Objek-georiënteerde en rolgebaseerde verspreide inligtingsekerheid in 'n oop transaksieverwerking omgewing.

Jaco van der Merwe

OBJEK-GEORIËNTEERDE EN ROLGEBASEERDE VERSPREIDE INLIGTINGSEKERHEID IN 'N OOP TRANSAKSIEVERWERKING OMGEWING.

deur

JACOBUS VAN DER MERWE

91 2261 8

VERHANDELING

voorgelê ter vervulling van die vereistes vir die graad

MAGISTER IN DIE NATUURWETENSKAPPE

in

REKENAARWETENSKAP

in die

FAKULTEIT NATUURWETENSKAPPE

aan die

RANDSE AFRIKAANSE UNIVERSITEIT

STUDIELEIER:	PROF. S.H. VON SOLMS
MEDESTUDIELEIER:	DR. M.S. OLIVIER

OKTOBER 1995

Aan Jonanda - baie dankie vir jou volgehoue belangstelling en ondersteuning in hierdie verhandeling. Aan my ouers - dankie vir julle motivering.

My opregte dank aan prof. Basie von Solms en Dr. Martin Olivier vir hulle leiding waarsonder hierdie verhandeling nie voltooi kon word nie.

"Aan die Here kom die lof toe, ... op Hom het ek vertrou. Hy het my gehelp. "
Psalm 28:6-7

Summary

Object-Oriented and role-based distributed information security in an open transaction processing environment.

Information is a valuable resource in any organisation and more and more organisations are realising this and want efficient means to protect it against disclosure, modification or destruction. Although relatively efficient security methods have been available almost as long as information databases, they all provide additional cost. This cost does not only involve money but also cost in terms of system performance and management of information security. Any new information security model must also provide better management of information security. In this dissertation we present a model that provides information security and aims to lower the technical skills required to manage information security using this approach.

In any business organisation we can describe each employee's duties. Put in other words, we can say that each employee has a specific business role in the organisation. In organisations with many employees there are typically many employees that have more or less the same duties in the organisation. This means that employees can be grouped according to their business roles. We use an employee's role as a description of his/her duties in a business organisation.

Each role needs resources to perform its duties in the organisation. In terms of computer systems, each role needs computer resources such as printers. Most roles need access to data files in the organisation's database but it is not desirable to give all roles access to all data files. It is obvious that roles have specific privileges and restrictions in terms of information resources.

Information security can be achieved by identifying the business roles in an organisation and giving these roles only the privileges needed to fulfill their business function and then assigning these roles to people (users of the organisation's computer system). This is called role-based security.

People's business functions are related, for example clerks and clerk-managers are related in the sense that a clerk-manager is a manager of clerks. Business roles are related in the same way. For an information security manager to assign roles to users it is important to see this relationship between roles. In this dissertation we present this relationship using a lattice graph which we call a role lattice. The main advantage of this is that it is eases information security management.

Creating new roles by using existing roles as templates or combining the privileges of existing roles also eases the management of information security. Object-oriented

design provides this by means of inheritance. We take advantage of this by implementing roles as objects.

The model for information security using distributed role profile objects (ORITO) is developed and formalised in the dissertation and this model and its advantages is described using a client/server environment. Security validation is done in a distributed manner using server computers. The model is described in the context of a transaction processing environment using transactions as the key information resource. The dissertation shows how the ORITO can be implemented in a distributed manner by distributing role profile objects between server computers.

The **main objectives of this dissertation are**:
To develop a model for distributed and object-oriented role-based information security in a transaction processing environment (ORITO). The model authorises access to information resources and eases the management of information security.
The model itself has the following advantages:

- It shows that role-based security can be implemented in an object-oriented manner. This is done by implementing role profiles as objects and creating new roles through inheritance. This eases the management of roles.
- It shows that role-based security can be distributed and used in a client/server environment. Distributed processing increases performance and reliability.
- The relationship between roles are presented with a role lattice graph and this graph is used to distributed roles between servers in a client/server environment. A role lattice eases the management and distribution of roles among servers.

Inhoudsopgawe

DEEL 1

DEEL 2

1. Inleiding en oorsig

1.1 Inleiding

Inligting is 'n kosbare hulpbron in enige organisasie. Meer en meer organisasies besef dat hulle besigheidsinligting van groot waarde is en wil hierdie inligting beskerm teen ongemagtigde bekendmaking, verandering of vernietiging. In hierdie verhandeling word inligtingsekerheid in 'n besigheidsorganisasie bestudeer.

Bestaande metodes om inligting in 'n besigheidsorganisasie te beskerm word bestudeer en voorstelle om hierdie metodes uit te brei en aan te pas vir die nuutse neigings in die rekenaarbedryf (soos objek-georiënteerdheid, klient/bediener en verspreide verwerking) word gegee.

Alhoewel daar verskeie metodes is om inligtingsekerheid in 'n organisasie te implementeer, bly die bestuur van inligtingsekerheid 'n probleem. Dikwels het die persone wat verwantwoordelik is vir die handhawing van inligtingsekerheid in die organisasie nie die nodige kundigheid nie [7]. In hierdie studie kyk ons hoe inligtingsekerheid geïmplementeer kan word op so 'n wyse dat die bestuur daarvan makliker is en gevolglik minder kundigheid vereis sover dit rekenaarsekerheid betref.

Klem word geplaas op rolgebaseerde inligtingsekerheid. In enige besigheidsorganisasie kan daar vir elke werknemer 'n beskrywing van sy/haar werkstake opgestel word. In ander woorde gestel: elke werknemer het 'n spesifieke besigheidsrol in die organisasie. 'n Groot organisasie het ook gewoonlik baie mense wat dieselfde tipe werkstaak verrig in die organisasie. Ons kan sê verskeie mense in die organisasie hoort tot dieselfde besigheidsrol. So kan ons nou na die besigheidsrol van werknemers verwys as ons hulle werkstaak wil beskryf. Die rolbenadering tot inligtingsekerheid het onder andere die voordeel dat dit die bestuur van inligtingsekerheid in 'n organisasie vergemaklik.

1.2 Doel van studie

In hierdie studie word 'n model ontwikkel wat die bestuur van inligtingsekerheid vergemaklik deur gebruik te maak van rolgebaseerde inligtingsekerheid en dit te kombineer met objek-georiënteerdheid en die voorstelling van rolverwantskappe met traliegrafieke.

Die model word later in die studie uitgebrei om die voordele van kliënt/bediener en verspreide omgewings te benut. Laasgenoemde maak die model 'n meer doeltreffende en meer betroubare model vir inligtingsekerheid.

Die studie word deurgaans gedoen uit die oogpunt van transaksieverwerking. Deur die model speisifiek vir transaksieverwerkers te beskryf is dit makliker om die werking van die model te beskryf en word die voordele wat die model bied meer sigbaar.

1.3 Nuwe begrippe in die studie

Alhoewel hoofstukke 2 tot 5 hoofsaaklik 'n agtergrondsbeskrywing is van bestaande begrippe in die rekenaarwetenskap literatuur, word daar nuwe konsepte ontwikkel en beskryf in hierdie hoofstukke. In hoofstukke 6 tot 9 word 'n nuwe model vir magtiging in 'n verspreide omgewing ontwikkel. Die model implementeer rolgebaseerde inligtingsekerheid op 'n objek-georiënteerde wyse.

Die belangrikste nuwe begrippe wat in hierdie verhandeling beskryf word is:

- Die gebruik van 'n rolprofiel om rolverwantskappe mee te stoor.
- Objek-georiënteerde sekerheidsprofiele en die skep van nuwe rolle deur oorerwing op rolobjekte toe te pas.
- Die gebruik van 'n roltralie om rolverwantskappe mee voor te stel en die bestuur van inligtingsekerheid te vergemaklik.
- Die model vir objek-georiënteerde en rolgebaseerde inligtingsekerheid in 'n transaksieverwerking omgewing (ORITO).
- Bostaande model in 'n verspreide omgewing ('n model vir magtiging in 'n verspreide omgewing soortgelyk aan Kerberos).
- Uitbreiding van ORITO vir meer doeltreffendheid en meer betroubaarheid.

1.4 Resultate bereik met die studie

Met die druk van hierdie verhandeling is die model vir objek-georiënteerde en rolgebaseerde inligtingsekerheid in 'n transaksieverwerking omgewing (ORITO) reeds aangebied deur die hoofouteur by 'n internasionale konferensie: *IT Sicherheit '95 Communications and Multimedia Security Conference*, September 1995.

'n Artikel, *Managing information security in a client/server environment with distributed, object-oriented role-based security*, is geskryf en gestuur na die IFIP/SEC '96 organiserings kommitee vir moontlike aanbieding by die konferensie in 1996. Die uitslag van die evaluering van die artikel word in Desember 1995 verwag.

Die model is gedeeltelik geïmplementeer as 'n eksterne sekerheidsmonitor vir die *CICS/6000* transaksieverwerker op 'n IBM RS/6000 rekenaar.

1.5 Oorsig

Die verhandeling bestaan uit twee hoofdele. Die eerste deel (hoodstukke 2 tot 5) gee 'n oorsig van terme en begrippe wat gebruik word in deel 2. Deel 1 vorm die agtergrond vir deel 2 van die verhandeling. In deel 2 (hoofstukke 6 tot 10) word die model vir objek-georiënteerde rolgebaseerde inligtingsekerheid in 'n transaksieverwerking omgewing (ORITO) geformuleer. Die model word uitgebrei na 'n model vir magtiging in verspreide en kliënt/bediener omgewings. Doeltreffendheid en betroubaarheid kan verhoog word met ORITO, hierdie en ander aspekte word ook in deel 2 bespreek.

Die inhoud van die hoofstukke en hoe hulle inmekaar pas word vervolgens beskryf.

1.5.1 DEEL 1

In **hoofstuk 2** word die rolbenadering tot inligtingsekerheid gedefinieer. Bestaande definisies word gegee en uitgebrei sodat ons rolgebaseerde inligtingsekerheid op 'n objek-georiënteerde wyse kan implementeer.

Rolle in 'n organisasie kan verband hou met mekaar. Nuwe rolle kan juis gevorm word deur eienskappe van ander rolle te gebruik. Hoofstuk 2 gee aandag aan die verwantskap tussen rolle.

Objek-georiënteerde programmering is 'n programmeringmetodiek wat toeneem in populariteit [17]. Een van die voordele van objek-georiënteerde programmering is dat dit 'n meer natuurlike wyse is om werklike wêreld situasies met 'n rekenaarprogram voor te stel [4, 17]. Om die voordele van objek-georiënteerdheid te benut toon ons hoe rolgebaseerde inligtingsekerheid op 'n objek-georiënteerde wyse geïmplementeer kan word.

Hoofstuk 3 gee 'n basiese beskrywing van objek-georiënteerdheid en toon hoe rolle as objekte gehanteer kan word. Die voordele hiervan word gegee en verskeie voorbeelde van rolgebaseerde sekerheidsmagtiging waar rolobjekte gebruik word, word bespreek.

Een van die belangrikste voordele van objek-georiënteerde rolle is dat nuwe rolle geskep kan word deur van oorerwing gebruik te maak. Dit beteken dat 'n nuwe rol geskep kan word deur twee ander rolle te kombineer. Die model wat in hierdie verhandeling geformuleer word maak intensief gebruik van oorerwing op rolle.

Soos reeds genoem kan rolle in 'n organisasie verband hou. So byvoorbeeld is dit moontlik dat 'n persoon wat die besigheidsrol van 'n Klerke_Bestuurder vervul toegang moet hê na al die stelselhulpbronne waarna 'n Klerk toegang het plus nog 'n paar ander. Ons kan sê dat die rol Klerke_Bestuurder meer voorregte het as 'n rol Klerk in 'n organisasie. Indien ons die verwantskap tussen rolle grafies kan voorstel is dit makliker om te besluit watter rolle om aan watter gebruikers toe te ken. Daar is verskeie maniere om die verwantskap tussen rolle grafies voor te stel. In hierdie verhandeling gebruik ons 'n spesiale grafiek, genaamd 'n traliegrafiek.

In **hoofstuk 4** gee ons 'n kort oorsig oor grafieke en definieer 'n traliegrafiek. Ons toon ook hoe 'n traliegrafiek gebruik kan word om die verwantskap tussen rolle voor te stel. Die voorstelling van rolle in 'n traliegrafiek word later in die verhandeling gebruik wanneer ons verspreide rolgebaseerde inligtingsekerheid verduidelik.

Hoofstuk 5 gee 'n kort oorsig oor transaksieverwerking. Hoofstuk 2 tot 5 gee die agtergrond vir hoofstuk 6. In hoofstuk 6 word 'n nuwe model vir magtiging in 'n verspreide omgewing geformuleer. Die terme in hoofstuk 5 en die voorafgaande hoofstukke word gebruik in die model wat in hoofstuk 6 geformuleer word.

1.5.2 DEEL 2

In **hoofstuk 6 word die model vir objek-georienteerde, rolgebaseerde inligtingsekerheid in 'n transaksieverwerker omgewing (ORITO)** geformuleer.

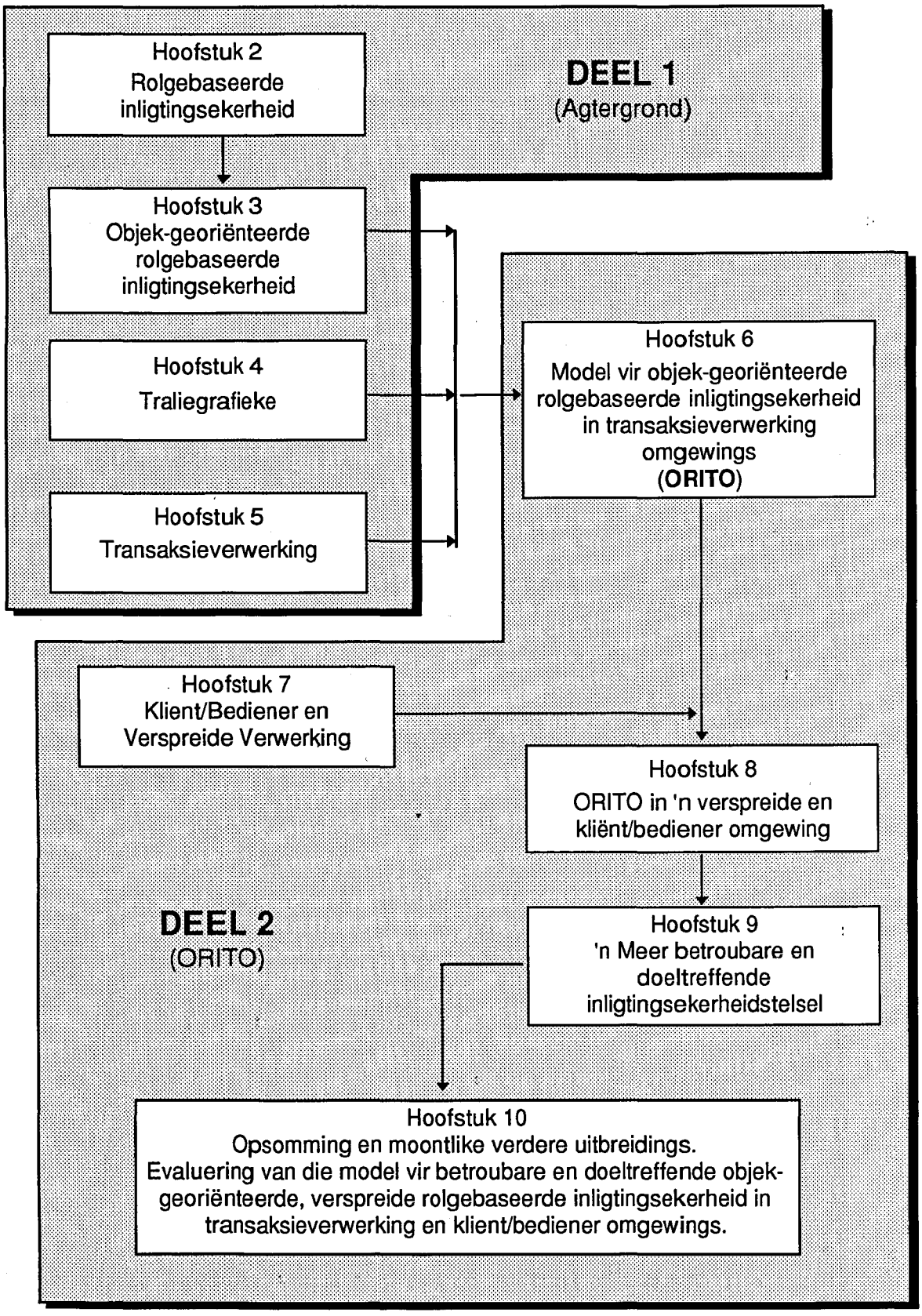
Die model gebruik die konsepte wat in die voorafgaande hoofstukke (Deel 1) verduidelik is om 'n model te formuleer wat die bestuur van inligtingsekerheid vergemaklik. Die voordele van die model asook 'n voorbeeld toepassing word gegee.

Al meer rekenaarstelsels neig om verspreide verwerking te doen. Verspreide verwerking en kliënt/bediener verwerking is nou verwant.
Hoofstuk 7 verduidelik wat verspreide verwerking en klient/bediener verwerking is.

Hoofstuk 8 gebruik die agtergrond wat in hoofstuk 7 gegee is om die model wat in hoofstuk 6 geformuleer is aan te pas vir 'n verspreide en klient/bediener omgewing. Om rolgebaseerde inligtingsekerheid verspreid te implementeer, versprei ons die rolobjekte wat in hoofstuk 3 gedefinieer is tussen bedieners in 'n klient/bediener omgewing. Die hoofstuk toon hoe die traliegrafiek, wat in hoofstuk 4 verduidelik is, gebruik kan word om dit makliker vir sekerheidsbestuurders te maak om te besluit hoe om rolle te versprei tussen bedieners.

In **hoofstuk 9** gee ons 'n kort bespreking van hoe die model uitgebrei kan word sodat dit 'n meer betroubare en meer doeltreffende inligtingsekerheidstelsel is. Ons wys hoe bedieners ander bedieners se werk tydelik kan oorneem wanneer een of meer bedieners buite werking raak.

Hoofstuk 10 sluit die verhandeling af met opsommende kommentaar en 'n bespreking van moontlike uitbreidings aan die model wat in hoofstuk 9 geformuleer is. 'n Evaluering van die verhandeling en die voordele van die studie word ook in hierdie hoofstuk bespreek. Figuur 1.1 toon die hoofstukke van hierdie verhandeling en toon hoe hulle inmekaar pas.



Figuur 1.1
Die hoofstukke van hierdie verhandeling.

DEEL 1

- Rolgebaseerde inligtingsekerheid
- Objek-georiënteerde rolgebaseerde inligtingsekerheid
- Traliegrafieke
- Transaksie verwerking

2. Rolgebaseerde inligtingsekerheid

2.1 Inleiding

In enige besigheidsorganisasie kan daar vir elke werknemer 'n beskrywing van sy/haar werkstake opgestel word. In ander woorde gestel: elke werknemer het 'n spesifieke *besigheidsrol* in die organisasie. 'n Groot organisasie het ook gewoonlik baie mense wat dieselfde tipe werkstaak verrig in die organisasie. Ons kan sê verskeie mense in die organisasie hoort tot dieselfde besigheidsrol. So kan ons nou na die besigheidsrol van werknemers verwys as ons hulle werkstaak wil beskryf.

Elke rol in 'n organisasie het sekere hulpmiddels nodig om sy take te kan verrig. In terme van rekenaarstelsels het elke rol ook sekere rekenaarbronne, byvoorbeeld drukkers nodig. Sekere rolle het ook toegang na konfidensiële datalêers in die organisasie se databasis nodig. Tog sal dit nie wenslik wees as alle rolle toegang het na die salarisse lêer nie. Rolle het dus sekere voorregte en sekere beperkings in terme van rekenaarbronne.

In 'n neutedop is die kern van rolgebaseerde inligtingsekerheid dat regte aan rolle toegeken word eerder as aan individuele gebruikers van 'n rekenaarstelsel. Gebruikers bekom dan hierdie regte deurdat hulle lidmaatskap van toepaslike rolle verkry. Hierdie eenvoudige idee maak die bestuur van toegang na hulpbronne aansienlik makliker. Die basiese idee agter rolgebaseerde inligtingsekerheid bestaan al van die gebruik van multi-gebruiker inligtingstelsels in die laat 1960s [15].

In hierdie verhandeling word rolgebaseerde inligtingsekerheid bestudeer en later gebruik in die model (ORITO) wat in hoofstuk 6 geformuleer word. Hierdie hoofstuk gee 'n oorsig van rolgebaseerde inligtingsekerheid. Let op dat sekere konsepte in hierdie hoofstuk nuwe idees is wat ingevoer word om die formulering van ORITO moontlik te maak.

2.2 Basiese definisie van rolle

Om rolgebaseerde inligtingsekerheid te beskryf, is dit nodig om eerstens 'n rol formeel te definieer. Ons gee in hierdie afdeling aandag aan die beskrywig en definiëring van die terme wat ons gebruik wanneer ons rolgebaseerde inligtingsekerheid beskryf. Rolgebaseerde inligtingsekerheid is nie 'n nuwe idee nie [2, 5, 7, 15], maar dit is nodig om van die bestaande definisies aan te pas of uit te brei sodat ons later in hierdie verhandeling 'n model vir verspreide en objek-georiënteerde rolgebaseerde inligtingsekerheid kan formuleer. Voordat ons die terme formeel definieer gee ons eers 'n kort beskrywing en voorbeeld van rolgebaseerde inligtingsekerheid.

'n Rol is 'n beskrywing van 'n spesifieke werkstaak in 'n organisasie. Afhangende van die tipe organisase kan 'n rol 'n beskrywing wees van 'n posisie van 'n werknemer in die organisasie; dit kan egter ook die take van al die persone in 'n kantoor van 'n

organisasie beskryf of bloot net die versameling aksies wat 'n houer van 'n sekere rol in 'n organisasie mag uitvoer [5]. 'n Rol beskryf dus wat gedoen kan en mag word deur 'n houer van rol, *ongeag wie die persoon* is aan wie die rol toegeken is [7].

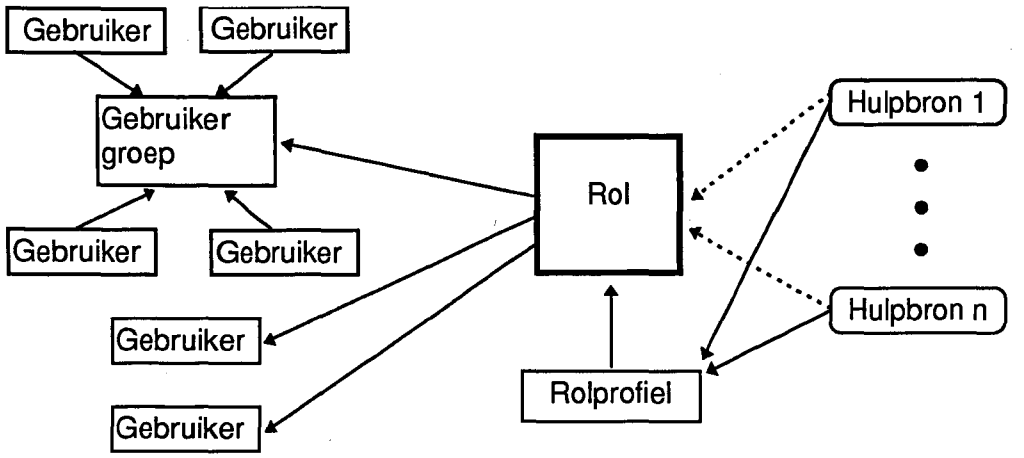
Volgens [5] kan *gebruikergroepe* ook aan 'n rol gekoppel word. In so 'n geval is daar tipies bestaande gebruiker groepe in 'n stelsel en die groep word aan 'n spesifieke rol gekoppel. Dit sal beteken dat al die gebruikers in dieselfde groep dieselfde toegang na bronne in die stelsel het. Gebruikergroepe kan ook handig gebruik word waar daar in 'n organisasie 'n aantal gebruikers is vir wie daar nie 'n duidelike werksbeskrywing is nie. Daar kan byvoorbeeld 'n gebruikergroep vir alle studente, wat vakansiewerk doen, in die organisasie wees; hierdie studente-groep kan dan byvoorbeeld aan die rol vir die Laagste_vlak_gebruiker gekoppel word.

Om gebruikergroepe aan rolle te koppel maak dit makliker om 'n rolgebaseerde sekerheidstelsel met bestaande bedryfstelsels soos Unix te integreer wat reeds van gebruikergroepe gebruik maak. Gebruikergroepe is dus nie noodsaaklik vir rolgebaseerde inligtingsekerheid nie, maar deur vir gebruikergroepe voorsiening te maak, is rolgebaseerde sekerheid meer versoenbaar met bestaande bedryfstelsels, soos Unix se ingeboude sekerheid.

In [2] word 'n *rolprofiel* gedefinieer as die versameling van alle bronne in 'n organisasie se rekenaarnetwerk wat nodig is of potensieel nodig kan wees om die take te verrig wat die rol uitmaak. Let op dat 'n rolprofiel egter ook die beperkings bevat waaronder 'n rol toegang mag hê na 'n bron, bv. 'n spesifieke datalêer is slegs toeganklik vir die rol in gewone werksure, nie andersins nie. 'n Rolprofiel is dus 'n spesifikasie van 'n rol wat die verantwoordelikhede, regte en pligte van 'n besigheidsrol beskryf. 'n Rolprofiel vir 'n klerk rol mag byvoorbeeld spesifiseer dat gebruikers wat gekoppel is aan die klerk rol 'n transaksie mag uitvoer wat fondse oordra van een rekening na 'n ander, mits die bedrag nie R5000 oorskry nie.

In hierdie verhandeling brei ons die definisie vir 'n rolprofiel volgens [2] uit. Ons vereis dat 'n rol en 'n rolprofiel as twee aparte datastrukture beskou moet word. 'n Rol is 'n datastruktuur wat beskryf hoe die rol met ander rolle verwant is (word later in meer besonderhede beskryf) en 'n rolprofiel is 'n datastruktuur wat beskryf watter bronne toeganklik is vir gebruikers wat aan so 'n rol gekoppel is en ook onder watter voorwaardes toegang tot die bron geldig of ongeldig is. Elke rol het dus presies een rolprofiel wat die rol beskryf.

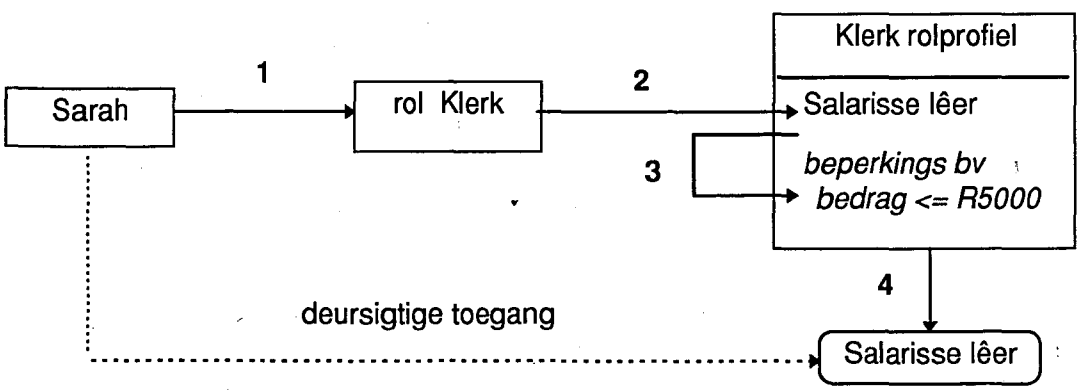
Figuur 2.1 toon bogenoemde benadering tot rolgebaseerde inligtingsekerheid. Gebruikers is gekoppel aan 'n spesifieke rol (direk of via 'n gebruikergroep). Die rol se beskrywing is in die rolprofiel wat gekoppel is aan die rol. Sekere hulpbronne in die stelsel is indirek (via die rolprofiel) gekoppel aan die rol. Die rolprofiel bevat ook die beperkings waaronder die gebruikers via die rol toegang tot die hulpbronne wat aan die rol gekoppel is kan kry.



Figuur 2.1.
'n Rolgebaseerde benadering tot sekerheid

2.2.1 Voorbeeld van magtiging in bostaande stelsel:

Veronderstel gebruiker Sarah is gekoppel aan die rol Klerk en sy wil 'n dataveld in die lêer Salarisse wysig. Die stelsel sal eerstens toets aan watter rol Sarah tans gekoppel is. Nadat die stelsel weet dat Sarah tans as 'n Klerk aangeteken is, sal getoets word of die rol Klerk toegang het tot die Salarisse lêer deur na die rol se rolprofiel te kyk. Sarah sal dus slegs toegelaat word om die bewerking uit te voer mits rol Klerk toegang het tot die Salarisse lêer **en** mits die rolprofiel nie 'n verdere beperking bevat nie. 'n Verdere beperking kan wees dat rol Teller slegs leesregte en nie ook skryfregte na die bron het nie - in so 'n geval sal Sarah se versoek geweier word. Figuur 2.2 toon die voorbeeld diagrammaties.



Figuur 2.2
Voorbeeld van magtiging met rolgebaseerde inligtingsekerheid.

In stap 1 in figuur 2.2 word bepaal dat Sarah tans as rol Klerk aangeteken is. Let op dat Sarah by aantekening op die stelsel moet aandui dat sy as rol Klerk wil aanteken (mits sy gemagtig is om as 'n Klerk aan te teken), die besonderhede hiervan word later gegee. By stap 2 word in die Klerk_rolprofiel gekyk of rol Klerk toegang het na die lêer Salarisse en in stap 3 word getoets of die omstandighede waaronder die versoek gemaak is geldig is, bv. is die bedrag kleiner R5000. Indien bogenoemde slaag, word

in stap 4 toegang verleen na die hulpbron Salarisse lêer. Al hierdie toetsings word deursigting vir gebruiker Sarah gedoen, vir haar lyk dit of sy direk toegang kry tot die Salarisse lêer soos wat die stippelpyl aandui.

Die skeiding van die konvensionele begrip van 'n rol in 'n rol en rolprofiel word later in die verhandeling gemotiveer wanneer die nodige agtergrond verskaf is. Die terme rol en rolprofiel word vervolgens meer formeel gedefinieer en daar word gewys op notasie wat gebruik word. Na elke definisie volg 'n informele beskrywing van die definisie.

2.2.2 Formele definisie van 'n rol

Ons gebruik die definisie soos gegee in [5] en pas dit aan om in te pas by die model wat later in hierdie verhandeling geformuleer word . Onthou dat 'n rol 'n rolprofiel het wat die rol se regte en beperkings beskryf. 'n Rolprofiel word gedefinieer in terme van voorregte en 'n voorreg word gedefinieer in terme van toegangsmetodes.

Definisie 2.1: *'n Voorreg is 'n paar (x, m) waar x verwys na 'n beskermde data item en m is 'n nie-leë versameling van toegangsmetodes vir objek x.*

x is enige beskermde data item soos byvoorbeeld 'n datalêer of enige hulpbron (soos 'n netwerk of drukker). *x* kan selfs 'n transaksie, bv. DEPONEER, ONTTREK, ens. in 'n transaksieverwerking omgewing wees. *x* is 'n naam of identifikasie wat die data item uniek spesifiseer in die stelsel.

Die toegangsmetodes *m* vir 'n beskermde data item beskryf die tipe toegang wat na 'n objek verleen word. Deur van toegangsmetodes gebruik te maak het ons 'n fyner benadering tot inligtingsekerheid deurdat ons nie sommer net toegang na 'n hulpbron verleen nie, maar die tipe toegang ook kan beperk na byvoorbeeld slegs lees en nie ook skryf nie. Tabel 2.1 toon 'n voorbeeld van drie voorregte met verskillende tipes objekte en hulle ooreenstemmende moontlike toegangsmetodes. In 'n stelsel met eenvoudige toegangsmetodes soos lees, skryf, uitvoer ens. is *m* 'n deelversameling van hierdie toegangsmetodes. *x* kan byvoorbeeld 'n datalêer soos DATA.TXT wees soos in voorreg 1 en *m* kan spesifiseer dat lees en skryf toegang na DATA.TXT verleen word. Waar *x* 'n objek in 'n objek-georiënteerde omgewing is soos in voorreg 2, kan *m* die uitvoerregte van een of meer lidfunksies (of metodes) van 'n objek-georiënteerde objek wees. *x* kan byvoorbeeld 'n objek VOERTUIG met verskeie attribute en lidfunksies wees maar toegang word slegs verleen na die lidfunksies Massa, Topsnelheid en Versnelling. In 'n transaksieverwerking omgewing, waar *x* 'n transaksie in die stelsel is (voorreg 3), kan *m* 'n lys van voorwaardes (bv. 'n beperking op die tye van die dag) wees waaronder *x* uitgevoer mag word. Volgens die voorbeeld in tabel 2.1 sal 'n gebruiker die transaksie DEPONEER kan uitvoer mits die bedrag tussen 0 en 5000 is en die transaksie vanaf 'n terminaal versoek is met id = TM1 of id = TM3 en mits die tyd van die dag tussen 9 uur en 6 uur is.

Voorreg	Data item (x)	Toegangsmetodes (m)
1	lêer DATA.TXT	lees, skryf.
2	OO objek VOERTUIG	lidfunksies Massa, Topsnelheid, Versnelling
3	transaksie DEPONEER	voorwaardes 0 <= bedrag <= 5000, Terminaal id ∈ {TM1, TM3 } 9:00 <= tyd <= 18:00

Tabel 2.1
Voorbeeld van verskillende tipes objekte en toegangsmetodes in 'n voorreg.

Let op dat die presiese inhoud van x en m afhang van die tipe toepassing sowel as die sekerheidsbeleid van 'n organisasie. In hoofstuk 6 formuleer ons 'n model waar transaksies die objek x in 'n voorreg is.

Definisie 2.2: *'n Rolprofiel is 'n paar (rp_naam, v_lys). v_lys is 'n versameling voorregte soos in definisie 2.1 gedefinieer is, rp_naam is 'n naam wat die rolprofiel uniek in die stelsel identifiseer.*

'n Rolprofiel het 'n naam en 'n lys van voorregte waar elke voorreg toegang gee na 'n objek in die stelsel onder sekere beperkings wat in die voorreg gespesifiseer is. Tabel 2.2 toon 'n voorbeeld van 'n Klerk_rolprofiel waar transaksies die objekte in die stelsel is wat beskerm moet word. Let op dat die rolprofiel dit moontlik maak vir gebruikers wat aan die rol Klerk gekoppel is om een van drie transaksies uit te voer. Elkeen van die transaksies in die rolprofiel se voorregtelys het toegangsmetodes of in ander woorde gestel: *toelaatbare omstandighede.*

rolprofiel Klerk		
Voorreg	Data item (x)	Toegansmetodes (m)
1.	transaksie DEPONEER	Terminaal_ID ∈ { TM1, TM2, TM3, TM4 }
2.	transaksie ONTTREK	Terminaal_ID ∈ { TM1, TM2 } en 0 <= bedrag <= 5000
3.	transaksie OORPLAAS	Terminaal_ID ∈ { TM1, TM2 } en 0 <= bedrag <= 10000 en 9:00 <= tyd <= 17:00

Tabel 2.2
Voorbeeld van 'n rolprofiel.

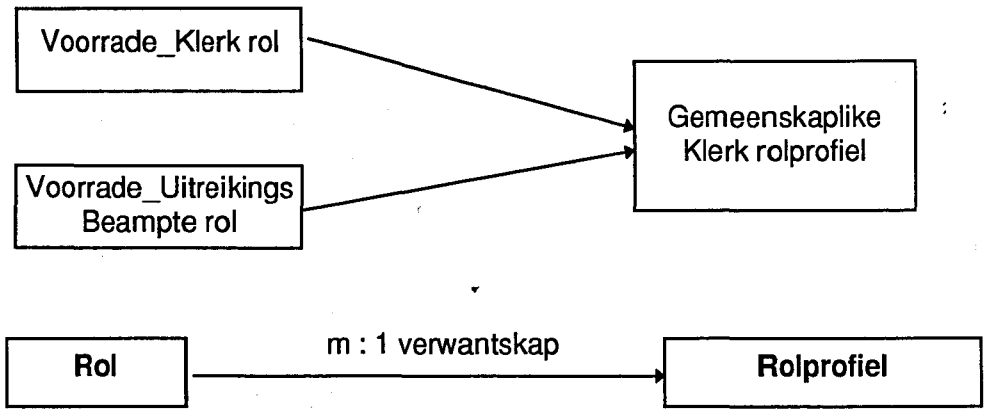
Ons is nou gereed om 'n rol te definieer. Tot dusver mag dit dalk nog onduidelik wees waarom 'n rol en sy ooreenstemmende rolprofiel geskei word, die rede hiervoor sal

egter eers duidelik wees wanneer elke komponent van die definisie vir 'n rol verduidelik is.

Definisie 2.3: *'n Rol is 'n veeltal (r_naam, rp_naam, o_lys, k_lys). r_naam is 'n unieke naam vir die rol in die stelsel. rp_naam is 'n unieke naam vir 'n rolprofiel in die stelsel wat hierdie spesifieke rol se voorregte beskryf. o_lys en k_lys word later beskryf.*

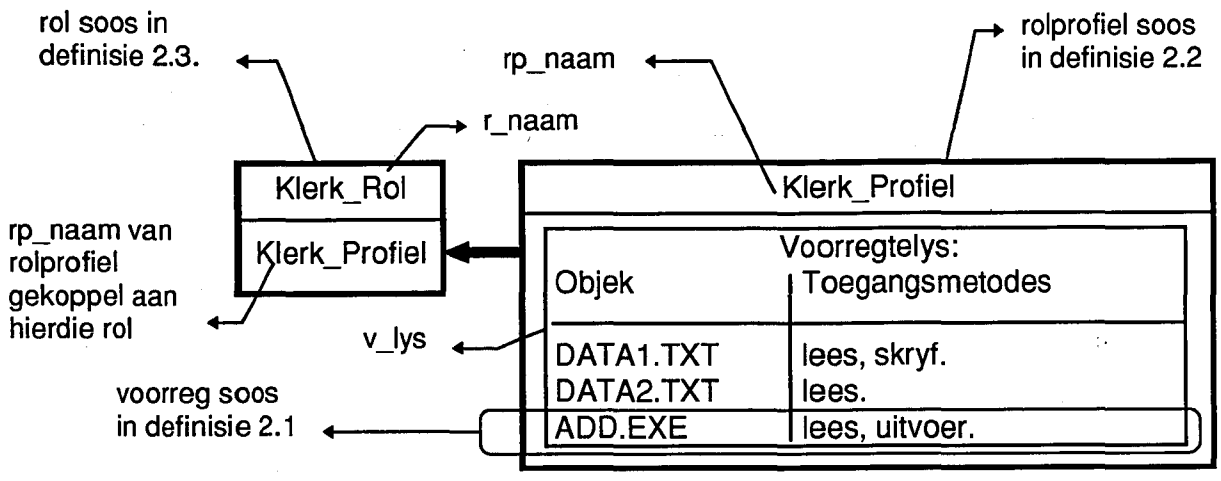
'n Rolprofiel word gekoppel aan 'n rol. Onthou dat ons gesê het 'n rolprofiel is die beskrywing van die regte en beperkings van 'n rol. Ons skei egter 'n rol en 'n rolprofiel deurdat 'n rol die beskrywing van die verwantskap tussen een rol en ander rolle in 'n rekenaarstelsel is en 'n rolprofiel is die beskrywing van die voorregte (soos in definisie 2.1) van 'n rol.

Rolle en rolprofiele is egter nie noodwendig op 'n een tot een basis gekoppel nie. Dit is moontlik dat daar twee of meer rolle in die organisasie is wat op logiese vlak as twee aparte rolle beskou word maar op implementering vlak gebruik hulle dieselfde rolprofiel. Elke rol word egter aan presies een rolprofiel gekoppel. 'n Voorbeeld van hierdie een tot meer verwantskap tussen rolle en rolprofiele is dat daar in 'n organisasie 'n Voorrade Klerk en 'n Voorrade Uitreikingsbeampte is, albei het egter presies dieselfde voorregte maar daar word verkies dat hulle as aparte besigheidsrolle beskryf word, tog het albei rolle dieselfde rolprofiel. Figuur 2.3 toon hierdie voorbeeld diagrammaties.



Figuur 2.3
Verwantskap tussen rolle en rolprofiele

Figuur 2.4 toon 'n voorbeeld waar ons die definisies wat ons tot dusver gedefinieer het gebruik. Die rol Klerk in die stelsel het 'n unieke naam Klerk_Rol. Die rol bevat 'n rolprofielnaam Klerk_Profiel. Die beskrywing van rol Klerk_Rol word gedoen in die Klerk_rolprofiel_Profiel. Klerk_Profiel bevat 'n versameling voorregte. In hierdie geval bestaan elke voorreg uit 'n eenvoudige stelselobjek (lêers) en die toegangsregte vir die spesifieke objek (lees, skryf of uitvoer).



Figuur 2.4.
'n Voorbeeld van 'n toepassing van definisies

'n Voorreg kan gesien word as 'n eenheid van toegangsregte wat by 'n rol gevoeg of verwyder kan word vir die administrasie van 'n rol. Let op dat as ons sê dat ons 'n voorreg by 'n rol voeg, dan beteken dit dat ons die voorreg by die beskrywing van die rol se regte voeg, m.a.w. ons voeg die voorreg by die *v_lys* van die rol se rolprofiel. Op dieselfde manier as ons na 'n rol se voorregtelys verwys, dan verwys ons na die voorregtelys van die rolprofiel wat aan die rol gekoppel is. Beskou weer figuur 2.4 as voorbeeld. Die voorreg ADD.EXE, {lees, uitvoer} is een logiese eenheid van toegangsregte wat by die rol Klerk_Rol gevoeg is. Die voorreg word egter gestoor in die rolprofiel wat aan Klerk_Rol gekoppel is, met ander woorde in Klerk_Profiel.

2.2.3 Notasie:

Gestel ons het 'n gedefinieerde rol r in die stelsel. Indien ons na r se rolprofiel wil verwys, skryf ons r.rp_naam . Indien ons na 'n rolprofiel rp se voorregtelys wil verwys skryf ons rp.v_lys . Ons kan natuurlik ook r.rp_naam.v_lys skryf om na die voorregtelys wat aan rol r toegeken is (via sy rolprofiel) te verwys. By die voorbeeld in figuur 2.4 is Klerk_Rol gekoppel aan Klerk_Profiel. Met Klerk_Rol.v_lys verwys ons na Klerk_Profiel.v_lys, die voorregtelys van Klerk_Profiel.

2.2.4 Onderskeid tussen hoë-vlak en lae-vlak gebruik van rolle en rolprofiele

Dit is belangrik om daarop te let dat die wyse waarop 'n sekerheidsbestuurder voorregte by rolle (en rolprofiele) voeg, nie noodwendig dieselfde is as die toekenning van voorregte aan rolle en rolprofiele op lae-vlak deur die stelsel nie. Hierdie opmerking word vervolgens bespreek.

Let eerstens daarop dat ons kan onderskei tussen hoë-vlak en lae-vlak gebruik van rolle en rolprofiele. Vir die doeleindes van hierdie verhandeling is hoë-vlak gebruik van rolle en rolprofiele die gebruik (toekenning, skepping, vernietiging, ens.) van rolle deur 'n persoon soos 'n sekerheidsbestuurder. Lae-vlak gebruik van rolle en rolprofiele is die aksies wat daarop uitgevoer word deur die stelsel (programatuur) wat die rolle en rolprofiele instand hou en in die stelsel stoor.

Indien ons bogenoemde benadering volg, mag dit so wees dat hoë-vlak gebruikers nooit direk met rolprofiele te doen kry nie. Dit kan eerder die lae-vlak stelsel se taak wees om rolprofiele te skep en instand te hou. By hierdie situasie sal die sekerheidsbestuurder byvoorbeeld regte by 'n rol voeg (hoë-vlak gebruik) en stelsel sal self die regte by die rol se rolprofiel gaan invoeg (lae-vlak gebruik). In laasgenoemde voorbeeld (2.2.3) is dit dus nie nodig om te onderskei tussen die byvoeging van voorregte aan rolle en byvoeging van voorregte aan rolprofiele nie. Vir die hoë-vlak gebruiker lyk dit dus of voorregte by rolle gevoeg word maar in der waarheid word die voorregte op lae-vlak by die rolprofiel gevoeg deur die stelsel.

In die res van die verhandeling gaan ons nie verder onderskeid tref tussen hoë-vlak en lae-vlak gebruik van rolle en rolprofiele nie. Let egter op dat die byvoeging van voorregte aan 'n rol kan uitloop op die byvoeging van voorregte aan 'n rolprofiel deur die stelsel. Op dieselfde wyse kan die byvoeging van voorregte aan 'n rolprofiel die oorsaak wees van 'n versoek om voorregte by 'n rol te voeg.

Tot dusver het ons 'n rol en 'n rolprofiel gedefinieer. Die opmerking is gemaak dat 'n rol die verwantskappe tussen homself en ander rolle beskryf. Hierdie verwantskappe is egter nog nie gedefinieer of verduidelik nie. In die volgende afdeling gee ons aandag aan hierdie verwantskappe.

2.3 Toegangsbeheer met behulp van rolle en rolverwantskappe

Rolle dien as 'n toegangsweg tot stelselinligting [5]. Die voorregtelys van 'n rolprofiel bepaal watter inligting, en onder watter voorwaardes, beskikbaar is in die stelsel via 'n rol gekoppel aan hierdie rolprofiel. In 'n rolgebaseerde sekerheidstelsel vind magtiging van toegang na inligting op meer as een vlak plaas. 'n Vlak van magtiging kan gesien word as nog 'n manier om magtiging vir gebruikers te gee of te onttrek. 'n Voorbeeld van twee vlakke van magtiging is om 'n gebruiker toegang te gee na 'n rol en om 'n rol se voorregtelys uit te brei. Deur hierdie vlakke te identifiseer kan ons toon dat rolgebaseerde inligtingsekerheid meer as een vlak bied waar regte direk of indirek aan gebruikers toegeken of onttrek kan word. In hierdie afdeling identifiseer ons vier vlakke en in die proses verduidelik ons die ouer_lys en kind_lys komponente van die definisie vir 'n rol soos gegee is in definisie 2.3. Hierdie komponente beskryf die verwantskap tussen 'n rol en ander rolle in die stelsel.

