - *AusschlRollen*(r_j)={alle Rollen, die sich mit r_j ausschließen}$\subseteq$Rollen
- Äquivalent für Aufgaben:
  - *AusschlAufgaben*(a_m)={alle Aufgaben, die sich mit a_m ausschließen}$\subseteq$Aufgaben
- und für Rollen-Aufgaben-Kombinationen:
  - *AusschlRollenUndAufgaben*(r_j, a_m)={alle Kombinationen, die sich mit (r_j, a_m) ausschließen}$\subseteq$Rollen$\times$Aufgaben.

Formale Definition. Ausführung. Dynamische Trennung von Pflichten.(1)

- Der Fall entspricht dem gleichzeitigen Agieren in mehreren Rollen.
- Wenn sich zwei oder mehr aktuelle Rollen nicht gegenseitig ausschließen, kann das Subjekt s_i gleichzeitig in diesen Rollen agieren.
  - *AusschlAktRollen(r_j)*={alle Rollen, die sich mit der Rolle r_j gegenseitig ausschließen (unabhängig von einer noch zu wählenden Aufgabe)} $\subseteq$ AutRollen(r_j) $\subseteq$Rollen
- Äquivalent für Aufgaben:
  - *AusschlAktAufgaben(a_m)*={alle Aufgaben, die sich mit der Aufgabe a_m gegenseitig ausschließen (unabhängig von einer noch zu wählenden Rolle)} $\subseteq$ AutAufgaben(a_m) $\subseteq$Aufgaben
- Äquivalent für Rollen-Aufgaben-Kombinationen:
  - *AusschlAktRollenUndAufgaben(r_j, a_m)*={alle RA-Kombinationen, die sich mit dem angegebenen Paar für ein Subjekt in diesem Zeitpunkt ausschließen }$\subseteq$Rollen$\times$Aufgaben.
- Dabei ist es unerheblich, in welcher Reihenfolge die Rollen und Aufgaben ausgewählt werden.

Formale Definition. Handlungsschritte: Prozeduren, Objekte, Zugriffe.(1)

- *Prozeduren*$=\{p_1, p_2, \ldots, p_p\}$.
  - Greifen auf Datenobjekte in kontrollierter Weise zu.
- *Objekte*$=\{o_1, o_2, \ldots, o_q\}$.
- *Zugriffe*$=\{[(p_p,o_q)]_1, [(p_r,o_s)]_2, \ldots, [(p_t,o_u)]_v\}$.
  - *AutZugriff*$(s_i,r_j,a_m)=[(p_p,o_q)]=$\{eine unleere Liste von Prozeduren und zugehörigen Objekten, über die den autorisierten Zugriff eines Subjekts in einer Rolle und mit einer Aufgabe beschrieben wird\}.
  - Die ganze Liste $[(p_p,o_q)] \leftrightarrow$ ein Handlungsmuster.
  - Eine Elemente der Liste $(p_p,o_q) \leftrightarrow$ ein Handlungsschritt.
  - *ErlaubterZugriff*$(s_i, r_j, a_m)=$\{wahr, wenn der Zugriff für das Objekt s_i in der Rolle r_j mit der Aufgabe a_m erlaubt ist\}.

Formale Definition. Konsistenzregeln.(1)

- *1)*Rollenautorisierung
  - *" s_i Î Subjekte:*

    AktRollen(s_i) Í AutRollen(s_i)
- *2)*Aufgabenautorisierung
  - *" s_i Î Subjekte:*

    AktAufgaben(s_i) Í AutAufgaben(s_i)
- *3)*Rollen- und Aufgabenautorisierung
  - *" s_i Î Subjekte:*

    AktRollenUndAufgaben(s_i) Í AutRollenUndAufgaben(s_i)

Formale Definition. Konsistenzregeln.(2)

- *4)* Statische Trennung von Pflichten
  - a) für Rollen

    $\forall s_i \in Subjekt, r_j, r_k \in Rollen, j \neq k:$

    $(r_j \in AutRollen(s_i) \wedge r_k \in AutRollen(s_i)) \Rightarrow r_j \notin AusschlRollen(r_k).$
  - b) für Aufgaben

    $\forall s_i \in Subjekt, a_m, a_n \in Aufgaben, n \neq m:$

    $(a_m \in AutAufgaben(s_i) \wedge a_n \in AutAufgaben(s_i)) \Rightarrow a_m \notin AusschlAufgaben(a_n).$
  - c) für RA-Kombinationen

    $\forall s_i \in Subjekt, a_m, a_n \in Aufgaben, r_j, r_k \in Rollen, (r_j, a_m) \neq (r_k, a_n):$

    $((r_j, a_m) \in AutRollenUndAufgaben(s_i) \wedge (r_k, a_n) \in AutRollenUndAufgaben(s_i)) \Rightarrow (r_j, a_m) \notin AusschlRollenAufgaben(r_k, a_n).$

Formale Definition. Konsistenzregeln.(3)

- 5)Dynamische Trennung von Pflichten
  - a)für Rollen

    " s_i Î Subjekt, r_j,r_k Î Rollen, j ¹ k:

    (r_j Î AktRollen(s_i) Ç r_k Î AktRollen(s_i)) Þ

    r_j Ï AusschAktlRollen(r_k).
  - b)für Aufgaben

    " s_i Î Subjekt, a_m,a_n Î Aufgaben, n ¹ m:

    (a_m Î AktAufgaben(s_i) Ç a_n Î AktAufgaben(s_i)) Þ

    a_m Ï AusschlAktAufgaben(a_n).
  - c)für RA-Kombinationen

    " s_i Î Subjekt, a_m,a_n Î Aufgaben, r_j,r_k Î Rollen, (r_j,a_m) ¹ (r_k,a_n):

    ((r_j,a_m) Î AktRollenUndAufgaben(s_i) Ç
    (r_k,a_n) Î AktRollenUndAufgaben(s_i)) Þ

    (r_j,a_m) Ï AusschlAktRollenAufgaben(r_k,a_n).