2.3.1 Gebruiker-rol magtiging

Die eerste vlak van magtiging in 'n rolgebaseerde inligtingsekerheidstelsel is die gebruiker-rol magtigingsvlak. Op hierdie vlak kry gebruikers toegang na stelselinligting deurdat elke gebruiker toegang kry na een of meer rolle. Onthou dat ons reeds gesê het dat rolgebaseerde inligtingsekerheid is meer versoenbaar met bestaande bedryfstelsels se inligtingsekerheid stelsels wanneer dit moontlik is om gebruikergroepe te koppel aan rolle. Deur gebruikers of gebruikergroepe aan rolle te koppel kry gebruikers of gebruikergroepe toegang na die stelselinligting via rolle. Dit word gedoen deur 'n tabel te onderhou wat vir elke gebruiker of groep gebruikers van die stelsel 'n lys van rolle bevat. Die lys rolle dui aan uit watter rolle 'n gebruiker mag

kies wanneer sy by die stelsel aanteken. Ons noem hierdie tabel voortaan 'n *toegangsbeheerlys* (TBL). Ons gee nou 'n formele definisie vir 'n TBL.

In enige stelsel waar inligting beskerm word deur een of ander sekerheidstelsel is dit nodig om gebruikers te laat aanteken op die stelsel. Wanneer 'n gebruiker aanteken op 'n stelsel verskaf hy sy unieke identifikasie aan die stelsel en die stelsel moet die gebruiker verifieer (seker maak die gebruiker is wie hy sê hy is) deur byvoorbeeld 'n wagwoord, wat slegs aan die gebruiker bekend is, van die gebruiker te lees. Die gebruiker se unieke identifikasie is enige string karakters wat aan die gebruiker verskaf is toe hy geregistreer is as 'n nuwe gebruiker van die stelsel. Ons noem hierdie string karakters 'n *gebruiker identifiseerder.* Let op dat gebruikers in groepe saam gegroepeer kan word soos byvoorbeeld 'n studente-groep van gebruikers. So 'n gebruikergroep kry ook 'n unieke identifikasie en ons noem hierdie identifikasie 'n *groep identifiseerder.*

Definisie 2.4: *Laat UID die versameling gebruiker identifiseerders in die stelsel en GID die versameling groep identifieerders in die stelsel wees, stel dan is ID = UID ∪ GID. 'n Toegangsbeheerlys (TBL) is 'n tabel met inskrywings waar elke inskrywing van die vorm [id_i, ROLLE] is waar $id_i \in ID$, $1<=i<=n$ (n = kardinalitieit(ID)) en ROLLE is 'n lys van rolname in die stelsel.*

Tabel 2.3 toon 'n voorbeeld van 'n toegangsbeheerlys. Let op dat elke inskrywing in die tabel 'n gebruiker identifiseerder bevat (bv. Sarah) en 'n lys van rolle (bv. Klerk, Teller)waaruit hierdie gebruiker mag kies wanneer sy op die stelsel aanteken.

TOEGANGSBEHEERLYS	
Gebruikernaam	Rolname
Sarah	Klerk, Teller
Peter	Klerk, Salarisse_Werker
Mary	Personeelbestuurder
Joe	Bankbestuurder

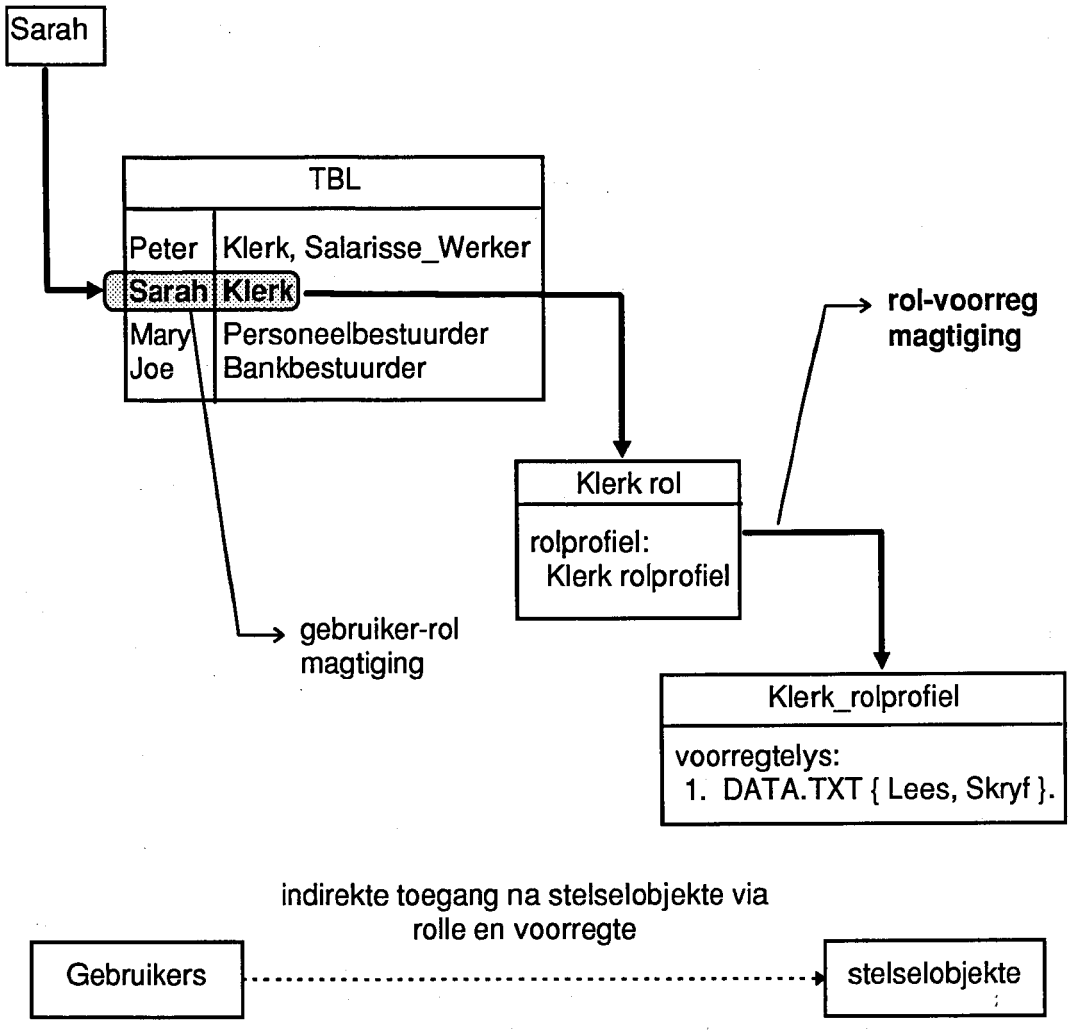
Tabel 2.3
Voorbeeld van 'n Toegangsbeheerlys.

Gebruiker-rol magtiging is een manier om gebruikers toegang na stelselinligting te gee. Elke rol het 'n rolprofiel wat die rol se voorregte beskryf. Wanneer 'n gebruiker aan 'n rol gekoppel word (deur 'n inskrywing in die TBL), kry die gebruiker die voorregte van die rol. Deur die voorregte in die rol se rolprofiel te verander, word die voorregte wat die gebruiker deur die gebruiker-rol magtiging verkry het ook indirek verander. Hierdie tipe magtiging, rol-voorreg magtigting, is die volgende vlak van magtiging en word vervolgens bespreek.

2.3.2 Rol-voorreg magtiging

Vir 'n gebruiker van 'n stelsel met rolgebaseerde inligtingsekerheid is dit nodig om aan 'n rol gekoppel te wees voordat toegang tot stelselobjekte verkry kan word. Die koppeling van gebruikers aan rolle, gebruiker-rol magtiging, is een manier van magting. Indien 'n gebruiker aan 'n rol gekoppel is kan ons die voorregte wat 'n gebruiker indirek deur 'n gebruiker-rol koppeling verkry het, wysig deur die voorregte van die rol te wysig. Hierdie tipe magtiging noem ons rol-voorreg. Hierdie vlak van magtiging behels dat magtiging verleen word deur 'n voorreg by die voorregtelys van 'n rol se rolprofiel gevoeg word.

Figuur 2.5 toon 'n voorbeeld van rol-voorreg magtiging. Gebruiker Sarah is aan die rol Klerk gekoppel deurdat haar gebruiker identifiseerder in die toegangbeheerlys voorkom en die rol Klerk is in die lys van rolle waaruit sy mag kies (gebruiker-rol magtiging). Die rol Klerk se rolprofiel, Klerk_rolprofiel, bevat in sy voorregtelys slegs een voorreg: lees- en skryfregte op lêer DATA.TXT. Gebruiker Sarah verkry hierdie voorreg deurdat sy aan die rol Klerk gekoppel is en die rol Klerk is aan die Klerk_rolprofiel gekoppel. Laasgenoemde noem ons *rol-voorreg magtiging*. Indien ons nog 'n voorreg, gestel leesregte op DATA2.TXT, by die voorregtelys van die Klerk_rolprofiel sou voeg, verkry gebruiker Sarah indirek ook hierdie voorreg. Laasgenoemde is weereens 'n voorbeeld van rol-voorreg magtiging.



Figuur 2.5
Gebruiker-rol en rol-voorreg magtiging.

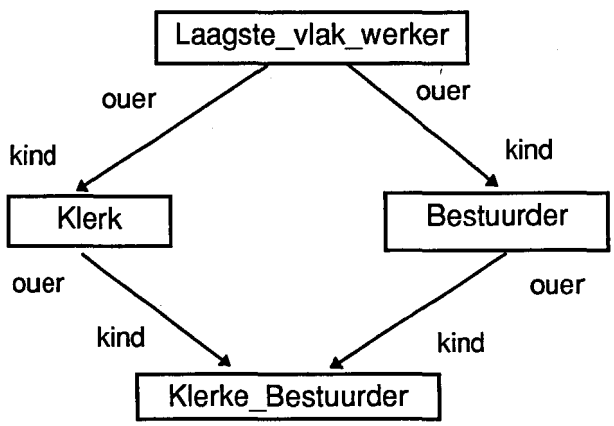
Die opmerkings is al gemaak dat 'n rol die beskrywing bevat van die rol se verwantskap met ander rolle asook die verwantskap met gebruikers van die stelsel. Ons het reeds die beskrywing van die verwantskap tussen rolle en gebruikers van die stelsel bespreek deur 'n toegangsbeheerlys te definieer. In die volgende afdeling bespreek ons hoe 'n rol sy verwantskap met ander rolle beskryf. Die bespreking is deel van die beskrywing van rol-rol magtiging.

2.3.3 Rol-rol magtiging

'n Rolgebaseerde inligtingsekerheidstelsel kan uitgebrei word tot 'n punt waar een rol die voorregte van 'n ander rol kan bekom, dit is byvoorbeeld moontlik dat 'n Klerk_Bestuurder rol die voorregte van 'n Klerk rol moet hê. Volgens hierdie tipe magtiging kry een rol toegang na 'n ander rol en ons noem dit rol-rol magtiging. Indien rol A toegang het na rol B, beteken dit dat gebruikers wat gekoppel is aan rol A, al die voorregte van rol B kry. Rol-rol magtiging is 'n aspek van rolle wat ons die rolstruktuur noem. Indien ons rol-rol magtiging in 'n rolgebaseerde stelsel toelaat, kan ons sê die rol waartoe 'n rol toegang het, is 'n ouer-rol van die rol. Op dieselfde wyse

kan ons sê dat alle rolle wat toegang het tot 'n spesifieke rol, is kind-rolle van daardie spesifieke rol. Indien ons nou ook vir elke rol sy ouer(s) en kind(ers) stoor, kan ons deur hierdie inligting te gebruik, die *verwantskap* tussen rolle voorstel. Indien 'n bestuurder die verwantskap tussen rolle kan sien, is dit makliker om te besluit watter rolle om aan 'n gebruiker toe te ken.

Figuur 2.6 toon 'n voorbeeld van 'n voorstelling van rolverwantskappe. Rol Klerk het rol Laagste_vlak_werker as ouer en het rol Klerke_Bestuurder as 'n kind rol. In hierdie voorbeeld het Laagste_vlak_werker die minste voorregte, Klerk en Bestuurder het albei al die voorregte van Laagste_vlak_werker en Klerke_Bestuurder het al die voorregte van Klerk en Bestuurder, dit beteken dat Klerke_Bestuurder die vereniging het van die voorregte van al die rolle in die stelsel.



Figuur 2.9.
Voorbleeld van voorstelling van rolverwantskappe

Indien ons die rolstruktuur (verwantskappe tussen rolle) van die stelsel wil stoor en benut, is dit nodig dat elke rol moet weet watter rolle is ouers en watter rolle is kinders van homself. Ons beskryf nou die *o_lys* en *k_lys* komponente van die definisie van 'n rol soos gegee is in definisie 2.3.

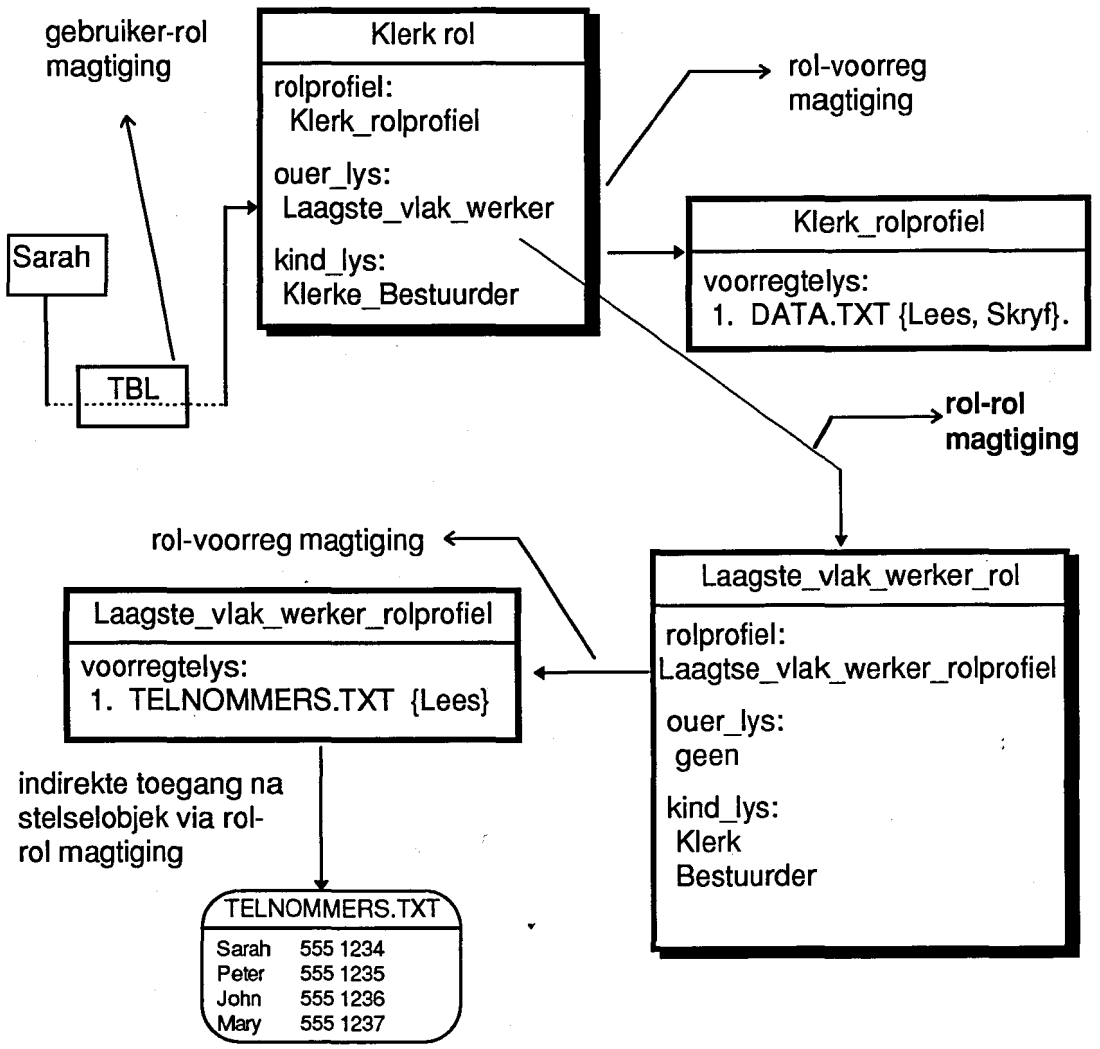
Vir 'n rol r is *o_lys* 'n lys rolle se name wat ouers is van van rol r en *k_lys* is 'n lys van rolle se name wat kinders is van rol r. Let op dat *o_lys* en *k_lys* ook leeg kan wees.

Ons gee weer die definiese vir 'n rol en sluit nou al die komponente in die veeltal se beskrywing in.

Definisie 2.3: *'n Rol is 'n veeltal (r_naam, rp_naam, o_lys, k_lys). r_naam is 'n unieke naam vir die rol in die stelsel. rp_naam is 'n unieke naam vir 'n rolprofiel in die stelsel wat hierdie spesifieke rol se voorregte beskryf. o_lys is 'n lys rolle wat ouers is van van hierdie rol. k_lys is 'n lys van rolle wat kinders is van hierdie rol. o_lys en k_lys kan ook leeg wees.*

Figuur 2.7 gee 'n diagrammatiese voorstelling van die definisie vir 'n rol. Die figuur toon ook rol-rol magtiging. Gebruiker Sarah is gekoppel aan die rol Klerk, maar het toegang na die voorregte van die Laagste_vlak_werker rol want die rol

Laagste_vlak_werker is 'n ouer-rol van Klerk. Sarah kry toegang na TELNOMMERS.TXT alhoewel dit nie deel is van rol Klerk se rolprofiel se voorregtelys nie. Toegang na TELNOMMERS.TXT word verkry via rol-rol magtiging (rol Klerk het toegang na rol Laagste_vlak_werker se voorregte).



Figuur 2.7.
'n Diagrammatiese voorstelling van die rol-definisie asook rol-rol magtiging.

Ons het nou die drie belangrikste vlakke van magtiging in 'n rolgebaseerde inligtingsekerheidstelsel bespreek. Indien ons transaksies as ons beskermde hulpbron beskou en toegang gee na transaksies, dan het ons ook 'n transaksie-bron vlak van magtiging. Ons bespreek hierdie vlak van magtiging vervolgens.

2.3.4 Transaksie-bron magtiging

Voordat ons hierdie vlak van magtiging verduidelik is dit nodig om die begrip *transaksie* te definieer. In hierdie afdeling word 'n kort oorsig van transaksies gegee. In hoofstuk 5 word daar in meer besonderhede gekyk na transaksieverwerking. In hoofstuk 6 word ORITO, die model vir verspreide, objek-georiënteerde en

rolgebaseerde inligtingsekerheid in *transaksieverwerking* omgewings bespreek. Dit is dus gepas om op hierdie stadium 'n kort bespreking te gee van rolgebaseerde inligtingsekerheid by transaksieverwerking. Hou egter in gedagte dat transaksieverwerking in meer besonderhede in hoofstuk 5 bespreek word.

2.3.5 'n Transaksie as eenheid van verwerking

Definisie 2.5: *'n Transaksie is 'n diskrete eenheid van verwerking en het toegang nodig na verskeie hulpbronne om suksesvol af te handel. In terme van voorregte, soos in definisie 2.1 gedefinieer is, is 'n transaksie 'n geordende versameling aksies op hulpbronne en 'n transaksie benodig 'n versameling voorregte waar elke voorreg van die vorm (x, m) is.*

Dit beteken vir 'n transaksie om suksesvol af te handel, moet alle opdaterings op hulpbronne suksesvol wees, andersins moet geen opdatering gedoen word nie ('n transaksie is 'n eenheid van verwerking). Om inligting van bronne te lees, of na bronne te skryf, het die transaksie (of die gebruiker wat die transaksie versoek), al die voorregte wat nodig is om toegang na die bronne te verkry, nodig. Figuur 2.8 toon 'n voorbeeld van 'n transaksie OORPLAAS. Die transaksie bevat twee stappe wat rekening A debiteer en rekening B krediteer. Die transaksie het skryfregte op albei rekeninge nodig.

transaksie OORPLAAS (A, B, Bedrag)

aksies/stappe:
1. Debiteer rekening A met Bedrag
2. Krediteer rekening B met Bedrag

voorregtelys:
1. rekening A { lees, skryf }
2. rekening B { lees, skryf }.

Figuur 2.11
'n Voorbeeld van 'n transaksie.

2.3.6 'n Transaksie as 'n eenheid van magtiging

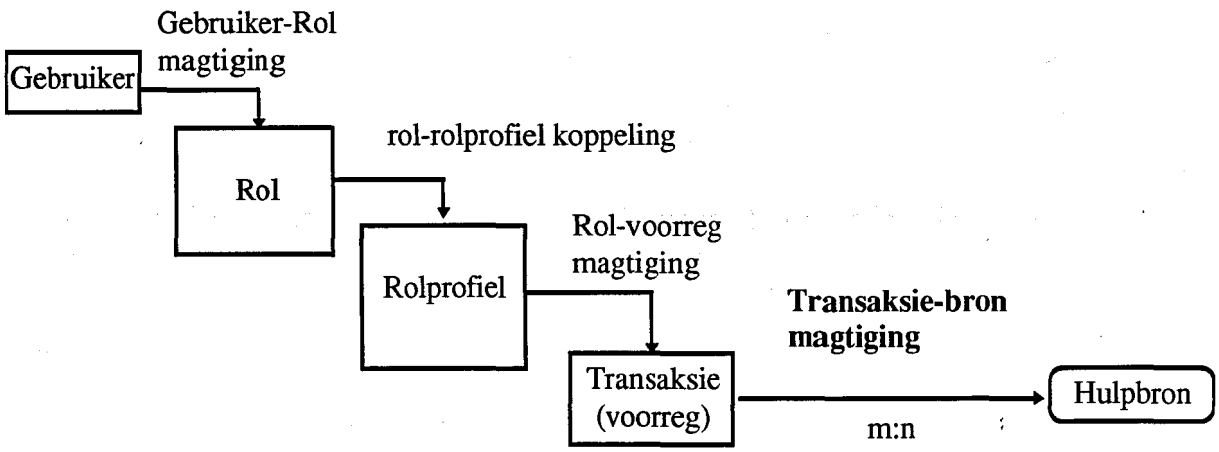
'n Transaksie kan gesien word as 'n eenheid van magtiging. 'n Gebruiker mag dalk nie die regte besit om twee bronne apart te wysig nie, maar mag dalk die reg besit om albei bronne gelyktydig te wysig deur 'n transaksie te gebruik. Deur transaksies te gebruik, kan daar beheer uitgevoer word oor die aksies wat 'n gebruiker mag uitvoer (**transaksies verleen beheerde toegang na hulpbronne**).

'n Voorbeeld van bogenoemde is 'n klerk in 'n bank - 'n klerk het dalk nie die regte om een rekening te debiteer of net 'n ander rekening te krediteer nie, maar die klerk mag dalk wel die regte besit om een rekening te krediteer en terselfde tyd 'n ander rekening met dieselfde bedrag te krediteer. So 'n versameling van aksies is dus 'n transaksie en die transaksie bevat die voorregte om twee bronne se inhoud te wysig ('n versameling van voorregte).

Dit is belangrik om daarop te let dat indien 'n gebruiker, of 'n rol in 'n rolgebaseerde stelsel, die reg het om 'n transaksie uit te voer, kry so 'n gebruiker of rol implisiet toegang na verskeie hulpbronne (maar wel onder die beheer van 'n transaksie).

2.3.7 Transaksie-bron magtiging

Figuur 2.9 toon hoe 'n gebruiker toegang na 'n hulpbron kan verkry deur 'n transaksie. Hierdie tipe toegang wat aan die gebruiker gegee word noem ons *transaksie-bron magtiging*. Elke transaksie verleen toegang na verskeie hulpbronne. Transaksies en hulpbronne het dus 'n m:n verwantskap. Deur dus 'n transaksie te wysig kan ons ook die regte van 'n gebruiker implisiet wysig. Die sekerheidsbestuurder moet egter sorg dat as 'n transaksie by 'n rol se rolprofiel gevoeg word, dan het die rol steeds net die regte wat hy nodig het om sy besigheidsfunksie te voltooi en nie meer as nodig nie.



Figuur 2.9
Transaksie-bron magtiging.

Let op dat in 'n transaksieverwerking stelsel waar daar toegang na transaksies gegee word is die objek x in die voorreg (x, m) 'n transaksie. Soos beskryf is in definisie 2.1, is die toegangsmetodes 'n lys van toelaatbare omstandighede soos 0 <= bedrag <= 5000 en terminiaal id $\in$ { TM1, TM3 }.

Die Transaksie-bron magtigingsvlak is bespreek omdat die model wat later geformuleer word, toegepas word in 'n transaksieverwerkerstelsel.

Ons het nou al die komponente van 'n rolgebaseerde inligtingsekerheidstelsel gedefinieer en beskryf. Om die werking van so 'n stelsel meer duidelik te maak word daar in die volgende afdeling 'n eenvoudige voorbeeld van rolgebaseerde sekerheid gegee.

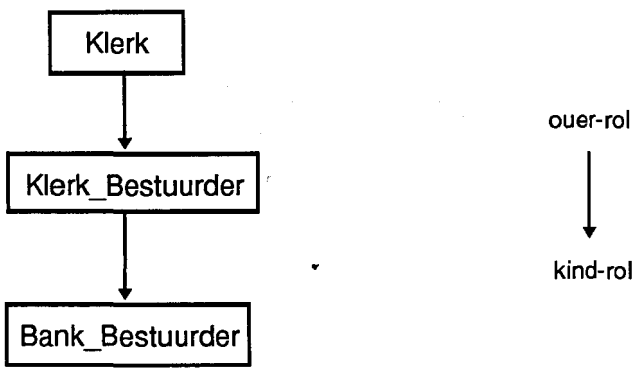
2.4 Voorbeeld van 'n rolgebaseerde inligtingsekerheidstelsel

Voordat magtiging in 'n rolgebaseerde inligtingsekerheidstelsel kan plaasvind moet die besigheidsrolle in 'n organisasie geïdentifiseer word en rolle met ooreenstemmende rolprofiele moet vir elke rol opgestel word. Die opstel van rolle en rolprofiele is 'n

taak wat gedoen word deur 'n persoon met kundigheid op die gebied van inligtingsekerheid. Die opstel van rolle en rolprofiele behels dat daar vir elke rol besluit moet word op die volgende:

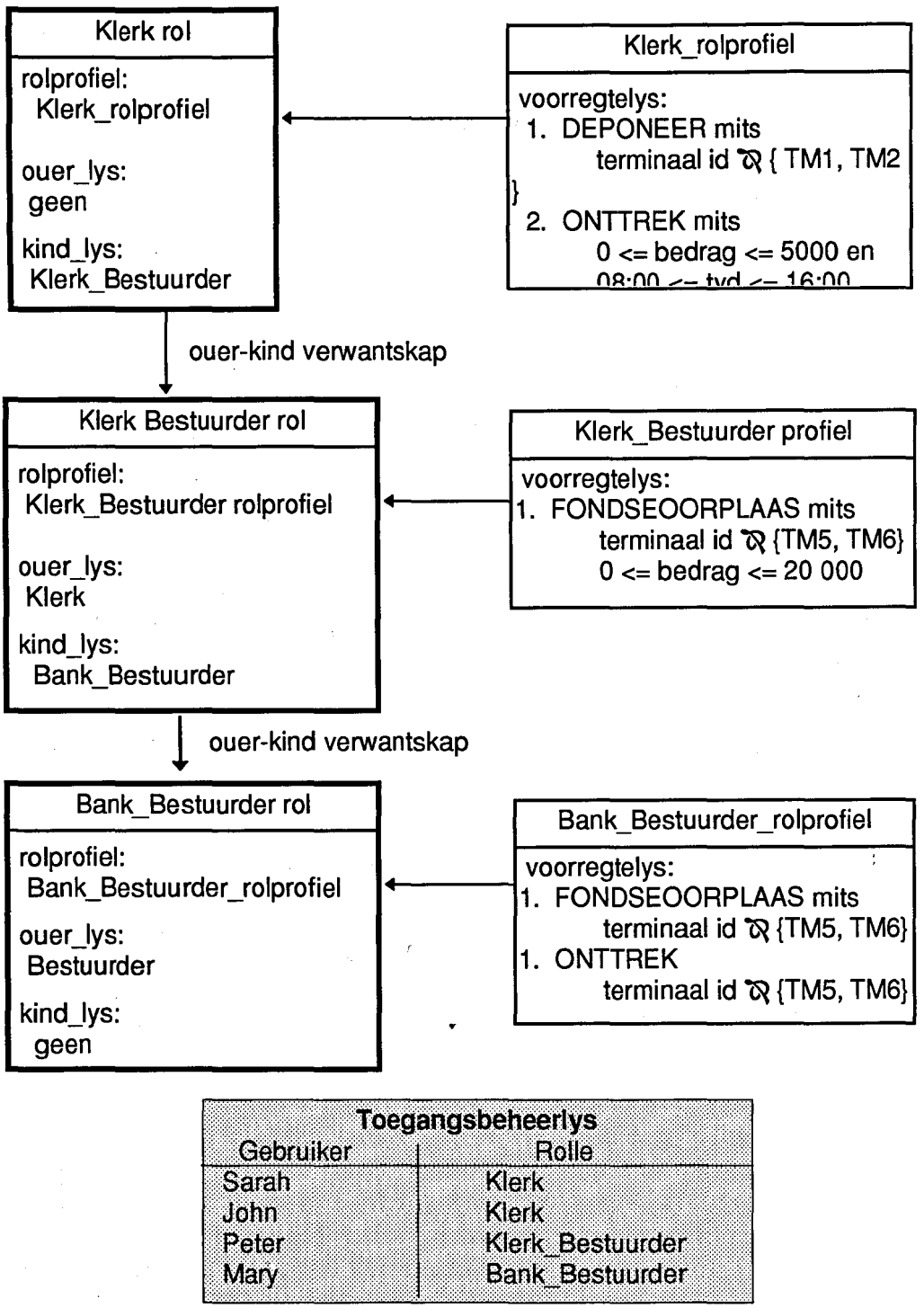
1. 'n unieke naam vir die rol.
2. watter stelselobjekte is toegangklik vir die rol en watter tipe toegang moet verleen word (bv lees en skryf). Die voorregte van die rol word in die rol se rolprofiel gestoor.
3. watter rolle se voorregte moet oorgedra word na hierdie nuwe rol. Daar moet 'n lys van ouer-rolle opgestel word.
4. watter rolle moet die voorregte van hierdie nuwe rol ontvang (kinder-rolle).
5. watter gebruikers moet toegang hê na hierdie nuwe rol. Die rolnaam moet by die gebruiker identifiseerders in die toegangsbeheerlys gevoeg word (slegs die gebruikers wat toegang het na die nuwe rol).
6. die naam van die rolprofiel wat die voorregte van hierdie rol stoor.

Ons gebruik in die voorbeeld transaksies as stelselobjekte in die voorregtelys van elke rol se rolprofiel. Dit beteken dan dat elke voorreg in 'n voorregtelys bestaan uit 'n transaksienaam en 'n lys van toelaatbare omstandigheide waaronder hierdie transaksie uitgevoer mag word. Om die voorbeeld eenvoudig te hou skep ons slegs drie rolle naamlik: Klerk, Klerk_Bestuurder en Bank_Bestuurder. Die rolle hou verband soos in figuur 2.10 geïllustreer word. Klerk_Bestuurder erf al die voorregte van rol Klerk en rol Bank_Bestuurder al die voorregte Klerk_Bestuurder.



Figuur 2.10.
Rolverwantskappe in die voorbeeld stelsel.

Die rolprofiele vir die drie rolle van ons stelsel is soos in figuur 2.11. Let op dat rol Bank_Bestuurder en rol Klerk_Bestuurder bevat albei die transaksie FONDSEOORPLAAS maar met verskillende toelaatbare omstandighede. In so 'n geval sê ons dat die voorreg ('n transaksie in hierdie voorbeeld) word *geoorskryf* in 'n kind rol, dit beteken dat alhoewel die ouer- en kindrol albei dieselfde transaksie het, is die transaksie geoorskryf in die kind-rol en geld die kind-rol se transaksie en toelaatbare omstandighede. Na die figuur volg 'n paar voorbeelde van magtiging in die voorbeeld stelsel.



Figuur 2.11
Die rolle en rolprofiele vir die voorbeeld stelsel.

Voorbeelde van magtigings in die stelsel:

1. Gestel gebruiker Sarah is aangeteken op die stelsel en is aangeteken as rol Klerk. Sarah versoek die transaksie ONTTREK en die bedrag betrokke is 2000 en sy doen die versoek vanaf terminaal TM1 en die tyd is 14:22. Die versoek word toegestaan omdat:
   - Sarah in die toegangsbeheerlys toegang verleen word na rol Klerk,
   - Die tranksasie is in die voorregtelys van Klerk_rolprofiel en
   - Die omstandighede waaronder die versoek gedoen is, is deel van die toelaatbare omstandighede van die voorreg. Let op dat daar geen beperking op die terminaal_id is nie.

2. Gestel gebruiker Mary is aangeteken op die stelsel en is aangeteken as rol Bank_Bestuurder. Mary versoek die transaksie DEPONEER en die bedrag betrokke is 1000 en sy doen die versoek vanaf terminaal TM1 en die tyd is 9:00. Die versoek word toegestaan omdat:
   - Mary in die toegangsbeheerlys toegang verleen word na rol Bank_Bestuurder,
   - Die tranksasie is nie in die voorregtelys van Bank_Bestuurder_rolprofiel, maar die rol Bestuurder is 'n ouer van rol Bank_Bestuurder; die stelsel kyk nou in die voorregtelys van Bestuurder. Die tranksasie is ook nie in die voorregtelys van die Bestuurder_rolprofiel nie maar rol Klerk is 'n ouer van rol Bestuurder. Die stelsel kyk in die voorregtelys van Klerk en transaksie DEPONEER is wel deel van die voorregtelys van Klerk_rolprofiel.
   - Die omstandighede waaronder die versoek gedoen is, is deel van die toelaatbare omstandighede van die voorreg in die voorregtelys van Klerk_rolprofiel.

3. Gestel gebruiker John is aangeteken op die stelsel en is aangeteken as rol Klerk. John versoek die transaksie DEPONEER en die bedrag betrokke is 1000 en hy doen die versoek vanaf terminaal TM1 en die tyd is 9:00. Die versoek word geweier omdat:
   - Die tranksasie DEPONEER nie in die voorregtelys van rol Klerk is nie of enige van rol Klerk se is ouers nie (rol Klerk het geen ouer-rolle nie).

Dit is belangrik om daarop te let dat die bostaande beskrywing van die wyse waarop getoets word of 'n transaksie gemagtig moet word of nie, is 'n logiese beskrywing. Wanneer die stelsel geïmplementeer word mag die stappe betrokke dalk anders wees. Een moontlike verskil is dat elke rolprofiel stoor by homself al die voorregte wat geërf is van ander rolle en verhoed so dat baie rolprofiele deursoek moet word vir elke magtiging. Hierdie aspek word weer aangespreek in hoofstuk 6 wanneer ons die model vir verspreide objek-georiënteerde rolprofiele beskryf.

Ten laaste is dit sinvol om die voordele van rolgebaseerde inligtingsekerheid te beskryf.

2.5 Voordele van rolgebaseerde inligtingsekerheidstelsels

Bestuurders moet daagliks mense in 'n besigheidsituasie toegangsregte gee na bronne in 'n rekenaarstelsel. In 'n stelsel met 'n goed gedefinieerde rolgebaseerde sekerheidstelsel hoef die bestuurder nie te weet watter bronne is nodig nie - dit is voldoende om te weet dat die rol bestaan en wat die rol beteken in besigheidsterme

(bv. rol Klerk gee toegang na presies die bronne wat 'n klerk in die organisasie sal nodig hê) [7]. Die bestuurder koppel dus slegs gebruikers en rolle en nie mense en bronne nie. Rolgebaseerde sekerheid *vergemaklik* dus *sekerheidsbestuur*.

Rolgebaseerde sekerheid verskaf *buigsaamheid* in die administrasie van gebruiker regte [5]. Gebruikerregte kan verander word deur eksplisiet 'n gebruiker toegang te gee na 'n ander rol of toegang na 'n rol te onttrek. Gebruikerregte kan ook implisiet verander word deur die rolprofiel van 'n rol waaraan 'n gebruiker gekoppel is te wysig. Hierdie buigsaamheid in administrasie ontstaan omdat 'n rolgebaseerde sekerheidstelsel verskillende vlakke van magtiging het soos in 2.3 bespreek is.

Rolgebaseerde sekerheid kan die beginsel van *minste regte* toepas deur 'n rol toegang te gee na slegs die bronne wat nodig is vir 'n besigheidsrol om sy taak te kan verrig in die organisasie.

2.6 Slot

Ons het gesien hoe inligtingsekerheid in 'n organisasie bereik kan word deur besigheidsrolle te identifiseer en die voorregte van sulke rolle te spesifiseer. Die voorregte van elke rol word in 'n rolprofiel gespesifiseer. Elke rol word dan met 'n rolprofiel gekoppel. 'n Rolprofiel is die beskrywing van die voorregte en beperkings van 'n rol in 'n organisasie terwyl 'n rol die beskrywing is van die verwantskap van 'n rol met ander rolle in die stelsel.

Gebruikers word gekoppel aan rolle deur 'n inskrywing in die toegangsbeheerlys te maak en kry sodoende toegang tot bronne in die organisasie se rekenaarstelsel. Ons het ook gesien dat rolle verband kan hou en deur rolle magtiging na ander rolle te gee kan gebruikers se regte implisiet gewysig word. Transaksies as stelselhulpbronne is bespreek en 'n voorbeeld stelsel waar rolgebaseerde inligtingsekerheid gebruik word is gegee. Die hoofstuk sluit af met die voordele van rolgebaseerde inligtingsekerheid.

In die volgende hoofstuk kyk ons hoe rolgebaseerde inligtingsekerheid op 'n objek-georiënteerde wyse geïmplementeer kan word. Objek-georiënteerde programmering is 'n belowende programmeringsmetodologie en al meer toepassings toon hierdie neiging, maar meer hieroor in die volgende hoofstuk.

3. Objek-georiënteerde rolgebaseerde inligtingsekerheid

3.1 Inleiding

In die vorige hoofstuk is rolgebaseerde inligtingsekerheid bespreek. Ons het gesien dat magtiging na stelselhulpbronne verleen kan word deur gebruikers aan rolle te koppel waar elke rol 'n rolprofiel het en so 'n rolprofiel beskryf die stelselhulpbronne waartoe die rol toegang het. Die rolprofiel beskryf ook die teogangsmetodes na 'n bron bv. slegs lees of lees en skryf of lees maar net tussen 9:00 en 17:00 op weeksdae.

In hierdie hoofstuk gaan ons meer aandag skenk aan rolprofiele en ons kyk hoe rolprofiele op 'n objek-georiënteerde wyse in 'n stelsel geïmplementeer kan word. Om dit te kan doen gee ons eers 'n oorsig oor objek-oriëntasie en daarna word rolprofielobjekte (objek-georiënteerde rolprofiele) beskryf.

Net soos wat rolle in 'n organisasie verband hou, kan van die rolle en rolprofiele in 'n inligtingsekerheidstelsel ook verband hou. Ons kyk later in die hoofstuk hoe rolprofielobjekte verband hou en hoe nuwe rolprofielobjekte geskep kan word deur hierdie verwantskappe in ag te neem.

Objek-georiënteerde rolprofiele hou sekere voordele in wat gewone rolprofiele nie het nie. Die voordele van rolprofielobjekte sluit die hoofstuk af.

3.2 Basiese objek-georiënteerde (OO) konsepte

Hierdie afdeling gee 'n oorsig oor objek-georiënteerde programmering en die konsepte wat verband hou daarmee. Die leser wat vertroud is met objek-georiënteerde programmering kan verder lees by paragraaf 3.3.

Objek-georiënteerde programmering is 'n belowende programmeringsmetodiek waarby ons die objekte waarmee ons in die werklike wêreld te doen kry kan modelleer as abstrakte objekte in 'n rekenaar. So kan ons byvoorbeeld vir 'n mens 'n objek in die rekenaar skep en alles wat ons omtrent 'n mens weet in die objek in die rekenaar stoor. 'n Objek in 'n rekenaar kan gesien word as 'n saamgestelde datatipe waar ons in die datatipe 'n hele paar datawaardes stoor. 'n Objek is egter anders as konvensionele saamgestelde datatipes in ten minste twee opsigte:

- 'n objek bevat datawaardes *en* funksies wat bewerkings op die data binne in die objek doen.
- 'n objek is 'n *instansie van 'n klas*. Dit beteken dat vir elke objek in die stelsel ons 'n raamwerk skep, wat ons 'n klas noem. Die raamwerk bevat plek vir datawaardes en bevat funksies maar stoor nie werklik enige data nie. 'n Klas is dus, baie eenvoudig gestel, 'n voorbeeld raamwerk vir 'n objek. Baie objekte in 'n stelsel kan instansies wees van dieselfde klas.

Ons kyk vervolgens in meer besonderhede na 'n klas en 'n objek.

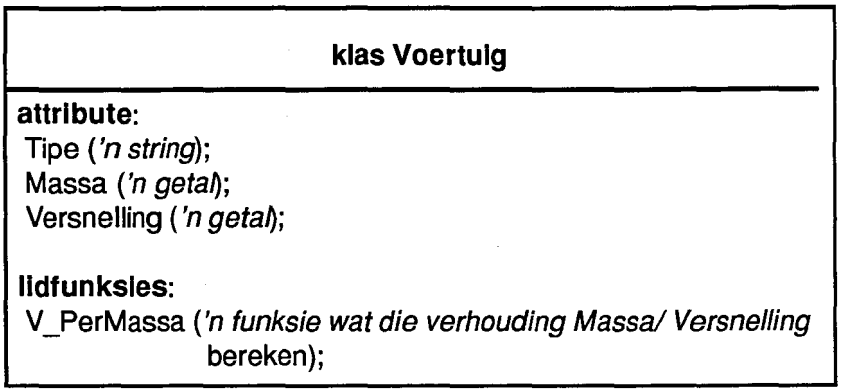
In objek-georiënteerde stelsels is 'n *objek* 'n entiteit met 'n unieke identifikasie [19]. Elke objek het 'n toestand en 'n gedrag. Die *toestand* van die objek is die waardes van die attribute van die objek en die *gedrag* van die objek is die versameling lidfunksies (of metodes) wat bewerkings doen op die toestand van die objek. Die attribute van 'n objek kan gesien word as datavelde waarin verskillende datawaardes van tyd tot tyd gestoor word. Omdat die inhoud van 'n objek se attribute kan verskil van tyd tot tyd sê ons die inhoud van 'n objek se attribute bepaal die toestand van die objek op 'n spesifieke tydstip. Die lidfunksies van 'n objek word saam met die attribute van die objek gestoor en doen bewerkings op die datawaardes van die objek. Die lidfunksies van 'n objek gee resultate wat afhang van die inhoud van die objek se attribute. Omdat die inhoud van 'n objek se attribute nie altyd dieselfde is nie, beteken dit dat die lidfunksies nie altyd dieselfde resultate lewer nie. Ons sê dat al die lidfunksies van 'n objek die gedrag van die objek vorm.

Alle objekte wat dieselfde versameling attribute en lidfunksies deel word gegroepeer in 'n *klas*. 'n Objek moet aan slegs een klas behoort as 'n instansie van daardie klas. 'n Klas kan gesien word as 'n nuwe saamgestelde datatipe in 'n rekenaarstelsel. Elke klas bevat egter data en funksies om bewerkings op die data te doen. In 'n objek-georiënteerde stelsel ontwerp ons klasse om entiteite wat saamhoort saam te groepeer. Elke entiteit word dan in die stelsel verteenwoordig as is 'n instansie van 'n spesifieke klas (of te wel 'n *objek*).

3.2.1 Voorbeeld van objekte en klasse

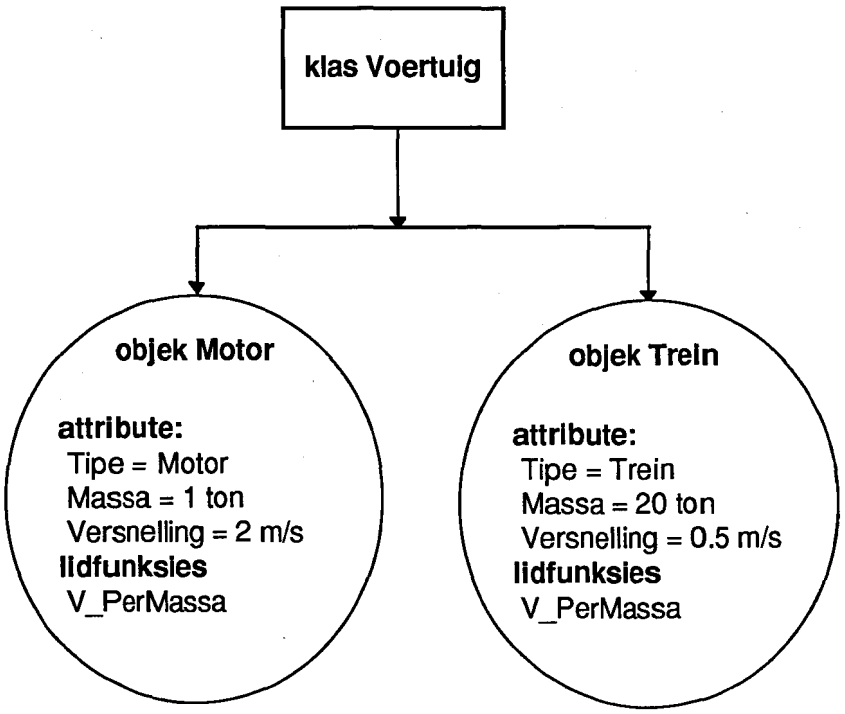
Om die verskil tussen 'n klas en 'n objek verder te illustreer neem die volgende voorbeeld. Gestel ons vorm 'n klas Voertuig en gee vir die Voertuig klas die attribute Tipe, Massa en Versnelling . Onthou dat die attribute van 'n klas is soos datavelde vir die klas waar ons al die datawaardes wat van toepassing is op so 'n tipe klas stoor. Vir die Voertuig klas wil ons drie datawaardes stoor, naamlik die tipe voertuig, die massa van die voertuig en die versnellingsvermoë van die voertuig. Ons stoor hierdie datawaardes in die attribute van die klas. Verder gee ons vir die Voertuig klas 'n lidfunksie V_PerMassa. Die V_PerMassa is 'n funksie wat bloot die verhouding Versnelling / Massa bereken en terugstuur. Onthou dat 'n klas anders is as konvensionele saamgestelde datatipes want 'n klas stoor datawaardes sowel as funksies wat bewerkings doen op die data.

Die Voertuig klas is dus 'n saamgestelde datatipe waar ons drie datawaardes: Tipe, Massa en Versnelling en een lidfunksie: V_PerMassa in die klas stoor. Figuur 3.1 toon die klas Voertuig. 'n Klas kan gesien word as 'n raamwerk vir 'n objek. In ons voorbeeld het ons 'n klas Voertuig met plek vir drie datawaardes en een lidfunksie. Die datawaardes word egter nie in 'n klas gestoor nie; wanneer ons datawaardes wil stoor skep ons 'n *instansie* van so 'n klas en stoor die datawaardes in die instansie van die klas. So 'n instansie noem ons 'n *objek*.



Figuur 3.1
Die Voertuig klas.

Die klas Voertuig kan nou gebruik word as raamwerk om objekte te vorm. Hierdie objekte is instansies van die klas. Gestel ons vorm nou 'n objek Motor met datawaardes: Tipe = Motor, Versnelling = 2 m/s en Massa = 1 ton. Ons skep nog 'n objek Trein met datawaardes Tipe = Trein, Versnelling = 0.5 m/s en Massa = 20 ton. Let nou op dat as ons die lidfunksie V_PerMassa vir die Motor objek roep (ons skryf Motor.V_PerMassa) stuur dit vir ons 'n waarde 2 terug maar Trein.V_PerMassa = 0.0025. Hieruit kan ons sien dat die waardes van die attribute van 'n objek die toestand bepaal van die objek op daardie tydstip. Albei objekte, Motor en Trein, is *instansies* van die klas Voertuig soos in figuur 3.2 getoon word. In hierdie verhandeling word 'n klas as 'n reghoekige blokke aangedui, en instansies daarvan word aangedui met sirkels of reghoeke met ronde hoeke.

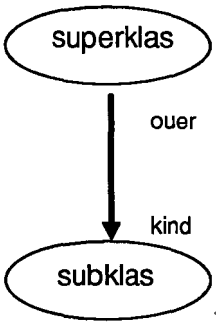


Figuur 3.2
Twee objekte as instansies van 'n klas Voertuig.

In figuur 3.2 toon ons die objek Motor en die objek Trein se attribute en lidfunksies. Let egter op dat in 'n objek-georiënteerde stelsel dit nie moontlik is om direk die inhoud van 'n objek se attribute te sien of te wysig nie. Die enigste manier waardeur die inhoud van 'n objek se attribute gelees of gewysig kan word is deur middel van die objek se lidfunksies. Laasgenoemde beteken dat die inhoud van 'n objek *geënkapsuleer* is. In ons voorbeeld is dit byvoorbeeld nie moontlik om vir objek Motor die waarde van Massa en die waarde van Versnelling te lees en dan self Massa / Versnelling te bereken nie. Die enigtse manier om Massa / Versnelling vir die objek Motor te bepaal is om die lidfunksie V_PerMassa te roep; ons skryf Motor.V_PerMassa. Enkapsulasie hou voordele in vir inligtingsekerheid; daar word later in die hoofstuk weer aandag hieraan geskenk.

3.2.2 Oorerwing tussen klasse

Objek-georiënteerde stelsels laat die gebruiker toe om nuwe klasse te skep deur bestaande klasse te gebruik. So 'n nuwe klas, genaamd 'n subklas van die bestaande klas, erf al die attribute en lidfunksies van die bestaande superklas (die nuwe klas kry fisies die attribute en metodes van die bestaande klas by), en die gebruiker mag nuwe lidfunksies en attribute spesifiseer vir die nuwe klas. Figuur 3.3 toon hoe oorerwing grafies in hierdie verhandeling voorgestel word. Let op dat die superklas in figuur 3.3 na verwys kan word as die ouer-klas en die subklas na verwys kan word as die kind-klas.

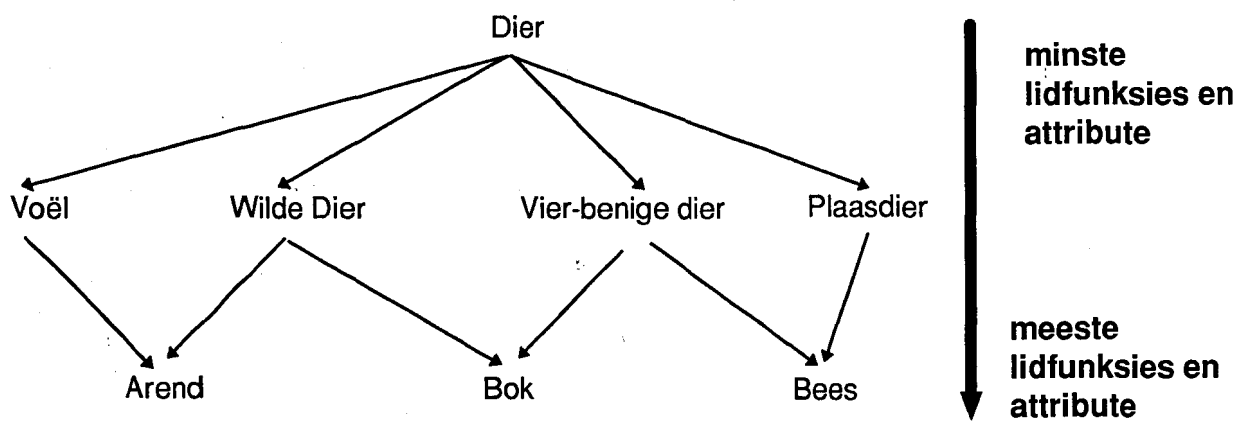


Figuur 3.3.
Voorstelling van enkel-oorerwing.

Let daarop dat oorerwing plaasvind op klasse, en nie op objekte nie. Dit is nie normaalweg in programmeertale wat objek-georiënteerdheid ondersteun, moontlik om 'n nuwe objek te vorm deur die datawaardes van ander objekte te kombineer nie. Ons spreek hierdie feit later weer aan maar vir nou is dit voldoende om te aanvaar dat nuwe objekte geskep kan word deur die datawaardes in die attribute van bestaande objekte te kombineer.

Sommige stelsels laat 'n klas toe om slegs van een klas te laat erf. Indien 'n stelsel toelaat dat 'n klas vanuit meer as een klas mag erf, laat die stelsel *meervoudige oorerwing* toe.

'n Stelsel wat slegs enkel-oorerwing toelaat se klasse vorm 'n hiërargie genaamd 'n klas hiërargie [1]. Indien 'n stelsel meervoudige oorerwing toelaat, is die klasse van so 'n stelsel gerangskik in 'n gewortelde gerigte asikliese grafiek. In die volgende hoofstuk gee ons aandag aan grafieke en sal ons 'n gewortelde gerigte asikliese grafiek definieer en verduidelik. Vir nou is dit voldoende om te weet hoe so 'n grafiek lyk. Figuur 3.4 toon 'n voorbeeld van 'n klas hiërargie; let op dat klasse laer af in die hiërargie gevorm is deur attribute en lidfunksies van klasse hoër op in die hiërargie te erf. Die klasse hoër op in die hierargie het dus tipies minder attribute en lidfunksies as klasse laer af in die hierargie.



Figuur 3.4.
'n Voorbeeld klas-hierargie.

In die algemeen kan 'n klas-hiërargie twee tipes verwantskappe tussen ouer- en kindklasse voorstel, naamlik *veralgemening-spesialisering* en *heel-deel* verwantskappe [14]. In hierdie verhandelig stel alle klas-hiërargie 'n veralgemenig-spesialisering verwantskap tussen ouer- en kindklasse voor. So byvoorbeeld is daar in die hierargie in figuur 3.4 ouer-klasse wat veralgemenings is van kind-klasse en die kind-klasse is spesialiserings van die ouer-klasse. So byvoorbeeld is die klas Bok 'n spesialisering van die klas Wilde Dier. Let op dat sekere klasse meer as een klas as ouer het. Sulke klasse is spesialiserings van meer as een klas, byvoorbleeld die Bok klas is 'n spesialisering van die algemene klas Wilde Dier en die algemene klas Vier-benige Dier.

Met die agtergrond van objek-georiënteerdheid wat ons nou het, is ons gereed om objek-georiënteerde rolprofiele te beskryf. Die beskrywing word gedoen want in die model wat in hoofstuk 6 geformuleer word, gebruik ons sulke objek-georiënteerde rolprofiele.

3.3 Rolprofiele as objekte (rolprofielobjekte)

Dit is moontlik om rolgebaseerde inligtingsekerheid op 'n objek-georiënteerde wyse te implementeer. In hierdie afdeling gaan ons kyk hoe ons 'n rolprofiel in die geheueruimte van 'n rekenaarstelsel kan stoor as 'n objek. Elkeen van hierdie objekte hanteer die magtiging van toegang na stelselhulpbronne vir 'n spesifieke rol in die stelsel. 'n Voordeel hiervan is dat nuwe rolprofielobjekte geskep kan word deur die voorregte wat in bestaande rolprofielobjekte gestoor is, te kombineer. Hierdie en ander voordele word bespreek.

'n Objek-georiënteerde rolprofiel ('n rolprofielobjek) is 'n entiteit op sy eie en ons kan, sonder om te sien watter voorregte in die objek gestoor is, die objek vra om toegang na 'n hulpbron te magtig deur 'n lidfunksie van die objek te roep. Ons begin deur 'n objek-georiënteerde rolprofiel (rolprofielobjek) te beskryf.

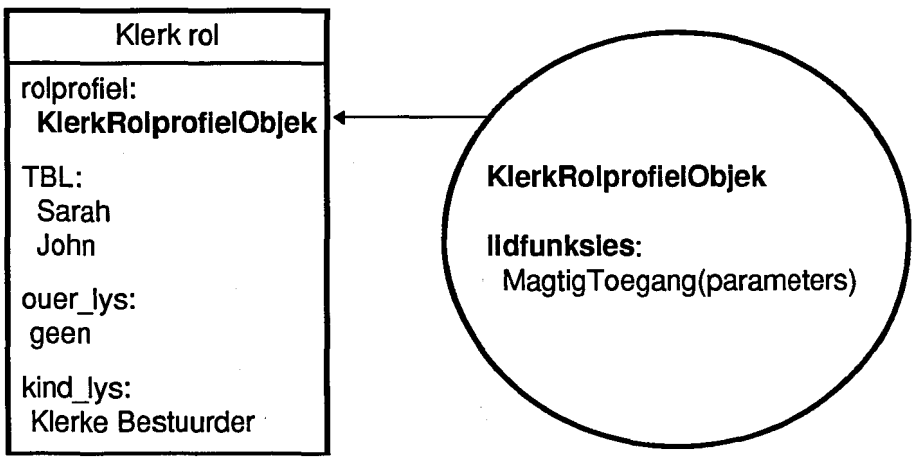
3.3.1 Rolprofielobjekte

Ons maak 'n rolgebaseerde inligtingsekerheidstelsel objek-georiënteerd deur te vereis dat die rolprofiele as objekte in die stelsel geïmplementeer moet word. Let op dat ons op hierdie stadium nie vereis dat 'n rol 'n objek is nie, ons vereis dat die rolprofiele objekte is. Presies hoe 'n rol geïmplementeer word, is nie nou belangrik nie - onthou net dat 'n rol byvoorbeeld 'n saamgestelde datatipe (soos 'n tabel) kan wees waarvan een inskrywing in die tabel die naam van 'n objek in die stelsel is en hierdie objek is 'n rolprofielobjek wat die voorregte vir hierdie rol stoor en magtiging vir die rol doen.

Onthou dat 'n rol soos voorheen gedefinieer bestaan uit 'n rolnaam, 'n naam van 'n rolprofiel in die stelsel sowel as 'n ouer- en kindlys. Voorheen het ons gesê dat die naam van die rolprofiel wat in 'n rol gestoor is, die naam is van 'n tabel in die stelsel waar die voorregte van 'n rol gestoor is. Die stelsel gebruik dan hierdie tabel om te bepaal of toegang na 'n stelselhulpbron gemagtig is indien 'n gebruiker aan so 'n rol gekoppel is. Van nou af is die naam van 'n rolprofiel in 'n rol nie meer die naam van 'n tabel in die stelsel nie, maar die naam van 'n objek in die stelsel. Die rolprofielnaam in

'n rol sal dus voortaan die unieke identifikasie wees van 'n objek (as instansie van 'n klas) in die stelsel.

Figuur 3.5 toon 'n voorbeeld van 'n rol Klerk en 'n rolprofielobjek KlerkRolprofielObjek.



Figuur 3.5
'n Voorbeeld rol met 'n rolprofielobjek.

Let op dat die rol Klerk 'n tabel is en een van die inskrywings in die tabel hou die naam van 'n rolprofielobjek, KlerkRolprofielObjek. Die rolprofielobjek is nie ook 'n tabel nie, maar 'n objek. Dit beteken ons kan nie die inhoud van die rolprofielobjek sien nie; al wat beskikbaar is, is lidfunksies om magtiging vir toegang na stelselhulpbronne te doen. In hierdie geval is slegs een lidfunksie, MagtigToegang, beskikbaar. Ons sal later kyk hoe die rolprofielobjek magtiging doen wanneer sy lidfunksies geroep word en watter datawaardes in die attribute van die rolprofiel gestoor word. In paragraaf 3.2 is gesê dat elke objek in 'n objek-georiënteerde stelsel is 'n instansie van 'n klas. In die volgende deel word die rolprofiel klas bespreek

3.4 Rolprofielklasse

In enige objek-georiënteerde stelsel is elke objek in die stelsel 'n instansie van 'n vooraf gedefinieerde klas soos wat op p.27 beskryf is. Die rolprofielobjek is 'n instansie van 'n rolprofielklas. Onthou dat ons gesê het dat 'n klas soos 'n raamwerk vir 'n objek is - 'n klas bevat attribute en lidfunksies wat bewerkings doen op die inhoud van die attribute, maar geen datawaardes is in die attribute gestoor nie (dit word in 'n objek gehou). Ons gaan vervolgens kyk hoe die rolprofielklas lyk.

Voordat ons die rolprofielklas bespreek, herhaal ons weer die definisie vir 'n voorreg en die definisie vir 'n rolprofiel .

'n Voorreg is 'n paar (x, m) waar x verwys na 'n beskermde data item en m is 'n nie-leë versameling van toegangsmetodes vir objek x.

x is enige beskermde data item soos byvoorbeeld 'n datalêer of enige hulpbron (soos 'n netwerk of drukker) ens. Die toegangsmetodes m beskryf die tipe toegang wat na die data item verleen word. In 'n stelsel met eenvoudige toegangsmetodes soos lees, skryf, uitvoer ens. is m 'n deelversameling van hierdie toegangsmetodes; x kan byvoorbeeld 'n datalêer soos DATA.TXT wees en m kan spesifiseer dat lees en skryf toegang na DATA.TXT verleen word.

'n Rolprofiel is 'n paar (rp_naam, v_lys). v_lys is 'n versameling voorregte soos hierbo beskryf is en rp_naam is 'n naam wat die rolprofiel uniek in die stelsel identifiseer.

Indien ons nie van objek-georiënteerde rolprofiele gebruik maak nie, sal die rolprofiele in 'n rolgebaseerde inligtingsekerheidstelsel 'n saamgestelde datatipe soos 'n tabel wees. Tabel 3.1 toon 'n voorbeeld van 'n rolprofiel wat as 'n tabel gestoor word. Let op dat die voorregtelys van die Klerk_rolprofiel bestaan uit transaksies en toelaatbare omstandighede waaronder die transaksies uitgevoer mag word.

rolprofiel Klerk		
Voorreg	Data item (x)	Toegansmetodes (m)
1.	transaksie DEPONEER	Terminaal_ID ∈ { TM1, TM2, TM3, TM4 }
2.	transaksie ONTTREK	Terminaal_ID ∈ { TM1, TM2 } en 0 <= bedrag <= 5000
3.	transaksie OORPLAAS	Terminaal_ID ∈ { TM1, TM2 } en 0 <= bedrag <= 10000 en 9:00 <= tyd <= 17:00

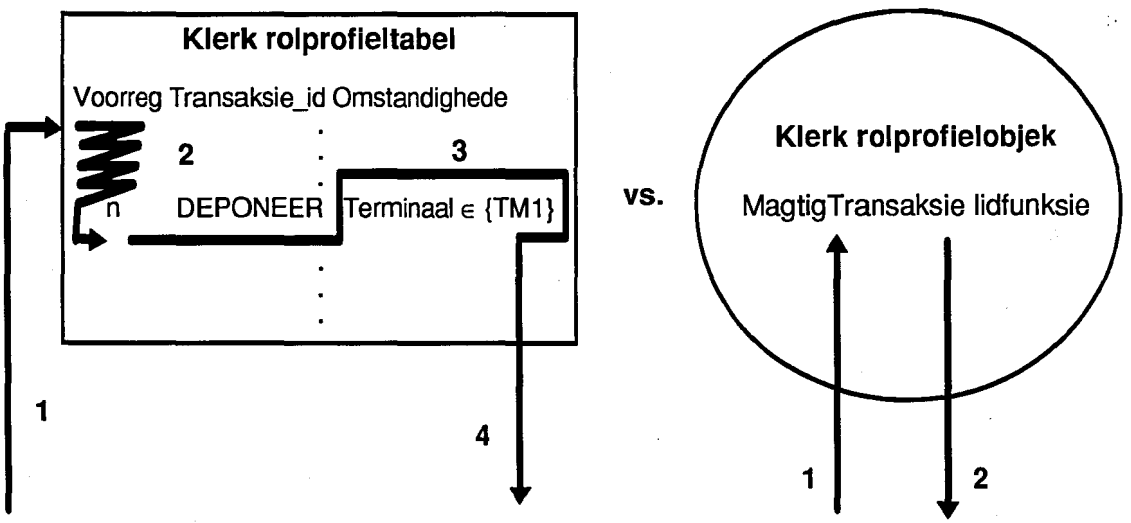
Tabel 3.1
Voorbeeld van 'n rolprofiel wat in 'n tabel gestoor word.

In hoofstuk 6 formuleer ons 'n model vir objek-georiënteerde rolgebaseerde inligtingsekerheid in 'n transaksieverwerking omgewing. Om die beskrywing van rolprofielobjekte makliker te maak en om beter in te pas by die model in hoofstuk 6, gaan ons rolprofielobjekte beskryf wat *magtiging doen vir die uitvoer van transaksies*. Dit beteken dat elke voorreg in die voorregtelys van 'n rolprofiel bestaan uit die naam van 'n transaksie en 'n lys van toelaatbare omstandighede waaronder die transaksie mag uitvoer.

Indien die stelsel rolprofiele as tabelle gestoor het soos in tabel 3.1, sou dit beteken dat elke keer wanneer 'n gebruiker 'n transaksie wil uitvoer, die stelsel moet bepaal aan watter rol die gebruiker gekoppel is en watter tabel die rolprofiel stoor. Hierna moet die stelsel deur die rolprofieltabel gaan soek vir die transaksie en self bepaal of die omstandighede waaronder die versoek gemaak is toelaatbaar is volgens die inskrywings in die rolprofieltabel. Wanneer ons egter van objek-georiënteerde rolprofiele gebruik maak, bepaal die stelsel nie self of 'n transaksie gemagtig is nie maar

roep 'n lidfunksie van die toepaslike rolprofielobjek. Die rolprofielobjek doen dan self die magtiging en stuur slegs 'n Magtig/Weier boodskap terug sonder dat die stelsel die voorregtelys van die objek hoef te deursoek. Ons noem hierdie eienskap van 'n objek *enkapsulasie*: die werking van die objek is nie bekend vir almal nie, die stelsel roep slegs lidfunksies en kry resultate terug sonder om die inhoud van die objek te sien.

Figuur 3.6 toon 'n diagrammatiese vergelyking van die stappe betrokke by die magtiging van 'n transaksie deur 'n rolprofiel as 'n tabel teenoor 'n rolprofiel as 'n objek. Let op dat die rolgebaseerde inligtingsekerheidstelsel meer van die werk self moet doen wanneer rolprofieltabelle gebruik word as wanneer rolprofielobjekte gebruik word.



Stappe met rolprofieltabel:

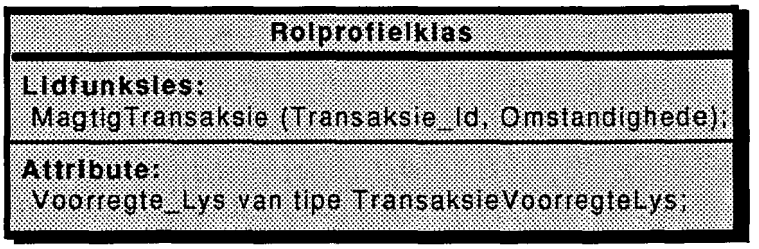
1. Vind naam van rolprofieltabel
2. Soek deur tabel vir transaksie
3. Soek deur toelaatbare omstandighede
4. Besluit of transaksie gemagtig is of nie

Stappe met rolprofielobjek:

1. Roep lidfunksie van rolprofielobjekte met nodige parameters.
2. Ontvang die Magtig of Weier boodskap

Figuur 3.6
Vergelyking van die stappe tussen magtiging met rolprofieltablle en rolprofielobjekte.

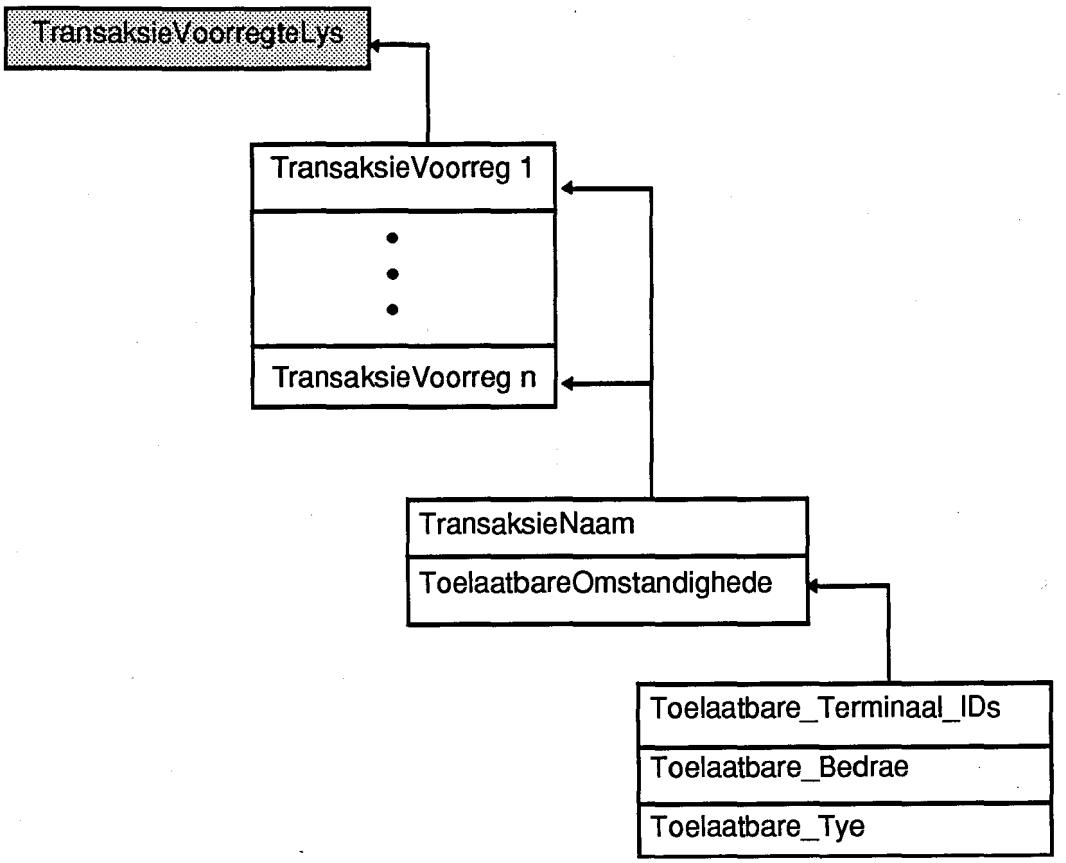
Vervolgens word die rolprofielklas beskryf. Die beskrywing van die rolprofielklas word stelselmatig uitgebrei, maar vir eers lyk die klas rolprofiel soos in figuur 3.7.



Figuur 3.7
Die rolprofielklas.

Let op dat dat die klas bestaan uit een lidfunksie en een attribuut. Die attribuut is 'n die voorregtelys vir hierdie rol. Ons het gesê ons gaan transaksies gebruik as stelselobjekte in die voorregtelys van ons rolprofielobjekte en daarom is die voorregtelys van die rolprofielklas 'n lys van transaksies en toelaatbare omstandighede vir elke transaksies.

Die voorregtelys vir die rolprofielklas is dus 'n saamgestelde datatipe. Figuur 3.8 toon die nuwe tipe TransaksieVoorregteLys diagrammaties en tabel 3.2 toon 'n voorbeeld van 'n TransaksieVoorregte_Lys. Die tipe TransaksieVoorregte_Lys is 'n lys waarin elke element van tipe TransaksieVoorreg is. Die tipe TransaksieVoorreg is 'n saamgestelde tipe en bevat die volgende: (TransaksieNaam, ToelaatbareOmstandighede). Die ToelaatbareOmstandighede tipe hang af van stelsel tot stelsel maar kan byvoorbeeld so lyk: (Toelaatbare_Terminaal_IDs, Toelaatbare_Bedrae, Toelaatbare_Tye).



Figuur 3.8
Diagrammatiese voorstelling van die tipe TransaksieVoorregteLys

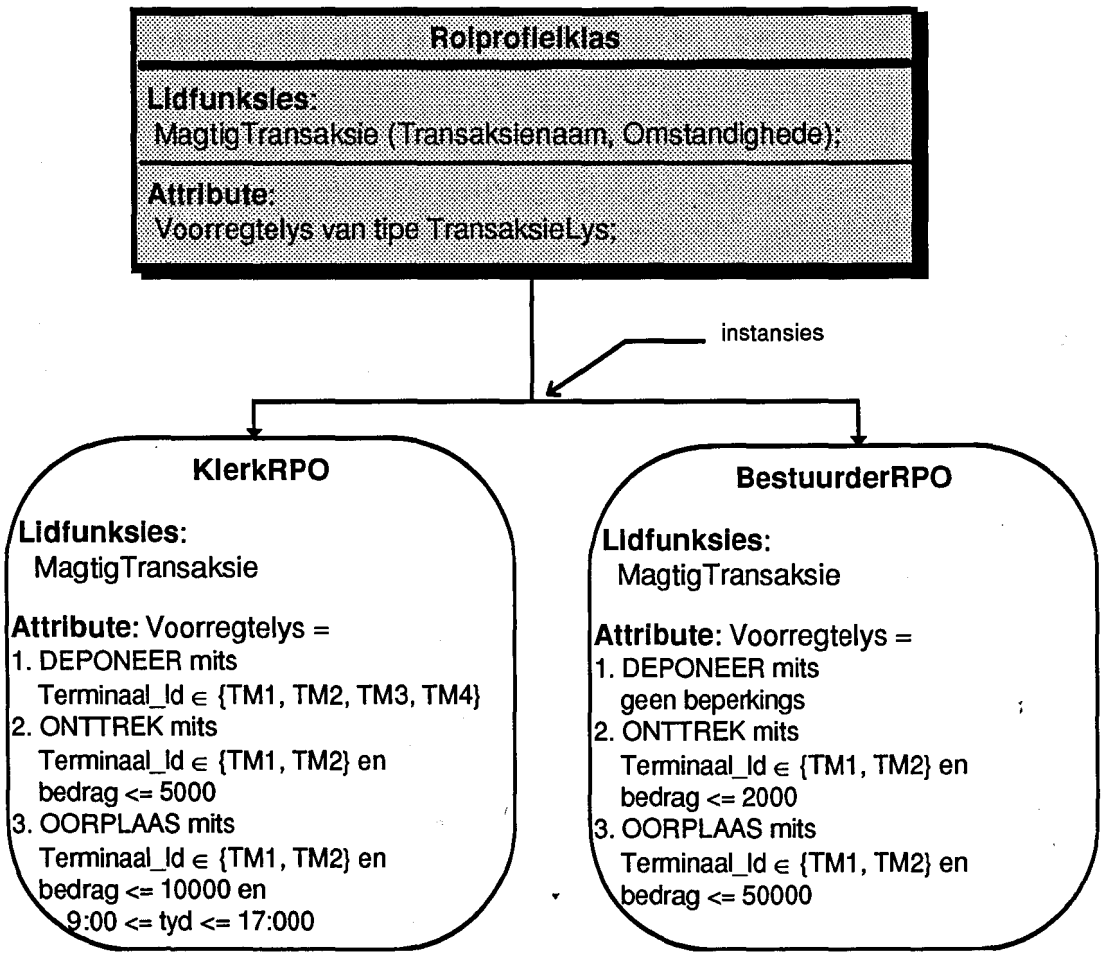
TransaksieVooregte_Lys		
Voorreg	TransaksieNaam	ToelaatbareOmstandighede
1.	transaksie DEPONEER	Terminaal_ID ∈ { TM1, TM2, TM3, TM4 }
2.	transaksie ONTTREK	Terminaal_ID ∈ { TM1, TM2 } en 0 <= bedrag <= 5000
3.	transaksie OORPLAAS	Terminaal_ID ∈ { TM1, TM2 } en 0 <= bedrag <= 10000 en 9:00 <= tyd <= 17:00

Tabel 3.2
Voorbeeld van TransaksieVooregte_Lys.

Nadat die rolprofielklas ontwerp is kan instansies van hierdie klas gevorm word. Elke instansie van die klas rolprofiel is 'n rolprofielobjek en verteenwoordig die voorregte van 'n rol in die stelsel. Die rolprofielobjek stoor in sy attribute die voorregte van 'n rol en het 'n lidfunksies MagtigTransaksie wat die uitvoer van transaksies magtig vir gebruikers wat gekoppel is aan 'n rol wat hierdie rol as rolprofielobjek het. Om bogenoemde te illustreer toon ons die volgende voorbeeld.

3.5 Magtiging met rolprofielobjekte

Veronderstel ons skep twee instansies van die rolprofielklas. Ons verwys voortaan na instansies van die rolprofielklas as rolprofielobjekte, of kortweg RPOs. Ons skep twee RPOs van die rolprofielklas en noem hullle KlerkRPO en BestuurderRPO. Figuur 3.9 toon die twee RPOs as instansies van die rolprofielklas.



Figuur 3.9.
'n Voorbeeld rolprofielklas en twee rolprofielobjekte.

Albei die RPOs (KlerkRPO en BestuurderRPO) is instansies van die rolprofielklas. Dit beteken albei het dieselfde attribute en lidfunksies maar verskillende waardes vir die attribute.

Elkeen van die twee RPOs het 'n lidfunksie MagtigTransaksie. Die lidfunksie ontvang as parameters die transaksienaam soos byvoorbeeld DEPONEER en 'n lys van omstandighede waaronder die transaksie versoek is. Hierdie lys kan byvoorbeeld wees {Bedrag = 1000, tyd = 15:30, terminaal = TM2}. Die lidfunksie MagtigTransaksie deursoek die voorregtelys en toets of die transaksie Transaksienaam in die vooregtelys voorkom. Indien die transaksie wel in die voorregtelys voorkom, toets dit of elke element van die lys Omstandighede deel is van die toelaatbare omstandighede van die transaksie in die voorregtelys. Indien

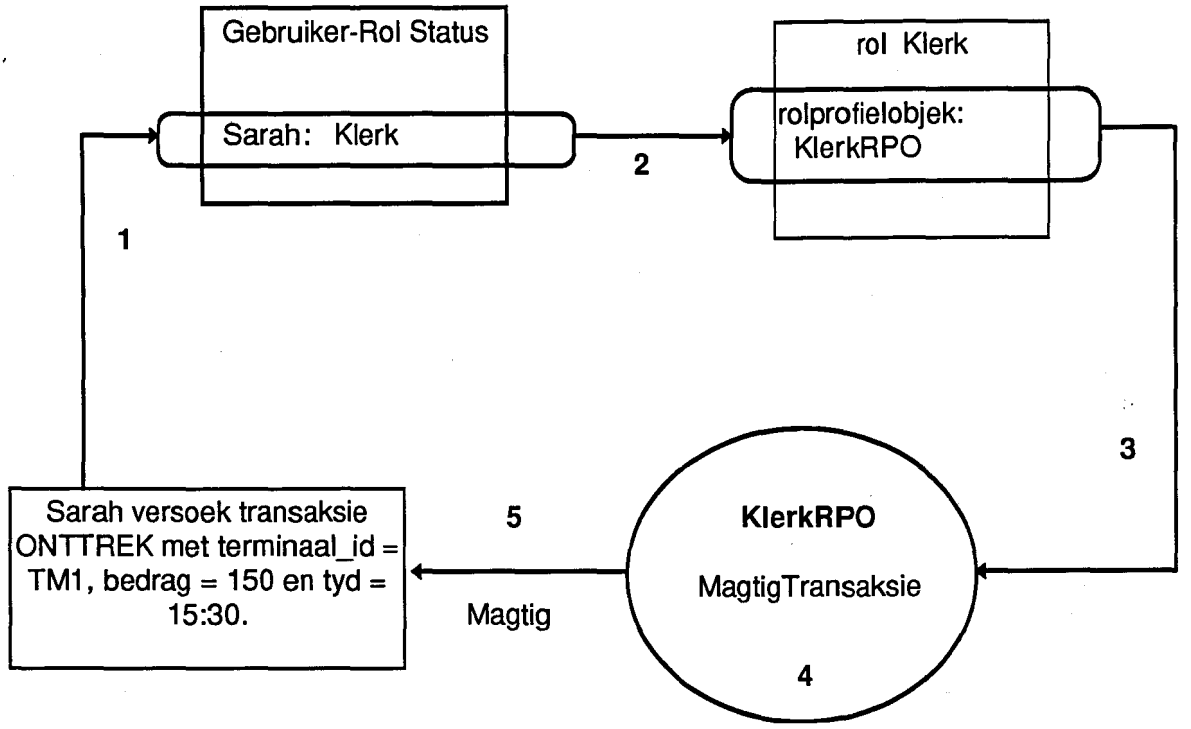
bogenoemde waar is stuur die funksie 'n Magtig boodskap terug, andersins word 'n Weier boodskap teruggestuur.

3.5.1 Voorbeeld van magtiging in 'n objek-georiënteerde rolgebaseerde stelsel:

Veronderstel gebruiker Sarah is gekoppel aan die rol Klerk en sy wil die transaksie ONTTREK uitvoer. Die omstandighede waaronder die transaksie versoek is, is Terminaal_id = TM2; bedrag = 150 en tyd = 15:00. Om die transaksie te magtig word die volgende stappe deur die stelsel gevolg:

1. Die stelsel lees in 'n tabel aan watter rol gebruiker Sarah tans gekoppel is. (Hierdie is die rol wat Sarah gekies het toe sy by die stelsel aangeteken het, nie die rolle in die toegangsbeheerlys nie). Die stelsel sien dat Sarah as rol Klerk in die stelsel aangeteken is.
2. Die volgende stap is om in die tabel waar inligting oor die rol Klerk gestoor word, die unieke identifikase van die RPO wat magtiging vir rol Klerk te lees. Die stelsel lees KlerkRPO as die RPO wat magtiging hanteer vir rol Klerk.
3. Vervolgens roep die stelsel die MagtigTransaksie lidfunksie van RPO KlerkRPO, ons skryf KlerkRPO.MagtigTransaksie en stuur as parameters die lys van omstandighede waaronder die transaksie versoek is.
4. Die rolprofielobjek KlerkRPO ontvang as parameters die transaksienaam (ONTTREK) en die lys van omstandighede waaronder die transaksie versoek is (Terminaal_Id = TM2; bedrag = 150 en tyd = 15:00). Die lidfunksie MagtigTransaksie deursoek die voorregtelys en toets of die transaksie ONTTREK in die vooregtelys voorkom. Omdat die transaksie wel voorkom, word getoets of elke element van die lys van omstandighede deel van die toelaatbare omstandighede van die transaksie in die voorregtelys.
5. Omdat die Terminaal_Id in die lys toelaatbare Terminaal_Ids is en die bedrag <= 5000 is, stuur die funksie 'n Magtig boodskap terug, andersins sou 'n Weier boodskap terug gestuur geword het.

Figuur 3.10 toon hierdie stappe diagrammaties.



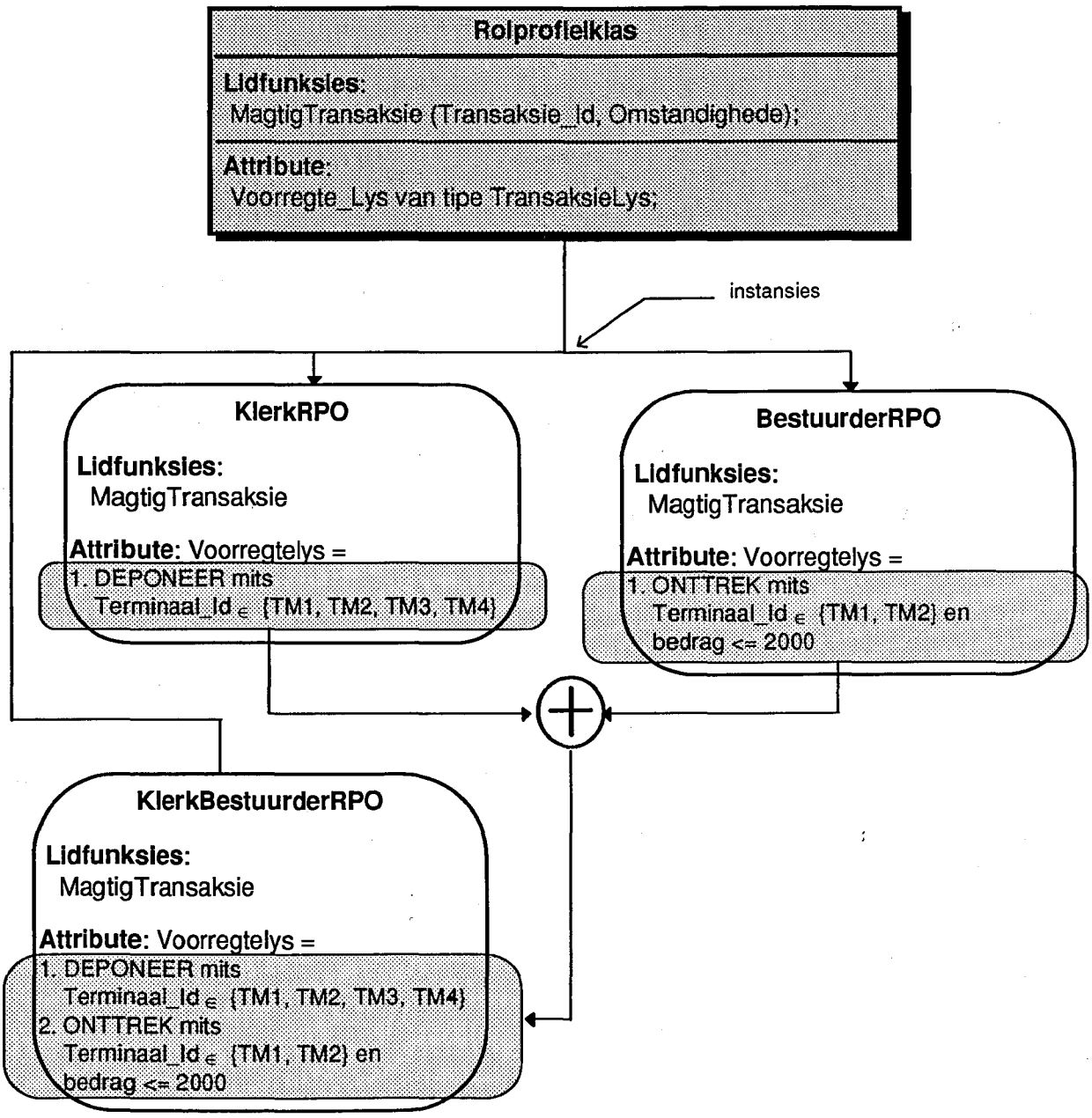
Figuur 3.10
Voorbeeld van magtiging met objek-georiënteerde rolgebaseerde inligtingsekerheid.

Tot dusver het ons nog nie aandag gegee aan die feit dat rolle verband kan hou nie. In die volgende afdeling kyk ons hoe ons die verwantskap tussen rolle en die verwantskap tussen rolprofielobjekte bymekaar uitbring.

3.6 Rolprofielobjekte wat datawaardes erf van ander rolprofielobjekte

Onthou dat ons reeds in hoofstuk 2 gesê het dat ons 'n rolgebaseerde inligtingsekerheidstelsel so kan uitbrei dat een rol toegang het na die voorregte van 'n ander rol. So is dit moontlik dat 'n rol KlerkBestuurder toegang het na die voorregte van 'n rol Klerk. In hierdie afdeling kyk ons wat die effek hiervan is op rolprofielobjekte.

In suiwer objek-georiënteerde stelsels kan oorerwing toegepas word op klasse. Dit beteken dat 'n nuwe klas gevorm kan word deur die attribute en lidfunksies van bestaande klasse saam te voeg. Let daarop dat 'n objek 'n instansie van 'n klas is, m.a.w. 'n objek is 'n klas definisie waarvoor daar werklike datawaardes in die attribuut velde van die klas ingevul is. In hierdie studie vereis ons dat, net soos wat nuwe klasse gevorm word deur bestaande klasse te kombineer, kan nuwe objekte gevorm word deur die attribute en hulle datawaardes van bestaande objekte te kombineer. Figuur 3.11 stel bogenoemde voor.



Figuur 3.11
Voorbeeld van die skep van rolprofielobjekte deur ander rolprofielobjekte se voorregte te erf.

In figuur 3.11 is daar drie rolprofielobjekte (KlerkRPO, BestuurderRPO en KlerkBestuurderRPO). Al drie die RPOs is instansies van die rolprofielklas. KlerkBestuurderRPO word geskep as 'n instansie van die rolprofielklas wat die voorregte van die RPOs KlerkRPO en BestuurderRPO erf. Let op dat KlerkBestuurderRPO se Voorregtelys die vereniging is van die Voorregtelys van die ander twee RPOs in die diagram.

Wanneer 'n nuwe RPO geskep word en die nuwe RPO se voorregte die vereniging van een of meer ander RPOs se voorregte is sê ons die nuwe RPO *erf* die ander RPOs se

voorregte. Die skep van nuwe rolprofielobjekte deur ander rolprofielobjekte se voorregte te erf maak dit makliker vir sekerheidsbestuurder om nuwe rolle en rolprofielobjekte te skep vir 'n rolgebaseerde inligtingsekerheidstelsel.

Om bogenoemde te motiveer, neem die voorbeeld waar daar in 'n rolgebaseerde stelsel reeds twee rolle is, naamlik Klerk met RPO KlerkRPO en Bestuurder met RPO BestuurderRPO. Die sekerheidsbestuurder kan maklik die nuwe rol, KlerkBestuurder vorm en die nuwe RPO KlerkBestuurderRPO se voorregte is bloot die vereniging van die voorregte in KlerkRPO en BestuurderRPO.

In hoofstuk 2 het ons 'n rol gedefinieer. Ons gee weer die formele definisie.

'n Rol is 'n veeltal (r_naam, rp_naam, o_lys, k_lys). r_naam is 'n unieke naam vir die rol in die stelsel. rp_naam is 'n unieke naam vir 'n rolprofiel in die stelsel wat hierdie spesifieke rol beskryf. o_lys is 'n lys rolle wat ouers is van van hierdie rol. k_lys is 'n lys van rolle wat kinders is van hierdie rol. o_lys en k_lys kan ook leeg wees.

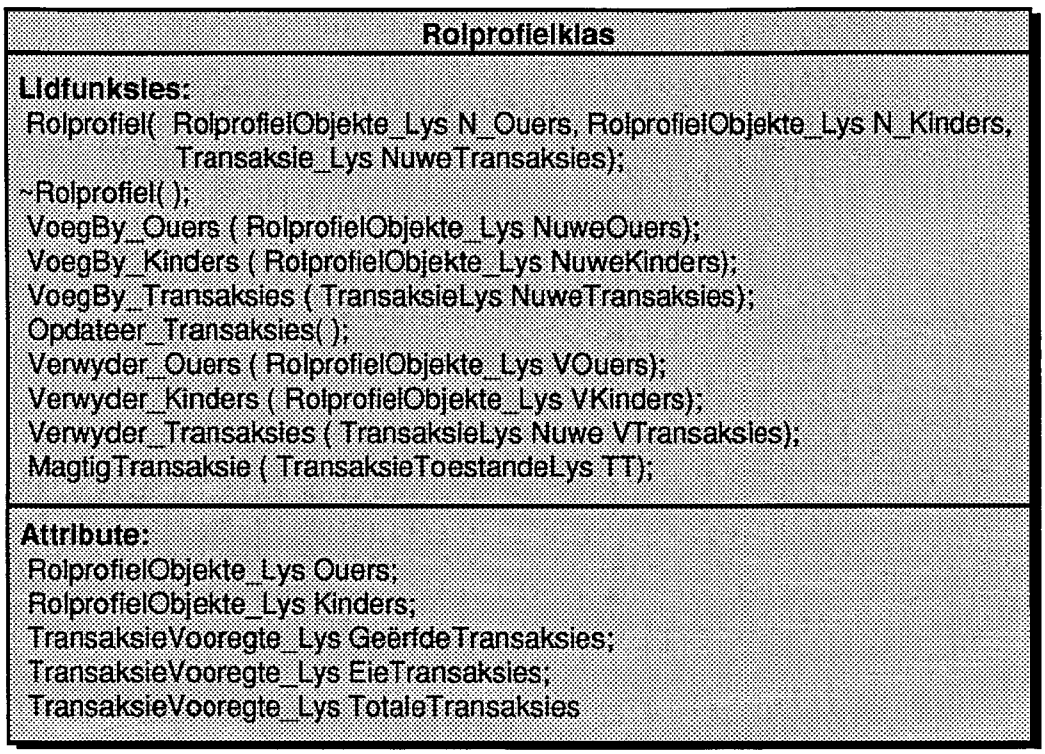
Let op dat 'n rol onder ander 'n lys bevat van ouer- en 'n lys van kindrolle. Onthou dat ouers van 'n rol rolle is waaruit die rol voorregte geërf het en kinders is rolle vir wie hierdie rol sy voorregte gee. Vir rolprofielobjekte geld dit ook: 'n rolprofielobjek erf voorregte van ouers en gee voorregte aan kinders. Om rolprofielobjekte in stand te hou is dit nodig dat 'n RPO weet watter RPOs sy ouers is en watter RPOs sy kinders is. Een voorbeeld waarom laasgenoemde nodig is, is dat wanneer 'n bestaande RPO 'n voorreg bykry, moet die RPO die nuwe voorreg ook oordra aan al sy kinders.