  Ein Subjekt darf keine Rollen-Aufgaben-Kombinationen mehrfach zur gleichen Zeit ausführen. Jede RA-Kombination schließt sich mit selbst aus.

Formale Definition. Konsistenzregeln.(4)

- *6)*Zugriffserlaubnis

  " s_i Î Subjekt, a_m Î Aufgaben, r_j Î Rollen:

  (ErlaubterZugriff(s_i, r_j, a_m)==wahr) Þ

  - *((r_j, a_m) Î AktRollenUndAufgaben(s_i)*

    Ç

  - *p_p Î Prozeduren, o_q Î Objekte:*

    [(p_p, o_q)] Î AutZugriff(s_i, r_j, a_m)).

Formale Definition. Überführungsfunktionen.(1)

- Die Überführungsfunktionen überführen von einem Zustand in den nächsten, wobei die entsprechenden Regeln (Konsistenzregeln) in jedem Zustand erfüllt sein müssen.
- *1)***WähleAufgabe(s_i, a_m)**
  - IF $a_m \in$ AutAufgaben(s_i) AND { AktAufgaben(s_i) $= \emptyset$ OR $\forall\ a_n \in$ AktAufgaben(s_i): $a_n \notin$ AusschlAktAufgaben(a_m) }
  - THEN AktAufgaben(s_i) $\leftarrow$ AktAufgaben(s_i) $\cup \{a_m\}$
  - ELSE *Fehlerbehandlung*

Formale Definition. Überführungsfunktionen.(2)

- *2)* **WähleRolleNachAufgabe(s_i, r_j, a_m)**
  - IF (r_j, a_m) $\in$ AutRollenUndAufgaben(s_i)
    AND
    {
    AktRollenUndAufgaben(s_i) = $\varnothing$
    OR
    $\forall$ (r_k, a_n) $\in$ AktRollenUndAufgaben(s_i), $j \neq k$:
    (r_k, a_n) $\notin$ AusschlAktRollenUndAufgaben(r_j, a_m)
    }
  - THEN AktRollen(s_i) $\leftarrow$ AktRollen(s_i) $\cup$ {r_j}

    AktRollenUndAufgaben(s_i) $\leftarrow$
    AktRollenUndAufgaben(s_i) $\cup$ {(r_j, a_m)}
  - ELSE *Fehlerbehandlung*

Formale Definition. Überführungsfunktionen.(3)

- *3)* **WähleRolle(s_i, r_j)**
  - IF $r_j \in$ AutAufgaben(s_i)
    AND
    {
    AktRollen(s_i) = $\emptyset$
    OR
    $\forall\ r_k \in$ AktRollen(s_i):
    $r_k \notin$ AusschlAktRollen(r_j)
    }
  - THEN AktRollen(s_i) $\leftarrow$ AktRollen(s_i) $\cup \{r_j\}$
  - ELSE *Fehlerbehandlung*

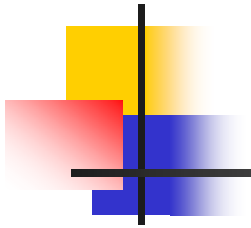
Formale Definition. Überführungsfunktionen.(4)

- *4)* **WähleAufgabeNachRolle(s_i, r_j, a_m)**
  - IF (r_j, a_m) $\in$ AutRollenUndAufgaben(s_i)
    AND
    {
    AktRollenUndAufgaben(s_i) = $\varnothing$
    OR
    $\forall$ (r_k, a_n) $\in$ AktRollenUndAufgaben(s_i), $j \neq k$:
    (r_k, a_n) $\notin$ AusschlAktRollenUndAufgaben(r_j, a_m)
    }
  - THEN AktAufgaben(s_i) $\leftarrow$ AktAufgaben(s_i) $\cup$ {a_m}
    AktRollenUndAufgaben(s_i) $\leftarrow$
    AktRollenUndAufgaben(s_i) $\cup$ {(r_j, a_m)}
  - ELSE *Fehlerbehandlung*

Formale Definition. Überführungsfunktionen.(5)

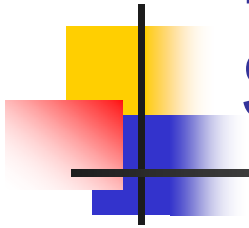
- *5)***FührеAus(s_i,r_j,a_m)**
  - IF ErlaubterZugriff(s_i,r_j,a_m)
  - THEN *Ausführen*
    AktRollenUndAufgaben(s_i)←
    AktRollenUndAufgaben(s_i)\{(r_j,a_m)}
    AktRollen(s_i)←AktRollen(s_i)\{r_j}
    AktAufgaben(s_i)←AktAufgaben(s_i)\{a_m}
  - ELSE *Fehlerbehandlung*
- Die Funktion führt ein spezielles Handlungsmuster aus, das für das Subjekt in der RA-Kombination festgelegt ist. Falls ein Subjekt mehrere aktuelle RA-Kombinationen ausgewählt hat, wird die Überführungsfunktion für jede einzelne RA-Kombination ausgeführt.

Formale Definition. Die Endlichkeitseigenschaft des Modells.(1)



- Das RA-Modell ist endlich. Die Endlichkeit ist dadurch gegeben, dass:
  - nur eine begrenzte Anzahl von Rollen (*n*) und eine begrenzte Anzahl von Aufgaben (*m*) existiert. Daraus ergibt sich, dass es maximal (*n*×*m)* RA-Kombinationen geben kann,
  - eine mehrfache Auswahl derselben RA-Kombination nicht erlaubt ist.

Beweis.
Skizze und Annahme.(1)

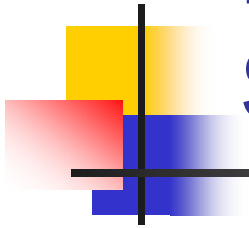


- Es kann gezeigt werden, dass die Überführungsfunktionen von einem gültigen Zustand (mit erfüllten Konsistenzregeln) nur in einen wieder gültigen Zustand überführen. Ein Zustand ist dann gültig, wenn die Konsistenzregeln 1-6 erfüllt sind.
- Der Beweis wird nach dem Prinzip der vollständigen Induktion geführt. Dies erfolgt in drei Schritten:
  - *Induktionsanfang:*
    - Zu zeigen ist, dass der Anfangszustand ein gültiger Zustand ist.
  - *Induktionsannahme:*
    - Annahme, dass ein Zustand z_s ein gültiger Zustand ist.
  - *Induktionsschritt:*
    - Zu zeigen ist, dass der Nachfolgezustand z_{s+1} ein gültiger Zustand ist.

Beweis.
Skizze und Annahme.(2)

- Die Menge der Zustände wird folgendermaßen beschrieben:
  - Zustände={z_1, ..., z_s}.
- Ein Zustand z_i wird durch alle Zustandsvariablen beschrieben:
  - z_i={ Subjekte, Rollen, Aufgaben,
    AutRollen, AktRollen,
    AutAufgaben, AktAufgaben,
    AutRollenUndAufgaben, AktRollenUndAufgaben,
    AusschlAufgaben, AusschlRollen,
    AusschlRollenUndAufgaben,
    AusschlAktAufgaben, AusschlAktRollen,
    AusschlAktRollenUndAufgaben,
    Prozeduren, Objekte,
    Zugriffe, AutZugriff, ErlaubterZugriff}

Beweis.
Skizze und Annahme.(3)



- Für den Anfangszustand z_0 gilt:
  - $\forall s_i \in$ Subjekte:
    - AktRollen(s_i)=$\varnothing$
    - AktAufgaben(s_i)=$\varnothing$
    - AktRollenUndAufgaben(s_i)= $\varnothing$
  - Alle anderen Mengen sind nicht leer.

Beweis. Induktionsanfang.(4)

- Es ist zu zeigen, dass der Anfangszustand z_0 ein gültiger Zustand ist.
  - *KR1. Rollenautorisierung.*
    - $\forall s_i \in$ Subjekte:
      $AktRollen(s_i) \subseteq AutRollen(s_i)$,
      weil $AktRollen(s_i) = \varnothing$
  - *KR2. Aufgabenautorisierung.*
    - $\forall s_i \in$ Subjekte:
      $AktAufgaben(s_i) \subseteq AutAufgaben(s_i)$,
      weil $AktAufgaben(s_i) = \varnothing$.
  - *KR3. Rollen- und Aufgabenautorisierung.*
    - $\forall s_i \in$ Subjekte:
      $AktRollenUndAufgaben(s_i) \subseteq AutRollenUndAufgaben(s_i)$,
      weil $AktRollenUndAufgaben(s_i) = \varnothing$.

Beweis. Induktionsanfang.(5)

- *KR4. Statische Trennung von Pflichten.*
  - a)für Rollen

$\forall s_i \in$ Subjekt, $r_j, r_k \in$ Rollen, $j \neq k$:

$(r_j \in \text{AutRollen}(s_i) \cap r_k \in \text{AutRollen}(s_i)) \Rightarrow r_j \notin \text{AusschlRollen}(r_k)$

per definitionem der Funktion AusschlRollen(r);

  - b)für Aufgaben

$\forall s_i \in$ Subjekt, $a_m, a_n \in$ Aufgaben, $n \neq m$:

$(a_m \in \text{AutAufgaben}(s_i) \cap a_n \in \text{AutAufgaben}(s_i)) \Rightarrow$
$a_m \notin \text{AusschlAufgaben}(a_n)$

per definitionem der Funktion AusschlAufgaben(a);

  - c)für RA-Kombinationen

$\forall s_i \in$ Subjekt, $a_m, a_n \in$ Aufgaben, $r_j, r_k \in$ Rollen,
$(r_j, a_m) \neq (r_k, a_n)$:

$((r_j, a_m) \in \text{AutRollenUndAufgaben}(s_i) \cap$
$(r_k, a_n) \in \text{AutRollenUndAufgaben}(s_i))$
$\Rightarrow (r_j, a_m) \notin \text{AusschlRollenUndAufgaben}(r_k, a_n)$

per definitionem der Funktion AusschlRollenUndAufgaben(r,a).