In die volgende afdeling brei ons die rolprofielklas uit sodat elke rolprofielobjek sy ouers en kinders in sy attribute stoor. Die rolprofielklas wat vervolgens bespreek word is die rolprofielklas wat gebruik word in die model (ORITO) wat in hoofstuk 6 geformuleer word.

3.7 Die uitgebreide rolprofielklas

Om voorsiening te maak vir die feit dat rolprofielobjekte voorregte kan erf van ouer-rolprofielobjekte brei ons nou die rolprofielklas uit. Ons voeg heelwat attribute en lidfunksies by wat elkeen beskryf word.

Figuur 3.12 toon die uitgebreide rolprofielklas vir die rolprofielobjekte wat geskep gaan word in die model wat in hoofstuk 6 geformuleer word. Rolprofielobjekte word geskep as instansies van hierdie klas. Ons toon later dat daar verskillende rolprofielklasse in 'n stelsel gebruik kan word.

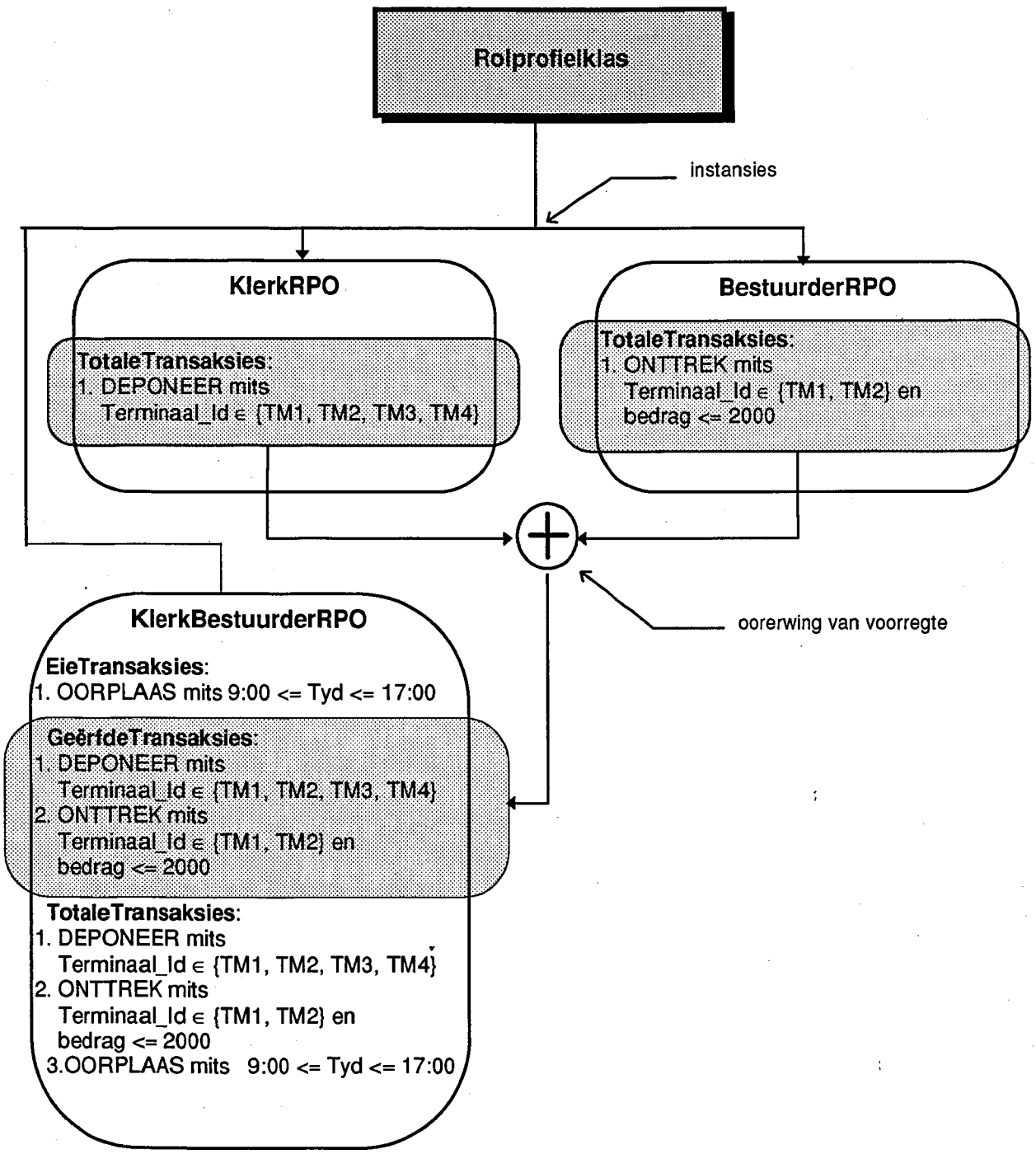


Figuur 3.12
Die uitgebreide rolprofielklas.

3.7.1 Verduideliking van die rolprofielklas

In die verduideliking gaan ons weer die voorbeeld gebruik wat in figuur 3.11 geskets is. Die voorbeeld word in 'n vereenvoudigde vorm in figuur 3.13 getoon. Let op dat daar drie rolprofielobjekte (KlerkRPO, BestuurderRPO en KlerkBestuurderRPO) is. Al drie die RPOs is instansies van die nuwe uitgebreide rolprofielklas soos in figuur 3.12. KlerkBestuurderRPO is ook 'n instansie van die rolprofielklas maar erf voorregte van die ander twee RPOs. Die voorregte wat geërf word, word in 'n nuwe attribuut GeërfdeTransaksies gestoor; dit word later volledig verduidelik.

In hierdie verduideliking verwys ons dikwels na attribute en lidfunksies van 'n rolprofielobjek. Let op dat wanneer ons skryf rpo_1.Ouers dan verwys ons na die lys van Ouers van die rolprofielobjek rpo_1. Onthou dat elkeen van die rolprofielobjekte wat in figuur 3.13 getoon word al die lidfunksies en attribute van die rolprofielklas in figuur 3.12 het. So byvoorbeeld het rolprofielobjek KlerkRPO 'n attribuut TotaleTransaksies. Indien dit nie duidelik is dat ons na KlerkRPO se TotaleTransaksies verwys nie, skryf ons KlerkRPO.TotaleTransaksies.



Figuur 3.13.
Drie RPOs as instansies van die uitgebreide rolprofielklas.

3.7.2 Attribute van die rolprofielklas:

Die attribute van die rolprofielklas word vervolgens bespreek. Tabel 3.3 gee 'n opsomming van die attribute. Na die tabel word die attribute in meer besonderhede bespreek.

Attribuut	Beskrywing
Ouers	Stoor die name van al die rolprofielobjekte gekoppel aan hierdie RPO as ouers.
Kinders	Stoor die name van al die rolprofielobjekte gekoppel aan hierdie RPO as kinders van die RPO.
GeërfdeTransaksies	'n Lys van transaksievoorregte wat geërf is van ouers RPOs.
EieTransaksies	'n Lys van transaksievoorregte wat nie geërf is van ander RPOs nie, maar ekstra bygevoeg is tot die rol se RPO.
TotaleTransaksies	Al die transaksievoorregte van die RPO. Die TotaleTransaksies is die vereniging van GeërfdeTransaksies en EieTransaksies.

Tabel 3.3
Opsomming van die attribute van die rolprofielklas.

Ouers is 'n lys van tipe RolProfielObjekte. Die tipe RolProfielObjek is 'n unieke naam van 'n instansie van die rolprofielklas. Die lys Ouers stoor die name van al die rolprofielobjekte wat gekoppel is aan hierdie RPO as ouers van die RPO. In die voorbeeld is KlerkRPO en BestuurderRPO albei ouers van KlerkBestuurderRPO. Dit beteken dat KlerkBestuurderRPO.Ouers = { KlerkRPO, BestuurderRPO }. 'n Rolprofielobjek se voorregte word aangevul met die voorregte van elke ouer rolprofielobjek in die lys Ouers; ons verduidelik dit later.

Kinders is ook 'n lys van tipe RolProfielObjekte. Die lys stoor die name van al die rolprofielobjekte wat gekoppel is aan hierdie RPO as kinders van die RPO. In die voorbeeld is KlerkBestuurderRPO 'n kind van KlerkRPO sowel as BestuurderRPO. Dit beteken dat KlerkRPO.Kinders = {KlerkBestuurderRPO} en BestuurderRPO.Kinders = {KlerkBestuurderRPO}. 'n Rolprofielobjek se kinders erf al sy transaksievoorregte.

GeërfdeTransaksies is van tipe TransaksieVoorregte_Lys en stoor die transaksievoorregte wat geërf is van ouer RPOs. In die uitgebreide rolprofielklas is dit nodig om te weet watter transaksievoorregte geërf is van ander rolprofielobjekte en watter transaksievoorregte eie is aan die rolprofielobjek. Om hierdie rede het ons 'n attribuut waar al die transaksies wat van ander rolprofielobjekte geërf is, gestoor word. In die voorbeeld is :

KlerkBestuurderRPO.GeërfdeTransaksies = {
1. DEPONEER mits
   Terminaal_Id ∈ {TM1, TM2, TM3, TM4}
2. ONTTREK mits
   Terminaal_Id ∈ {TM1, TM2} en
   bedrag <= 2000 }.

Die tipe TransaksieVoorregte_Lys is 'n lys waarin elke element van tipe TransaksieVoorreg is. Die tipe TransaksieVoorreg is 'n saamgestelde tipe en bevat die volgende:
(TransaksieNaam, ToelaatbareOmstandighede).

Die ToelaatbareOmstandighede tipe hang af van stelsel tot stelsel maar kan byvoorbeeld so lyk: (Toelaatbare_Terminaal_IDs, Toelaatbare_Bedrae, Toelaatbare_Tye). In figuur 3.8 op p.36 is die tipe TransaksieVoorregte_Lys diagrammaties getoon.

EieTransaksies is van tipe TransaksieVoorregte_Lys en stoor die transaksievoorregte wat nie geërf is van ouer RPOs nie maar ekstra bygevoeg is tot die rol se RPO. In die voorbeeld het die rolprofielobjek KlerkBestuurderRPO geërfde transaksievoorregte en eie transaksievoorregte,
KlerkBestuurderRPO.EieTransaksies = {OORPLAAS mits 09:00 <= Toelaatbare_Tye <= 17:00 }.

TotaleTransaksies is ook van tipe TransaksieVoorregte_Lys en stoor al die transaksies van 'n rol se RPO. TotaleTransaksies = EieTransaksies ∪ GeërfdeTransaksies.
In die voorbeeld is:
KlerkBestuurderRPO.TotaleTransaksies = {
1. DEPONEER mits
   Terminaal_Id ∈ {TM1, TM2, TM3, TM4}
2. ONTTREK mits
   Terminaal_Id ∈ {TM1, TM2} en
   bedrag <= 2000
3. OORPLAAS mits 09:00 <= Toelaatbare_Tye <= 17:00 }.

3.7.3 Lidfunksies van die rolprofielklas:

Die lidfunksies van die rolprofielklas word vervolgens bespreek. Tabel 3.4 gee 'n opsomming van die lidfunksies. Na die tabel word die lidfunksies in meer besonderhede bespreek.

Lidfunksie	Beskrywing
konstruktor Rolprofiel	Skep 'n nuwe instansie van die rolprofielklas.
destruktor ~Rolprofiel	Vernietig 'n rolprofielobjek
VoegBy_Ouers	Voeg nuwe ouer-RPOs by 'n RPO se lys van Ouers.
VoegBy_Kinders	Voeg nuwe kind-RPOs by 'n RPO se lys van kinders.
VoegBy_Transaksie	Voeg nuwe (eie) transaksievoorregte by 'n RPO.
Opdateer_Transaksies	Sorg dat 'n RPO se Geërfde- en Totale transaksievoorregte op datum is.
Verwyder_Ouers	Ontkoppel ouer-RPOs van 'n rolprofielobjek
Verwyder_Kinders	Ontkoppel kind-RPOs van 'n rolprofielobjek
Verwyder_Transaksies	Onttrek eie transaksievoorregte van 'n rolprofielobjek
MagtigTransaksie	Magtig die uitvoer van 'n transaksie vir gebruikers gekoppel aan 'n rol

Tabel 3.4
Opsomming van die lidfunksies van die rolprofielklas.

Die **konstruktor Rolprofiel** skep 'n nuwe instansie van die Rolprofielklas. Die konstruktor ontvang as parameters 'n lys van ouers (N_Ouers) en 'n lys van kinders (N_Kinders) sowel as 'n lys van nuwe voorregte (parameter Voorregte). Die ouers is rolprofielobjekte wie se voorregte oorgeërf moet word na die nuwe RPO wat geskep word. Die kinders is RPOs aan wie die nuwe objek se voorregte oorgedra moet word nadat dit geskep is. Die konstruktor stel die nuwe RPO se attribuut EieTransaksies se inhoud gelyk aan die inhoud van die parameter NuweTransaksies.
In die voorbeeld sal dit beteken dat toe die rolprofielobjek KlerkBestuurderRPO geskep is, is die rolprofielklas se konstruktor geroep met parameters N_Ouers = { KlerkRPO, BestuurderRPO} , N_Kinders = { } en Voorregte = { OORPLAAS mits 09:00 <= Toelaatbare_Tye <= 17:00 }.

Om te sorg dat die attribute GeërfdeTransaksies, Ouers en Kinders van alle rolprofielobjekte in die stelsel op datum bly, word die nuwe RPO se lidfunksies VoegBy_Ouers(N_Ouers), en VoegBy_Kinders(N_Kinders) geroep. Hierdie lidfunksies word later verduidelik.

Sodra die nuwe RPO geskep is sorg die konstruktor dat die VoegBy_Ouers lidfunksie van elke RPO in die Kinders lys geroep word. Die nuwe RPO se naam word as parameter gestuur (Meer hieroor later).

Die **destruktor ~Rolprofiel** lidfunksies word geroep wanneer 'n RPO vernietig word en sal tipies gebeur wanneer 'n rol nie meer in 'n organisasie bestaan of deur 'n ander rol vervang word. Die destruktor roep die Verwyder_Ouers lidfunksie van elke RPO in sy Kinders lys en stuur as parameter sy naam. Op soortgelyke wyse word die Verwyder_Kinders lidfunksie van elke RPO in sy Ouers lys geroep.

Die **VoegBy_Ouers** lidfunksie van 'n rolprofielobjek neem die lys Nuwe_Ouers ('n parameter) en voeg dit by die waarde van sy attribuut Ouers. Hierna moet gesorg word dat die attribuut GeërfdeTransaksies en TotaleTransaksies steeds op datum is. Om dit te doen word hierdie RPO se lidfunksie Opdateer_Transaksies geroep.

Die **VoegBy_Kinders** lidfunksie moet sorg dat die waarde van die RPO se attribuut Kinders op datum is en dat elke kind RPO in die lys Kinders weet dat hierdie RPO sy ouer is. Om dit te doen word eers bepaal watter kinders in die parameter Nuwe_Kinders is werklik nuwe kinders:

Werklik_Nuwe_Kinders = Nuwe_Kinders - Kinders.

Veronderstel dat in die voorbeeld was KlerkBestuurderRPO nie geskep met KlerkRPO as ouer nie. Ons veronderstel egter dat op 'n latere stadium word die VoegBy_Kinders lidfunksie van KlerkRPO geroep om KlerkBestuurderRPO een van sy kinders te maak. KlerkRPO.VoegBy_Kinders(Nuwe_Kinders = KlerkBestuurderRPO). Die lidfunksie bepaal dat
Werklik_Nuwe_Kinders = {KlerkBestuurderRPO} - {ϕ} = {KlerkBestuurderRPO}

Hierna word die lidfunksie VoegBy_Ouers van elke RPO in die lys Werklik_Nuwe_Kinders geroep en as parameter word hierdie RPO (waarvoor

VoegBy_Kinders geroep is) se naam gestuur. Dit beteken in die voorbeeld wat hierbo veronderstel is, dat KlerkBestuurderRPO.VoegBy_Ouers(Nuwe_Ouers = KlerkRPO) geroep word.
Laastens word gesorg dat
Kinders = Nuwe_Kinders ∪ Kinders vir KlerkRPO.

Die lidfunksie **VoegBy_Transaksies** neem die parameter NuweTransaksies en stel
EieTransaksies = NuweTransaksies ∪ EieTransaksies en
TotaleTransaksies = EieTransaksies ∪ GeërfdeTransaksies.
Hierna word die lidfunksie Opdateer_Transaksies vir elke RPO in die RPO se lys Kinders geroep.

Opdateer_Transaksies sorg dat die waardes van 'n RPO attribute GeërfdeTransaksies en TotaleTransaksies steeds op datum is. Om dit te doen stel die lidfunksie die waardes van die attribute as volg:

Gestel dit geld vir 'n RPO dat Ouers = (rpo_1, rpo_2, ... , rpo_n) , dan is
GeërfdeTransaksies = rpo_1.TotaleTransaksies ∪ rpo_2. TotaleTransaksies ∪ ... ∪ rpo_n. TotaleTransaksies, en
TotaleTransaksies = GeërfdeTransaksies ∪ EieTransaksies.

Indien die waarde van TotaleTransaksies verander het, word die lidfunksies Opdateer_Transaksies vir elke kind RPO in die die RPO se lys Kinders geroep om elkeen se TotaleTransaksies attribuut op datum te hou.

Verwyder_Ouers word vir 'n RPO geroep indien dit ontkoppel word van sekere ouer RPOs. Dit beteken dat hierdie RPO nie meer die ouer RPOs se transaksies mag erf nie. Die funksies stel eers

Ouers = Ouers - VOuers (VOuers is 'n lys van ouer RPOs wat as parameter gestuur is)

Om te sorg dat hierdie RPO se attribuut GeërfdeTransaksies en TotaleTransaksies sowel as elke kind van hierdie RPO se attribute op datum bly, word Opdateer_Transaksies vir hierdie rolprofielobjek geroep.

Verwyder_Kinders word vir 'n RPO geroep indien dit ontkoppel word van sekere kind RPOs. Dit beteken dat hierdie RPO nie meer sy transaksies mag oordra na kinders in die lys Vkinders ('n parameter vir die funksie) nie. Die funksies stel eers

Kinders = Kinders - VKinders

Om te sorg dat die waardes van die attribute GeërfdeTransaksies en TotaleTransaksies van elke kind RPO op datum bly, word die lidfunksie Verwyder_Ouers van elke RPO in die lys Vkinders geroep. As parameter word hierdie objek se identifikasie gestuur.

Die lidfunksie **Verwyder_Transaksies** word geroep om al die transaksies in die lys VTransaksies ('n parameter vir die funksie) uit hierdie RPO se attribuut EieTransaksies te verwyder. Die funksie stel:

EieTransaksies = EieTransaksies - VTransaksies en
TotaleTransaksies = GeërfdeTransaksies ∪ EieTransaksies.

Hierna word die Opdateer_Transaksies lidfunksie vir elke RPO in die lys Kinders geroep.

Al die lidfunksies wat tot dusver bespreek is word gebruik om die rolprofielobjek se attribuutwaardes in stand te hou. Die lidfunksie **MagtigTransaksie** word gebruik om 'n transaksie te magtig. Hierdie lidfunksie ontvang die naam van 'n transaksie en 'n lys van omstandighede wat aandui waar en hoe die transaksie versoek is, byvoorbeeld die tyd van die transaksie en die identifikasie van die terminaal waar dit versoek is. Voorlopig sê ons MagtigTransaksie toets of die transaksie voorkom in die rolprofielobjek se attribute TotaleTransaksies en kyk dan of die transaksie aan die beperkings van die transaksie (wat ook in die lys TotaleTransaksies gestoor word) voldoen en stuur 'n Waar terug om aan die dui die transaksie is gemagtig, anderins word 'n Onwaar terug gestuur aan die objek/stelsel wat MagtigTransaksie geroep het om aan te dui die transaksie nie uitgevoer mag word nie. Die formele beskrywing van magting van transaksies word in hoofstuk 6 gedoen.

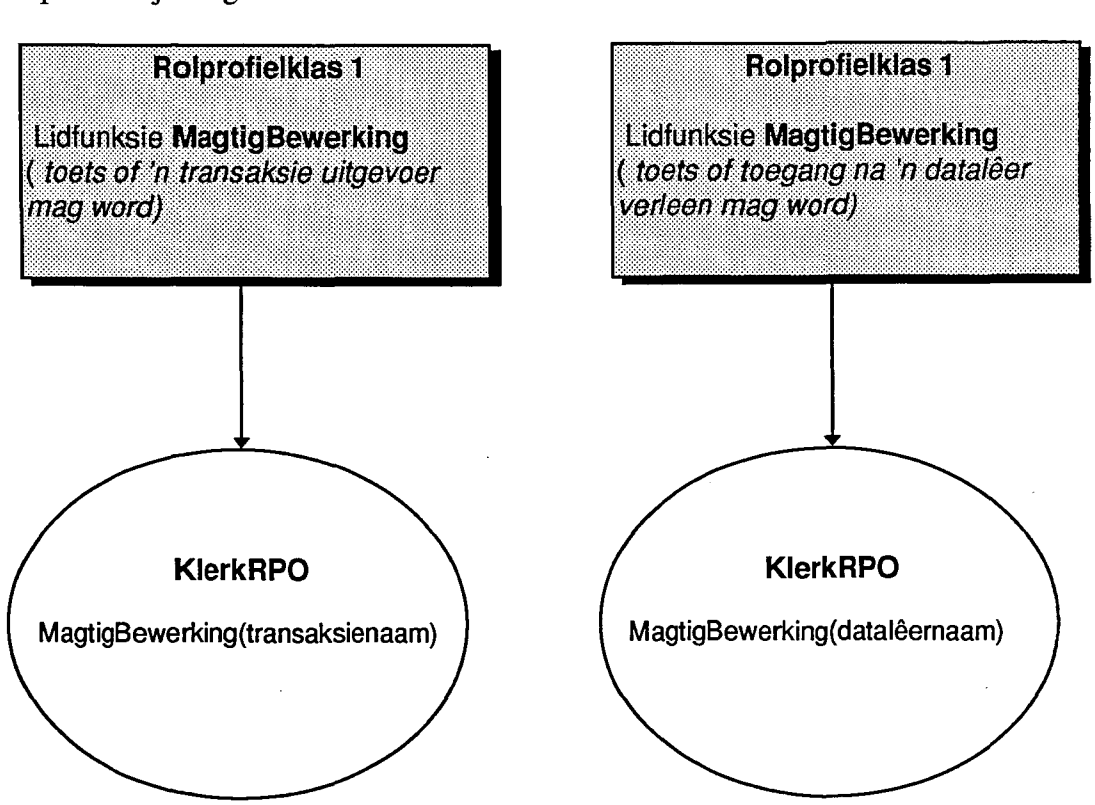
In die voorbeeld is dit moontlik dat die MagtigTransaksie lidfunksie van KlerkRPO geroep kan word met parameters TransaksieToestandeLys = (Transaksienaam = DEPONEER, Terminaal = TM1, bedrag = 500 en tyd = 15:30).
KlerkRPO sal 'n Magtig boodskap terugstuur want die transaksie DEPONEER is in die TotaleTransaksieLys en Terminaal_id = TM1 ∈ {TM1, TM2, TM3, TM4}. Daar is nie 'n beperking op die bedrag of tyd nie.

3.8 Die gebruik van verskillende rolprofielklasse.

Alle rolprofielobjekte in die stelsel hoef nie instansies van dieselfde klas te wees nie. In so 'n stelsel kan dit moontlik wees dat sekere rolle slegs transaksies mag uitvoer en ander rolle programme mag uitvoer. Ons sal moontlik 'n rolprofielklas hê met 'n MagtigBewerking lidfunksie wat toets of 'n rol 'n transaksie mag uitvoer en 'n ander rolprofielklas met 'n MagtigBewerking lidfunksie wat toets of 'n rol toegang het na 'n stelselhulpbron soos 'n programlêer. Die stelsel roep egter steeds net die MagtigBewerking lidfunksie van die toepaslike rolprofielklas.

Figuur 3.14 toon 'n voorbeeld van 'n stelsel waar meer as een rolprofielklas gebruik word. Let op dat daar twee rolprofielobjekte in die stelsel is (KlerkRPO en BestuurderRPO). Albei die rolprofielobjekte het 'n lidfunksie MagtigBewerking maar die lidfunksie reageer verskillend in elkeen van die rolprofielobjekte. By die KlerkRPO toets MagtigBewerking of 'n transaksie uitgevoer mag word en by BestuurderRPO toets MagtigBewerking of toegang na 'n datalêer geldig is. Die stelsel hoef egter nie te weet wat binne in 'n rolprofielobjek aangaan nie. Die stelsel roep die MagtigBewerking

lidfunksie en hoef nie eens te weet dat die magtiging verskillend in verskillende rolprofielobjekte gedoen word nie.



Figuur 3.14
Rolprofielobjekte as instansies van verskillende rolprofielklasse.

Bostaande toon 'n voorbeeld van veelmorfinisme ("*polymorphism*") - waar twee objekte dieselfde lidfunksienaam (MagtigBewerking) het, maar die lidfunksie verskillend optree in verskillende objekte [19].

3.9 Voordele van rolprofielobjekte

Omdat die rolprofiele objekte is, is al die voordele van objek-georiënteerdheid dadelik beskikbaar. Dit beteken nou dat nuwe rolle kan geskep word deur die rolprofielobjekte van bestaande rolle te kombineer (te erf). Objek-georiënteerdheid verleen ook enkapsulasie. Dit beteken dat die inhoud van 'n rolprofielobjek nie beskikbaar is vir enige deel van die stelsel nie. 'n Proses in die stelsel kan slegs by die inligting in 'n rolprofielobjek uitkom deur die objek se lidfunksies te gebruik. Die inligting is dus slegs beskikbaar op 'n beheerde manier. Verder gee die rolprofielobjek slegs die inligting wat nodig is aan die roepende proses, en niks meer nie. So word inligting ook beskerm.

'n Ander belangrike voordeel van objek-georiënteerde rolprofiele is dat die sekerheidskomponent van die stelsel in 'n objek-georiënteerde taal geskryf kan word. Dit is voordelig wanneer die res van die stelsel ook objek-georiënteerd ontwikkel is en daar gevolglik nie koppelvlak probleme is tussen die sekerheidskomponent en die res van die stelsel nie [10]. Al meer stelsels word objek-georiënteerd soos byvoorbeeld objek-georiënteerde transaksieverwerkers. By sulke stelsels kan rolgebaseerde

sekerheid 'n integrale deel uitmaak van die stelsel omdat daar nie verskillende programmeringsmetodieke in die stelsel is nie. Sekerheid word dan deel van die stelsel en nie 'n deel wat later bygevoeg is nie.

3.10 Slot

In hierdie hoofstuk is beskryf hoe rolgebaseerde inligtingsekerheid op 'n objek-georiënteerde wyse gedoen kan word. Dit is gedoen deur rolprofiele (soos in hoofstuk 2 gedefinieer is) as objekte te implementeer. Die hoofstuk het gewys hoe rolprofielobjekte geskep word as 'n instansie van 'n rolprofielklas en hoe 'n rolprofielobjek die voorregte van ander rolprofielobjekte kan erf. Die hoofstuk het 'n uitgebreide rolprofielklas verduidelik. Die rolprofielklas het transaksievoorregte gestoor in sy voorregtelys. Die rolprofielobjekte wat in hierdie hoofstuk verduidelik is, word in hoofstuk 6 gebruik waar 'n model vir objek-georiënteerde rolgebaseerde inligtingsekerheid in klient/bediener en transaksieverwerking omgewings geformuleer word. Voordat hierdie model geformuleer kan word, is dit nodig om traliegrafieke te verduidelik en om 'n oorsig van klient/bediener en transaksieverwerking te gee. Laasgenoemde word in die volgende twee hoofstukke gedoen.

4. Traliegrafieke

4.1 Inleiding

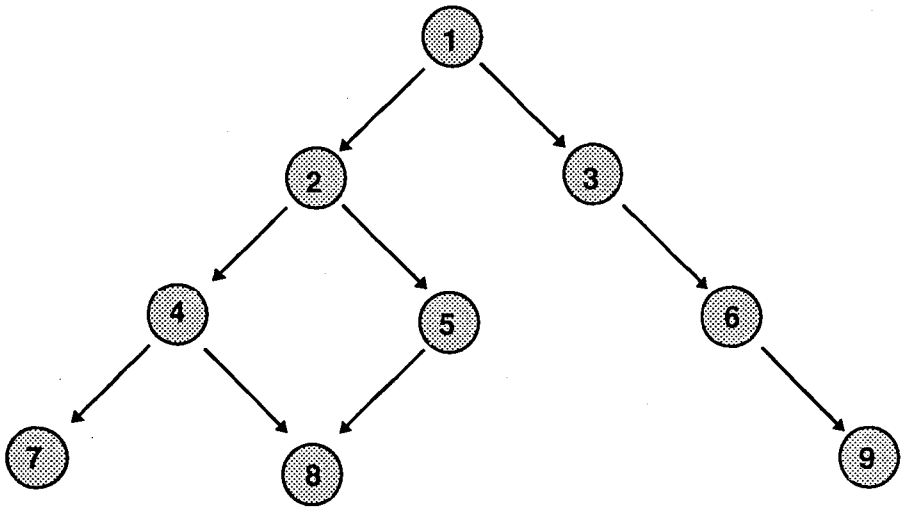
Dikwels wil 'n mens die verwantskap tussen entiteite grafies voorstel. Grafieke is 'n goeie metode om sulke verwantskappe mee voor te stel. In hoofstuk 6 stel ons die verwantskap tussen rolle in 'n rolgebaseerde inligtingsekerheidstelsel grafies voor. Daar is verskeie tipes grafieke waarmee verwantskappe tussen entiteite voorgestel kan word. In hierdie verhandeling word 'n traliegrafiek gebruik om hierdie verwantskappe mee voor te stel.

Die gebruik en die voordele van traliegrafieke word in hierdie hoofstuk gegee. Voordat ons 'n traliegrafiek kan definieer, is dit nodig om die definisie van 'n paar grafiekteorie terme te gee en te verduidelik; dit word vervolgens gedoen.

4.2 Basiese grafiekteorie definisies

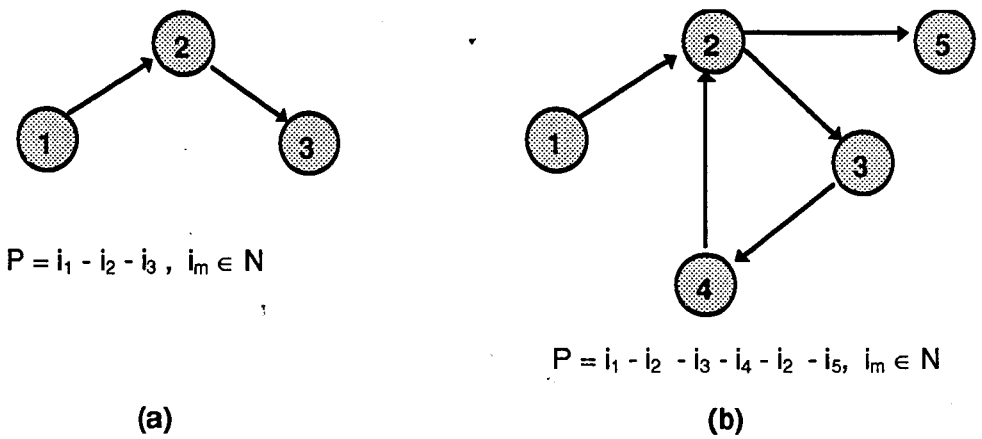
Die volgende paar definisies vorm deel van grafiekteorie. In hierdie verhandeling doen ons nie 'n studie van grafiekteorie nie, maar omdat ons later in die verhandeling 'n traliegrafiek in die model (ORITO) wat ons formuleer gebruik, is dit belangrik dat ons die basiese definisies wat gebruik word om 'n traliegrafiek mee te definieer, formeel weergee.

Definisie 4.1: *'n Gerigte grafiek G = (N, A), bestaan uit 'n versameling punte N, die punte of nodusse van die grafiek, en 'n versameling A, die boë van die grafiek. Die boë van die grafiek is elkeen 'n geordende paar punte. Figuur 4.1 toon 'n voorbeeld van 'n gerigte grafiek. Vir hierdie grafiek is N = {1, 2, 3, 4, 5, 6, 7, 8, 9} en A = {(1, 2), (2, 4), (4, 7), (2, 5), (5, 8), (1, 3), (3, 6), (6, 9) } [4].*



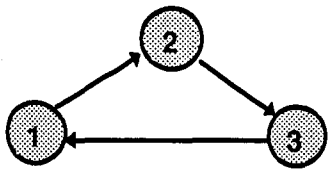
Figuur 4.1.
'n Gerigte grafiek

Definisie 4.2: *'n Gerigte pad P in 'n grafiek G = (N, A) is 'n reeks punte i_1 - i_2 - ··· - i_{r-1} - i_r waar vir elke twee opeenvolgende punte in die pad P, is die boog $(i_k, i_{k+1}) \in A$ en geen punte word herhaal in die pad nie. 'n Pad waarin een of meer punte herhaal word, word 'n gerigte lynry genoem [4]. Figuur 4.2(a) toon 'n voorbeeld van 'n gerigte pad en figuur 4.2(b) toon 'n voorbeeld van 'n gerigte lynry.*



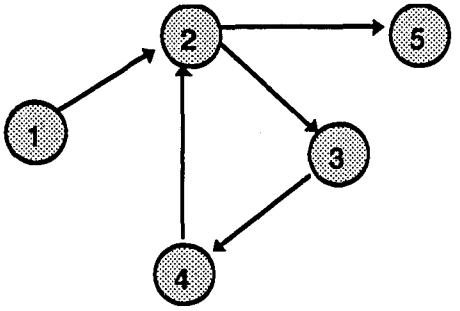
Figuur 4.2
'n Gerigte pad en 'n gerigte lynry.

Definisie 4.3: *'n Gerigte siklus S in 'n gerigte grafiek G = (N, A) is 'n gerigte pad i_1 - i_2 - ··· - i_r saam met die boog (i_r, i_1) [4]. Figuur 4.3(a) en (b) toon twee gerigte grafieke met elke 'n 'n gerigte siklus S.*



$S = i_1 - i_2 - i_3 - i_1, \quad i_m \in N$

(a)



$S = i_2 - i_3 - i_4 - i_2, \quad i_m \in N$

(b)

Figuur 4.3
Twee gerigte grafieke wat elk 'n gerigte siklus S bevat.

Definisie 4.4: *'n Gerigte asikliese grafiek is 'n gerigte grafiek wat geen siklusse bevat nie. Die grafiek in figuur 4.1 is 'n gerigte asikliese grafiek.*

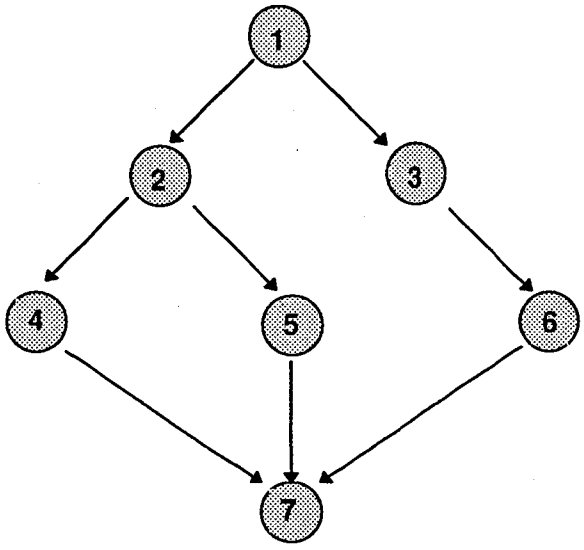
In die res van die verhandeling word dikwels verwys na twee punte in 'n grafiek as 'n ouer-kind of kind-ouer verwantskap. Die definisies vir ouer-punte en kind-punte in 'n grafiek word vervolgens gegee.

Definisie 4.5: *'n Punt i_1 in 'n gerigte grafiek $G = (N, A)$ is 'n ouer van 'n ander punt i_2 as en slegs as $(i_1, i_2) \in N$. Op dieselfde wyse is 'n punt i_1 in 'n gerigte grafiek $G = (N, A)$ 'n kind van 'n ander punt i_2 as en slegs as $(i_2, i_1) \in N$. In figuur 4.3(a) is punt 1 'n ouer van punt 2 en punt 3 is 'n kind van punt 2.*

Die laaste grafiekteorie definisie wat in hierdie hoofstuk ingesluit word, is die definisie van 'n gerigte traliegrafiek, of kortweg 'n tralie. Hierdie definisie is belangrik omdat ons later in die verhandeling rol-verwantskappe met 'n tralie gaan voorstel.

Definisie 4.6: *'n Gerigte traliegrafiek, of kortweg 'n tralie, is 'n gerigte asikliese grafiek waar presies een punt geen ouers het nie en een of meer kinders het , en presies een punt geen kinders het nie en een of meer ouers. Alle ander punte het ten minste een ouer en ten minste een kind.*

Figuur 4.4 toon 'n voorbeeld van 'n tralie. Punt 1 het geen ouers en twee kinders, punt 7 het geen kinders en drie ouers, al die ander punte het ten minste een ouer en ten minste een kind.



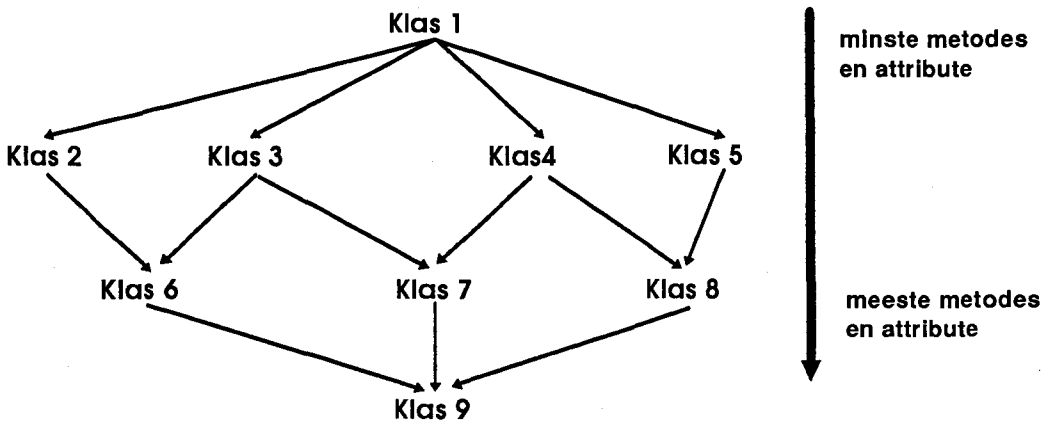
Figuur 4.4.
'n Gerigte traliegrafiek

'n Traliegrafiek het baie gebruike en voordele. In hierdie verhandeling benut ons die voordele van 'n traliegrafiek intensief. Ons gee 'n kort beskrywing van die gebruike en voordele van traliegrafieke.

4.3 Gebruike en voordele van traliegrafieke

'n Traliegrafiek kan gebruik word om die verwantskap tussen entiteite voor te stel. Objekte en klasse in 'n stelsel hou verband. So byvoorbeeld is daar 'n ouer en kind verwantskap tussen sommige klasse in 'n stelsel. Die verwantskap tussen klasse en die verwantskap tussen objekte kan op verskeie maniere grafies voorgestel word. In hierdie verhandeling gebruik ons 'n traliegrafiek om sulke verwantskappe voor te stel.

Figuur 4.5 toon hoe die verwantskap tussen klasse in 'n stelsel wat meervoudige oorerwing toelaat voorgestel kan word. Let op dat die klasse so geskep kan word dat dit 'n tralie vorm; so 'n tralie word soms 'n **klastralie** genoem [1]. Let op dat in meeste stelsels sal klas 9 in figuur 4.5 nie noodwendig nodig wees nie, maar word geskep om die klasse te rangskik in 'n tralie. Klas 9 is dus 'n klas wat die meeste lidfunksies en attribute het terwyl klas 1 die minste het.

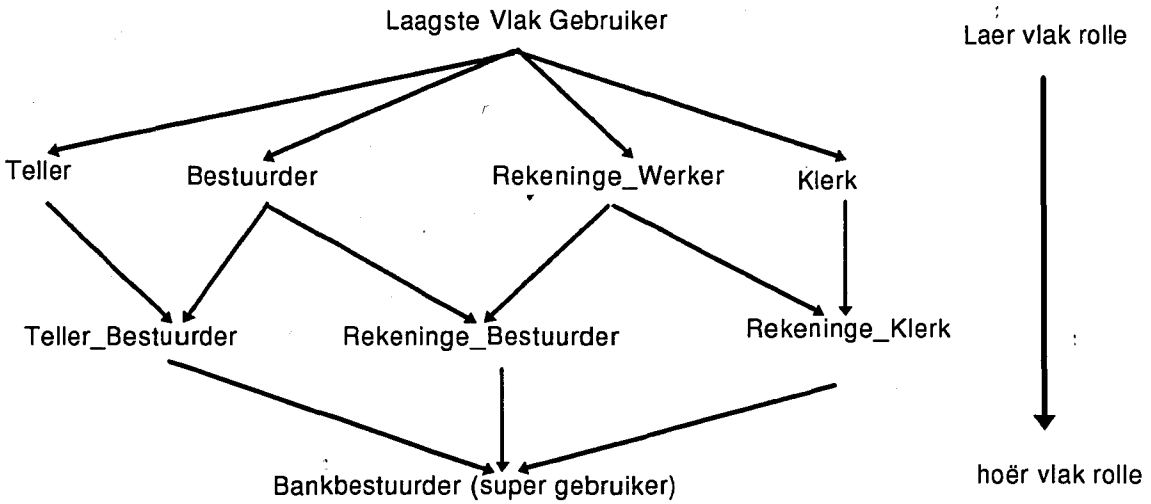


Figuur 4.5.
'n Voorbeeld klastralie.

4.3.1 Voorstelling van rolverwantskappe met 'n traliegrafiek

Net soos wat die verwantskappe tussen klasse in 'n objek-georiënteerde stelsel grafiek voorgestel kan word met 'n klastralie kan die verwantskap tussen rolle in 'n rolgebaseerde inligtingsekerheidstelsel grafies voorgestel word met 'n traliegrafiek.

Figuur 4.6 toon 'n voorbeeld van 'n traliegrafiek wat die verwantskap tussen rolle voorstel. Ons noem so 'n traliegrafiek 'n roltralie.



Figuur 4.6
'n Voorbeeld roltralie.

Figuur 4.6 toon die verwantskap tussen rolle. Let op dat uit die roltralie ons grafies maklik kan sien dat 'n rol soos Teller_Bestuurder meer voorregte het as 'n rol soos Klerk omdat die rol Klerk hoër-op in die tralie voorkom. In hoofstuk 6 gee ons meer aandag aan roltralies.

Daar is verskeie voordele aan die gebruik van 'n traliegrafiek om klasse of objekte se verwantskap mee voor te stel, ons noem vervolgens 'n paar.

4.3.2 Voordele van die gebruik van 'n traliegrafiek

Wanneer nuwe klasse geskep moet word is dit handig indien die verwantskap tussen bestaande klasse grafies gesien kan word. Dit vergemaklik die keuse oor watter klasse om te gebruik as ouer klasse vir die nuwe klas.

Wanneer klasse of objekte in 'n tralie gerangskik is, is dit maklik om grafiekteorie algoritmes op die tralie toe te pas en die tralie te deursoek om sodoende byvoorbeeld die gemeenskaplike attribute van twee klasse in die tralie te bepaal.

Net soos wat klasse in 'n tralie gerangskik kan word, is dit moontlik om die verwantskap tussen rolle ook grafies met 'n traliegrafiek voor te stel. Dit is een van die belangrikste redes hoekom ons 'n tralie definieer en beskryf.

Die voordele van die gebruik van 'n traliegrafiek sal meer duidelik word in die volgende hoofstuk wanneer ons 'n roltralie en die gebruik daarvan beskryf. Ons gee in die volgende hoofstuk meer aandag hieraan.

4.4 Slot

Hierdie hoofstuk het 'n gerigte traliegrafiek gedefinieer. Om dit te kon doen, was dit nodig om 'n paar ander grafiekteorie terme se definisies te gee en te beskryf. Die gebruike en voordele van 'n traliegrafiek is gegee. Die belangrikste rede waarom ons traliegrafieke gedefinieer en beskryf het is sodat ons gereed is om in hoofstuk 6 te sien hoe rolverwantskappe met 'n traliegrafiek voorgestel kan word. Die voorafgaande hoofstukke tesame met die volgende hoofstuk oor transaksieverwerking gee die agtergrond en beskryf die basiese terme wat in hoofstuk 6 gebruik word. Die model vir rolgebaseerde inligtingsekerheid in 'n transaksieverwerking omgewing word in hoofstuk 6 geformuleer.

5. Transaksieverwerking

5.1 Inleiding

In hierdie verhandeling word 'n model vir objek-georiënteerde rolgebaseerde inligtingsekerheid in transaksieverwerking en kliënt/bediener omgewings (ORITO) geformuleer. Om dit te kan doen moet die nodige agtergrond gegee word. Hierdie hoofstuk tesame met hoofstukke 2 tot 4 gee die nodige agtergrond.

In hierdie hoofstuk word 'n oorsig gegee van die term transaksieverwerking. Die term word gedefinieer en verduidelik met voorbeelde. Inligtingsekerheid in transaksieverwerking omgewings word bespreek.

5.2 Transaksieverwerking

Transaksieverwerking is fundamenteel, 'n metode waarvolgens toepassingsprogramme wat toegang na stelselhulpbronne verkry weet dat hierdie toegang sekere eienskappe het [10] (ons bespreek die eienskappe later). Hierdie eienskappe word verkry deur toegang tot stelselhulpbronne te doen deur van transaksies gebruik te maak.

Definisie 5.1: *'n Transaksie, in die konteks van 'n transaksieverwerker, is 'n logiese eenheid van werk. Wanneer die transaksie uitvoer word daar voldoen aan die AKID eienskappe* [10].

5.2.1 Die AKID eienskappe van transaksies:

5.2.1.1 Atoomheid

Die opdaterings tussen die begin en die einde van 'n transaksie is atomies. Dit beteken dat die toepassing wat die transaksie versoek kan seker wees dat indien die transaksie suksesvol afhandel is elkeen van die opdaterings gedoen. Indien die transaksie nie een of meer van die opdaterings kon doen nie, sal geeneen van die opdaterings wat deel maak van die transaksie gedoen word nie.