Beweis. Induktionsanfang.(6)

- *KR5. Dynamische Trennung von Pflichten.*
  - a)für Rollen

$\forall s_i \in$ Subjekt, $r_j, r_k \in$ Rollen, $j \neq k$:

$(r_j \in \text{AktRollen}(s_i) \cap r_k \in \text{AktRollen}(s_i)) \Rightarrow$
$r_j \notin \text{AusschAktRollen}(r_k)$,

weil $\text{AktRollen}(s_i) = \varnothing$;

  - b)für Aufgaben

$\forall s_i \in$ Subjekt, $a_m, a_n \in$ Aufgaben, $n \neq m$:

$(a_m \in \text{AktAufgaben}(s_i) \cap a_n \in \text{AktAufgaben}(s_i)) \Rightarrow$
$a_m \notin \text{AusschlAktAufgaben}(a_n)$,

weil $\text{AktAufgaben}(s_i) = \varnothing$;

  - c)für RA-Kombinationen

$\forall s_i \in$ Subjekt, $a_m, a_n \in$ Aufgaben, $r_j, r_k \in$ Rollen, $(r_j, a_m) \neq (r_k, a_n)$:

$((r_j, a_m) \in \text{AktRollenUndAufgaben}(s_i) \cap$
$(r_k, a_n) \in \text{AktRollenUndAufgaben}(s_i)) \Rightarrow$
$(r_j, a_m) \notin \text{AusschlAktRollenAufgaben}(r_k, a_n)$,

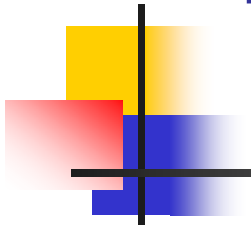
weil $\text{AktRollenUndAufgaben}(s_i) = \varnothing$.

Beweis. Induktionsanfang.(7)

- *KR6. Zugriffserlaubnis.*

  $\forall s_i \in$ Subjekt, $a_m \in$ Aufgaben, $r_j \in$ Rollen:
  (ErlaubterZugriff(s_i, r_j, a_m)==wahr) $\Rightarrow$
  ((r_j, a_m) $\in$ AktRollenUndAufgaben(s_i)
  $\cap$
  $\exists p_p \in$ Prozeduren, $o_q \in$ Objekte:
  [(p_p, o_q)] $\in$ AutZugriff(s_i, r_j, a_m));
  da AktRollenUndAufgaben(s_i)=$\varnothing$, ist
  ErlaubterZugriff=falsch, d.h., der Zugriff ist nicht erlaubt.

- Damit ist der Anfangszustand z_0 im Sinne der sechs KRn gültig.



Beweis. Induktionsannahme.(8)

- Annahme, dass der Zustand z_s im Sinne der sechs KRn gültig ist.

Beweis. Induktionsschritt.(9)

Es ist zu zeigen, dass der Folgezustand z_{s+1} im Sinne der sechs KRn gültig ist.

- *1)Hilfsmittel.*

  Eine allgemeine Überführungsfunktion lässt sich jetzt definieren:

  $$\text{ÜF} : \text{Zustände} \times \text{Operationen} \times 2^{\text{Parameter}} \rightarrow \textit{Zustände},$$

  wobei:

  - die Menge Operationen besteht aus den Überführungsfunktionen

    (WähleRolle, WähleAufgabeNachRolle, WähleAufgabe, WähleRolleNachAufgabe, FührеAus),

    Operationen={op_1, op_2, ..., op_x},
  - die Menge Parameter besteht aus den Eingabeparametern, die für die jeweiligen Überführungsfunktionen erforderlich sind:

    Parameter={$para_1$, $para_2$, ..., $para_y$}.

Beweis. Induktionsschritt.(10)

- *2)Annahme.*

  $\forall z_s \in$ Zustände, $op_x \in$ Operationen, $para_w$, ..., $para_y \in$ Parameter:

  $z_{s+1} = ÜF(z_s, op_x, para_w, ..., para_y)$.

  Seien alle der Operation op_x entsprechenden Vorbedingungen bezüglich der Parameter erfüllt.

  Dann ist es zu zeigen, dass der Zustand z_{s+1} unter diesen Annahmen gültig ist.

- *3)Induktionsschritt (anhand einer Überführungsfunktion).*

  Betrachte die Überführungsfunktion **WähleRolle**.

Parameter:	s_i, r_j
Aktion:	$AktRollen(s_i) \leftarrow AktRollen(s_i) \cup \{r_j\}$
Vorbedingung 1:	$r_j \in AutRollen(s_i)$
Vorbedingung 2:	$(AktRollen(s_i) = \emptyset) \cup$
	$(\forall r_k \in AktRollen(s_i): r_k \notin AusschlAktRollen(r_j))$

Beweis. Induktionsschritt.(11)

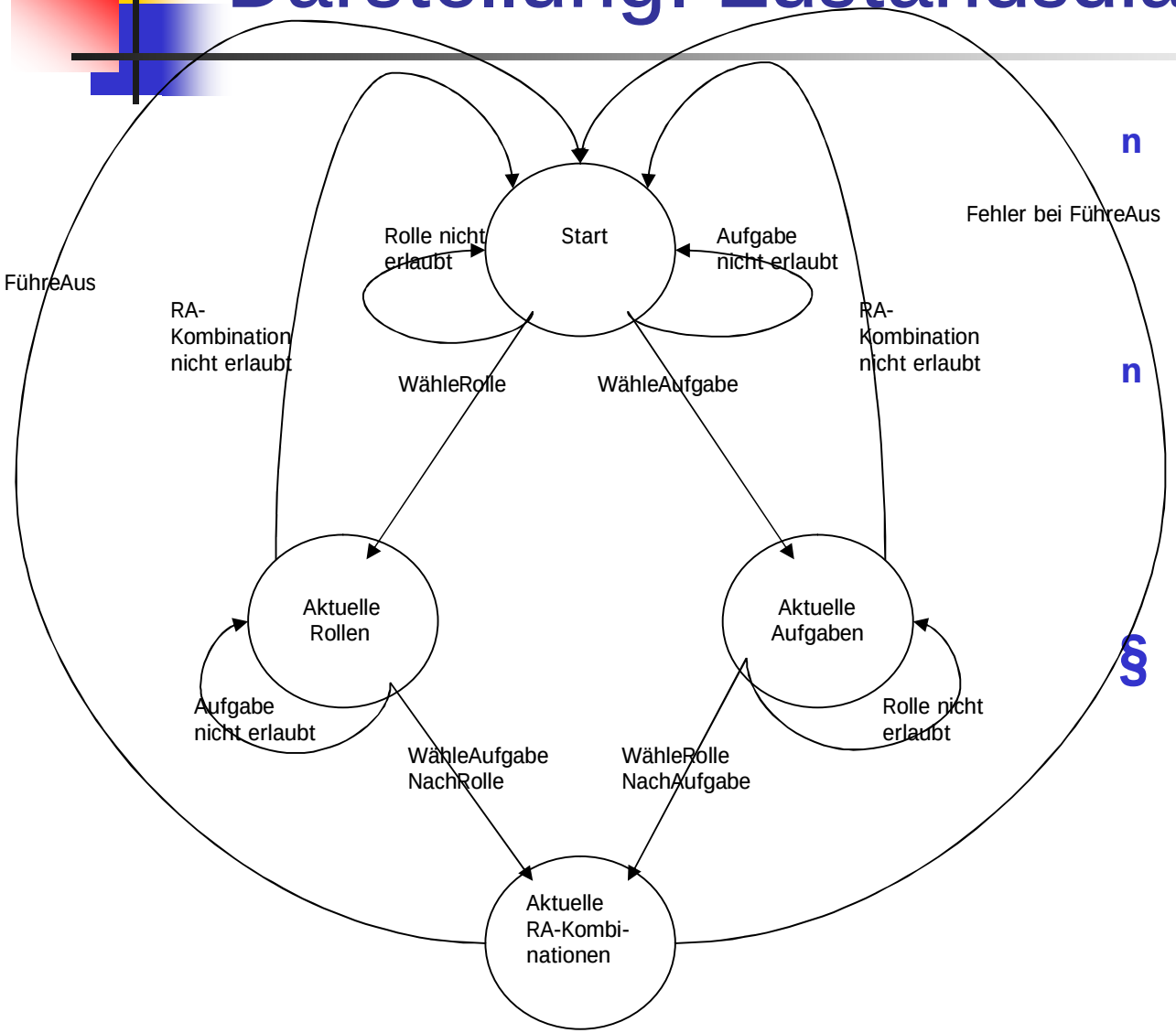
- Da Konsistenzregeln 2,3,4,5 und 6 in keiner Weise von der Überführungsfunktion WähleRolle abhängen, wird nur die Konsistenzregel 1 betrachtet:

  $AktRollen(s_i) \subseteq AutRollen(s_i)$

- Zwei Fälle können hier unterschieden werden:
  - bisher hat das Subjekt keine aktuelle Rollen: $AktRollen(s_i)=\emptyset$,
  - es hat bereits aktuelle Rollen.
- Dann gilt:
  - $r_j \in AutRollen(s_i)$ (nach der Vorbedingung); da es vor dem Vorgang gilt, dass $AktRollen(s_i) =\emptyset$, und die Aktion die Vereinigung von $AktRollen(s_i)$ und r_j ist, gilt: $\{ r_j\}=AktRollen(s_i) \subseteq AutRollen(s_i)$;
  - da die Konsistenzregeln im Zustand z_s gelten, gilt in diesem Zustand:

    $AktRollen(s_i) \subseteq AutRollen(s_i)$.
- Dann gilt es im Zustand z_{s+1} $AktRollen(s_i) \subseteq AutRollen(s_i)$, weil $r_j \in AutRollen(s_i)$ nach der 1sten Vorbedingung.
- D.h., der Zustand z_{s+1} ist nach der Ausführung der Überführungsfunktion *WähleRolle* gültig. Der Beweis für die restlichen Überführungsfunktionen ist äquivalent.□

Darstellung. Zustandsdiagramm.



- Beginnt mit dem Startzustand (das Subjekt bereits authentisiert).
- Der Startzustand= der Endzustand (nach der Ausführung eines Handlungsmusters)
- Das Modell lässt sich durch einen endlichen Automaten darstellen.