5.2.1.2 Konsistensie

Die toepassing wat 'n transaksie roep, kan daarop staat maak dat wanneer 'n transaksie eindig, hetsy suksesvol of onsuksesvol, dan is alle hulpbronne waarna die transaksie toegang gehad het in 'n geldige toestand. Indien konsistentheid nie verseker word nie, kan dit gebeur dat 'n hulpbron in 'n onwaar toestand is wanneer twee transaksies byna gelyktydig opdaterings doen op dieselfde hulpbron.

5.2.1.3 Isolasie

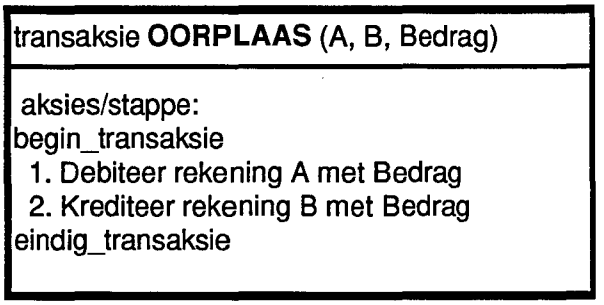
Die aksies wat deel maak van 'n transaksies (die opdaterings op hulpbronne) is nie sigbaar vir die res van die stelsel voordat die transaksie suksesvol voltooi nie.

5.2.1.4 Duursaamheid

Wanneer 'n transaksie suksesvol voltooi, is die opdaterings wat die transaksie gemaak het permanent en word nie beïnvloed deur stelselfalings nie.

Vir 'n stelsel om volledige transaksieverwerking te bied moet aan al bogenoemde eienskappe voldoen word elke keer wanneer 'n transaksie uitgevoer word.

Figuur 5.1 toon 'n voorbeeld van 'n transaksie met die naam OORPLAAS. Die transaksie bevat slegs twee stappe wat rekening A debiteer en rekening B krediteer met dieselfde bedrag. Die transaksie het skryfregte op albei rekeninge nodig.



Figuur 5.1
'n Voorbeeld van 'n transaksie.

Let op dat die twee aksies van die transaksie albei suksesvol moet wees, andersins moet nie een uitgevoer word nie. Dit is byvoorbeeld nie aanvaarvaar om rekening B te krediteer terwyl rekening A nie gedebiteer is nie.

Transaksieverwerking kan produktiwiteit by die ontwikkeling van stelsels verhoog omdat minder kode geskryf hoef te word - transaksies van die transaksieverwerker word geroep in plaas van om die opdaterings self te doen.

5.3 Sekerheid in transaksieverwerkingstelsels

'n Transaksie is 'n eenheid van werk. 'n Transaksie kan ook gesien word as 'n eenheid van magtiging. Dit beteken dat indien 'n gebruiker 'n transaksie mag uitvoer is dit vanselfsprekend dat die gebruiker ook toegang het na die stelselhulpbronne waartoe die transaksie toegang het. Daar is egter 'n verskil tussen toegang na transaksies en toegang na stelselhulpbronne: 'n Transaksie bied *beheerde* toegang tot stelselhulpbronne. Neem die voorbeeld in figuur 5.1: 'n gebruiker mag die OORPLAAS transaksie uitvoer maar mag dalk nie enige van die aksies van die transaksies alleen uitvoer nie (die gebruiker mag fondse oorplaas van een rekening na 'n ander maar mag nie een rekening debiteer sonder om 'n ander te krediteer nie).

In 'n transaksieverwerkerstelsel verkry gebruikers toegang tot stelselhulpbronne deur middel van transaksies. Daar is egter transaksies wat nie deur alle gebruikers uitgevoer mag word nie. Die stelsel waarbinne 'n transaksieverwerker uitvoer moet dus sorg dat gebruikers slegs transaksies uitvoer indien hulle gemagtig is om dit te doen.

Indien ons sekerheidsapekte in ag neem, kan ons sê dat 'n transaksie 'n logiese eenheid van verwerking is met 'n aantal stappe waar elke stap moontlik toegang na stelselhulpbronne nodig het. Vir die doeleindes van hierdie verhandeling definieer ons 'n transaksie as volg:

Definisie 5.2 *'n Transaksie, in die konteks van inligtingsekerheid, is 'n diskrete eenheid van verwerking en het toegang nodig na verskeie hulpbronne om suksesvol af te handel. In terme van voorregte, soos in definisie 2.1 gedefinieer is (herhaal hieronder), is 'n transaksie 'n geordende versameling aksies op hulpbronne en 'n transaksie benodig 'n versameling voorregte waar elke voorreg van die vorm (x, m) is.*

Ons gee weer die definisie vir 'n voorreg:
'n Voorreg is 'n paar (x, m) waar x verwys na 'n beskermde data item en m is 'n nie-leë versameling van toegangsmetodes vir objek x.

Dit beteken vir 'n transaksie om suksesvol af te handel, moet alle opdaterings op hulpbronne suksesvol wees, andersins moet geen opdatering gedoen word nie ('n transaksie is 'n eenheid van verwerking). Om inligting van bronne te lees, of na bronne te skryf, het die transaksie (of die gebruiker wat die transaksie versoek), al die voorregte wat nodig is om toegang na die bronne te verkry, nodig. Figuur 5.2 toon die voorbeeld van 'n transaksie OORPLAAS. Die transaksie bevat twee stappe wat rekening A debiteer en rekening B krediteer. Die transaksie het skryfregte op albei rekeninge nodig.

transaksie OORPLAAS (A, B, Bedrag)

aksies/stappe:
1. Debiteer rekening A met Bedrag
2. Krediteer rekening B met Bedrag

voorregtelys:
1. rekening A { lees, skryf }
2. rekening B { lees, skryf }.

Figuur 5.2
'n Voorbeeld van 'n transaksie met voorregte.

5.4 Transaksiemonitors

'n *Transaksiemonitor* is 'n program wat die transaksieverwerker omgewing verskaf waarbinne transaksies uitgevoer word en sorg onder meer vir die sekerheidsaspekte van transaksieverwerking (om te sorg dat gebruikers gemagtig is om transaksies te mag uitvoer).

Die vlak van inligtingsekerheid wat 'n transaksiemonitor bied hang af van die spesifieke produk soos byvoorbeeld die CICS of TUXEDO transaksiemonitors. CICS/6000, die weergawe van CICS vir die IBM RS/6000 masjien, bied transaksiesekerheid deur elke transaksie in die stelsel 'n sleutelnommer vanaf 2 tot 64 te gee (nommer 1 dui op 'n

onbeskermde/publieke transaksie) [16]. Elke gebruiker wat aanteken op die stelsel kry 'n stel sleutelnommers (ook waardes vanaf 2 tot 64). CICS/6000 toets of die sleutelnommer van 'n transaksie in die sleutellys van 'n gebruiker is voordat die gebruiker die transaksie mag uitvoer. Beskou die volgende as voorbeeld. Gestel daar is vyf transaksies met ooreenstemmende sleutelnommers soos in tabel 5.3 en drie gebruikers met sleutelnommers soos in tabel 5.4.

Transaksie	Sleutelnommer
AANTEKEN	0
DEPONEER	1
ONTTREK	3
NAVRAAG	1
OORPLAAS	2

Tabel 5.3
Transaksie met sleutelnommers.

Gebruiker	Sleutelnommers
Sarah	1
Peter	2,3
John	1,2

Tabel 5.4
Gebruikers en hulle sleutelnommers.

Tabel 5.5 toon die transaksies wat elke gebruikers kan uitvoer. Let op dat gebruiker Peter sleutelnommers 2 en 3 het. Dit beteken dat alle transaksies op sleutelvlak 2 en 3 (ONTTREK en OORPLAAS) sowel as transaksies op sleutelvlak 0 (AANTEKEN) deur Peter uitgevoer mag word.

Gebruiker	Gemagtigde Transaksies
Sarah	AANTEKEN, DEPONEER, NAVRAAG
Peter	AANTEKEN, OORPLAAS, ONTTREK
John	AANTEKEN, DEPONEER, NAVRAAG, OORPLAAS

Tabel 5.5
Gebruikers en hulle gemgtigde transaksies.

CICS/6000 bied ook die opsie om die magtiging van transaksies deur 'n eksterne program te doen. Hierdie opsie maak dit moontlik om die magtiging van die uitvoer van transaksies op 'n ander manier te doen as wat CICS/6000 dit normaalweg doen. Deur van hierdie opsie gebruik te maak is dit byvoorbeeld moontlik om die uitvoer van transaksies te magtig deur van rolgebaseerde inligtingsekerheid gebruik te maak. Die volgende hoofstuk toon hoe rolgebaseerde inligtingsekerheid in 'n transaksieverwerking omgewing gedoen kan word.

5.5 Slot

In hierdie hoofstuk is transaksieverwerking bespreek. Sekerheid in transaksieverwerkingstelsels is bespreek en die wyse waarop transaksies gemagtig word in die CICS/6000 transaksiemonitor is verduidelik. Hierdie hoofstuk tesame met die vorige hoofstukke gee die nodige agtergrond vir die model vir rolgebaseerde inligtingsekerheid in transaksieverwerking omgewings wat in die volgende hoofstuk geformuleer word.

DEEL 2

- Die model vir objek-georiënteerde rolgebaseerde inligtingsekerheid in 'n transaksieverwerking omgewing (ORITO)
- Verspreide en Kliënt/bediener omgewings
- ORITO in 'n verspreide en kliënt/bediener omgewing
- 'n Meer doeltreffende en meer betroubare verspreide inligtingsekerheidstelsel
- Evaluering en toekomstige uitbreidings aan ORITO

6. Die model vir objek-georiënteerde rolgebaseerde inligtingsekerheid in 'n transaksieverwerking omgewing (ORITO)

6.1 Inleiding

Daar is verskeie maniere om inligting in 'n organisasie te beskerm teen ongemagtigde vernietiging, bekendmaking of verandering [1, 2, 12, 17]. In hierdie verhandling bestudeer ons 'n paar metodes en formuleer 'n nuwe model vir inligtingsekerheid. Ons noem hierdie metode die model vir objek-georiënteerde rolgebaseerde inligtingsekerheid in 'n transaksieverwerking omgewing (ORITO).

Die model is gebou uit begrippe wat in die voorafgaande hoofstukke beskryf is. Alhoewel baie konsepte en terme in die model gebruik word kan ons 'n paar uitsonder. Die model is hoofsaaklik 'n model vir rolgebaseerde inligtingsekerheid. Rolgebaseerde inligtingsekerheid is geensins 'n nuwe benadering tot inligtingsekerheid nie [2, 5, 6, 7]. ORITO is egter 'n nuwe model vir inligtingsekerheid in ten minste twee opsigte. Eerstens is ORITO 'n objek-georiënteerde rolgebaseerde model en tweedens word die verwantskap tussen rolle voorgestel met 'n traliegrafiek en word hierdie grafiek gebruik om die bestuur van inligtingsekerheid te vergemaklik en om rolobjekte te versprei tussen bediener rekenaars. In die volgende hoofstukke word ORITO uitgebrei en daar word getoon dat ORITO gebruik kan word vir inligtingsekerheid in 'n verspreide kliënt/bediener omgewing.

In beginsel kan ORITO in verskeie omgewings suksesvol toegepas word, maar in hierdie hoofstuk word ORITO verduidelik as 'n model vir inligtingsekerheid in transaksieverwerking omgewings. Laasgenoemde vergemaklik die beskrywing van die model deurdat daar van konkrete voorbeelde gebruik gemaak kan word. Hou egter in gedagte dat die model redelik algemeen toepasbaar is.

Die hoofstuk begin met 'n verduideliking van hoe die model gebruik word in 'n transaksieverwerking omgewing. 'n Formele beskrywing van die begrip 'magtiging van transaksies deur ORITO' word gegee en daarna word aandag geskenk aan die voorstelling van die verwantskappe tussen rolle en onder andere wys ons op die voordele van 'n roltralie en waarom sekere ander voorstellings minder geskik is. ORITO se werking word beskryf aan die hand van 'n paar voorbeelde.

6.2 Kort beskrywing van ORITO

In hierdie gedeelte word 'n kort beskrywing van die model gee. Daar word gewys op die hoofkomponente van ORITO.

Die model vir objek-georiënteerde, rolgebaseerde inligtingsekerheid in transaksieverwerker omgewing (ORITO) gebruik begrippe wat in die vorige hoofstukke beskryf is. In van die volgende hoofstukke gaan ORITO uitgebrei word,

maar in hierdie hoofstuk gee ons die basiese komponente van ORITO en wys hoe hulle inmekaar steek.

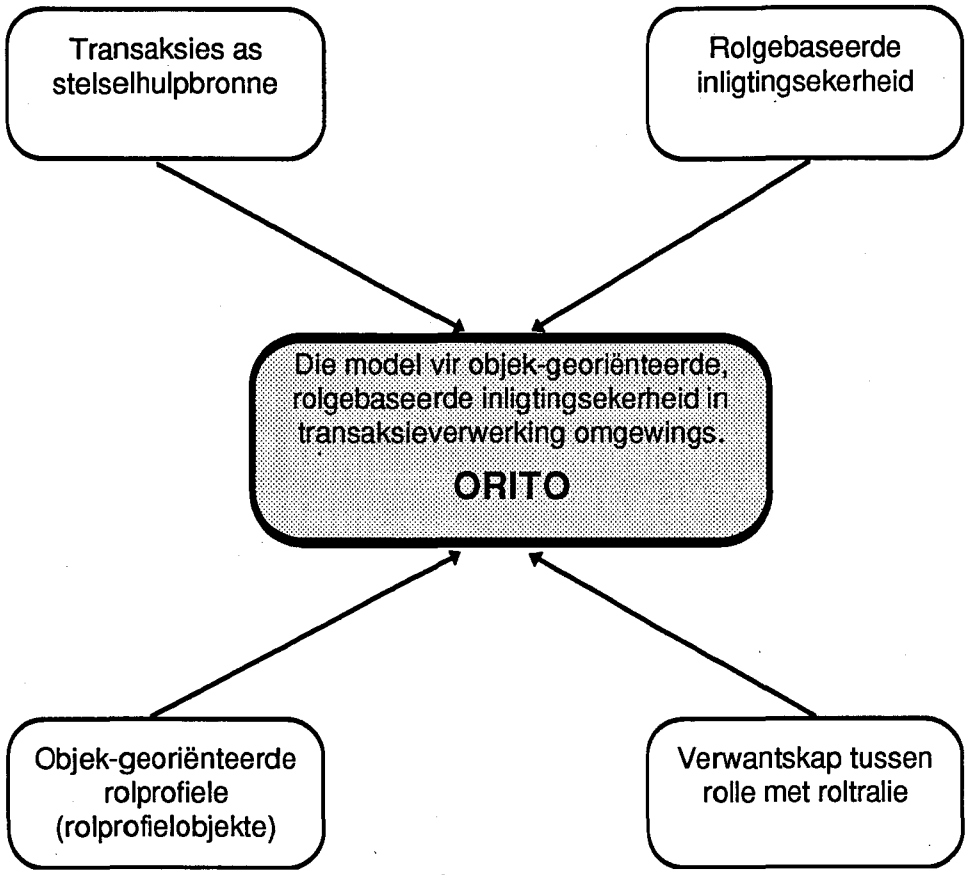
6.2.1 Die komponente van ORITO

Eerstens is ORITO 'n model vir inligtingsekerheid by transaksieverwerkers en is die basiese objek wat in die stelsel beskerm moet word transaksies. Die model sorg dat gebruikers transaksies uitvoer slegs indien hulle gemagtig is om dit te doen.

Tweedens is ORITO 'n model vir rolgebaseerde inligtingsekerheid. Gebruikers word aan rolle gekoppel en mag transaksies uitvoer indien die rol waaraan hulle gekoppel is die transaksies mag uitvoer.

Die derde hoofkomponent van ORITO is die gebruik van objek-georiënteerde rolprofiele (of rolprofielobjekte in kort). 'n Rolprofielobjek verteenwoordig die voorregte van 'n rol in die stelsel en lidfunksies van so 'n rolprofielobjek word geroep elke keer wanneer 'n gebruiker wat aan 'n rol gekoppel is 'n transaksie wil uitvoer.

Die laaste komponent van ORITO is die gebruik van roltralies om die verwantskap tussen rolle voor te stel. Die gebruik van 'n roltralie vergemaklik die bestuur van rolgebaseerde inligtingsekerheid en is veral nuttig wanneer nuwe rolle geskep word. Figuur 6.1 toon die vier komponente van ORITO wat in hierdie hoofstuk bespreek word.



Figuur 6.1
Die komponente van ORITO.

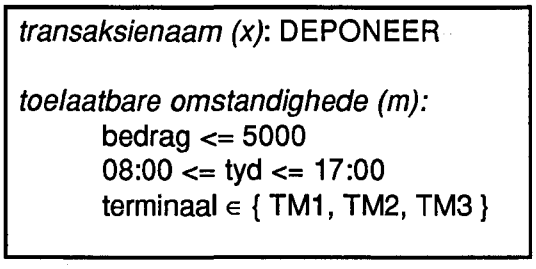
6.3 Transaksies as stelselhulpbronne

Soos reeds genoem is, is ORITO 'n model vir inligtingsekerheid in 'n transaksieverwerking omgewing. In die vorige hoofstuk is transaksieverwerkers bespreek en ORITO kan gebruik word om die transaksies wat 'n transaksieverwerker moet uitvoer te magtig. Dit beteken dat wanneer ORITO gebruik word, word gebruikers gemagtig om **transaksies** uit te voer via rolle. Onthou dat in die vorige hoofstuk gesê is dat 'n transaksie toegang het na verskeie hulpbronne. 'n Transaksie bestaan dus uit 'n aantal logiese stappe en 'n versameling van toegangsmetodes na stelselhulpbronne (bv, lees, skryf, uitvoer, ens.) soos in figuur 5.2 op p.59. In die vorige hoofstuk is ook genoem dat 'n transaksie 'n eenheid van verwerking is en indien 'n gebruiker 'n transaksie mag uitvoer beteken dit dat 'n gebruiker al die aksies wat die transaksie opmaak mag uitvoer as 'n geheel (nie noodwendig een afsonderlik nie). In ORITO aanvaar ons dat transaksies oordeelkundig opgestel is en indien 'n gebruiker die reg kry om 'n transaksie uit te voer beteken dit dat so 'n gebruiker net die regte kry wat nodig is om sy besigheidsfunksie te verrig en nie onnodig meer regte nie.

In definisie 2.1 is 'n voorreg gedefinieer as 'n paar (x, m). Ons gee weer die definisie:

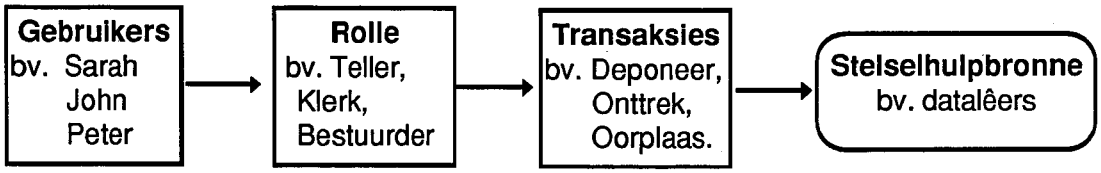
'n Voorreg is 'n paar (x, m) waar x verwys na 'n beskermde data item en m is 'n nie-leë versameling van toegangsmetodes vir objek x.

In ORITO is *x* transaksies en *m* is 'n beskrywing van *toelaatbare omstandighede* waaronder toegang na transaksie *x* verleen mag word. *m* kan bevoorbeeld spesifiseer dat *x* uitgevoer mag word tussen 13:00 en 17:00 bedags en slegs vanaf spesifieke terminale. In ORITO gebruik ons sulke voorregte en noem hulle *transaksievoorregte.* Figuur 6.2 gee 'n voorbeeld van 'n transaksievoorreg.



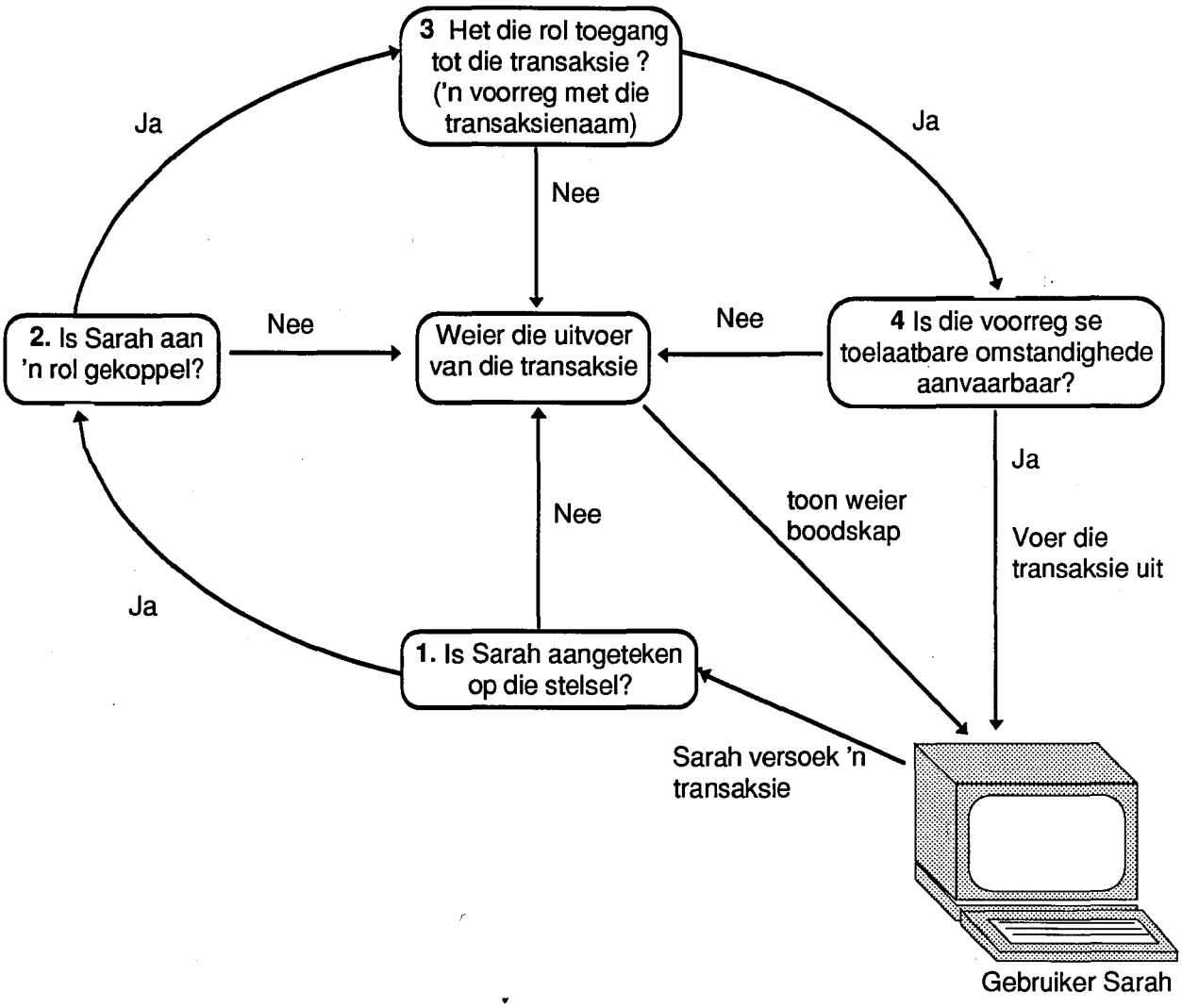
Figuur 6.2
Voorbeeld van 'n transaksievoorreg.

In ORITO word rolgebaseerde inligtingsekerheid gebruik. Dit beteken dat gebruikers die reg kry om transaksies uit te voer deurdat hulle aan rolle gekoppel is en die rolle het die reg om sekere transaksies onder sekere toelaatbare omstandighede uit te voer. Figuur 6.3 toon hierdie konsep diagrammaties.



Figuur 6.3
'n Rolgebaseerde benadering tot inligtingsekerheid.

ORITO is dus 'n model vir 'n inligtingsekerheidstelsel waar gebruikers aangeteken is by 'n transaksieverwerker en transaksies versoek. Die versoekte transaksies mag slegs uitgevoer word indien 'n gebruiker wat so 'n transaksie versoek, gekoppel is aan 'n rol in die stelsel en hierdie rol 'n transaksievoorreg het met die naam van die transaksie en die transaksievoorreg se toelaatbare oomstandighede laat die uitvoer van die transaksie toe. Figuur 6.4 toon 'n vereenvoudigde weergawe van die magtiging van transaksies in ORITO. Let op dat die model uitgebrei word in hierdie en volgende hoofstukke. In die volgende deel beskryf ons hierdie magtiging van die uitvoer van transaksies in ORITO formeel.



Figuur 6.4
Magtiging van transaksies in ORITO.

6.4 Formele beskrywing van magtiging van die uitvoer van transaksies in ORITO.

In die voorafgaande gedeeltes het ons op 'n informele wyse beskryf hoe magtiging in rolgebaseerde stelsels gedoen word en ook gekyk hoe die uitvoer van transaksies gemagtig word deur ORITO. In hierdie gedeelte gaan ons 'n formele beskrywing gee van die magtiging van transaksies deur ORITO. 'n Basiese definisie vir magtiging word gegee en uitgebrei vir ORITO.

6.4.1 Basiese definisie van magtiging

In [1] word die basiese *definisie vir 'n magtiging* gegee as 'n 3-tal (s, o, a) waar

$s \in S$,
die versameling subjekte (gebruikers) is van die stelsel;

$o \in O$,
die versameling objekte (bv. datalêers) in die stelsel; en

$a \in$ A,
die versameling van magtigingstipes (bv. lees, skryf, uitvoer, ens.).

'n Funksie *f* is gedefineer om te bepaal of die magtiging (*s*, *o*, *a*) Waar of OnWaar is:

$$f: S \times O \times A \rightarrow \{\text{Waar, Onwaar}\}.$$

Gegee 'n 3-tal (*s*, *o*, *a*), as *f*(*s*, *o*, *a*) = Waar, beteken dit subjek (of gebruiker) *s* het die magtigingstipe *a* (bv. lees) op objek *o* (bv. 'n spesifieke datalêer).

Voorbeeld:
Beskou die toegangsmatriks in tabel 6.1. 'n Toegangsmatriks is 'n tabel wat aantoon watter gebruikers watter bewerkings op watter objekte in 'n stelsel mag uitvoer. Hierdie tabel toon 'n aantal toelaatbare (*s*, *o*, *a*) drie-talle vir 'n inligtingsekerheidstelsel. Let op die versamelings S, O en A wat die tabel voorafgaan.

Versameling subjekte S = {Sarah, John, Peter}.
Versameling objekte O = {DATA1.TXT, DATA2.TXT, PROGRAM.EXE}.
Versameling magtigingstipies A = {lees, skryf, uitvoer}.

Subjek s	Objek o	Magtigingstipe a
Sarah	DATA1.TXT	lees
Sarah	DATA2.TXT	lees,skryf
John	PROGRAM.EXE	lees, skryf, uitvoer
Peter	PROGRAM.EXE	lees, uitvoer

Tabel 6.1
'n Voorbeeld toegangsmatriks.

Uit tabel 6.1 kan ons nou sien dat *f(Sarah, DATA1.TXT, lees)* = Waar en dit beteken dat gebruiker Sarah mag 'n Lees bewerking op die objek DATA1.TXT uitvoer. Daar is meer as een manier hoe die stelsel kan bepaal of die funksie *f* waar of onwaar is maar in hierdie voorbeeld toets die stelsel gewoon of die toegangsmatriks 'n inskrywing bevat met subjek = Sarah, objek = DATA1.TXT en lees $\in$ magtigingstipe.

Veronderstel gebruiker Peter wil 'n skryf bewerking op die objek PROGRAM.EXE uitvoer. Die stelsel sal hierdie bewerkings weier omdat *f(Peter, PROGRAM.EXE, skryf)* = Onwaar (die inskrywing Peter, PROGRAM.EXE, skryf kom nie in die toegangsmatriks voor nie).

Die stelsel hoef nie noodwendig 'n toegangsmatriks te gebruik nie. Wat belangrik is, is dat daar 'n funksie *f* is wat bepaal of 'n 3-tal (*s*, *o*, *a*) waar of onwaar is en dienooreenkomstig 'n versoek toestaan of weier. In die volgende deel sien ons hoe die funksie *f* gedefinieer word in ORITO ('n rolgebaseerde inligtingsekerheidstelsel).

6.4.2 Magtiging in ORITO

In ORITO gebruik ons bogenoemde konsep van magtiging, maar omdat ORITO 'n model vir magtiging in 'n transaksieverwerking omgewing met rolgebaseerde inligtingsekerheid is, pas ons die konsep as volg aan:

Let op dat ons nie nou die rolprofiele as objekte beskou nie; ons gee later in die hoofstuk aandag hieraan.

'n magtiging vir 'n transaksie in ORITO is 'n 4-tal (u, r, t, a) waar

$u \in U$,
die versameling van gebruiker identifiseerders (soos in hoofstuk 2 beskryf is) in die stelsel;

$r \in R$,
die versameling rolle in die stelsel;

$t \in T$,
die versameling transaksies wat deur die transaksiesverwerker(s) van die stelsel uitgevoer kan word; en

$$a = (a_1, a_2, ..., a_n) \in A = (A_1, A_2, ..., A_n).$$

A is 'n vektor waarvan die elemente die domein van toelaatbare omstandighede op transaksies is. a is 'n vektor waarvan die elemente $(a_1, a_2, ..., a_n)$ aandui onder watter omstandighede is 'n transaksie versoek, bv. vanaf watter terminaal en vir watter bedrag.

6.4.3 Voorbeeld

Tabel 6.2 toon 'n voorbeeld van A, die domein van toelaatbare omstandighede.

A_1 terminaal_id	A_2 bedrag van transaksie	A_3 tyd van die dag
{TM1, TM2, TM3, TM4}	bedrag $\in \Re$	00:00 <= tyd 24:00

Tabel 6.2
'n Voorbeeld van A, die domein van toelaatbare omstandighede.

Volgens tabel 6.2 bestaan A in hierdie geval uit drie versamelings. Die eerste versameling A_1, gee die domein van alle terminaal_identifiseerders in die stelsel. A_2 spesifiseer dat die bedrag van 'n transaksie enige reële getal kan wees en die tyd van 'n transaksie kan enige geldige tyd van die dag wees. Let op dat A is slegs die domein van toelaatbare omstandighede en niks sê van die toelaatbare omstandighede vir 'n spesifieke transaksie nie.

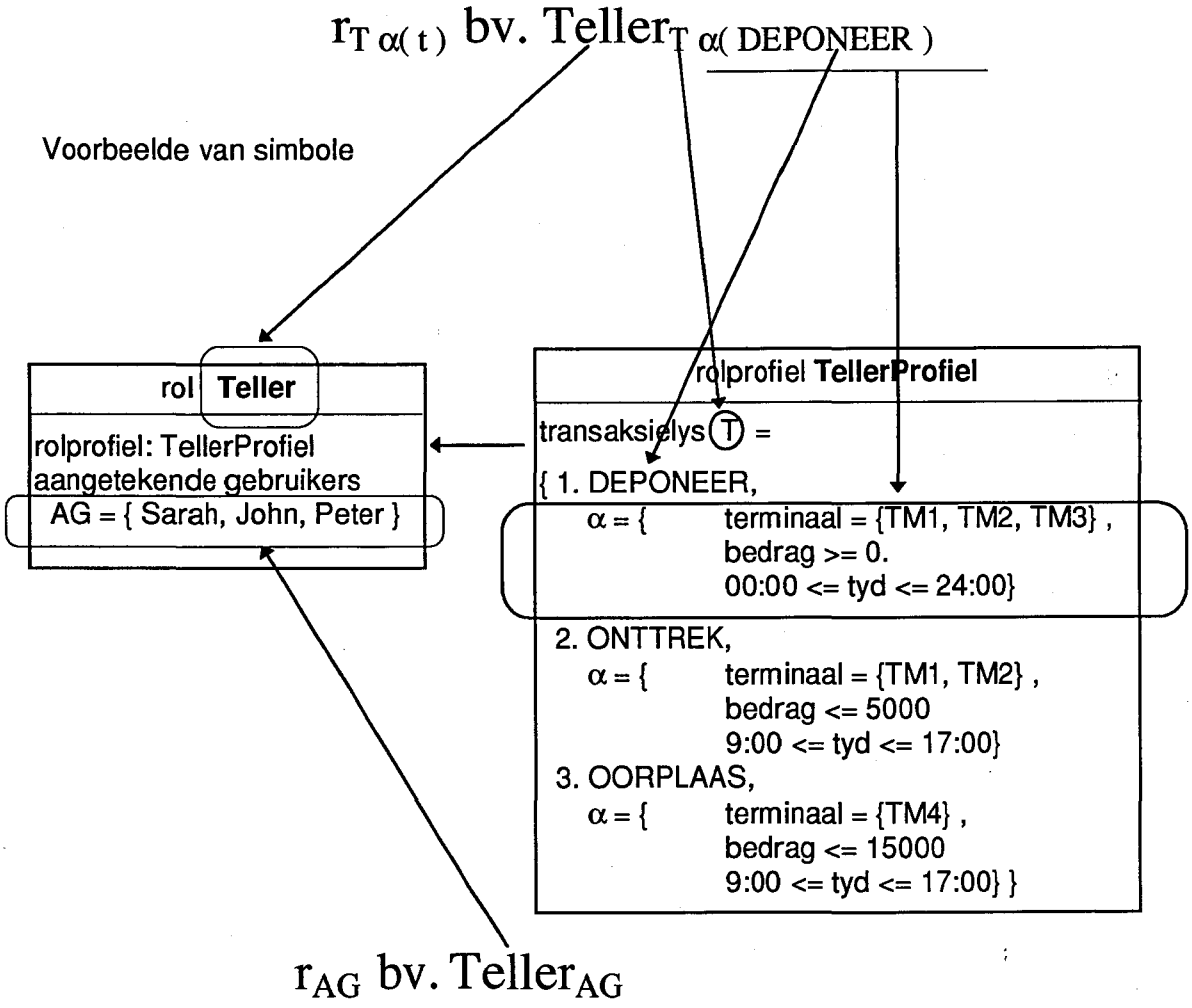
Indien ons hou by die voorbeeld in tabel 6.2 is 'n geldige veeltal a vir enige transaksie t, a = (terminaal_id = TM2, bedrag = 5000 , tyd = 15:34). Let op dat $a_1 \in A_1$, $a_2 \in A_3$, ... , $a_n \in A_n$, dit beteken dus as a_1 die terminaal is van waar 'n

transaksie uitgevoer is, is A_1 die domein van alle terminaal_id's in die stelsel en in die voorbeeld is a_1 = TM2 $\in A_1$ = {TM1, TM2, TM2, TM4}.

Let op dat ons hier ten minste drie verskillende vektore van omstandighede onderskei. Die eerste is die *domein van toelaatbare omstandighede* wat ons aandui met A, die tweede is die *spesifieke omstandighede* waaronder 'n transaksie versoek is en ons dui dit aan met *a*. Die laaste vektor is die *toelaatbare omstandighede* van 'n spesifieke transaksie in 'n spesifieke rol se rolprofiel, ons dui so 'n vektor van toelaatbare omstandighede aan met α.

6.4.4 Notasie

Voordat ons die funksie wat magtiging in ORITO doen beskryf, let op die volgende notasie. Onthou dat ons in die vorige hoofstukke gesê het dat elke rol 'n rolprofiel het en 'n rolprofiel het 'n lys transaksies wat uitgevoer mag word en elke transaksie in die lys het 'n lys toelaatbare omstandighede vir die transaksie. Wanneer ons na die transaksielys van 'n rol se rolprofiel wil verwys skryf ons r_T. Indien ons na die toelaatbare omstandighedeα van transaksie *t* in die transaksielys T van rol *r* se rolprofiel wil verwys skryf ons $r_{T_\alpha(t)}$. Let verder daarop dat elke rol 'n lys van gebruikers het wat tans aangeteken is op die stelsel en gekoppel is aan die rol. Ons verwys na die lys van aangetekende gebruikers vir rol *r* as r_{AG}. Ons is nou gereed om die magtigingsfunksie *f* te definieer. Figuur 6.5 toon die notasie 'n voorbeeld van die notasie.



Figuur 6.5
'n Voorbeeld van die notasie wat gebruik word.

Let op dat ons hier die aangetekende gebruikers in die rol stoor. Later in die dokument vereis ons egter dat die aangetekende gebruikers in 'n aparte tabel gestoor word. Laasgenoemde vergemaklik die implementasie in 'n verspreide omgewing. Om egter 'n formele definisie vir magtiging in ORITO te gee, hou ons by eersgenoemde.

'n Funksie *f* is gedefineer om te bepaal of die magtiging (*u*, *r*, *t*, *a*) Waar of Onwaar is:

f: U x R x T x A $\rightarrow$ (Waar, Onwaar).

Gegee 'n 4-tal (*u*, *r*, *t*, *a*), die funksie *f* moet bepaal of gebruiker *u* gekoppel is aan rol *r* en of rol *r* gemagtig is om transaksie *t* uit te voer onder die omstandighede wat in *a* beskryf word.

Met ander woorde, gegee (*u*, *r*, *t*, *a*) dan is $f(u, r, t, a)$ = Waar, indien

$u \in r_{AG}$ *(gebruiker met identifiseerder u is gekoppel aan rol r en aangeteken as 'n gebruiker van tipe rol r),*

en

$t \in r_T$ *(transaksie t is in die transaksielys van rol r se rolprofiel)*,

en

$a \in r_{T_\alpha(t)}$ *(die omstandighede waaronder transaksie t versoek is, is deel van die toelaatbare beperkings α van transaksie t in die transaksielys T van rol r)*.

Let op dat $a \in r_{T_\alpha(t)}$ beteken dat elkeen van die omstandighede (a_1, a_2, ..., a_n) van transaksie *t* in die ooreenkomstige toelaatbare omstandighede (α_1, α_2, ... , α_n) van die transaksie *t* in die transaksielys van rol *r* se rolprofiel is. Byvoorbeeld, gestel 'n transaksie *t* word versoek onder die omstandighede a_1 = TM2 (terminaal_id), a_2 = 1000 (bedrag van transaksie) en a_3 = 10:33 (tyd van transaksie). Indien $a \in r_{T_\alpha(t)}$ dan is TM2 $\in \alpha_1$, 1000 $\in \alpha_2$ en 10:33 $\in \alpha_3$.

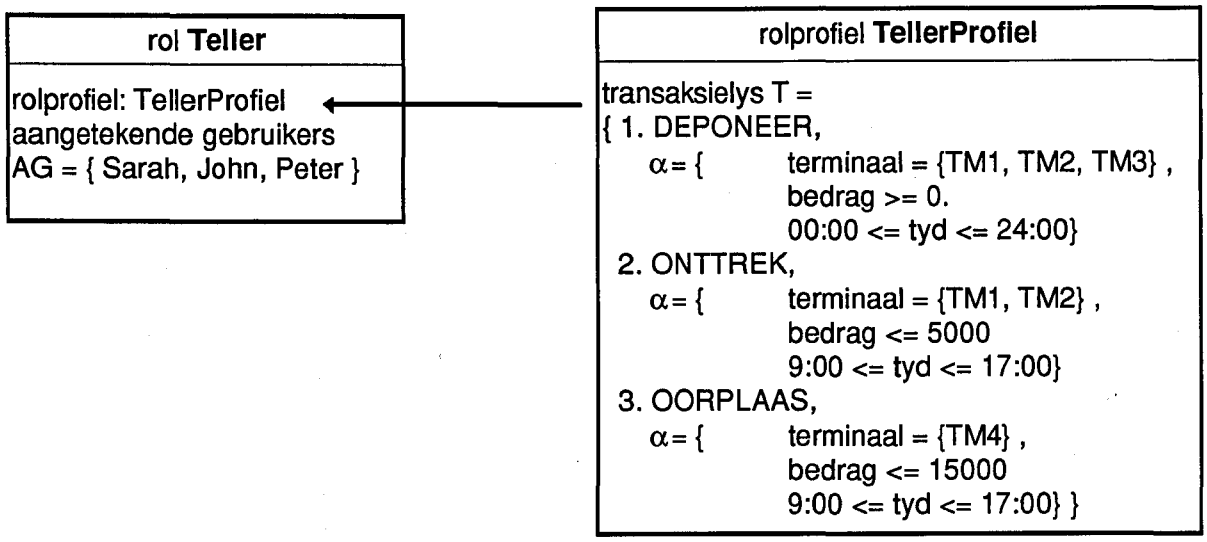
Let op dat *f* nie alleen toets of rol *r* transaksie *t* mag uitvoer nie, maar ook of *t* uitgevoer mag word onder die omstandighede waaronder *t* versoek is soos in *a* beskryf is.

Indien enige een van die vier voorwaardes hierbo nie geldig is nie, is die **resultaat van *f* Onwaar.**

6.4.5 Voorbeeld

Om die magtiging van transaksies deur ORITO te illustreer beskou die volgende voorbeeld.

Figuur 6.6 toon die voorbeeld diagrammaties. Gestel rol Teller bestaan en TellerProfiel is die rolprofiel vir rol Teller. Verder het rol Teller 'n lys van gebruikers AG, wat aangeteken is en gekoppel is aan rol Teller, hierdie lys noem ons AG = {Sarah, Peter, John}. Die rolprofiel TellerProfiel bevat 'n transaksieys T met transaksiename en toelaatbare omstandighede. $Teller_{AG}$ is 'n lys van gebruikers wat, toe hulle by die stelsel aangeteken het, die rol Teller gekies het.



Figuur 6.6
'n Voorbeeld van 'n rol en ooreenstemmende rolprofiel met 'n transaksielys.

Deur nou die voorbeeld in figuur 6.6 te beskou kan die volgende magtigingsversoeke opstel en vasstel of die versoeke gemagtig of geweier word:

1. Gestel gebruiker John is aangeteken op die stelsel as 'n Teller en wil 'n transaksie DEPONEER vanaf terminaal TM1 vir 'n bedrag van 250 uitvoer en die tyd is 10:15. Die omstandighede waaronder die transaksie versoek is, is a = (TM1, 250, 10:15). Die 4-tal wat funksie f benodig lyk dus as volg: (John, Teller, DEPONEER, a). Wat is die resultaat van f(John, Teller, DEPONEER, a)?
   Al drie die voorwaardes van die funksie moet waar wees:
   i) John $\in$ Teller_{AG} is waar,
   ii) DEPONEER $\in$ TellerProfiel_{T} is waar en
   iii) $a \in \text{TellerProfiel}_{T_\alpha(\text{DEPONEER})}$ is waar.
   Al drie die voorwaardes is waar, dus f is waar en gevolglik mag die transaksie wat deur gebruiker John versoek is uitgevoer word.

2. Gestel gebruiker Sarah is aangeteken op die stelsel as 'n Teller en wil 'n transaksie ONTTREK vanaf terminaal TM5 vir 'n bedrag van 1000 uitvoer en die tyd is 9:00. Die omstandighede waaronder die transaksie versoek is, is a = (TM5, 1000, 9:00). Die 4-tal wat funksie f benodig lyk dus as volg: (Sarah, Teller, ONTTREK, a). Wat is die resultaat van f(Sarah, Teller, ONTTREK, a)?
   Al drie die voorwaardes van die funksie moet waar wees:
   i) Sarah $\in$ Teller_{AG} is waar,
   ii) ONTTREK $\in$ TellerProfiel_{T} is waar en
   iii) $a \in \text{TellerProfiel}_{T_\alpha(\text{ONTTREK})}$ is onwaar omdat die toelaatbare omstandighede vir transaksie ONTTREK nie terminaal TM5 toelaat nie.
   Al drie die voorwaardes is nie waar nie, f is onwaar en gevolglik mag die transaksie wat deur gebruiker Sarah versoek is nie uitgevoer word nie.

In hierdie gedeelte is 'n formele beskrywing gegee vir die magtiging van transaksies deur ORITO. Bogenoemde is dus die formele grondslag vir die model. In die volgende dele van hierdie hoofstuk word aandag gegee aan die ander twee

hoofkomponente van ORITO, naamlik magtiging van transaksies deur rolprofielobjekte en die voorstelling van rolverwantskappe met 'n roltralie.

6.5 Magtiging van transaksies in ORITO deur rolprofielobjekte (RPOs)

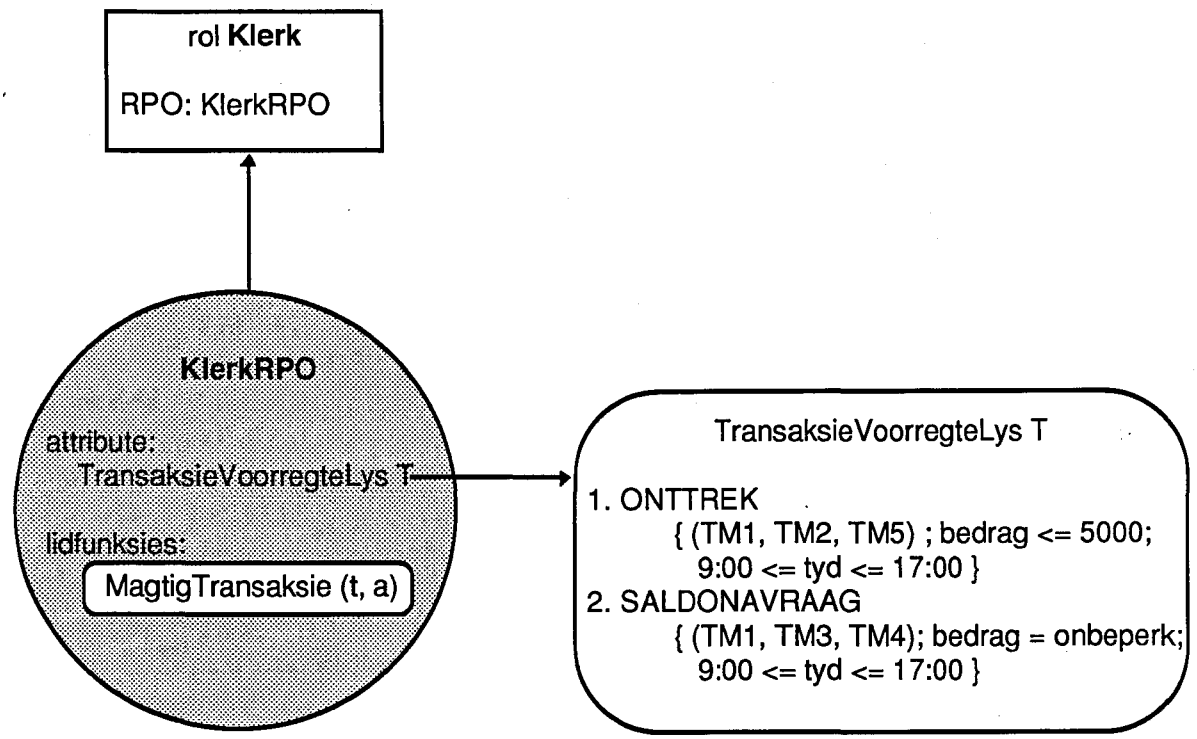
In hoofstuk 3 het ons die begrip rolprofielobjek (RPO) bespreek. Objek-georiënteerdheid hou baie voordele in en die voordele van die gebruik van rolprofielobjekte is in 3.9 op p.49 bespreek. In hierdie gedeelte kyk ons hoe ORITO die voordele van rolprofielobjekte benut deur gebruik te maak van rolprofielobjekte wat die uitvoer van transaksies in 'n transaksieverwerkerstelsel magtig.