Anwendung des Modells in Chipkarten.(1)

- Auf der Basis des formalen RA-Modells werden RA-Anwendungen für eine multifunktionale Chipkarte (RA-Chipkarte) im elektronischen Zahlungsverkehr modelliert.
- In diesem Anwendungsbeispiel kann die Chipkarte an unterschiedlichen Orten eingesetzt werden. Diese können sein:
  - Ein SB-Terminal,
  - Der Kassenschalter einer Bank,
  - Der eigene Computer mit einer Netzverbindung zur eigenen Bank,
  - Ein bankenunabhängiges Zahlungsterminal (beispielweise in einem Geschäft).

Anwendung des Modells in Chipkarten.(2)

- Verwendete Symbole:

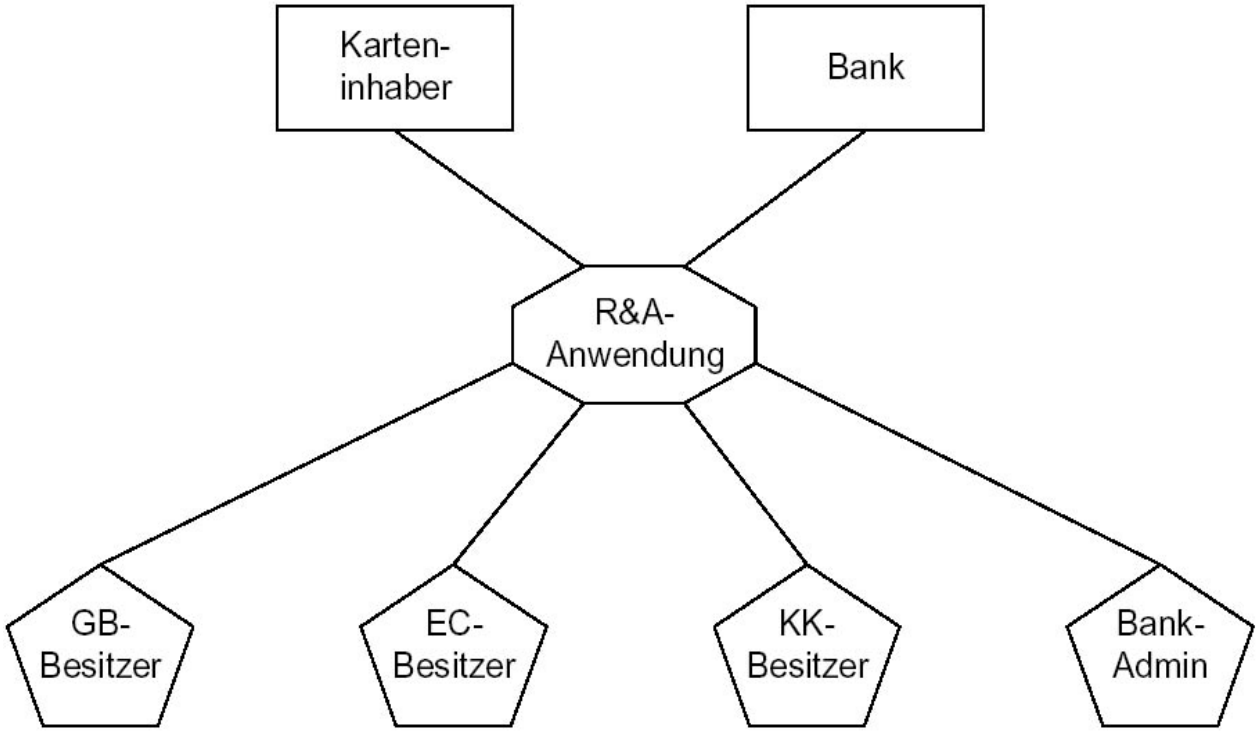
Anwendung des Modells in Chipkarten.(3)

- Jetzt sind Subjekte, Rollen und Aufgaben zu definieren:
  - *Subjekte*

Subjekte	Beschreibung
Karteninhaber	Inhaber beziehungsweise der Benutzer der Chipkarte.
Bank	Vertreter der kartenausgebenden Stelle (Bankangestellter).

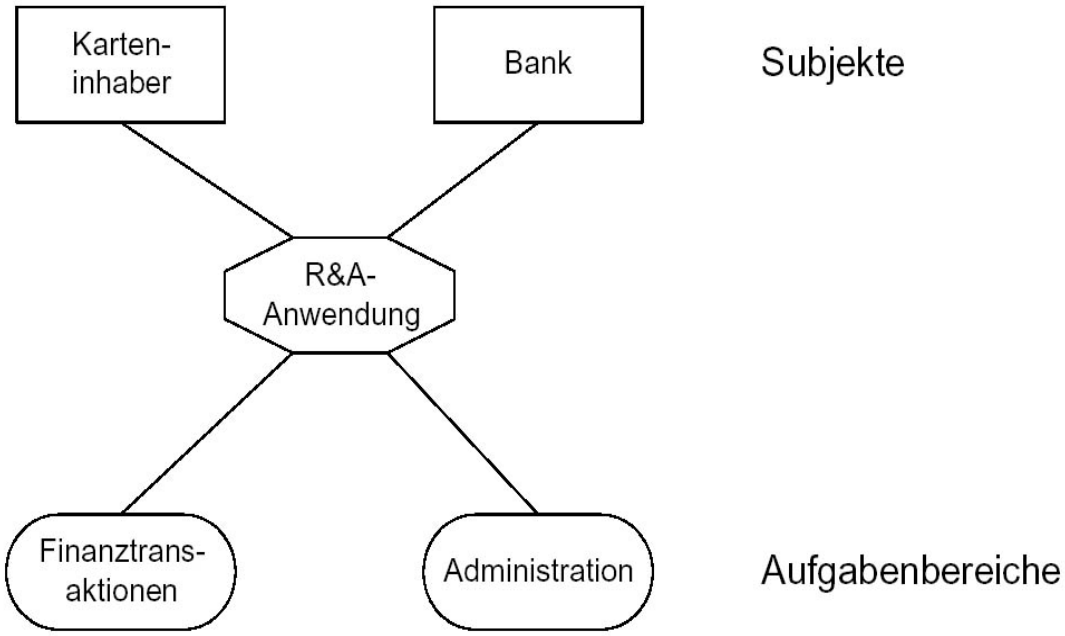
Anwendung des Modells in Chipkarten.(4)

- *Rollen*



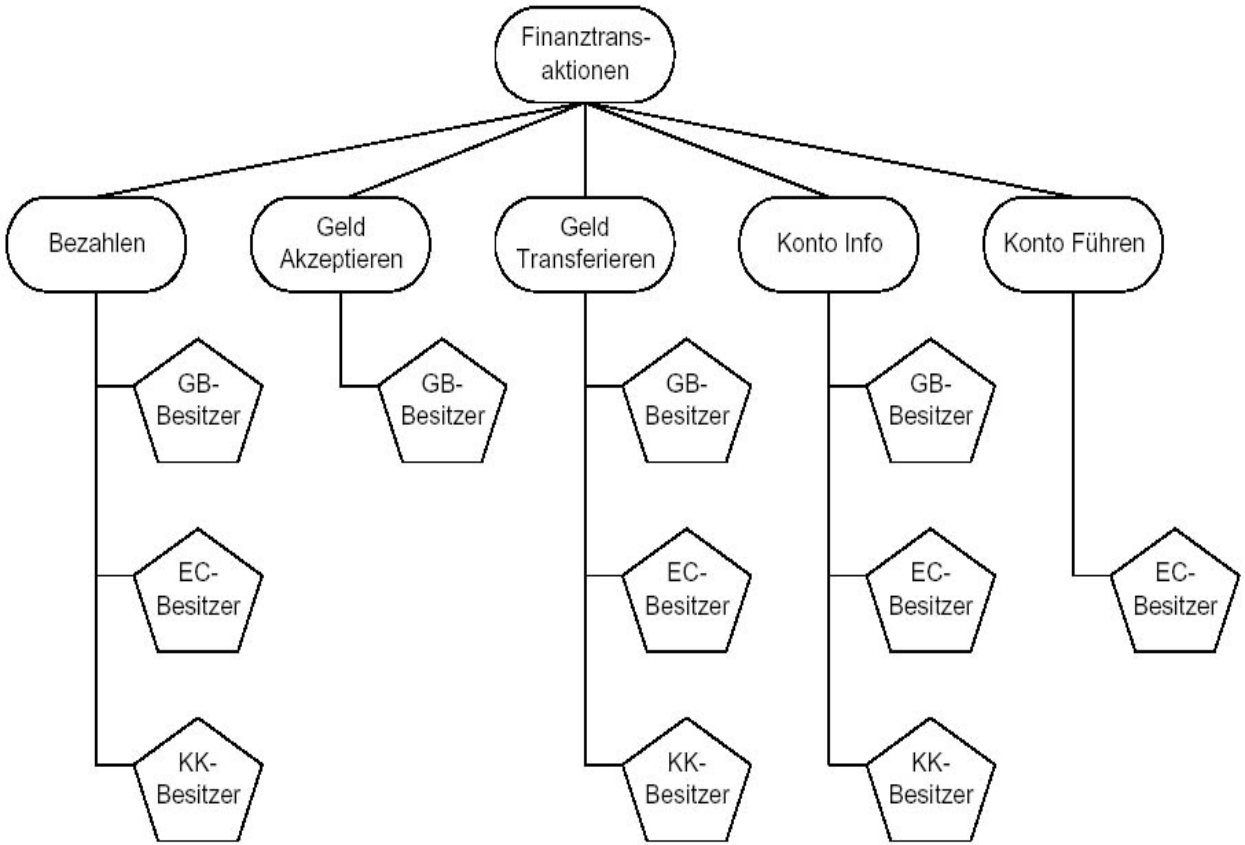
Anwendung des Modells in Chipkarten.(5)