Onthou dat in hoofstuk 3 is gesê dat in 'n stelsel waar rolgebaseerde inligtingsekerheid gebruik word, daar vir elke rol in die organisasie 'n rol geïdentifiseer word. Elkeen van hierdie rolle bevat die basiese inligting van die rol, met ander woorde inligting soos watter rolle is ouers of kinders van hierdie spesifieke rol. Die voorregte (die transaksies wat 'n rol mag uitvoer) word nie in die rol self gestoor nie maar in 'n rolprofiel. Vir elke rol in die stelsel is daar dus 'n rolprofiel wat (in terme van transaksies) beskryf watter transaksies deur so 'n rol en onder watter toelaatbare omstandighede uitgevoer mag word.

Ons het gesê dat 'n rolprofielobjek 'n objek-georiënteerde objek is wat die magtiging vir 'n spesifieke rol doen. In hierdie hoofstuk is ons besig om 'n model vir objek-georiënteerde, rolgebaseerde inligtingsekerheid in 'n transaksieverwerking omgewing te formuleer en te bekryf. In die vorige afdeling het ons die magting van transaksies in ORITO formeel beskryf. In hierdie afdeling kyk ons hoe hierdie magtiging gedoen word wanneer rolprofielobjekte gebruik word.

Onthou dat die inligting van enige OO objek geënkapsuleer is. Dit beteken dat wanneer 'n stelsel die uitvoer van 'n transaksie wil magtig deur 'n rolprofielobjek te gebruik kan die stelsel nie die lys van voorregte (transaksies met toelaatbare omstandighede) binne in die RPO sien nie. Die stelsel *roep 'n lidfunksie* van die RPO en kry slegs 'n magtig of weiering boodskap terug.

Om die werking van ORITO te illustreer in terme van rolprofielobjekte beskou die volgende voorbeeld. Gestel in 'n organisasie is daar 'n rol Klerk. Die rol Klerk het 'n rolprofielobjek KlerkRPO wat die uitvoer van transaksies magtig vir hierdie rol. Die transaksies wat rol Klerk mag uitvoer en hulle toelaatbare omstandighede word in die attribute van KlerkRPO gestoor. Figuur 6.7 toon die voorbeeld diagrammaties.



Figuur 6.7
Voorbeeld van 'n rol met 'n rolprofielobjek

Let op dat in figuur 6.7 die transaksievoorregtelys van die rolprofielobjek KlerkRPO getoon word. Dit is egter nie moontlik vir stelselkomponente om die inhoud van die transaksievoorregtelys te sien nie; toegang tot hierdie lys is slegs moontlik deur die lidfunksie MagtigTransaksie.

Veronderstel nou gebruiker Peter wil transaksie ONTTREK uitvoer en transaksie ONTTREK word uitgevoer onder die omstandighede a (onthou $a \in A$ is 'n versameling omstandighede (eienskappe) van die transaksie, bv. die bedrag van transaksie en die tyd van die transaksie). Gestel a = { terminaal = TM1, bedrag = 1000, tyd = 9:55 } en Peter is aangeteken as rol Klerk op die stelsel.

Wanneer is Peter volgens ORITO gemagtig om die transaksie ONTTREK onder omstandighede a uit te voer indien ons van rolprofielobjekte gebruik maak?

In terme van die funksie f soos in die vorige afdeling gedefinieer is, is gebruiker u gemagtig om transaksie t met omstandighede a uit te voer indien daar 'n rol r bestaan sodanig dat

$f(u, r, t, a)$ = Waar, met f gedefinieer soos in 6.4.2,

Andersins is G nie gemagtig om t onder a uit te voer nie.

Vir die voorbeeld sal dit beteken dat die stelsel eerstens moet bepaal aan watter rol Peter tans gekoppel is, die besonderhede hiervan word later bespreek. Gestel nou die stelsel het bepaal dat Peter aan rol Klerk gekoppel is. Die stelsel bepaal vervolgens

watter RPO in die stelsel hanteer hierdie rol se magtig. Dit word gedoen deur in rol Klerk se rolprofielnaam veld te kyk en sodoende 'n geheuewyser na die rol se rolprofielobjek te kry. Die rolprofielobjek KlerkRPO hanteer magtiging vir rol Klerk. Die stelsel stuur 'n boodskap aan die rolprofielobjek KlerkPRO om die lidfunksie MagtigTransaksie uit te voer en stuur aan die rolprofielobjek as parameters die naam van die transaksie en die lys van omstandighede waaronder die transaksie versoek is. In kort skryf ons KlerkRPO.MagtigTransaksie(ONTTREK, a) waar a = { terminaal = TM1, bedrag = 1000, tyd = 9:55 } .

Die rolprofielobjek KlerkRPO ontvang die MagtigTransaksie boodskap en toets of die transaksie ONTTREK in sy transaksievoorregtelys T voorkom (ONTTREK $\in$ $Klerk_T$), indien wel toets dit of die lys van omstandighede in *a* deel is van die toelaatbare omstandighede van die transaksie in sy transaksievoorregtelys (a $\in$ $Klerk_{T_a(ONTTREK)}$). Indien wel, stuur die objek 'n Magtig boodskap terug, andersins word 'n Weier boodskap teruggestuur aan die roepende objek/stelsel eenheid. Volgens die voorbeeld sal die transaksie ONTTREK gemagtig word want ONTTREK is deel van die transaksievooregtelys van KlerkRPO en die omstandighede a waaronder die transaksie versoek is, is deel van die toelaatbare omstandighede vir transaksie ONTTREK in die transaksievoorregtelys van KlerkRPO.

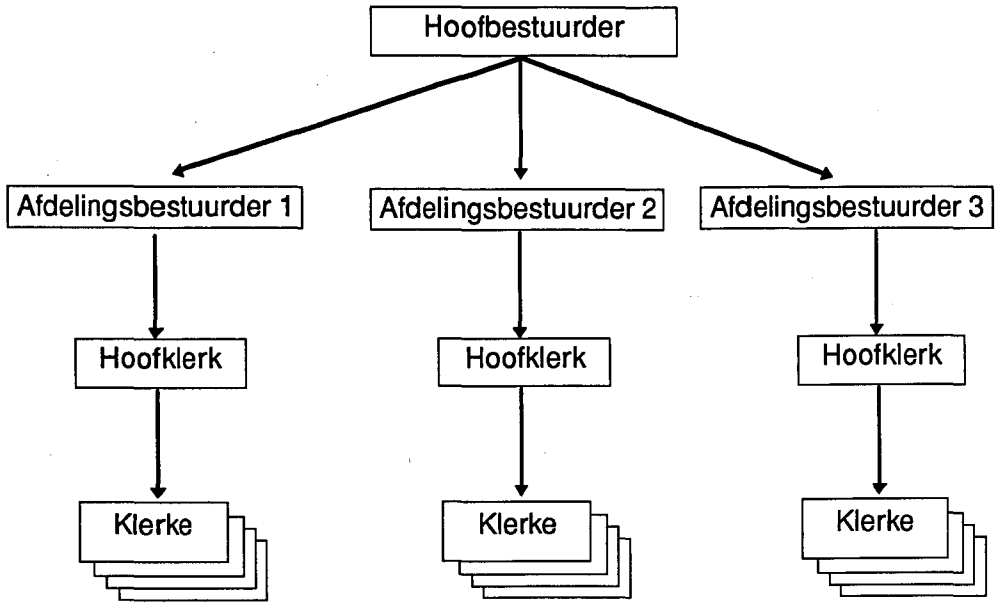
In hierdie gedeelte is getoon hoe ORITO van die voordele van objek-georiënteerdheid gebruik kan maak deur rolprofielobjekte te gebruik om die uitvoer van transaksies te magtig. In die volgende gedeelte gee ons aandag aan die laaste komponent van ORITO, naamlik om roltralies te gebruik om die verwantskap tussen rolle voor te stel.

6.6 Rangskikking van rolle in ORITO

(Voorstelling van die verwantskap tussen rolle met roltralies)

In hierdie gedeelte word beskryf hoe die verwantskap tussen rolle in ORITO voorgestel word deur van 'n traliegrafiek gebruik te maak. Die voordele van so 'n grafiek word bespreek en daar word ook gewys op moontlike ander voorstellings wat oorweeg kan word en waarom op 'n traliegrafiek vir ORITO besluit is.

In meeste organisasies is besigheidsrolle in een of ander hiërargie gerangskik soos byvoorbeeld die hiërargie in figuur 6.8. Let op dat daar nie mense in die hiërargie voorkom nie - die hiërargie bestaan uit *besigheidsrolle* wat van tyd tot tyd deur verskillende mense opgeneem kan word.



Figuur 6.8
Voorbeeld van 'n hiërargie van besigheidsrolle.

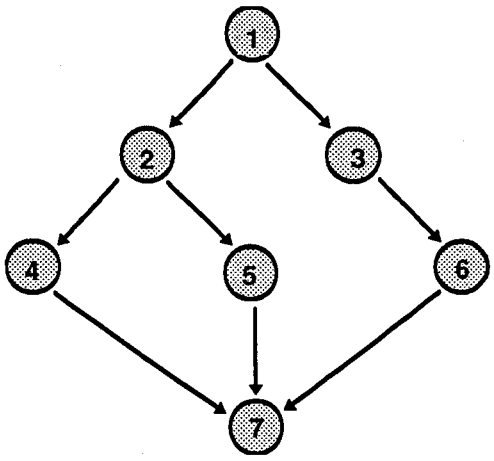
In ORITO word die rolle in die stelsel ook volgens 'n hiërargie gerangskik. Die rolle word gerangskik volgens 'n traliegrafiek. In [1] word rolle ook volgens 'n traliegrafiek gerangskik. ORITO gebruik van die beginsels in [1] maar gebruik 'n gewysigde vorm van die traliegrafiek in [1].

In hoofstuk 4 het is van die voordele van traliegrafieke uitgewys. Die gebruik van die traliegrafiek om die verwantskap tussen rolle voor te stel maak dit makliker om die verwantskap tussen die rolle met een oogopslag te sien.

6.6.1 'n Roltralie

In hierdie gedeelte beskryf ons 'n traliegrafiek wat die verwantskap tussen rolle voorstel. Ons herhaal weer die definisie vir 'n traliegrafiek uit hoofstuk 4.

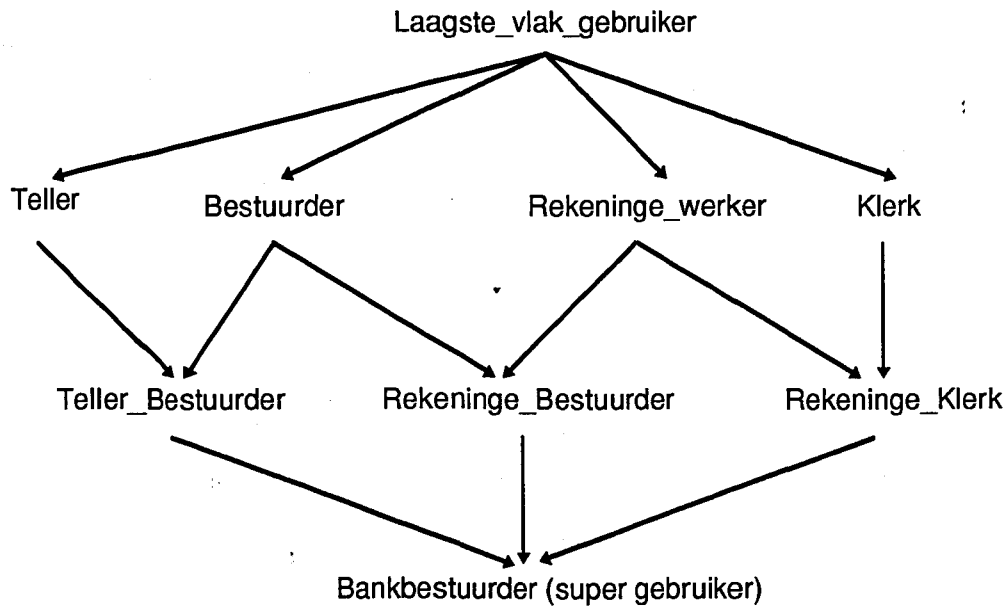
'n Gerigte traliegrafiek, of kortweg 'n tralie, is 'n gerigte asikliese grafiek waar presies een punt geen ouers het nie en een of meer kinders het , en presies een punt het geen kinders nie en een of meer ouers. Alle ander punte het ten minste een ouer en ten minste een kind. Figuur 6.9 toon 'n voorbeeld van 'n tralie. Punt 1 het geen ouers en twee kinders, punt 7 het geen kinders en drie ouers, al die ander punte het ten minste een ouer en ten minste een kind.



Figuur 6.9.
'n Gerigte traliegrafiek.

Definisie 6.1: *'n Roltralie is 'n gerigte traliegrafiek, waar elke nodus in die grafiek 'n rol in die stelsel voorstel. Die roltralie dui die verwantskap tussen rolle aan. 'n Gerigte boog van rol A in die roltralie na rol B in die roltralie dui aan dat rol B geskep word deur voorregte onder andere te erf van rol A. Dit beteken dat rol B al die transaksies kan uitvoer wat rol A kan, want as rol A transaksie T bevat, bevat rol B nou ook transaksie T.*

Beskou nou die volgende voorbeeld roltralie in figuur 6.10 en die verduideliking wat daarop volg.



Figuur 6.10
'n Voorbeeld roltralie.

Let op dat die roltralie in figuur 6.10 voldoen aan die vereistes van 'n gerigte traliegrafiek deurdat die nodus vir rol Laagste_vlak_gebruiker geen ouers bevat en

vier kinders terwyl die nodus vir rol Bankbestuurder geen kinders bevat en drie ouers, al die ander nodusse bevat ten minste een kind en ten minste een ouer.

Die gerigte boog van nodus Teller na nodus Teller_Bestuurder beteken dat die rol Teller_Bestuurder van die transaksievoorregte van rol Teller erf. Laasgenoemde verwantskap geld vir alle rolle in die roltralie wat met 'n boog verbind is. Dit beteken dus dat rolle hoër-op in die roltralie minder voorregte het as rolle laer-af in die roltralie. Rol Laagste_vlak_gebruiker het die minste voorregte en die rol Bankbestuurder het die vereniging van alle voorregte van rolle in die roltralie.

Die oorerwing van 'n transaksie ONTTREK van rol Teller na rol Teller_Bestuurder beteken ook dat die beperkings van transaksie ONTTREK by rol Teller oorgeërf word na rol Teller_Bestuurder. Byvoorbeeld, gestel die transaksie ONTTREK by rol Teller die beperking het dat die transaksies slegs uitgevoer mag word vir bedrag <= 1000, dan sal die beperking vir die transaksies ook geld by rol Teller_Bestuurder. By rol Teller_Bestuurder kan egter nog transaksies gevoeg word.

Let ook op dat transaksies wat in Teller is in Teller_Bestuurder gewysig kan voorkom. Dit is byvoorbeeld moontlik dat rol Teller transaksie OORPLAAS bevat en sekere beperkings vir OORPLAAS, bv. bedrag van transaksie <= 10 000. Rol Teller_Bestuurder kan egter reeds transaksie OORPLAAS bevat maar met ander beperkings, bv. die bedrag van transaksie <= 20 000. Om te hou by die benadering dat rolle laer-af in roltralie meer voorregte bevat as rolle hoër-op in die roltralie vereis ons dat wanneer laasgenoemde geval voorkom, is die beperkings vir OORPLAAS wat in Teller_Bestuurder gevoeg word, die minimum van die twee beperkings (die vereniging van die voorregte). Dit beteken dat OORPLAAS in Teller_Bestuurder sal die beperking bevat dat bedrag <= 20 0000.

In hierdie afdeling is beskryf hoe die verwantskap tussen rolle in ORITO voorgestel word. In die volgende afdeling word die formele beskrywing van magtiging deur ORITO uitgebrei om die rangskikking van rolle in 'n roltralie in ag te neem.

6.6.2 Formele beskrywing van magtiging deur ORITO vir rolle gerangskik in 'n roltralie

Wanneer rolle gerangskik is in 'n roltralie beteken dit dat sekere rolle voorregte erf van ander ouer-rolle. Wanneer ORITO die uitvoer van 'n transaksie vir 'n gebruiker wat aan 'n rol gekoppel is moet magtig, is dit nodig om te kyk na die ouer-rolle van die rol waaraan die gebruiker gekoppel is. Dit is moontlik dat 'n gebruiker gekoppel is aan 'n rol wat nie die nodige voorregte het nie, maar 'n ouer-rol het wat wel oor die nodige voorregte beskik. In so 'n geval moet die transaksie gemagtig word.

Die formele beskrywing van 'n magtiging vir 'n transaksie in ORITO word nou herhaal en uitgebrei waar nodig om van 'n roltralie gebruik te maak.

'n magtiging vir 'n transaksie in ORITO waar rolle in 'n roltralie gerangskik is, is 'n 4-tal (u, r, t, a) waar

$u \in U$,

die versameling van gebruiker identifiseerders (soos in hoofstuk 2 beskryf is) in die stelsel;

$r \in R$,
die versameling rolle in die stelsel;

$t \in T$,
die versameling transaksies wat deur die transaksiesverwerker(s) van die stelsel uitgevoer kan word; en

$a = (a_1, a_2, ..., a_n) \in A = (A_1, A_2, ..., A_n)$.

A is 'n vektor waarvan die elemente die domein van toelaatbare omstandighede op transaksies is. a is 'n vektor waarvan die elemente $(a_1, a_2, ..., a_n)$ aandui onder watter omstandighede is 'n transaksie versoek, bv. vanaf watter terminaal en vir watter bedrag.

Ons gebruik dieselfde notasie soos wat in 6.4.4 beskryf is.

Let op dat ons hier die aangetekende gebruikers in die rol stoor. Later in die dokument vereis ons egter dat die aangetekende gebruikers in 'n aparte tabel gestoor word. Laasgenoemde vergemaklik die implementasie in 'n verspreide omgewing. Om egter 'n formele definisie vir magtiging in ORITO te gee, hou ons by eersgenoemde.

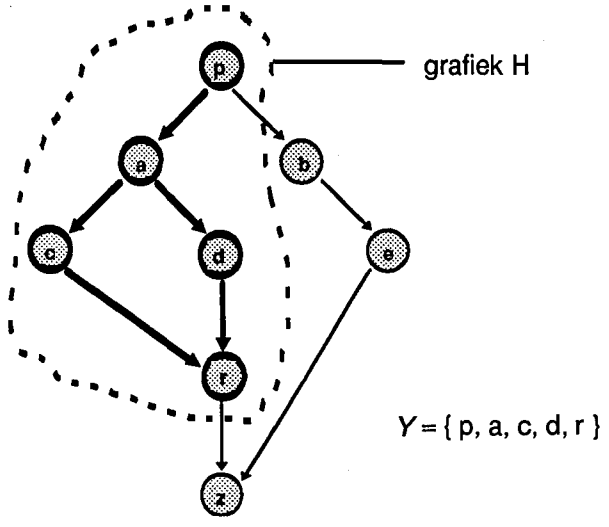
'n Funksie f is gedefineer om te bepaal of die magtiging (u, r, t, a) Waar of Onwaar is:
$f: U \times R \times T \times A \rightarrow$ (Waar, Onwaar).

Gegee 'n 4-tal (u, r, t, a), die funksie f moet bepaal of gebruiker u gekoppel is aan rol r en of rol r gemagtig is om transaksie t uit te voer onder die omstandighede wat in a beskryf word.

Met ander woorde, gegee (u, r, t, a)

laat p = die rol in die roltralie met geen ouers (die rol op vlak 0 in die roltralie).
laat H = die grafiek van die vereniging van alle gerigte paaie in die roltralie vanaf rol q na rol r.
laat Y = die versameling van alle rolle in die grafiek H.

Figuur 6.11 toon 'n voorbeeld van bogenoemde.



Figuur 6.11.
'n Voorbeeld roltralie

dan is $f(u, r, t, a)$ = Waar, indien

daar 'n rol p bestaan sodanig dat $p \in Y$ en

$u \in r_{AG}$ (gebruiker met identifiseerder u is gekoppel aan rol r en aangeteken as 'n gebruiker van tipe rol r),

en

$t \in p_T$ (*transaksie t is in die transaksielys van rol p se rolprofiel*),

en

$a \in p_{T_\alpha(t)}$ (*die omstandighede waaronder transaksie t versoek is, is deel van die toelaatbare beperkings* α *van transaksie t in die transaksielys T van rol p*).

Let op dat $a \in p_{T_\alpha(t)}$ beteken dat elkeen van die omstandighede (a_1, a_2, ..., a_n) van transaksie t in die ooreenkomstige toelaatbare omstandighede (α_1, α_2, ... , α_n) van die transaksie t in die transaksielys van rol p se rolprofiel is. Byvoorbeeld, gestel 'n transaksie t word versoek onder die omstandighede a_1 = TM2 (terminaal_id), a_2 = 1000 (bedrag van transaksie) en a_3 = 10:33 (tyd van transaksie). Indien $a \in p_{T_\alpha(t)}$ dan is TM2 $\in \alpha_1$, 1000 $\in \alpha_2$ en 10:33 $\in \alpha_3$.

Let op dat f nie alleen toets of rol r transaksie t mag uitvoer nie, maar ook of enige een van die rolle waaruit rol r voorregte erf transaksie t mag uitvoer.

Indien enige een van die vier voorwaardes hierbo nie geldig is nie, is die **resultaat van** f **Onwaar.**

Bogenoemde magtiging sê dus dat die funksie *f* waar is indien 'n gebruiker aan rol *r* gekoppel is en die gebruiker wil 'n transaksie uitvoer waarvan die transaksie en die omstandighede waaronder die transaksie versoek is in die rol *r* of in enige van *r* se ouers is.

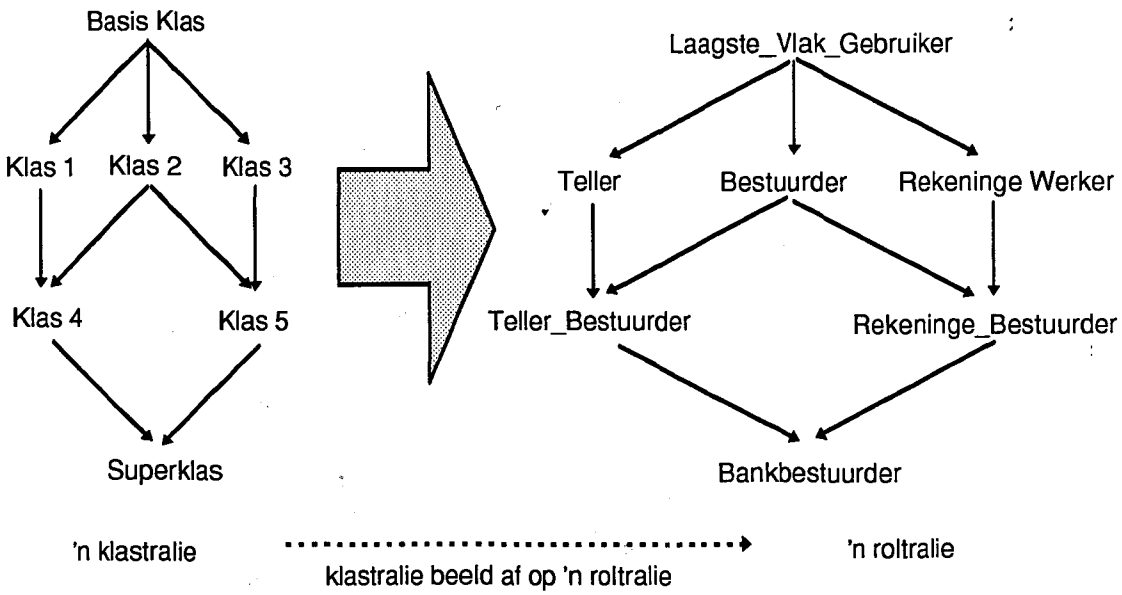
In die volgende afdeling kyk ons na die voordele van die rangskikking van rolle in 'n roltralie.

6.7 Voordele van die rangskikking van rolle in 'n roltralie

Daar is verskeie metodes waarvolgens die verwantskappe tussen rolle in 'n rolgebaseerde inligtingsekerheidstelsel voorgestel kan word. 'n Traliegrafiek het sekere definitiewe voordele soos reeds in hoofstuk 4 beskryf is. In hierdie afdeling gee ons weer aandag aan die voordele van die rangskikking van rolle in 'n roltralie.

6.7.1 'n Roltralie en 'n klastralie is nou verwant.

In hoofstuk 4 is getoon dat 'n traliegrafiek gebruik kan word om die verwantskap tussen klasse voor te stel in 'n objek-georiënteerde stelsel wat meervoudige oorerwing toelaat. Let op dat die rolgebaseerde inligtingsekerheidstelsel wat in ORITO gebruik word ook meervoudige oorerwing toelaat - nuwe rolle kan geskep word deur die transaksievoorregte van 'n aantal rolle te kombineer. Figuur 6.12 toon 'n voorbeeld van 'n klastralie en toon 'n roltralie wat daarop afbeeld. Let op dat dat 'n roltralie dusverwant is aan 'n klastralie en omdat ORITO dit moontlik maak om nuwe rolle te skep deur meervoudige oorerwing, net soos in die klastralie, beteken dit dat 'n traliegrafiek 'n goeie voorstelling is vir die verwantskap tussen rolle in ORITO.

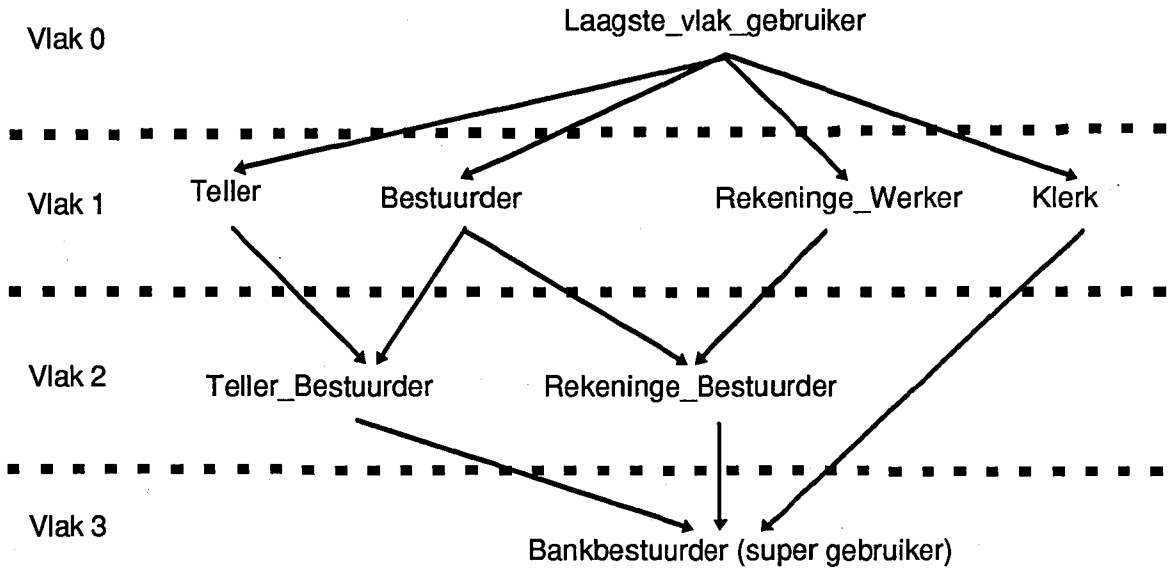


Figuur 6.12
'n Voorbeeld klastralie en 'n roltralie wat daarop afbeeld.

6.7.2 'n Roltralie maak die koppeling van rolle aan gebruikers makliker

Die grafiese voorstelling van die roltralie help die sekerheidsbestuurder om te besluit watter rolle om aan watter gebruikers toe te ken. In figuur 6.13 word getoon dat die

rolle in die roltralie in vlakke verdeel is. Hoër vlakke se rolle het meer voorregte as laer vlakke se rolle (vlak 0 is die rolle wat bo aan die roltralie voorkom). Die sekerheidsbestuurder kan dus maklik sien watter vlak rol sy aan 'n gebruiker toeken. Byvoorbeeld, 'n Rekeninge_Bestuurder en 'n Teller_Bestuurder is op dieselfde vlak maar 'n Teller is op 'n laer vlak en het minder voorregte.

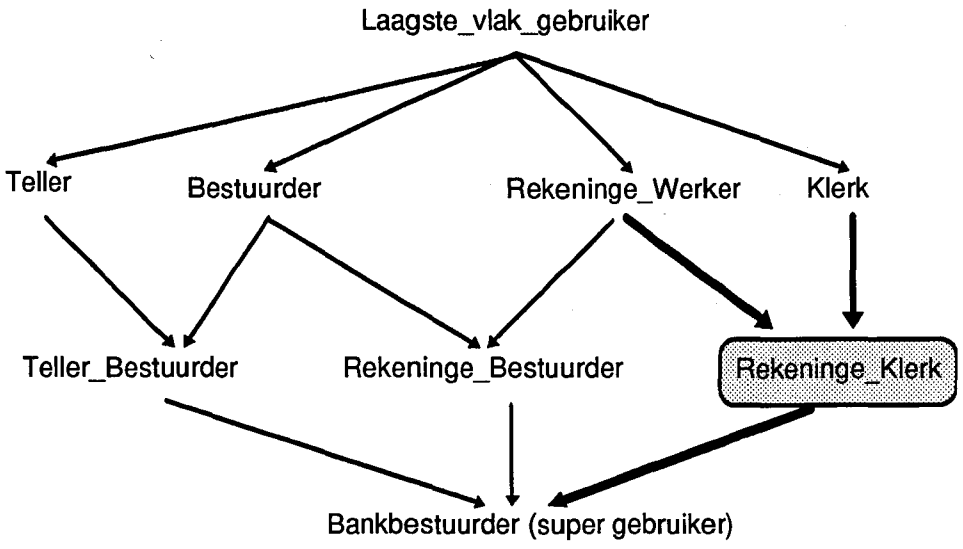


Figuur 6.13
Vlakke in 'n roltralie.

In sommige organisasies hang die voorregte van 'n rol af van een of ander vlak (in 'n militêre opset kan dit byvoorbeeld van rang afhang)

6.7.3 'n Roltralie maak dit makliker om nuwe rolle te skep.

Deur die roltralie te beskou, kan die sekerheidsbestuurder ook makliker besluit watter ouer- en kindrolle om aan 'n nuwe rol toe te ken. Beskou weer die roltralie in figuur 6.13. Gestel daar moet 'n nuwe rol geskep word en hierdie rol verteenwoordig 'n persoon in die organisasie wat die pligte van 'n Klerk en 'n Rekeninge_Werker sal hê en moontlik ook ander pligte. Die sekerheidsbestuurder kan maklik 'n nuwe rol, sê Rekeninge_Klerk, skep deur die rolle Klerk en Rekeninge_Werker te kombineer en 'n paar transaksievoorregte by te voeg. Die rol Bankbestuurder erf dan die transaksievoorregte van hierdie nuwe rol om te hou by ons tralie benadering. Figuur 6.14 toon die roltralie in figuur 6.13 met die nuwe rol ingevoeg.



Figuur 6.14
Die byvoeging van 'n rol in 'n roltralie.

6.7.4 Ontleding van rolle deur rolanalises met die roltralie te doen

Omdat 'n roltralie 'n gerigte asikliese grafiek is, kan interessante rolanalises gedoen word, bv. watter transaksies twee rolle in die roltralie in gemeen het, deur grafiekteorie algoritmes op die roltralie toe te pas.

Soos voorheen genoem maak die rangskikking van rolle in 'n roltralie die toekenning en beskouing van rolverwantskappe maklik. 'n Roltralie kan ook handig gebruik word wanneer ons sekere rol analises wil doen soos byvoorbeeld om te bepaal uit watter rolle 'n bestaande rol geërf het. Ons beskou 'n paar voorbeelde.

Indien daar 'n gerigte pad van rol A na rol N in die roltralie bestaan, beteken dit dat A 'n *superrol* van N is. Dit beteken dat elke transaksie wat A kan uitvoer moet N ook kan uitvoer.

Gestel ons het 'n redelike uitgebreide roltralie en wil bepaal of 'n rol Teller die regte het van 'n rol Klerk. Omdat die roltralie 'n gerigte asikliese grafiek is kan ons nou die eienskappe van die traliegrafiek gebruik om die analise te bepaal. Ons hoef bloot te kyk of daar 'n gerigte pad vanaf die nodus Klerk na die nodus Teller in die grafiek bestaan. Daar is verskeie grafiekteorie algoritmes beskikbaar om sulke soektogte deur 'n gerigte grafiek te doen, byvoorbeeld die *topologiese ordening soek* algoritme soos in [4]. Indien daar dus 'n gerigte pad vanaf Klerk na Teller in die roltralie bestaan het Teller van rol Klerk geërf en het die regte van rol Klerk.

Ons kan deur soortgelyke algoritmes op die roltralie toe te pas ander eienskappe van die rangskikking van die rolle in die roltralie bepaal. Soos byvoorbeeld 'n lys van alle rolle wat in die oorerwingsgrafiek van 'n rol is of ons kan bepaal of twee rolle in die roltralie identies is en 'n oorbodige rol uitskakel ens.

In 'n organisasie kan verdeling van pligte tot gevolg hê dat 'n persoon byvoorbeeld in 'n posisie is waar hy 'n bestelling mag plaas maar nie mag goedkeur ook nie. Dit is

moontlik om deur 'n statiese ontleding van rolle wat aan 'n gebruiker toegeken mag word en die roltralie te verseker dat sulke situasies (wat in die sekerheidsbeleid uitgespel behoort te wees) nie voorkom nie. Indien dit wel voorkom, kan die roltralie gebruik word om vas te stel wat die mees bevoorregte rolle is waar die kombinasie van regte nie voorkom nie en 'n aanbeveling kan aan die sekerheidsbestuurder gemaak word.

Die super gebruiker rol in 'n roltralie se transaksievoorregte is altyd die vereniging van die transaksievoorregte van al die rolle. Dit is dus 'n maklike manier om 'n lys van alle voorregte in die stelsel te kry.

6.7.5 'n Roltralie kan die sekerheidsmagtiging meer ekonomies maak

Indien regte by 'n rol gevoeg word, kan dit aanleiding gee tot die skepping van 'n nuwe rolprofiel vir die betrokke rol (sien p.13). Die rangskikking van die rolprofiele in 'n tralie-grafiek maak die stelsel meer ekonomies. Beskou die volgende as motivering. Wanneer 'n nuwe rolprofiel geskep (of 'n bestaande een verander word) kan dit met ouer- en kindrolprofiele vergelyk word om te sien of dit dalk moontlik is om rolprofiele te integreer. Dit kan laer looptydkoste tot gevolg hê teen ietwat hoër koste tydens verandering aan rolle.

6.7.6 Gebruik van roltralie om rolprofielobjekte tussen rekenaars te versprei

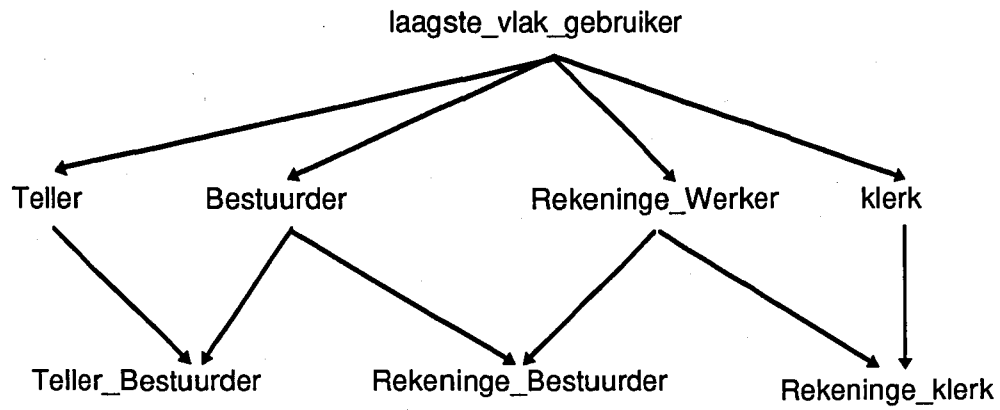
Later in die verhandeling gaan ons kyk hoe ons rolprofielobjekte tussen rekenaars kan versprei sodat in 'n groot organisasie die magtiging nader plaasvind aan die rekenaar wat die magtiging versoek het. Hiervoor sal die roltralie handig gebruik kan word om te sien watter rolle het logiese verwantskappe. Ons sal dus die roltralie kan opdeel en die nodige ouer- en kindrolprofielobjekte oor rekenaars kan versprei. Die verspreiding van rolprofielobjekte word egter in meer besonderhede in volgende hoofstukke bespreek. Neem egter kennis dat 'n roltralie hierdie verspreiding vergemaklik.

6.8 Ander oorwegings vir die voorstelling van rolverwantskappe

ORITO gebruik 'n traliegrafiek om rolverwantskappe mee voor te stel. Daar is egter 'n paar moontlike manier om hierdie verwantskappe mee voor te stel. In hierdie afdeling bespreek ons 'n paar van hierdie moontlikhede en vergelyk hulle met 'n roltralie.

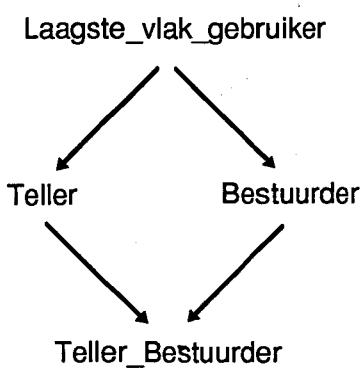
6.8.1 Roltralie vs. 'n gewone rolgrafiek

Ons het reeds gesê dat nuwe rolle gevorm kan word deur voorregte (of transaksies en hulle toelaatbare omstandighede) van ouer rolle te erf. Deur hierdie verwantskap grafies voor te stel kan ons 'n traliegrafiek van die verwantskap tussen rolle teken. In meeste stelsels sal ons egter nooit die super gebruiker soos in figuur 6.14 getoon is aan 'n gebruiker van die stelsel toeken nie. Indien ons ook nie die super gebruiker rol in die rolgrafiek insluit nie kry ons 'n *gewone rolgrafiek* soos in figuur 6.15.



Figuur 6.15
'n Rolgrafiek sonder 'n super gebruiker ('n gewone rolgrafiek)

In hierdie gedeelte word die insluit van die super gebruiker rol in die rolgrafiek gemotiveer. Eerstens, indien ons nie die super gebruiker rol in die rolgrafiek insluit nie, vorm die grafiek nie 'n roltralie nie maar 'n gewone rolgrafiek. Dit is nie 'n probleem nie, maar beskou die volgende situasie: Gestel ons wil uit die rolgrafiek in figuur 6.15 die rolle Laagste_vlak_gebruiker, Teller, Bestuurder en Teller_Bestuurder onttrek. Ons kan dit byvoorbeeld doen wanneer ons hierdie rolle se sekerheidsmagtiging op 'n aparte rekenaar wil doen (word in meer besonderhede bespreek in hoofstuk 8). Hierdie vier rolle vorm nou die rolgrafiek soos in figuur 6.16.



Figuur 6.16
'n Sub-rolgrafiek (roltralie)

Let op dat die rolgrafiek in figuur 6.16 is 'n subgrafiek van figuur 6.15 en vorm 'n tralie en nie 'n hiërargie nie.

Dit is altyd moontlik om uit 'n gewone rolgrafiek 'n roltralie te vorm: voeg bloot 'n nuwe super gebruiker onder aan die rolgrafiek by waarvan hierdie nuwe rol se voorregte die vereniging van al die ander rolle in die stelsel is. Dit is egter nie altyd moontlik om 'n rolgrafiek uit 'n roltralie te vorm nie: dit mag dalk nodig wees om 'n rol te verwyder wat wel in die stelsel gebruik word. 'n Roltralie kan dus vir meer gevalle gebruik word as 'n gewone rolgrafiek. Ons gaan die opdeling van rolgrafieke intensief bestudeer in volgende hoofstukke en om konsistentheid te behou (te sorg dat al die grafieke in die proses traliegrafieke bly), vereis ons dat elke rolgrafiek 'n roltralie is en daarom is ons aanvanklike rolgrafiek 'n traliegrafiek en nie 'n gewone rolgrafiek nie.

Die insluit van die super gebruiker rol maak dit ook vir ons maklik om die vereniging van voorregte in die stelsel te bepaal. Die super gebruiker rol se voorregte is juis die vereniging van al die voorregte in die stelsel.

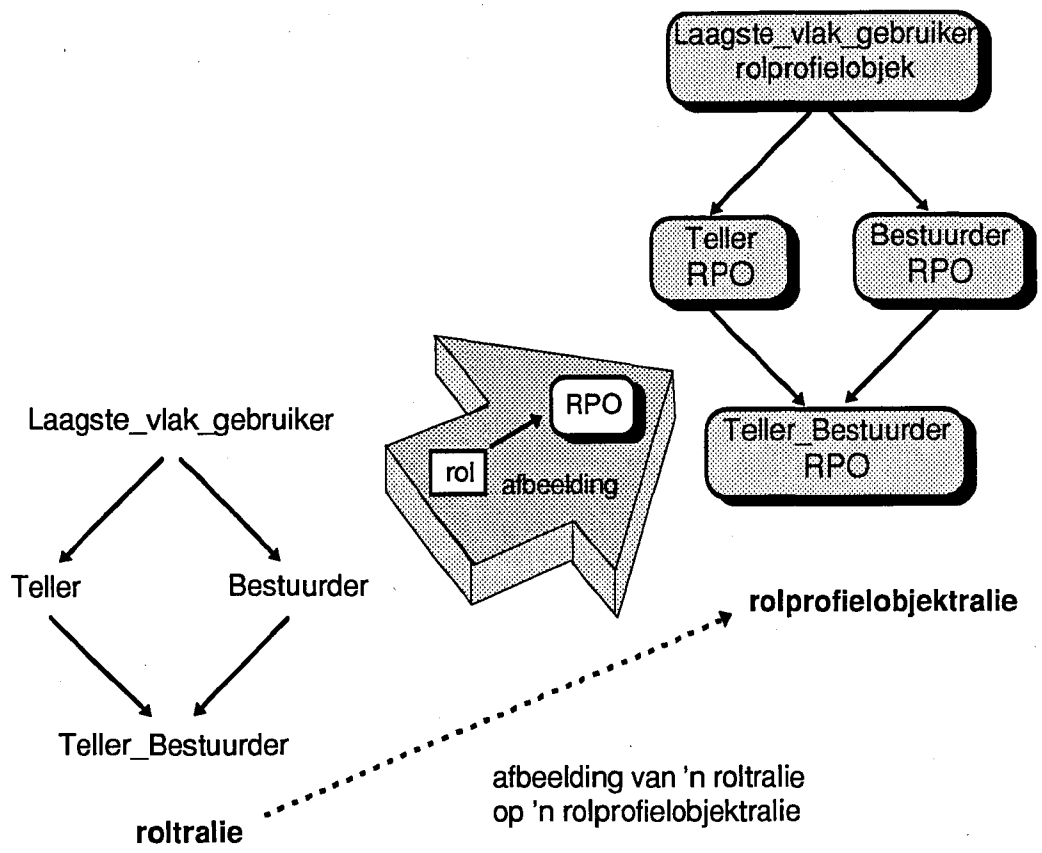
Tot dusver het ons nog net gekyk na die voorstelling van rolverwantskappe. Onthou dat 'n rol se voorregte eintlik in 'n objek-georiënteerde rolprofiel, 'n rolprofielobjek gestoor word. In die volgende gedeelte word 'n vergelyking getref tussen die voorstelling van rolverwantskappe met 'n roltralie en die voorstelling van rolprofielobjekverwantskappe met 'n rolprofielobjektralie.

6.8.2 Roltralie vs. rolprofielobjektralie

Vir elke rol in die stelsel is daar 'n rolprofiel wat die transaksievoorregte vir die rol stoor. In ORITO word die rolprofiele as objekte geïmplementeer (rolprofielobjekte). Is dit nie dalk beter om die verwantskap tussen rolprofielobjekte aan te dui eerder as die verwantskap tussen rolle nie? Ons beantwoord die vraag in hierdie gedeelte.

Definisie 6.2: *'n Rolprofielobjektralie is 'n gerigte traliegrafiek, waar elke nodus in die grafiek 'n rolprofielobjek in die stelsel voorstel. Die rolprofielobjektralie dui die verwantskap tussen rolprofielobjekte aan. 'n Gerigte boog van rolprofielobjek A in die roprofielobjektralie na rolprofielobjek B dui aan dat rolprofielobjek B geskep word deur transaksievoorregte onder andere te erf van rolprofielobjek A. Dit beteken dat rol B al die transaksies kan uitvoer wat rol A kan, want as rolprofielobjek A transaksie T bevat, bevat rolprofielobjek B nou ook transaksie T.*

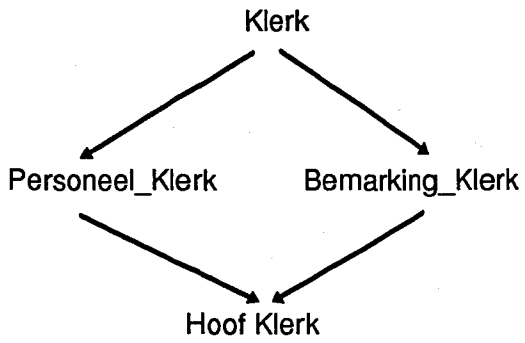
Onthou dat 'n rol inligting stoor soos die verwantskap van 'n rol met ander rolle terwyl 'n rolprofielobjek die transaksies stoor wat 'n rol mag uitvoer en die toelaatbare omstandighede waaronder dit gedoen moet word. Dit is dus moontlik om vir 'n elke roltralie 'n ooreenstemmende rolprofielobjektralie te skep. Figuur 6.17 toon dit aan.