- *Aufgaben*



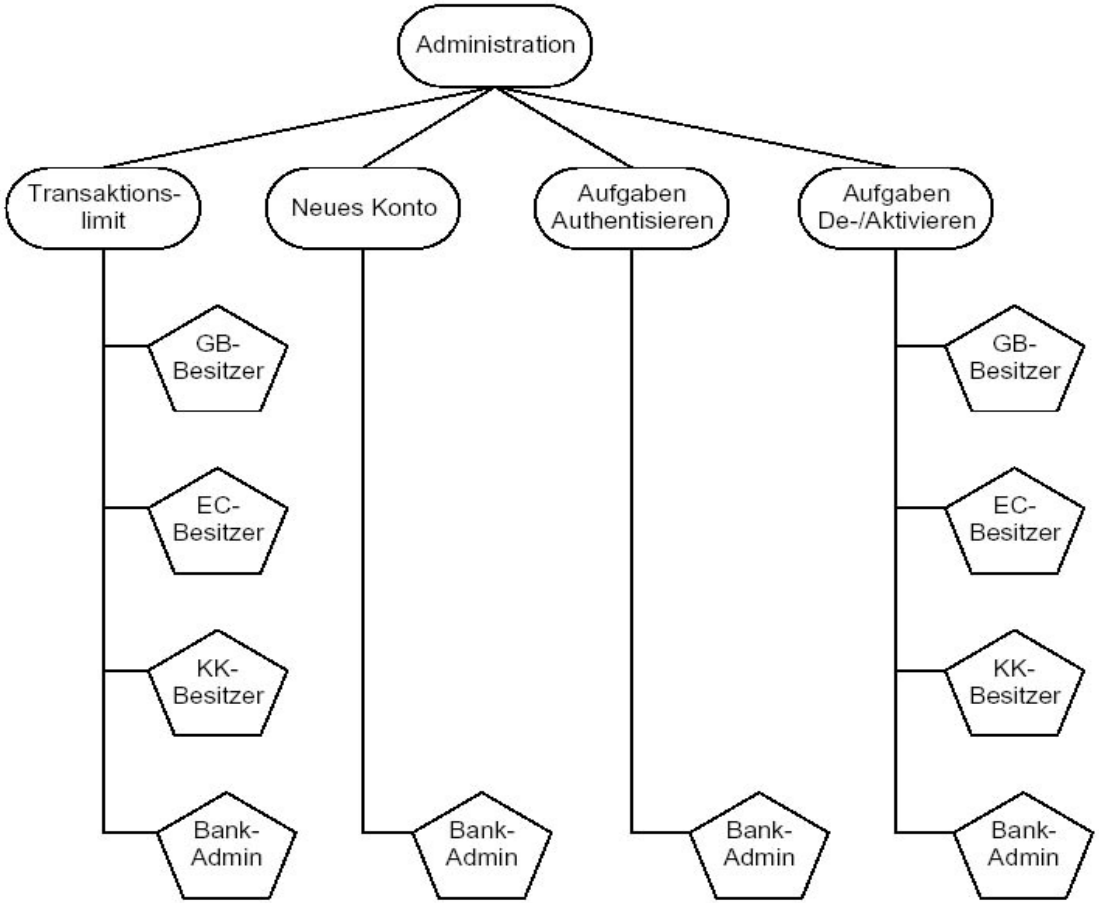
Anwendung des Modells in Chipkarten.(6)

- *Aufgaben*



Anwendung des Modells in Chipkarten.(7)

- *Aufgaben*



Anwendung des Modells in Chipkarten.(8)

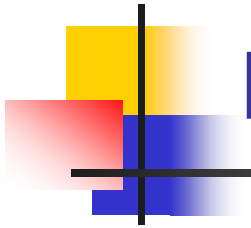
- *Autorisierte RA-Kombinationen für das Subjekt Karteninhaber.*

(s_1) Inhaber Rollen Aufgaben	(r_1) GB-Besitzer	(r_2) EC-Besitzer	(r_3) KK-Besitzer	(r_4) Bank-Admin
(a_1) Bezahlen	✓	✓	✓	
(a_2) Geld Akzeptieren	✓			
(a_3) Geld Transferieren	✓	✓	✓	
(a_4) Konto Info	✓	✓	✓	
(a_5) Konto Führen		✓	✓	
(a_6) Transaktionslimit	✓	✓	✓	
(a_7) Neues Konto				
(a_8) R&A Authent.				
(a_9) R&A De-/Aktiv.	✓	✓	✓	

Anwendung des Modells in Chipkarten.(9)

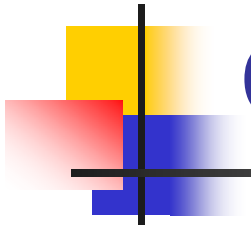
- *Autorisierte RA-Kombinationen für das Subjekt Bank.*

(s_2) Bank Rollen / Aufgaben	(r_1) GB-Besitzer	(r_2) EC-Besitzer	(r_3) KK-Besitzer	(r_4) Bank-Admin
(a_1) Bezahlen				
(a_2) Geld Akzeptieren				
(a_3) Geld Transferieren				
(a_4) Konto Info				
(a_5) Konto Führen				
(a_6) Transaktionslimit				✓
(a_7) Neues Konto				✓
(a_8) R&A Authent.				✓
(a_9) R&A De-/Aktiv.				✓



Bewertung und Zusammenfassung.

- Bewertung des RA-Modells:
  - Frei Wahl von Rollen und Aufgaben durch zweidimensionale Struktur,
  - Feine Granularität der Zugriffe,
  - Zugriff über wohl definierte Prozeduren auf Objekte,
  - Gleichzeitige Ausführung von konfliktfreien Anwendungen,
  - Statische und dynamische Trennung von Pflichten,
  - Mininisierung der notwendige Rechte,
  - Leichte Administration.



Quellen.

- Schier, Kathrin. Vertrauenswürdige Kommunikation im elektronischen Zahlungsverkehr. Ein formales Rollen- und Aufgabenbesiertes Sicherheitsmodell für Anwendung mit multifunktionalen Chipkarten. Dissertation Universität Hamburg, Juni 1999
- Eckert, Claudia: IT-Sicherheit – Konzepte, Verfahren, Protokolle. R. Oldenbourg Verlag, 2000