Figuur 6.17
'n Roltralie en sy ooreenstemmende rolprofielobjektralie.

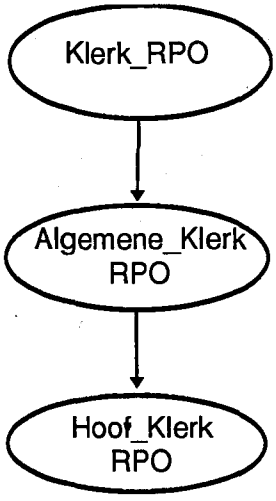
Die voorbeeld in figuur 6.16 laat dit lyk asof die rolle in 'n roltralie op 'n een-tot-een basis afbeeld op die rolprofielobjekte in 'n rolprofielobjektralie maar dit is nie noodwendig so nie. Beskou die volgende as motivering.

Gestel ons het 'n rol Klerk met rolprofielobjek Klerk_RPO , verder het ons 'n rol Pesoneel_Klerk met rolprofielobjek Personeel_Klerk_RPO en 'n rol Bemarking_Klerk met rolprofielobjek Bemarking_Klerk_RPO. Die rolle Personeel_Klerk en Bemarking_Klerk word gevorm deur die voorregte van die rol Klerk te erf. Laastens vorm ons rol Hoof_Klerk met rolprofielobjek Hoof_Klerk_RPO. Die rol Hoof_Klerk erf al die voorregte van rolle Bemarking_Klerk en Personeel_Klerk. Die roltralie vir hierdie rolle lyk soos in figuur 6.18.



Figuur 6.18.
'n Voorbeeld roltralie.

Dit kan nou gebeur dat rol Bemarking_Klerk en Personeel_Klerk se rolprofielobjekte presies dieselfde attribute en lidfunksies het. Die stelselbestuurder mag dalk weet dat rol Bemarking_Klerk en Personeel_Klerk in beginsel presies dieselfde is, maar verkies om hulle verskillende name te gee om te hou by die werklike besigheidsrolle wat in die organisasie bestaan. By die implementering van ORITO kan egter besluit word dat rol Bemarking_Klerk en rol Personeel_Klerk kry dieselfde rolprofielobjek (sê Algemene_Klerk_RPO). In so 'n geval is die rangskikking van die rolprofielobjekte van die stelsel soos in figuur 6.19.



Figuur 6.19.
'n Rolprofielobjekttrale vir rolle in figuur 6.18.

Bostaande dui aan dat die rolprofielobjektralie eenvoudiger kan wees as die roltralie. Die rolprofielobjektralie verloor egter inligting omdat dit nie noodwendig 'n rolprofielobjek vir elke rol bevat nie. Ons gebruik dus voortaan slegs die roltralie. Hou egter in gedagte dat vir elke roltralie daar 'n rolprofielobjektralie is wat moontlik eenvoudiger is.

Daar bestaan verskeie maniere om rolverwantskappe mee voor te stel (roltralies, rolhiërargieë, rolprofielobjekte, ens.). Elkeen van die voorstellings het sy eie voordele en nadele. Ons volstaan met bogenoemde twee voorbeelde van waar 'n roltralie 'n

meer geskikte voorstelling is. In die volgende deel word aandag gegee aan die koppeling van gebruikers aan rolle in die stelsel.

6.9 Koppeling van gebruikers aan rolle in 'n stelsel wat ORITO implementeer

Wanneer 'n gebruiker 'n transaksie wil uitvoer word 'n sekerheidsversoek aan 'n program wat ORITO implementeer gestuur. Tot dusver het ons aangeneem dat die program wat ORITO implementeer 'n gebruikernaam ontvang, die rol waaraan die gebruiker tans gekoppel is asook die verlangde transaksie en die transaksie omstandighede. Ons het egter nog nie aandag gegee aan die koppeling van 'n gebruiker en 'n rol 'n werkende stelsel nie. Of in ander woorde gestel, hoe weet die program watter rol 'n gebruiker gekies het wanneer hy of sy by die stelsel aangeteken het?

Wanneer 'n gebruiker by 'n terminaal aanteken moet hy in meeste stelsels 'n gebruiker identifikasie en wagwoord verskaf en word dan geverifieer as 'n geldige gebruiker. In 'n stelsel waar ORITO geïmplementeer is, sal dit dit steeds nodig wees om die gebruiker se identiteit te verifieer en verder sal die gebruiker ook 'n rol moet kies vir sy huidige sessie. Dit kan as volg gebeur: Die gebruiker teken aan by die stelsel (deur sy identifikasie en wagwoord te verskaf). Voordat die gebruiker kan begin om transaksies uit te voer moet hy registreer by die sekerheidsbestuurderprogram. Dit word gedoen deurdat die stelsel se sekerheidsbestuurderprogram aan die gebruiker 'n lys van gemagtigde rolle verskaf waaruit hy een kies.

Die rekenaar waarop die sekerheidsmagtiging vir transaksies gedoen word, moet 'n tabel onderhou waar daar vir elke gebruiker van die stelsel 'n lys van rolle is waaruit so 'n gebruiker mag kies wanneer hy aanteken. Tabel 6.3 toon 'n voorbeeld van so 'n tabel.

GEBRUIKER_ROLLE	
Gebruikernaam	Rolname
Sarah	Klerk, Teller
Peter	Klerk, Salarisse_Werker
Mary	Personeelbestuurder
Joe	Bankbestuurder

Tabel 6.3
Voorbeeld van 'n GEBRUIKER_ROLLE tabel.

Let op dat indien 'n gebruiker nie meer die rekenaarstelsel gaan gebruik nie (soos wanneer 'n werknemer bedank) dan is al wat nodig is, om die gebruiker se naam uit die GEBRUIKER_ROLLE tabel te onttrek.

Sodra die gebruiker 'n rol gekies het maak die sekerheidsbestuurderprogram 'n aantekening van hierdie keuse van die gebruiker in 'n aparte tabel. In 2.3.1 op p.15 is hierdie tabel bespreek. Die tabel is 'n toegangsbeheerlys genoem. Let op dat daar

vervolgens na hierdie tabel verwys word as die ROLKEUSE tabel. Tabel 6.4 toon 'n voorbeeld van so 'n tabel.

ROLKEUSE	
Gebruikernaam	Huidige Rol
Sarah	Teller
Peter	*geen*
Mary	*geen*
Joe	Bankbestuurder

Tabel 6.4
Voorbeeld van 'n ROLKEUSE tabel ('n toegangsbeheerlys tabel).

Indien die gebruiker nie 'n rol uit die lys wil kies nie, is sy enigste ander keuse om af te teken van die stelsel. In Tabel 6.4 is gebruikers Peter en Mary nie aangeteken nie en gevolglik ook nie aan 'n rol gekoppel nie. Elke keer as 'n sekerheidsmagtiging gedoen moet word sal daar dus eers gekyk word aan watter rol die gebruiker gekoppel is volgens die ROLKEUSE tabel.

Wanneer die gebruiker afteken maak die sekerheidsbestuurder program 'n aantekening in die ROLKEUSE tabel dat die gebruiker tans nie aan 'n rol gekoppel is nie, bv. Peter en Mary in tabel 6.4.

Ons gee later weer aandag aan die koppeling van gebruikers aan rolle wanneer ons die gebruik van ORITO in 'n stelsel met meer as een sekerheidsbestuurderprogram beskou.

6.10 Die gebruik van die roltralie

In hierdie afdeling word opmerkings gemaak oor hoe die roltralie gebruik word in ORITO en watter verband dit hou met die rolprofielobjekte van die stelsel.

Ons weet dat rolle bevat nie werklik enige transaksievoorregte nie, dit word gestoor in rolprofielobjekte. Watter nut dien rolle in ORITO? 'n Belangrike nut van rolle in ORITO is dat dit die roltralie moontlik maak deurdat elke rol sy ouers en kinders stoor en uit hierdie inligting kan 'n roltralie opgebou word. Die voordele van 'n roltralie is reeds bespreek in 6.6.2.

Deur uit die ouer- en kindlys van elke rol in die stelsel 'n roltralie op te bou kan ons 'n grafiese beeld vorm van hoe rolle verband hou in die stelsel. Hierdie beeld help dan onder andere met die volgende:

- Die keuse vir watter rolle om aan watter gebruikers toe te ken. Die sekerheidsbestuurder gebruik die roltralie van 'n stelsel wat ORITO implementeer om te besluit watter rolle om by 'n gebruiker se lys van rolle te voeg waaruit hy kan kies. Onthou dat ons in die vorige afdeling gesê het dat hierdie lys word in 'n tabel GEBRUIKER_ROLLE gestoor.
- Die keuse vir watter rolle om te gebruik as ouers of kinders van nuwe rolle wat geskep word.

Die belangrike punt om hier na op te let is dat rolle en roltralies nie veel te make het met die werklike magtiging van transaksies in ORITO nie maar eerder help met die toekenning van rolle en die skep van nuwe rolle. 'n Roltralie is amper soos 'n masker vir die stelsel en gee 'n vinnige opsomming vir wat werklik agter die masker gebeur. In die volgende afdeling gee ons 'n kort samevatting van die hoofstuk en wys hoe die komponente inmekaar skakel.

6.11 Samevatting

Hierdie hoofstuk het 'n hele paar begrippe gedefinieer om uiteindelik die komponente van die model vir objek-georiënteerde rolgebaseerde inligtingsekerheid in transaksieverwerking omgewings te beskryf. Om die model meer duidelik te maak kyk ons opsommend hoe die komponente inmekaar pas.

Onthou dat ORITO hoofsaaklik een doel het: om die magtiging van transaksies te doen in 'n transaksieverwerker stelsel deur rolgebaseerde inligtingsekerheid op 'n objek-georiënteerde wyse te gebruik.

Vervolgens word die stappe bespreek wat gevolg word om ORITO te implementeer in 'n transaksieverwerker omgewing.

6.11.1 Stappe by die implementering van ORITO

6.11.1.1 Opstelling van die transaksieverwerker

Eerstens moet die transaksieverwerker so opgestel word dat vir elke transaksie wat uitgevoer moet word, die stelsel 'n program wat ORITO implementeer vra of die transaksie mag uitvoer of nie. In transaksieverwerkers soos CICS/6000 is dit moontlik om die opstelling so te doen dat alle magtiging van transaksies deur 'n eksterne sekerheidsmonitor ('n ESM) gedoen word [11]. Alle sekerheidsmagtiging word dus aan die eksterne program oorgelaat.

6.11.1.2 Skep van rolle

Ons verwys voorts na die program wat ORITO implementeer as die eksterne sekerheidsmonitor of ESM. Vir die ESM om die magtiging te kan doen moet 'n paar rolle gedefineer word. Dit behels dat daar eerstens 'n basiese of laagste vlak rol geskep moet word; hierdie rol bevat slegs die transaksies wat deur elke gebruiker van die stelsel uitgevoer mag word.

Die *skep* van 'n rol behels 'n paar dinge:

1. Die ESM skep 'n nuwe rolprofielobjek as instansie van die rolprofielklas soos voorheen beskryf is
2. Daar word besluit op 'n lys van ouers (al die rolle waarvan die nuwe rol transaksies erf) en 'n lys van kinders (al die rolle aan wie hierdie rol se transaksies oorgedra moet word). Hierdie twee lyste word gestoor in die attribute van die rolprofielobjek en in 'n tabel genaamd ROLLE.
3. Die ESM moet in die ROLLE tabel 'n inskrywing gaan byvoeg. Die inskrywing bevat die naam van die rol, die identifikasie van die rolprofielobjek vir die rol, die lys van ouer-rolle en 'n lys van die kind-rolle. Tabel 6.5 toon 'n voorbeeld van so 'n tabel.

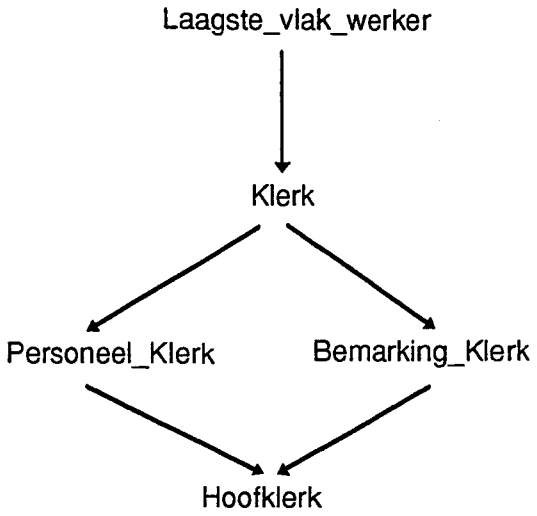
ROLLE			
Rolnaam	Rolprofielobjek	Ouers	Kinders
Klerk	Klerk_RPO	Laagste_vlak _Werker	Personeel Klerk, Bemarking_Klerk
Personeel_Klerk	Pers_Klerk_RPO	Klerk	Hoof_Klerk
Bemarking_ Klerk	Bem_Klerk_RPO	Klerk	Hoof_Klerk
HoofKlerk	Hoof_Klerk_RPO	Personeel_Klerk Bemarking_Klerk	geen

Tabel 6.5
Voorbeeld van 'n ROLLE tabel.

4. Met elke rol wat opgestel word, word besluit watter nuwe transaksies 'n rol moet hê en watter ander rolle se transaksies geërf moet word. Vir elke transaksie wat by 'n nuwe rol gevoeg word, moet besluit word wat is die toelaatbare omstandighede vir die transaksie, byvoorbeeld maksimum bedrag. Die verskillende tipes omstandighede (bedrag, tyd, ens.) hang af van die spesifieke stelsel. Hierdie inligting word in die attribute van die rolprofielobjek vir die rol gestoor.

'n Unieke rol word nou vir elke rol in die besigheid opgestel en die stappe soos in die vorige paragraaf beskryf is word gedoen om die ROLLE tabel uit te brei en die nodige rolprofielobjekte te skep.

Let op dat die roltralie intensief gebruik word in hierdie stap. By elke keuse van 'n nuwe rol word die bestaande roltralie beskou om te besluit watter rolle om te gebruik as ouers of kinders van die nuwe rol. Elke keer as 'n nuwe rol geskep is, word die rol in die roltralie gevoeg om die roltralie op datum te hou. Die roltralie vir die rolle wat in tabel 6.5 voorkom, word getoon in figuur 6.20.



Figuur 6.20.
Die roltralie vir die rolle in tabel 6.5.

6.11.1.3 Toekening van rolle aan gebruikers

Nou bestaan daar vir elke besigheidsrol 'n rol en ooreenkomstige rolprofielobjek in die ESM. In hierdie stap besluit die sekerheidsbestuurder watter rolle om aan watter gebruikers toe te ken. Soos reeds voorheen gesê, onderhou die ESM 'n tabel GEBRUIKER_ROLLE. Hierdie tabel stoor vir elke gebruiker van die stelsel 'n lys van gemagtigde rolle waaruit hy of sy kan kies by aantekening by die stelsel.

Let weer daarop dat die sekerheidsbestuurder by hierdie stap gebruik maak van die roltralie om te kan besluit watter rolle om aan 'n gebruiker toe te ken. Deur die roltralie te beskou is dit maklik om te sien hoe rolle verwant is en watter vlak van rolle in die roltralie aan 'n gebruiker toegeken kan word.

6.11.1.4 Aantekening van gebruikers (registrasie by die ESM)

Wanneer gebruikers die transaksieverwerker wil gebruik, moet hulle aanteken deur 'n naam en wagwoord te verskaf. Nadat die stelsel die gebruiker geverifieer het, word die ESM geroep om die nuwe gebruiker by die ESM te registreer. Die ESM aanvaar die gebruiker is geverifieer as 'n geldige gebruiker en kyk in sy GEBRUIKER_ROLLE tabel watter rolle aan hierdie gebruiker gemagtig is. Hierdie lys rolle word aan die gebruiker getoon waaruit hy een kies. Die ESM stoor hierdie keuse in 'n ROLKEUSE tabel vir latere gebruik. Tabel 6.6 toon 'n voorbeeld van so 'n tabel.

ROLKEUSE	
Gebruikernaam	Huidige Rol
Sarah	Teller
Peter	*geen*
Mary	*geen*
Joe	Bankbestuurder

Tabel 6.6
Voorbeeld van 'n ROLKEUSE tabel.

6.11.1.5 Magtiging van transaksies

Op hierdie punt kan die gebruiker nou 'n transaksie versoek, die transaksieverwerker roep dadelik die ESM en stuur die gebruiker se identifikasie, transaksienaam en die lys omstandighede waaronder die transaksie versoek is aan die ESM. Die ESM kyk in sy ROLKEUSE tabel aan watter rol die gebruiker gekoppel is. Indien die gebruiker volgens die ROLKEUSE tabel aan geen rol gekoppel is nie, word die transaksie dadelik geweier. Indien alles sover slaag lees die ESM die identifikasie van die rol se rolprofielobjek in die ROLLE tabel. Die MagtigTransaksie lidfunksie van die RPO word geroep en die transaksienaam en omstandighede word gestuur as parameters. Die RPO doen die magtiging soos in hierdie en vorige hoofstukke beskryf is en stuur 'n Magtig/Weier boodskap terug. Die transaksie word slegs toegelaat as die RPO 'n Magtig resultaat lewer.

6.11.1.6 Aftekening van gebruikers

Indien 'n gebruiker afteken van die stelsel, stel die transaksieverwerker die ESM in kennis wat op sy beurt die ROLKEUSE tabel opdateer. Die ROLKEUSE tabel toon dat dat die gebruiker aan geen rol gekoppel is op die oomblik nie.

6.11.1.7 Instandhouding van rolle

Die sekerheidsbestuurder mag besluit om van tyd tot tyd rolle te wysig of nuwe rolle by te voeg. Die toepaslike tabelle en rolprofielobjekte word dan aangepas soos in hierdie en vorige hoofstukke bespreek is.

Die roltralie kan gebruik word om rolanalises van tyd tot tyd te doen. Soos voorheen beskryf is, kan uit die roltralie bepaal word watter rol oorbodig is, byvoorbeeld twee identiese rolle, en met ander rolle vervang kan word. Baie ander soortgelyke rolanalises kan met behulp van die roltralie gedoen word.

6.12 Slot

In hierdie hoofstuk het ons baie van die terme en begrippe van die vorige hoofstukke gebruik om die model vir objek-georiënteerde, rolgebaseerde inligtingsekerheid in 'n transaksieverwerking omgewing (ORITO) te formuleer en te beskryf.

Die hoofstuk het die vier hoofkomponente van ORITO, naamlik magtiging van die uitvoer van transaksies, rolgebaseerde inligtingsekerheid, objek-georiënteerde rolprofiele en roltralies om die verwantskap tussen rolle voor te stel, beskryf en gewys hoe hulle inmekaar pas.

In die hoofstuk is aandag gegee aan twee moontlike ander voorstellings van rolverwantskappe en daar is ook gewys hoe die roltralie gebruik word om die bestuur van inligtingsekerheid te vergemaklik.

Die hoofstuk is afgesluit met 'n beskrywing van die stappe betrokke by die implementering van ORITO om te wys hoe die komponente van ORITO gebruik word.

ORITO vergemaklik die bestuur van inligtingsekerheid in transaksieverwerkers. Dit is egter moontlik om die model uit te brei en te benut in verskeie ander omgewings soos in verspreide en kliënt/bediener omgewings. In die volgende hoofstuk gee ons 'n kort oorsig van verspreide en kliënt/bediener omgewings en wys dan in latere hoofstukke hoe ORITO uitgebrei en benut kan word in sulke omgewings.

7. Verspreide en kliënt/bediener omgewings

7.1 Inleiding

In die vorige hoofstuk het ons 'n model vir inligtingsekerheid geformuleer (ORITO). Hierdie model vergemaklik die bestuur van inligtingsekerheid in 'n transaksieverwerking omgewing. Daar is gewys hoe die model geïmplementeer kan word in 'n stelsel waar 'n enkele rekenaar die model implementeer en sodoende die magtiging van transaksies hanteer. Die model het egter die potensiaal om in ander omgewings gebruik te kan word. Een moontlike omgewing wat in latere hoofstukke bespreek gaan word is kliënt/bediener omgewings. Ons gaan later ook sien hoe die model funksioneer in verspreide omgewings waar die magtiging van transaksies versprei gaan word tussen rekenaars.

Om bogenoemde te kan doen is dit nodig 'n kort oorsig te gee van kliënt/bediener en verspreide omgewings. In hierdie hoofstuk doen ons laasgenoemde en ons kyk ook na sekerheid in sulke omgewings en bespreek een of meer bestaande metodes vir die handhawing van inligtingsekerheid in sulke omgewings.

7.2 Verspreide omgewings

In die afgelope dekade was daar 'n vinnige groei in data kommunikasie netwerke [17]. Netwerke laat gebruikers van rekenaars toe om inligting op 'n elektroniese wyse uit te ruil. Dit maak dit moontlik om rekenaarhulpbronne te deel en ons sien al meer en meer hoe hierdie geleentheid benut word. Dit term "verspreide verwerking" word gebruik om die potensiaal van 'n netwerk van rekenaarstelsels te beskryf [16].

Daar is 'n neiging op die oomblik in die publieke en privaat sektore van die mark na meer afgeplatte organisasie strukture waarvolgens groepe en individue meer outonomiteit en meer finansiële magte gegee word [8]. Hierdie benadering maak die besigheid meer gevoelig vir die behoeftes van die kliënt en verhoog produktiwiteit deurdat individue meer aanspreeklik gehou kan word vir hulle aksies in die besigheid. In so 'n omgewing is samewerking tussen individue en groepe in die organsasie steeds nodig en word bewerkstellig deur inligtingstegnologie.

Verspreide stelsels laat gebruikers toe om toegang te verkry na inligting en selfs die uitvoer van programme te bewerkstellig op stelsels waaraan die gebruiker verbind is deur middel van 'n netwerkverbinding. Gebruikers is tipies verbind met sulke afgeleë rekenaarstelsels deur een of ander rekenaarnetwerk. Let egter op dat daar nie 'n beperking is op die fisiese ligging van die afgeleë rekenaar nie. Dit kan reg langs die gebruiker se rekenaar wees of selfs in 'n ander land.

Volgens [10] is die term verspreide verwerking een van die mees misbruikte terme in rekenaarwetenskap die afgelope paar dekades en hang dit af van wat werklik versprei word oor 'n netwerk. Een moontlikheid is om die *data* te versprei tussen rekenaars, byvoorbeeld 'n databasis program mag sy data dalk stoor op verskeie rekenaars regoor

die wêreld. Dit is ook moontlik om die *verwerking* van data te versprei tussen rekenaars soos waar een rekenaar die wiskundige bewerkings doen en 'n ander rekenaar hanteer die afvoer op skerm. In die volgende hoofstuk gaan ons kyk hoe *inligtingsekerheid* versprei word tussen rekenaars deur van ORITO gebruik te maak. ORITO word uitgebrei na 'n model waar die taak om die uitvoer van transaksies te magtig, versprei word na 'n paar rekenaars. Let op dat die uitgebreide ORITO inligtingsekerheid verspreid hanteer en sodoende die voordele van verspreide verwerking benut.

Daar is baie voordele aan verspreide verwerking. Verspreiding van data stem ooreen met die strukture van organisasies waar die organisasie se besigheidsfunksies en data fisies versprei word [10]. Verspreide verwerking maak dikwels ekonomies meer sin deurdat rekenaarstelsels meer doeltreffend benut word [10]. In die volgende hoofstukke gaan ons sien hoe ORITO 'n model vir inligtingsekerheid verskaf wat makliker is om te bestuur, meer doeltreffend en betroubaar is deur van verspreide verwerking gebruik te maak en die model te implementeer in 'n verspreide kliënt/bediener omgewing.

7.3 Sekerheid in verspreide omgewings

Die voordele van verspreide verwerking is al baie gepubliseer en daar word gereeld nuwe voordele uitgewys [9, 10] . Daar is egter 'n prys om te betaal vir die voordeel om inligting te versprei en te verwerk op verskeie rekenaars sonder direkte beheer oor elkeen van hierdie rekenaars. Die prys is natuurlik dat jy nie direkte beheer het oor die bestuur van jou inligting en jou inligting mag blootgestel wees aan ongemagtigde toegang, verandering of uitwissing [16]. Verspreide stelsels vereis dus meer aandag aan sekerheid om die stelsel te beskerm.

Inligtingsekerheid in verspreide omgewings is al goed bestudeer [8, 17, 18] en daar word gereeld nuwe metodes en sagteware ontwikkel om verspreide omgewings te beskerm. *Distributed Computing Environment* (DCE) wat deur *Open System Foundation* (OSF) verskaf word is sagteware wat dit moontlik maak om toegang te kry na hulpbronne op enige rekenaar onder enige bedryfstelsel [9, 18]. Hierdie toegang word gemagtig en die nodige sekerheid word deur DCE verskaf [9]. OSF se DCE verskaf inligtingsekerheid wat gebaseer is op die Kerberos magtigingstelsel en 'n paar ander sekerheidskomponente word bygevoeg [18]. Die Kerberos magtigingstelsel word later in hierdie hoofstuk volledig bespreek.

ORITO word in die volgende hoofstuk uitgebrei om verspreide verwerking te benut. Let daarop dat ORITO nie 'n nuwe model is vir inligtingsekerheid in verspreide omgewings nie, ORITO moet eerder gesien word as 'n model wat inligtingsekerheid in 'n transaksieverwerking omgewing verskaf deur verspreide verwerking te benut. Die uitgebreide ORITO wat in hoofstuk 8 en 9 bespreek word benut verspreide verwerking en is 'n nuwe model vir magtiging in kliënt/bediener omgewings, soortgelyk aan Kerberos.

7.4 Kliënt/bediener omgewings

Kliënt/bediener stelsels neem die idee van verspreide stelsels een stap verder. In effek word die tradisionele hoofraamstelsel in twee gedeel [8]. Die gebruikerkoppelvlak en

toepassingverwerking word verskaf deur die gebruiker se persoonlike rekenaar of werkstasie of *kliënt*, en die databasis bestuur word verskaf deur 'n aparte rekenaar wat geoptimeer is juis vir die spesifieke doel en hierdie rekenaar word die *bediener* genoem.

In 'n rekenaarstelsel kan daar verskeie soorte bedieners wees, soos byvoorbeeld netwerkbedieners en drukkerbedieners. Deur die toepassing of werkstaak te verdeel tussen die gebruiker se rekenaar en 'n bediener rekenaar kan elke rekenaar die taak doen waarvoor dit die beste geskik is.

'n Belangrike deel van enige kliënt/bediener stelsel is boodskappe [9]. Die kliënt en bediener word verbind deur boodskappe wat deur die netwerk waarmee hulle fisies verbind is gestuur word om te sorg dat hulle logies verbind bly.

Kliënt/bediener toepassings is buigsaam: die stelsel kan maklik uitgebrei word om meer gebruikers te akkommodeer. Tesame met verspreide stelsels tegnologie, kan gebruikers toegang verkry na verskeie databassise op verkillende rekenaarstelsels, almal deur een kragtige grafiese kopplevlak wat nie beskikbaar sou wees op dom terminale in tradisionele hoofraam stelsels nie.

Verspreide en kliënt/bediener stelsels het duidelik baie voordele, maar hierdie tipe stelsels bring weer nuwe sekerheidsvraagstukke te wee [8]. In die volgende hoofstuk kyk ons na die implementering van ORITO in verspreide en kliënt/bediener stelsels as 'n manier om inligtingsekerheid verspreid te doen en inligtingsekerheid in kliënt/bediener omgewings aan te spreek.

7.5 Sekerheid in kliënt/bediener omgewings

Inligtingsekerheid is 'n belangrike aspek van kliënt/bediener omgewings. Gebruikers by kliënt-rekenaars en programme wat uitvoer op kliënt-rekenaars moet gemagtig word om die dienste wat bediener-rekenaars lewer te gebruik.

Om inligtingsekerheid in 'n kliënt/bediener omgewing te bewerkstellig is dit nodig om 'n protokol ('n stel reëls) te gebruik waarvolgens kliënte en bedieners kommunikeer. *Kerberos* is 'n protokol waarvolgens kliënte en bedieners in 'n kliënt/bediener omgewing kan kommunikeer om te sorg dat kliënte gemagtig is om die dienste van bedieners te bekom. Die model wat in die volgende hoofstuk geformuleer word ('n uitbreiding van ORITO) kan ook gesien word as 'n protokol vir inligtingsekerheid in 'n kliënt/bediener omgewing. Die model gebruik Kerberos beginsels. Die *Kerberos* protokol word vervolgens bespreek.

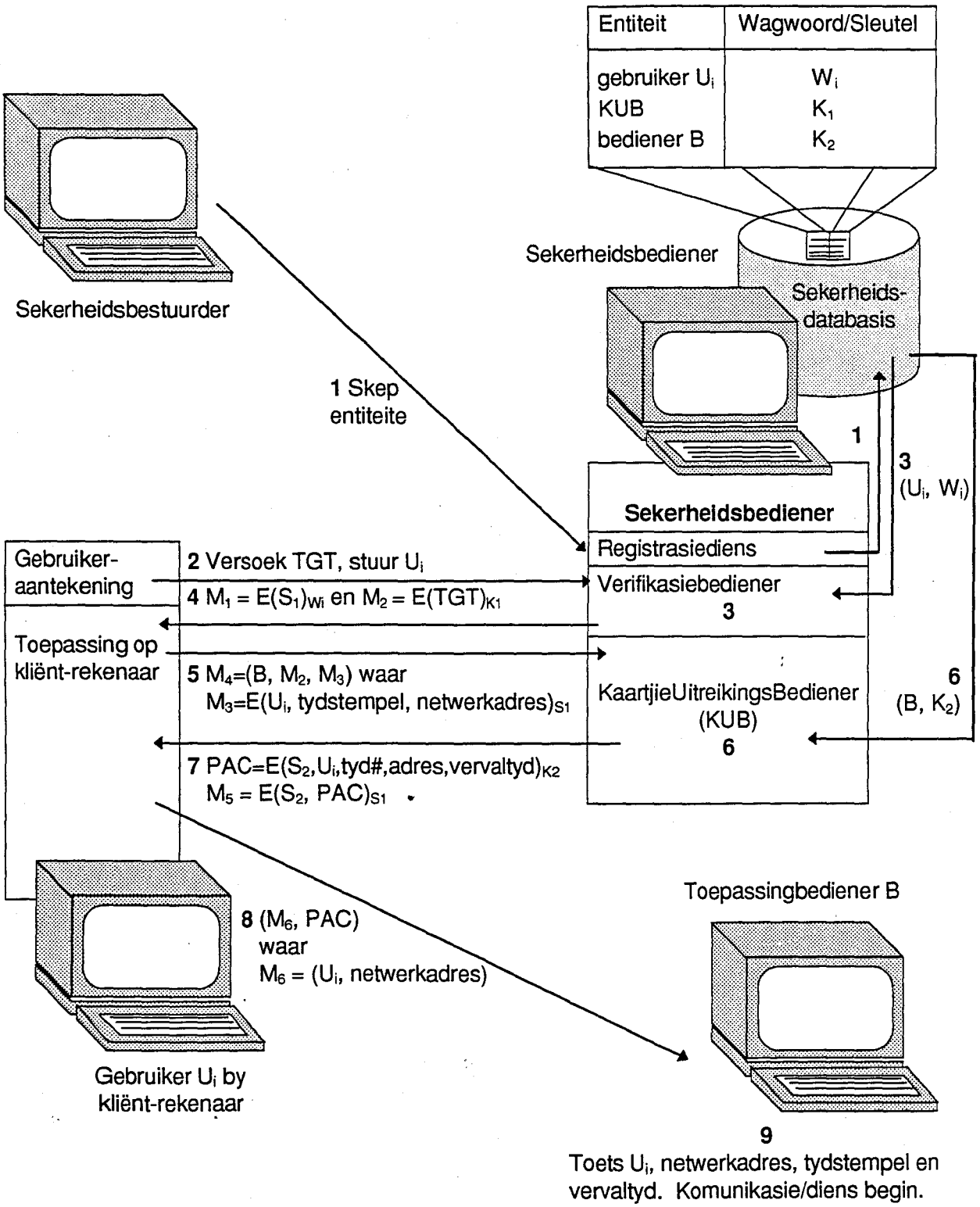
7.5.1 Die Kerberos-magtigingstelsel

In hierdie gedeelte word die Kerberos magtigingsprotokol bespreek. Lesers wat vertroud is met Kerberos kan voortlees by paragraaf 7.6.

Kerberos is 'n protokol vir inligtingsekerheid in 'n kliënt/bediener omgewing en is ontwerp deur MIT [9]. Kerberos is vernoem na die metologiese driekoppige monster, wat volgens metologie, die hekke van Hades bewaar het. Die Kerberos "monster" wat deur MIT ontwerp is bestaan uit drie "koppe": Die Verifikasiebediener (VB),

Sekerheidsdatabasis (SDB) en die Kaartjieuitreikingsbediener (KUB). Al drie komponente word op een bediener geïmplementeer en staan gesamentlik bekend as die Sekerheidsbediener (SB).

Die volgende stappe toon hoe Kerberos funksioneer om inligtingsekerheid te verskaf in 'n kliënt/bediener omgewing. Figuur 7.1 toon die stappe diagrammaties.



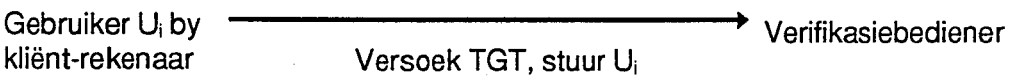
Figuur 7.1
Stappe tydens magtiging deur Kerberos.

Beskrywing van stappe:

1. Eerstens moet 'n sekerheidsbestuurder of netwerkbestuurder 'n program uitvoer wat die Sekerheidsdatabasis (SDB) op die Sekerheidsbediener (SB) skep. Die Sekerheidsdatabasis (SDB) bevat die name en wagwoorde van alle kliënt/bediener

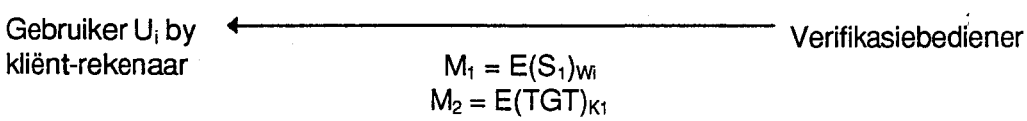
entiteite op die netwerk (kliënte, bedieners en gebruikers by kliënt-rekenaars). Wagwoorde word slegs in die die SDB gestoor.

2. Wanneer 'n gebruiker U_i by 'n kliënt-rekenaar aanteken, stuur die kliënt-rekenaar 'n kaartjie-toekenningsversoek na die Verifikasiebediener (VB) op die Sekerheidsbediener. Die gebruiker se naam (nie wagwoord nie) word na die VB gestuur. Die gebruiker se wagwoord kan dus nie vanaf die netwerkverbindings gelees (afgeluister) word nie.

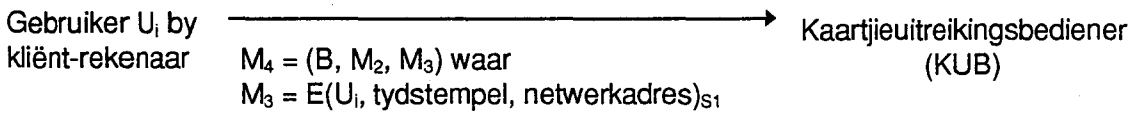


3. Die Verifikasiebediener se doel is om twee entiteite in die kliënt/bediener omgewing aan mekaar te verifieer (te sorg dat albei seker is hulle kommunikeer met wie dit dink hulle kommunikeer). Wanneer die VB 'n kaartjie-toekenningsversoek ontvang, lees dit gebruiker U_i se wagwoord (W_i) uit die Sekerheidsdatabasis. Indien die gebruiker in die SDB gevind word genereer die VB 'n sessiesleutel S_1 wat gebruik word vir kommunikase tussen die kliënt-rekenaar en die Kaartjieuitreikingsbediener (KUB). Die VB enkripteer hierdie sessie sleutel met die gebruiker se wagwoord en vorm so boodskap M_1. 'n Kaartjiemagtigings-kaartjie ("*Ticket Granting Ticket*" (TGT)) word gegeneer en bevat onder andere die sessiesleutel S_1, gebruiker id U_i, 'n tydstempel, die leeftyd van die kaartjie, ens. Die VB lees die private sleutel K_1 van die Kaartjieuitreikingsbediener en enkripteer die TGT en vorm so boodskap M_2.

4. Die Verifikasiebediener stuur boodskappe M_1 en M_2 wat in stap 3 gegenereer is aan die kliënt-rekenaar.

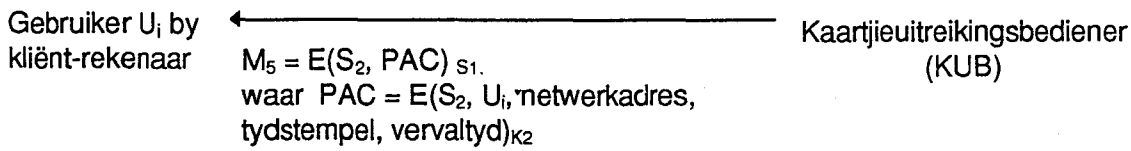


5. Die kliënt-rekenaar dekripteer boodskap M_1 en verkry die sessiesleutel S_1 ($S_1 = D(M_1)_{WI}$). Gebruiker U_i kan egter nie M_2 dekripteer om TGT te lees nie. Die kliënt-rekenaar vorm boodskap $M_3 = E(U_i, \text{tydstempel, netwerkadres})_{S1}$. Hierna word boodskap M_4 gevorm en gestuur na die Kaartjieuitreikingsbediener (KUB). $M_4 = (B, M_2, M_3)$ waar B die naam van die bediener is waar 'n diens gebruik wil word. Onthou dat M_2 die geënkripteerde TGT is.

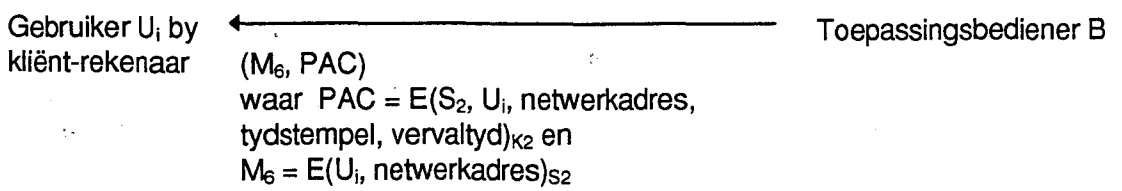


6. Die Kaartjieuitreikingsbediener (KUB) dekripteer boodskap M_2 met sy private sleutel K_1 en lees so die TGT. Onthou dat TGT = (S_1, U_i, tydstempel, leeftyd, ens.) Die KUB gebruik die sessiesleutel S_1 en dekripteer M_3 om (U_i, tydstempel, netwerkadres) te lees. Die KUB toets of die inligting in M_3 ooreenstem met die inligting in die TGT. Onder andere word getoets of die gebruiker se netwerkadres ooreenstem met die adres vanwaar die versoek gestuur is en of die tydstempel ooreenstem met die huidige tyd. Indien alles aanvaarbaar is mag die versoek voortgaan.

7. Die KUB genereer 'n nuwe sessiesleutel S_2 vir gebruik tussen die kliënt-rekenaar en bediener B. Die KUB skep 'n nuwe kaartjie, 'n gemagtigde voorreg sertifikaat of "*Privilege Authorisation Certificate*" (PAC). $PAC = E(S_2, U_i, \text{netwerkadres, tydstempel, vervaltyd})_{K2}$. K_2 is die enkripsie sleutel van bediener B wat slegs bekend is aan B en KUB. Boodskap M_5 word gevorm en bevat die nuwe sessielsutel S_2 en die PAC. M_5 word geënkripteer onder sessiesleutel S_1 en gestuur aan die kliënt-rekenaar..



8. Die kliënt-rekenaar dekripteer boodskap M_5 met sessiesleutel S_1 en verkry sessiesleutel S_2 en die geënkripteerde PAC. Gebruiker U_i is nou gereed om met bediener B te kommunikeer. Die kliënt-rekenaar skep boodskap $M_6 = E(U_i, \text{netwerkadres})_{S2}$. M_6 en die PAC word aan bediener B gestuur.



9. Bediener B dekripteer die PAC lees die sessiesleutel S_2 en gebruik die sleutel om boodskap M_6 te dekripteer. Die toepassingsbediener B toets nou onder andere of die inligting in M_6 ooreenstem met van die inligting in die PAC en of die PAC nog geldig is (nie verval het nie). Indien al die toetse slaag kan gebruiker U_i en bediener

B kommunikeer met sleutel S_2. Die gebruiker kan nou dienste van bediener B aanvra.

Bostaande stappe verduidelik die stappe hoe Kerberos sekerheidsmagtiging doen in 'n kliënt/bediener omgewing. In die volgende hoofstuk formuleer ons 'n model soos Kerberos wat magtiging doen in kliënt/bediener omgewings. Let op dat ORITO nie Kerberos gebruik nie maar is 'n model soos Kerberos om inligtingsekerheid te verskaf in kliënt/bediener omgewings waar toegang na stelselhulpbronne verkry word deur middel van transaksies.

7.6 Slot

In hierdie hoofstuk is 'n kort oorsig gegee van die terme *verspreide verwerking* en *kliënt/bediener* omgewings. Daar is ook gewys op die maniere om inligting te beskerm in sulke omgewings. Onder andere is Kerberos bespreek as 'n protokol om inligtingsekerheid te verskaf in sulke omgewings. In die volgende hoofstuk gaan ons sien dat ORITO uitgebrei kan word om inligtingsekerheid in 'n transaksieverwerker en kliënt/bediener omgewing te verskaf en die inligtingsekerheid verspreid te doen. ORITO is interessant in die sin dat dit inligtingsekerheid in kliënt/bediener omgewings verskaf en self gebruik maak van verspreide en kliënt/bediener tegnologie.

8. ORITO in 'n verspreide en kliënt/bediener omgewing

8.1 Inleiding

In hierdie hoofstuk sien ons hoe ORITO kliënt/bediener en verspreide omgewings kan benut. ORITO word uitgebrei as 'n model wat inligtingsekerheid verskaf deur van kliënt/bediener beginsels gebruik te maak. Hierdie uitbreiding is nodig sodat ons later in die hoofstuk kan toon hoe ORITO van verspreide en kliënt/bediener omgewings gebruik maak en 'n model vir inligtingsekerheid verskaf waar transaksies gemagtig word in 'n potensieel verspreide omgewing. Daar word getoon hoe die roltralie wat in vorige hoofstukke beskryf is opgedeel word en die ooreenstemmende rolprofielobjekte versprei word tussen bediener rekenaars.

Bogenoemde word gedoen om 'n stelsel te skep waar die bestuur van inligtingsekerheid makliker is en die stelsels meer doeltreffende inligtingsekerheid verskaf. Die hoofstuk bespreek hierdie en ander voordele.

8.2 Implementering van ORITO in 'n kliënt/bediener omgewing

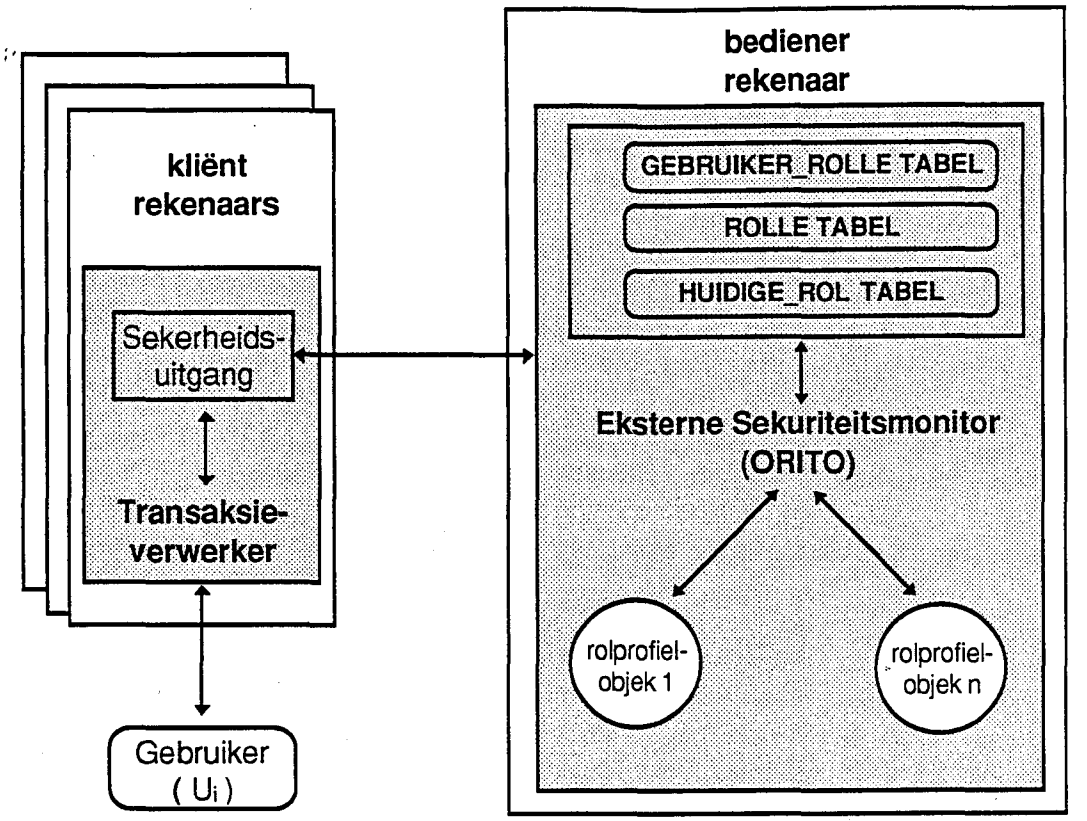
In die vorige hoofstuk het ons beskryf hoe die komponente van ORITO inmekaar skakel om objek-georiënteerde rolgebaseerde inligtingsekerheid te verskaf. Ons gaan nou kyk hoe ORITO in 'n kliënt/bediener omgewing geïmplementeer kan word.

Let op dat ORITO funksioneer in 'n omgewing waar alle toegang na stelselhulpbronne geskied deur middel van transaksies. Die enigste manier vir 'n gebruiker om dienste van ander bediener rekenaars te verkry, is deur 'n transaksie uit te voer. Elke transaksie kan slegs uitvoer indien dit gemagtig word deur 'n program wat ORITO implementeer en hierdie program voer uit op 'n bediener wat die inligtingsekerheid vir die stelsel hanteer.

Die bedieners hanteer dus die magtiging van die uitvoer van transaksies en kliënt rekenaars benodig hierdie diens. Die uitbreiding van ORITO in hierdie gedeelte stel 'n model daar wat inligtingsekerheid op 'n kliënt/bediener metode lewer.

8.2.1 Komponente van ORITO in 'n kliënt/bediener omgewing

Figuur 8.1 toon 'n eenvoudige opstelling waar ORITO in so 'n omgewing funksioneer. Die figuur toon die interaksie tussen die komponente van die stelsel en word vervolgens beskryf.



Figuur 8.1
ORITO in 'n kliënt/bediener omgewing.

Op die kliënt rekenaar loop 'n transaksieverwerker en op die bediener rekenaar loop 'n program wat die uitvoering van transaksies op hierdie en ander kliënt rekenaars magtig.

Dit is belangrik om te onderskei tussen komponente van die transaksieverwerker en komponente van die program wat die magtiging van die uitvoer van transaksies hanteer. Op die kliënt rekenaar loop 'n transaksieverwerker. Gebruikers is by die kliënt rekenaar aangeteken by die transaksieverwerker en versoek transaksies. Voordat die transaksieverwerker 'n transaksie uitvoer moet dit eers bepaal of die gebruiker gemagtig is om die transaksie uit te voer. Die transaksieverwerker doen nie self hierdie magtiging nie maar gebruik 'n eksterne program hiervoor. In figuur 8.1 word laasgenoemde aangedui as 'n *sekerheidsuitgang*. Hiermee word bedoel dat die transaksieverwerker al die nodige inligting (transaksienaam, gebruikernaam, ens.) na 'n aparte program/proses stuur op die kliënt rekenaar en verwag van hierdie program/proses om die magtiging te doen. Hierdie program op sy beurt hanteer weer nie self die magtiging nie, maar roep 'n program op 'n bediener. Die bediener loop dus nie self 'n transaksieverwerker nie, net 'n program wat sekerheidsmagtiging vir transaksieverwerkers doen.

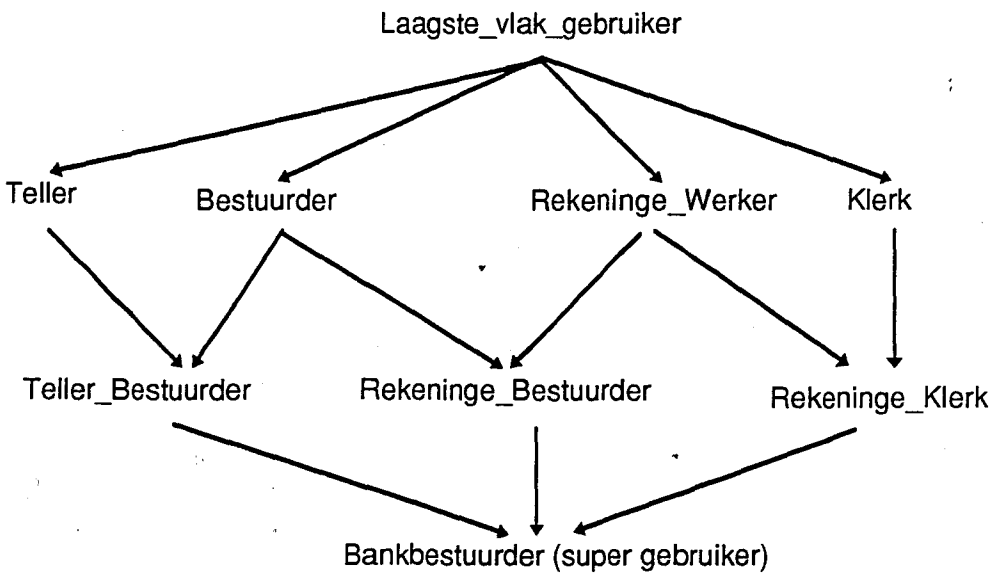
Bostaande konsep van eksterne programme wat die magtiging van die uitvoer van transaksies vir transaksieverwerkers hanteer word reeds in kommersiële transaksieverwerkers gebied. CICS/6000 bied hierdie opsies by wyse van 'n Eksterne Sekerheidsmonitor (ESM) [11]. CICS for OS/2 bied die opsie by wyse van *User*

Exits, waarvolgens prosedures geskryf kan word wat sekerheid hanteer vir die transaksieverwerker [9].

Ons aanvaar vir nou dat die bediener 'n program loop wat reageer op boodskappe vanaf transaskieverwerkers wat transaksies gemagtig wil hê. Ons noem die program weer die eksterne sekerheidsmonitor (ESM). Die ESM implementeer dus effektief ORITO. Die eksterne sekerheidsmonitor (ESM) waarna ons hier verwys is nie noodwendig dieselfde ESM as waarna verwys word in die CICS/6000 handleidings nie. Alhoewel dit in baie opsigte soortgelyk is, is die grootse verskil dat die ESM in hierdie verhandeling op 'n aparte bediener rekenaar uitvoer en is nie deel van die transaksieverwerker soos in CICS/6000 nie.

Neem ook kennis dat daar toepassings bestaan waar een transaksieverwerker die sekerheidsdienste van 'n ander transaksieverweker versoek [9]. ORITO is nie 'n model vir laasgenoemde situasie nie - ORITO word geïmplementeer op 'n aparte rekenaar as 'n aparte program.

Die ESM bevat 'n paar tabelle. Voorbeelde van hierdie tabelle word in tabel 8.1, 8.2 en 8.3 getoon en is gebaseer op die roltralie in figuur 8.2. Ons bespreek later hoe hierdie tabelle op die bediener rekenaar as deel van die ESM gestoor word. Al die tabelle wat in die volgende bespreking as voorbeelde gegee word, word later weer gebruik om die werking van die model te beskryf.



Figuur 8.2
Die roltralie wat gebruik word as voorbeeld om ORITO in 'n kliënt/bediener omgewing te verduidelik.

- GEBRUIKER_ROLLE. Hierdie tabel bevat vir elke gebruiker in die stelsel 'n lys van rolle waaruit die gebruiker kan kies wanneer hy of sy by die stelsel aanteken om transaksies uit te voer. Tabel 8.1 toon 'n voorbeeld van so 'n tabel.

GEBRUIKER_ROLLE	
Gebruiker Identifikasie	Gemagtigde Rolle
John	Klerk Rekeninge_Klerk
Mary	Klerk Rekeninge_Klerk
Peter	Rekeninge_Bestuurder Rekeninge_Klerk
.	
.	
.	
Sarah	Bankbestuurder

Tabel 8.1
Voorbeeld van GEBRUIKER_ROLLE tabel.

Die GEBRUIKER_ROLLE tabelle word opgestel deur 'n persoon in die organisasie wat verantwoordelik is vir inligtingsekerheid in die organisasie. So 'n persoon moet die nodige magte hê om rolle aan gebruikers toe te ken. Indien die rolle reeds gedefinieer en geskep is (deur 'n persoon met die nodige kundigheid), het hierdie persoon minimale kundigheid op die gebied van inligtingsekerheid nodig.

- ROLLE. 'n Tabel soos in hoofstuk 6 beskryf is met 'n inskrywing vir elke rol en bevat die naam van 'n rol, die identifikasie van 'n rolprofielobjek vir die rol, 'n lys van ouer-rolle en 'n lys van kinder-rolle. Tabel 8.2 toon 'n voorbeeld.

ROLLE			
Rolnaam	Rolprofielobjek	Ouers	Kinders
Klerk	Klerk_RPO	Laagste_Vlak_Werker	Rekeninge_Klerk
Rekeninge_Bestuurder	RekBest_RPO	Bestuurder	Rekeninge_Werker Bankbestuurder
Rekeninge_Klerk	RekKlerk_RPO	Klerk Rekeninge_Werker	Bankbestuurder
.			
.			
.			
Bankbestuurder	BankBest_RPO	Teller_Bestuurder Rekeninge_Bestuurder Rekeninge_Klerk	-

Tabel 8.2
Voorbeeld van ROLLE tabel.

Die rolprofielobjek kolom in die ROLLE tabel bevat die identifikasie van 'n rolprofielobjek vir die rol. Klerk_RPO is dus die unieke identifikasie van die objek Klerk_RPO in die stelsel. Die objek is 'n instansie van die rolprofielklas soos voorheen beskryf is. Deur die objek se identifikasie te stoor kan ons later by die objek se lidfunksies uitkom, 'n voorbeeld in C++ notasie is Klerk_RPO.MagtigTransaksie(*id, t, a*).

- HUIDIGE_ROL. Bevat vir elke gebruiker die rol waaraan hy of sy tans gekoppel is (of geen rol). Wanneer 'n gebruiker aanteken by die stelsel moet sy 'n rol kies wat sy wil verteenwoordig vir die sessie werk. Dit is dus moonltik dat sekere gebruikers gemagtig is om uit meer as een rol te kies by aantekenning by die stelsel. Tabel 8.3 toon 'n voorbeeld van so 'n tabel.

HUIDIGE_ROL	
Gebruiker Identifikasie	Huidige Rol
John	Klerk
Mary	Rekeninge_Klerk
Peter	*(Nie aangeteken nie)*
.	
.	
.	
Sarah	Bankbestuurder

Tabel 8.3
Voorbeeld van HUIDIGE_ROL tabel.

Die ESM bevat ook 'n rolprofielobjek vir elke rol in die ROLLE tabel. Elke rolprofielobjek (RPO) is 'n instansie van die klas rolprofiel soos in die vorige hoofstuk beskryf is. Die ESM roep lidfunksies van hierdie rolprofielobjekte om magtiging van transaksies te doen. Dit word in die volgende afdeling beskryf.

8.2.2 Stappe by die magtiging van 'n transaksie deur ORITO in 'n kliënt/bediener omgewing

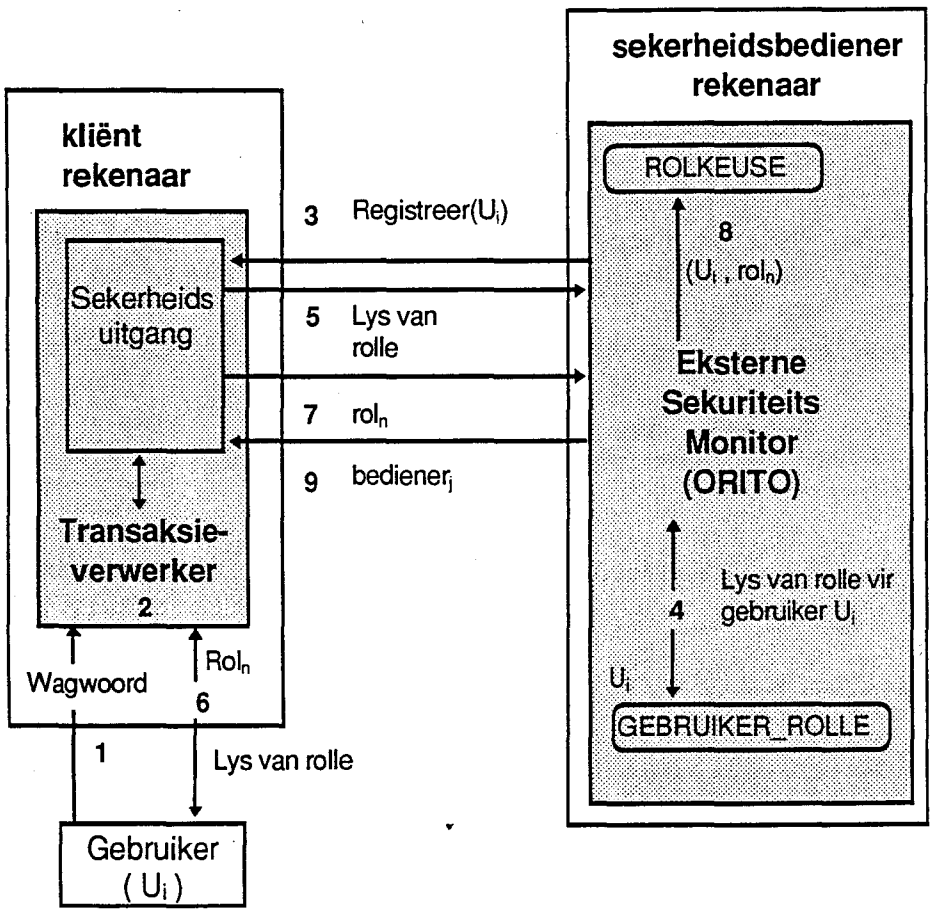
Om die werking van die ESM in hierdie omgewing te beskryf beskou ons die stappe waardeur 'n gebruiker van die stelsel gaan. Elke gebruiker gaan in een sessie deur drie hoof fases. Ons noem hierdie die *Registrasie-*, *Magtiging-* en *Aftekeningfases.*

Die bediener rekenaar wag op 'n boodskap van 'n kliënt rekenaar en reageer daarop. Die bediener kan op basies drie boodskappe reageer naamlik:

- **Registreer 'n nuwe gebruiker.** Hierdie boodskap word aan die bediener gestuur sodat die bediener kan weet dat 'n gebruiker by 'n kliënt rekenaar aangeteken het en te sorg dat die gebruiker aan 'n rol gekoppel word.
- **Magtiging van 'n transaksie.** Die bediener ontvang hierdie boodskap tesame met die gebruiker se naam, die transaksienaam en transaksieomstandighede en besluit dan of die transaksie mag uitvoer of nie.
- **Verval van registrasie.** Die boodskap word gestuur wanneer 'n gebruiker afteken by 'n kliënt rekenaar sodat die bediener sy rekords kan opdateer.

8.2.2.1 Die Registrasie fase

Tydens die Registrasie fase teken die gebruiker aan by die transaksieverwerker en moet by die sekerheidsbediener registreer om transaksies te kan uitvoer. Dit behels die stappe wat in figuur 8.3 getoon word.



Figuur 8.3
Registrasie van 'n gebruiker deur ORITO in 'n kliënt/bediener omgewing.

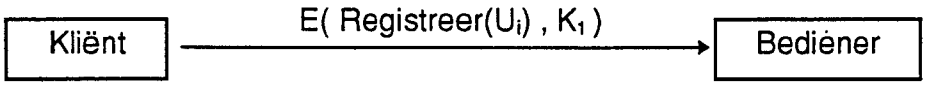
Stappe in die Registrasie fase:

1. Die gebruiker verskaf sy identifikase, U_i, en wagwoord aan die transaksieverwerker.
2. Die transaksieverwerker verifieer die gebruiker. Verifikasie van die gebruiker vind plaas deurdat die gebruiker sy naam en wagwoord aan die transaksieverwerker verskaf. Die transaksieverwerker toets of die gebruiker naam en wagwoord geldig is.

   Let op: Alle kommunikasie tussen die transaksieverwerker en die sekerheidsbediener vind plaas *via die sekerheidsuitgang* van die transaksieverwerker. Op p.106 is reeds verwys na die sekerheidsuitgang van transaksieverwerkers. In kort is 'n sekerheidsuitgang van 'n transaksieverwerker deel van die transaksieverwerker en 'n metode waarvolgens magtiging van transaksies deur eksterne programme gedoen word (nie deur die transaksieverwerker self nie). Die transaksieverwerker stuur dus al die nodige inligting (transaksienaam, gebruikernaam, ens.) aan sy sekerheidsuitgang en verwag van die sekerheidsuitgang om die transaksie te magtig. In ORITO vereis ons dat die

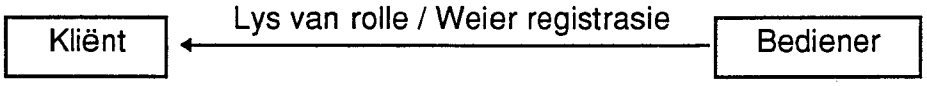
sekerheidsuitgang van die transaksieverwerker nie self die transaksie magtig nie maar op sy beurt weer die sekerheidsbediener roep om die transaksie te magtig.

3. Die transaksieverwerker stuur (via sy sekerheidsuitgang) vir die ESM op die bediener 'n Registrasie boodskap met die gebruiker se identifikasie (U_i). Alle boodskappe word geënkripteer onder 'n sleutel (bv. K_1) wat die kliënt en bediener deel.

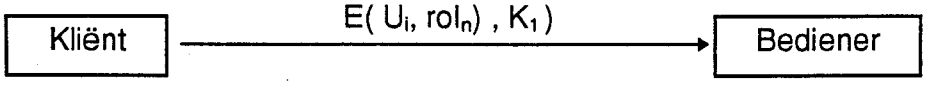


4. Die ESM ontvang die boodskap, dekripteer dit, lees in die GEBRUIKER_ROLLE tabel die lys van rolle waaraan hierdie gebruiker gekoppel mag wees.

5. Die ESM stuur die lys van gemagtigde rolle vir gebruiker U_i terug aan die kliënt.



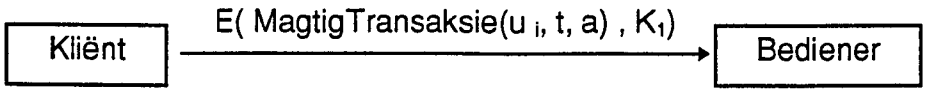
6. Die sekerheidsuitgang (via die transaksieverweker) op die kliënt vertoon aan die gebruiker die lys, die gebruiker maak 'n keuse (sê rol_n) en die keuse word aan die bediener teruggestuur (via die sekerheidsuitgang) in stap 7.



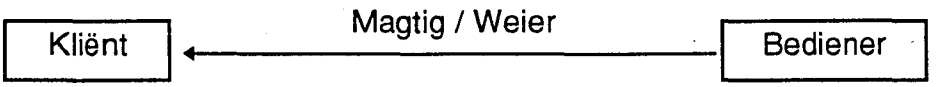
8. Die ESM op die sekerheidsbediener opdateer die ROLKEUSE tabel sodat die gebruiker se rolkeuse later gebruik kan word. Gebruiker U_i is nou geregistreer by die bediener as 'n gebruiker van tipe rol_n en kan begin om transaksies uit te voer.

8.2.2.2 Die Magtiging fase

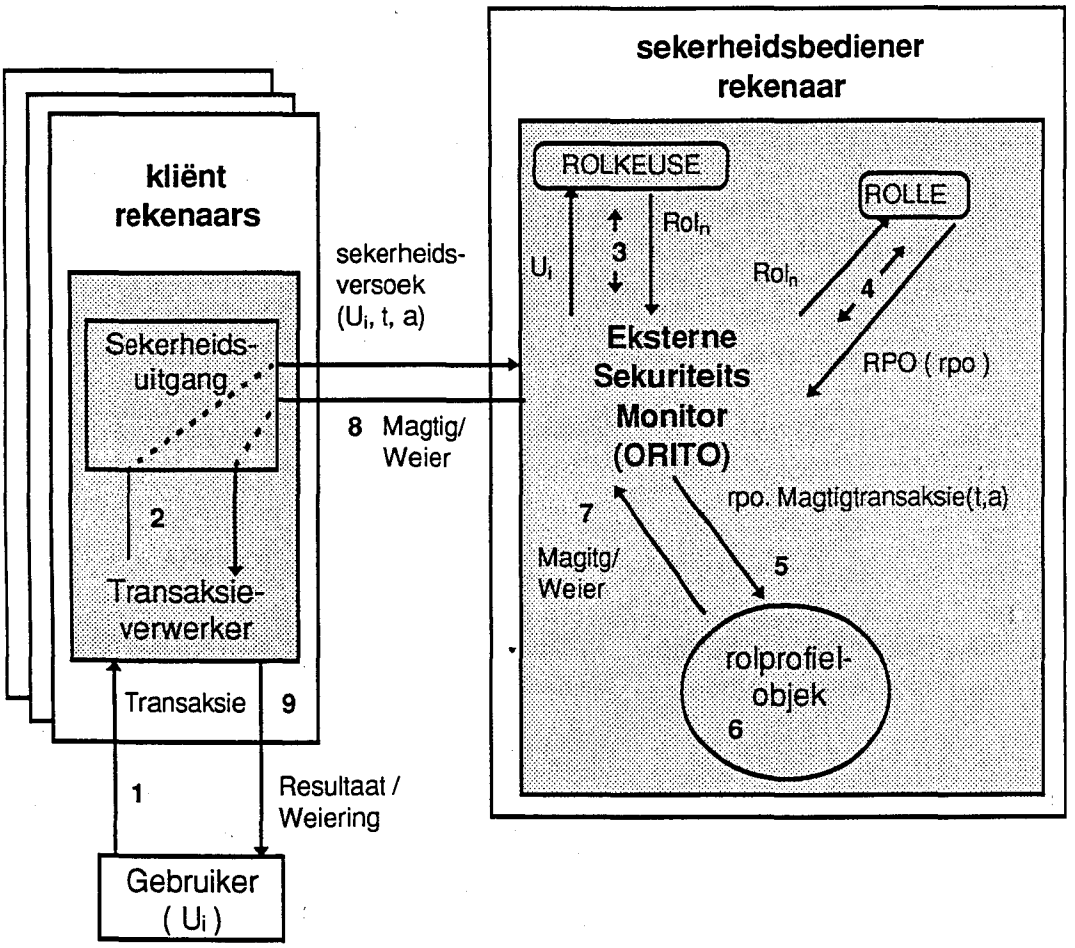
In die Magtiging fase versoek die gebruiker by die kliënt rekenaar transaksies wat deur die sekerheidsbediener gemagtig moet word. Vir elke transaksie wat versoek word, stuur die transaksieverwerker op die kliënt rekenaar die MagtigTransaksie boodskap aan die bediener tesame met die gebruiker identifikasie, transaksienaam en transaksie omstandighede.



Die bediener ontvang die boodskap en lees in die HUIDIGE_ROL tabel aan watter rol die gebruiker tans gekoppel is. Hierna word die rol waaraan die gebruiker gekoppel is se inskrywing in die ROLLE tabel opgesoek. Die MagtigTransaksie lidfunksie van die rol se rolprofielobjek word geroep. Die resultaat wat die RPO terugstuur word net so aan die kliënt rekenaar aangestuur en die transaksie word slegs uitgevoer as dit 'n magtig resultaat is.



Figuur 8.4 toon 'n diagrammatiese voorstelling van die stappe betrokke by die Magtiging fase van 'n transaksie deur 'n ESM wat ORITO implementeer in 'n kliënt/bediener omgewing waar slegs een bediener gebruik word. Die stappe word na die diagram bepreek.



Figuur 8.4
Magtiging deur ORITO in 'n kliënt/bediener omgewing met een bediener.

Stappe in die Magtiging fase:

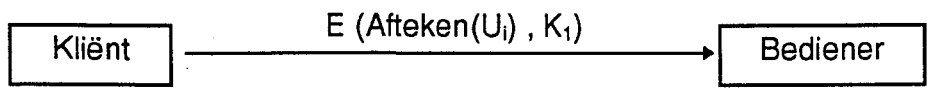
1. Die gebruiker versoek 'n transaksie (sê t) deur middel van die transaksieverwerker.
2. Die transaksieverwerker weet dat die transaksie slegs mag uitvoer indien dit deur die ESM gemagtig is en versoek sy sekerheidsuitgang om die transaksie te magtig. Die transaksieverwerker se sekerheidsuitgang vorm 'n sekerheidsversoek bestaande uit die Magtigtransaksie boodskap en die gebruiker se identifikasie (U_i in die geval) en die transaksienaam (t) asook die omstandighede waaronder die transaksie versoek is (a) en stuur hierdie versoek aan die bediener rekenaar.
3. Die bediener rekenaar ontvang die Magtigtransaksie versoek en stuur dit aan die ESM (wat ORITO implementeer) . Die ESM moet eerstens uitvind aan watter rol

gebruiker U_i tans gekoppel is en onttrek die inligting uit die ROLKEUSE tabel (gestel gebruiker U_i is gekoppel aan rol_n).

4. Die ESM onttrek nou die nodige inligting van die rol uit die ROLLE tabel
5. In stap 4 is die unieke identifikasie van rol_n se rolprofielobjek uit die ROLLE tabel gelees (gestel dit is rpo). Die ESM roep nou die MagtigTransaksie lidfunksie van die rolprofielobjek en as parameters word die transaksienaam (t) en transaksieomstandighede (a) na die rolprofielobjek gestuur (rpo. MagtigTransaksie(t, a) in C++ notasie). Let op dat vir die rolprofielobjek dit nie saak maak watter gebruiker versoek die transaksie nie.
6. Die rolprofielobjek toets of transaksie t in sy transaksielys voorkom en of die omstandighede a 'n deelversameling is van die toelaatbare omstandighede van die transaksie soos in hoofstuk 6 beskryf is. Let op dat hierdie magtiging geënkapsuleer is in die objek en die ESM het nie direkte beheer hieroor nie. Dit maak die stelsel veiliger.
7. Indien die transaksie t nie in die transaksielys voorkom nie of a is nie 'n deelversamling van die toelaatbare omstandighede van die transaksie in die transaksie lys van rpo nie, word 'n Weier boodskap teruggestuur aan die ESM, andersins word 'n Magtig boodskap terug gestuur.
8. Die ESM neem die resultaatboodskap van die rolprofielobjek en stuur dit net so aan die kliënt rekenaar.
9. Die transaksieverwerker voer slegs die transaksie uit indien die bediener 'n Magtig boodskap terug gestuur het, andersins word 'n sekerheidsweiering boodskap aan die gebruiker vertoon.

8.2.2.3 Die Aftekening fase

In die Aftekening Fase teken 'n gebruiker af by 'n kliënt rekenaar en die aftekening boodskap tesame met die gebruiker se identifikasie word aan die bediener gestuur. Die ESM op die bediener opdateer die ROLKEUSE tabel om hierdie aftekening in die toestand van sy tabelle te weerspieel.



8.3 ORITO in 'n verspreide stelsel

Let op dat in bostaande voorbeeld ORITO in 'n stelsel geïmplementeer is waar daar slegs een bediener rekenaar is. Die stelsel kan egter uitgebrei word na 'n stelsel waar daar meer as een bediener is. In so 'n stesel kan ORITO op elke bediener gedupliseer word. Kliënt rekenaars kan dus na verskillende bedieners gaan om rolgebaseerde sekerheidsmagtiging te doen. So 'n stelsel verhoog doeltreffendheid deurdat die kanse dat een bediener 'n bottelnek vorm verlaag. Die nadeel is egter dat al die rolle en hulle ooreenkomstige rolprofielobjekte op elke bediener in stand gehou moet word en dit is 'n probleem uit 'n administrasie- sowel as 'n sekerheidsoogpunt.

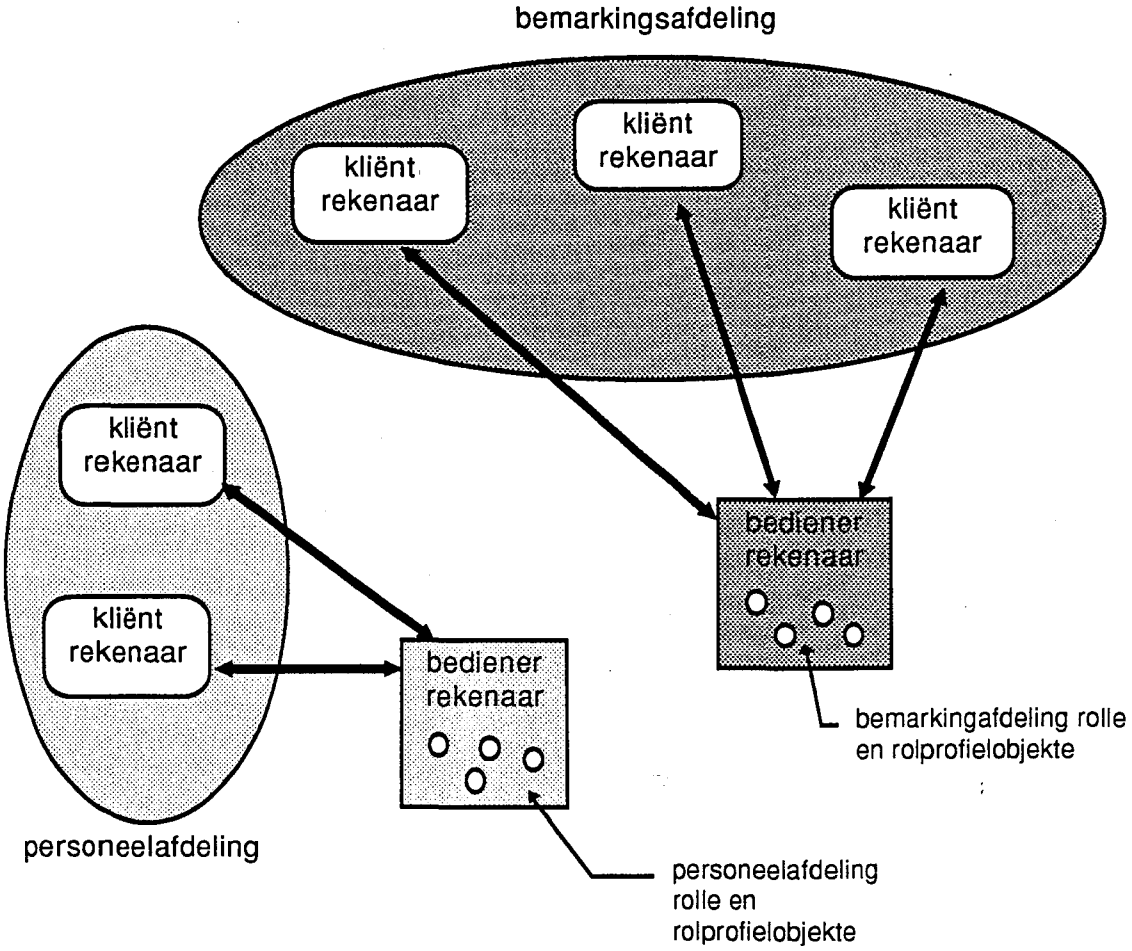
'n Meer interessante stelsel is waar ORITO *opgebreek* word tussen bedieners, m.a.w. verskillende bedieners bevat verskillende rolle en hulle rolprofielobjekte.

8.3.1 Voordele van die verspreiding van rolle en rolprofielobjekte tussen bedieners

- Die verspreiding van rolle en rolprofielobjekte tussen bedieners het die voordeel dat alhoewel elke bediener beskikbaar is vir enige kliënt, sal een bediener normaalweg 'n spesifieke groep kliënte bedien wat 'n *meer doeltreffende stelsel* is.
- *Rolle en rolprofielobjekte word egter nie gedupliseer nie.* Wysigings in rolle hoef dus net op een bediener gedoen te word.
- 'n Verdere voordeel is dat in 'n groot organisasie met baie besigheidsrolle daar 'n sekerheidsbestuurder vir elke bediener kan wees wat die rolle op die bediener in stand hou. So 'n benadering *verskaf vertikale skaleerbaarheid* [9], die vermoë om maklik die werksvermoë van die bedieners te verhoog deur nog bedieners by die stelsel te voeg eerder as om een bediener aanhoudend uit te brei en te vergroot.

8.3.2 Voorbeeld van 'n omgewing met meer as een bediener.

Wanneer daar besluit word om rolle en rolprofielobjekte tussen biedeners te versprei kan die verspreiding logies en verwant aan die struktuur van die organisasie gedoen word. Die stelling word na figuur 8.5 gemotiveer. Figuur 8.5 toon 'n omgewing met twee bedieners waar sekere kliënt-rekenaars hoofsaaklik met een van twee bedieners kommunikeer.



Figuur 8.5
'n Kliënt/bediener omgewing met meer as een bediener.

Let op dat in figuur 8.5 die rolprofielobjekte versprei is tussen twee sekerheidsbedieners. Gebruikers in die personeelafdeling gaan na die personeelafdeling bediener vir die magtiging van transaksies wat hulle wil uitvoer en gebruikers in die bemarkingsafdeling gaan na die bemarkingsafdeling bediener vir die magtiging van transaksies wat hulle wil uitvoer.

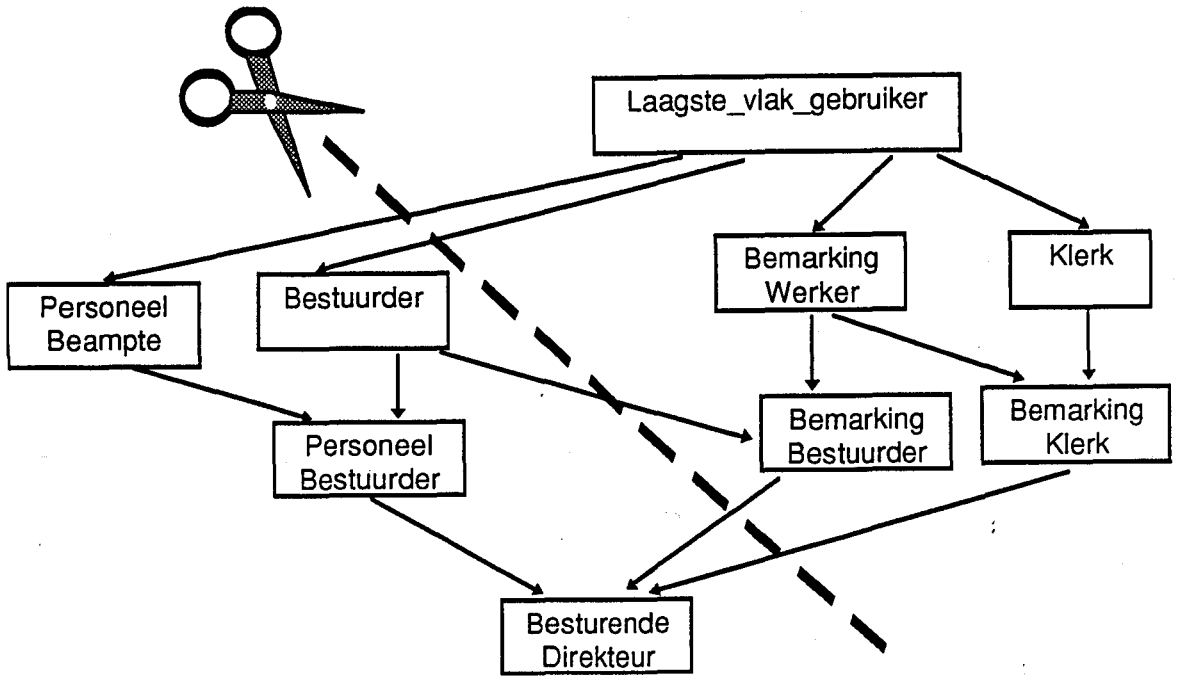
8.3.3 Gebruik van die roltralie om rolprofielobjekte te versprei

Die roltralie kan handig gebruik word om te bepaal watter rolprofielobjekte en rolle op watter bediener lê. Die roltralie word dusverdeel in kleiner subtralies en hierdie subtralies se rolprofielobjekte sowel as die rol self word oor bedieners vesprei (onthou die rol self bevat 'n verwysing na die rolprofielobjek maar ook ander inligting soos 'n lys van ouer- en kindrolle).

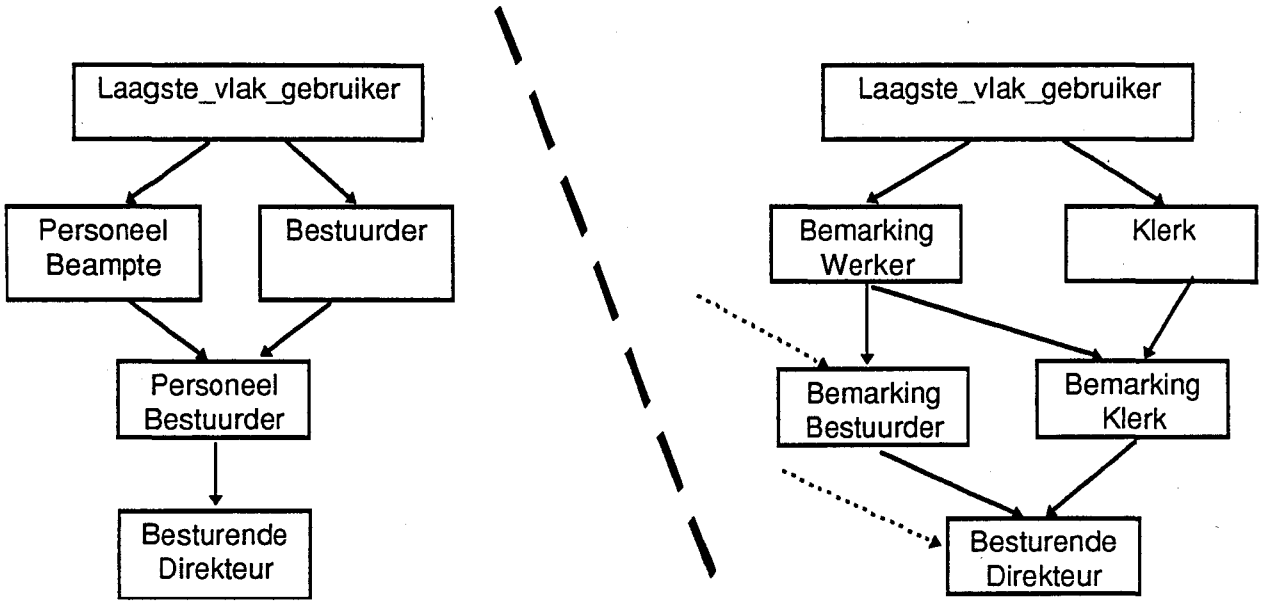
Die verdeling van die roltralie in subtralies sal normaalweg verband hou met die uiteensetting van die organisasie waarvoor die roltralie geskep is. So byvoorbeeld sal die roltralie rolle vir 'n personeel- en bemarkingafdeling bevat. Daar sal egter subtralies vir die personeel- en bemarkingafdeling onderskeidelik geskep word. Hierdie subtralies se rolle en rolprofielobjekte word dus na verskillende bedieners

versprei. Die bediener met die personeelafdeling se rolle sal dus normaalweg sekerheidsversoeke van transaksies uit die personeelafdeling hanteer.

Figuur 8.6a toon 'n roltralie en die donker snydingslyn toon die moontlike opdeling van die rolle tussen twee subtralies. In Figuur 8.6b word die resultaat van so 'n opdeling getoon. Let op dat die resultaat weer twee tralies is (maar subtralies in hierdie geval).



Figuur 8.6a
'n Voorbeeld Roltralie en moontlike opdeling



Figuur 8.6b
Die resultaat van die opdeling van 'n tralie: twee subtralies

Figuur 8.6 het getoon hoe die rolle in 'n organisasie opgedeel kan word en oor twee bedieners versprei kan word. Die bemarkingafdeling en die personeelafdeling gebruik dusverskillende bedieners. Sekere rolle mag op meer as een bediener voorkom om te hou by 'n roltralie rangskikking van rolle maar ook omdat sulke rolle tipies ewe veel by albei afdelings gebruik word en op albei bedieners meer doeltreffendheid sal bewerkstellig.

Let ook op dat in figuur 8.6b daar by rol Bemarkingbestuurder en rol Besturende_Direkteur 'n stippellyn-pyl aangebring word. Dit toon dat hierdie rolle gevorm is uit onder andere 'n ander rol wat nie in hierdie tralie (of op hierdie bediener) voorkom nie. Die naam van so 'n ouer-rol en die bediener waarop hy lê sal egter steeds in die rol gestoor moet word om later wysigings van die rol moontlik te maak.

8.3.4 Implikasies van 'n verspreide stelsel op ORITO

Let op dat indien daar meer as een bediener in die stelsel is sal die transaksieverwerker op 'n kliënt moet weet watter bediener om te gebruik om sy sekerheidsversoeke te hanteer. Dit beteken die transaksieverwerker se sekerheidsuitgang op die kliënt rekenaar moet weet watter bediener die sekerheidsversoeke hanteer. 'n Ander opsie is om alle versoeke na 'n sentrale bediener te stuur wat dit weer roeteer na die korrekte bediener. So 'n stelsel loop egter die risiko dat die sentrale bediener 'n bottelnek kan veroorsaak en indien die sentrale bediener nie funksioneer nie kan geen sekerheidsversoek by sy korrekte bediener uitkom nie (al funksioneer die bediener wel).

'n Verdere probleem wat ontstaan in die verspreide stelsel is dat die kliënt rekenaar nie noodwendig weet by watter bediener om 'n gebruiker wat nuut aanteken te registreer nie. Onthou dat die bediener wat verantwoordelik is vir die magtiging van transaksies vir 'n gebruiker afhang van die rol wat die gebruiker kies wanneer hy aanteken. 'n

Gebruiker mag byvoorbeeld soms aanteken as 'n Bemarking_Werker en ander tye weer as 'n Personeel_Beampte. In eersgenoemde geval sal die bemarkingsafdeling se bediener die magtigings hanteer en wanneer die gebruiker as 'n Personeel_Beampte aanteken sal die personeelafdeling se bediener die magtigings moet hanteer. Die punt om op te let is dat die stelsel eers weet nadat die gebruiker sy rolkeuse gemaak het watter bediener om te gebruik vir die hantering van die magtiging van transaksies vir die gebruiker.

Ons los hierdie registrasie probleem op deur te vereis dat 'n kliënt na enige bediener kan gaan om registrasie van 'n nuut aangetekende gebruiker te hanteer. Dit vereis eerstens dat elke bediener 'n kopie van die GEBRUIKER_ROLLE moet tabel hê en hierdie tabel moet op datum wees. Hierdie is 'n redelike beperking op die stelsel omdat wanneer die GEBRUIKER_ROLLE tabel by 'n bediener verander word, stuur so 'n bediener 'n boodskap aan elke ander bediener in die stelsel om die verandering ook in sy tabel aan te bring. Wanneer 'n gebruiker by 'n kliënt rekenaar aanteken sal die transaksieverwerker op die kliënt se sekerheidskomponent dus vir enige bediener in die stelsel kan vra om die gebruiker te registreer. Elke kliënt kan natuurlik 'n voorkeurlys van bedieners hê wat registrasie van 'n gebruiker hanteer.

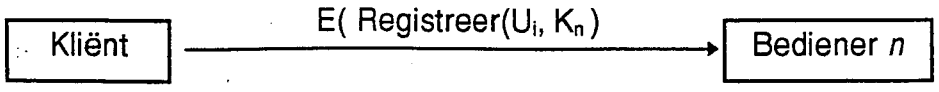
8.3.5 Die Registrasie fase in 'n verspreide stelsel

Die registrasie fase word eers informeel beskryf en dan stapsgewys aan die hand van 'n diagram.

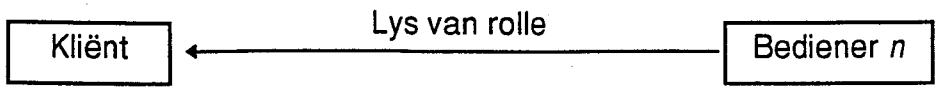
Die gebruiker verskaf sy identifikase en wagwoord aan die transaksieverwerker. Die transaksieverwerker verifieer die gebruiker en om te registreer by 'n sekerheidsbediener kies die transaksieverwerker 'n bediener uit sy voorkeurlys van bedieners en stuur vir die ESM op die daardie bediener (gestel dit is bediener n) die Registreer boodskap tesame met die gebruiker se identifikasie.

Kommunikasie tussen die transaksieverwerker en die sekerheidsbediener geskied weer deur middel van die transaksieverwerker se sekerheidsuitgang soos op p.106 beskryf is.

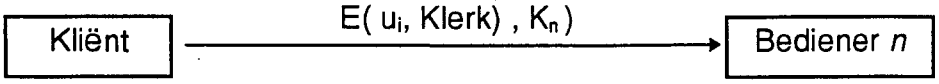
Alle boodskappe word geënkripteer onder 'n sleutel (bv. K_n)wat die kliënt en bediener deel. Die sleutels van die verskillende bedieners moet natuurlik ook by elke kliënt gehou en beskerm word. Vir die beskerming en verspreiding van sleutels bestaan daar goed nagevorsde algoritmes soos byvoorbeeld die publieke sleutel stelsel [12].



Die ESM op bediener n ontvang die boodskap, dekripteer dit, lees in sy GEBRUIKER_ROLLE tabel die lys van rolle waaraan hierdie gebruiker gekoppel mag wees en stuur hierdie lys terug aan die kliënt.



Die transaksieverwerker op die kliënt vertoon aan die gebruiker die lys, die gebruiker maak 'n keuse en die keuse word aan die bediener terug gestuur. Gestel die gebruiker kies rol Klerk.

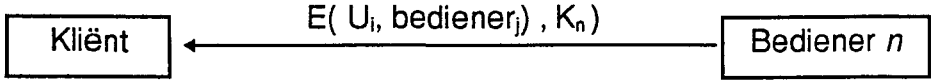


Nou weet bediener n aan watter rol gebruiker U_i gekoppel is. Ons vereis nou verder dat elke bediener 'n tabel ROLLE_BEDIENER bevat. Hierdie is 'n tabel wat aandui watter bediener die magtiging van watter rol hanteer. Tabel 8.5 toon 'n voorbeeld van so 'n tabel.

ROLLE_BEDIENER	
Rolnaam	Bediener
Personeel_Beampte	Bediener 2
Klerk	Bediener *j*
rol *m*	Bediener *n*

Tabel 8.5
Voorbeeld van 'n ROLLE_BEDIENER tabel.

Die bediener kyk nou in sy ROLLE_BEDIENER tabel watter bediener hanteer die magtiging van hierdie rol; gestel dit is bediener j. Bediener n stuur 'n boodskap aan die kliënt rekenaar om aan te dui watter bediener gebruik moet word om gebruiker U_i se transaksies te magtig.

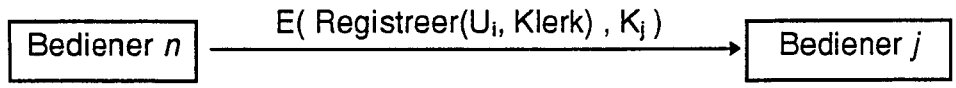


Die kliënt rekenaar ontvang die boodskap en voeg in 'n lokale tabel GEBRUIKER_BEDIENER die gebruiker se identifikasie en die bediener om te gebruik wanneer transaksies vir gebruiker U_i gemagtig moet word. Tabel 8.6 toon 'n voorbeeld van so 'n tabel. Elke kliënt rekenaar het so 'n tabel nodig om te weet watter sekerheidsbediener om te gebruik vir watter gebruiker.

GEBRUIKER_BEDIENER	
Gebruiker identifikasie	Bediener
U_k	Bediener *l*
Ui	Bediener *j*

Tabel 8.6
Voorbeeld van 'n GEBRUIKER_BEDIENER tabel.

Om die registrasie fase te voltooi moet bediener j weet van gebruiker U_i. Dit word gedoen deurdat **bediener n aan bediener j 'n boodskap stuur** om gebruiker U_i wat aangeteken is as Klerk by bediener j te registreer.

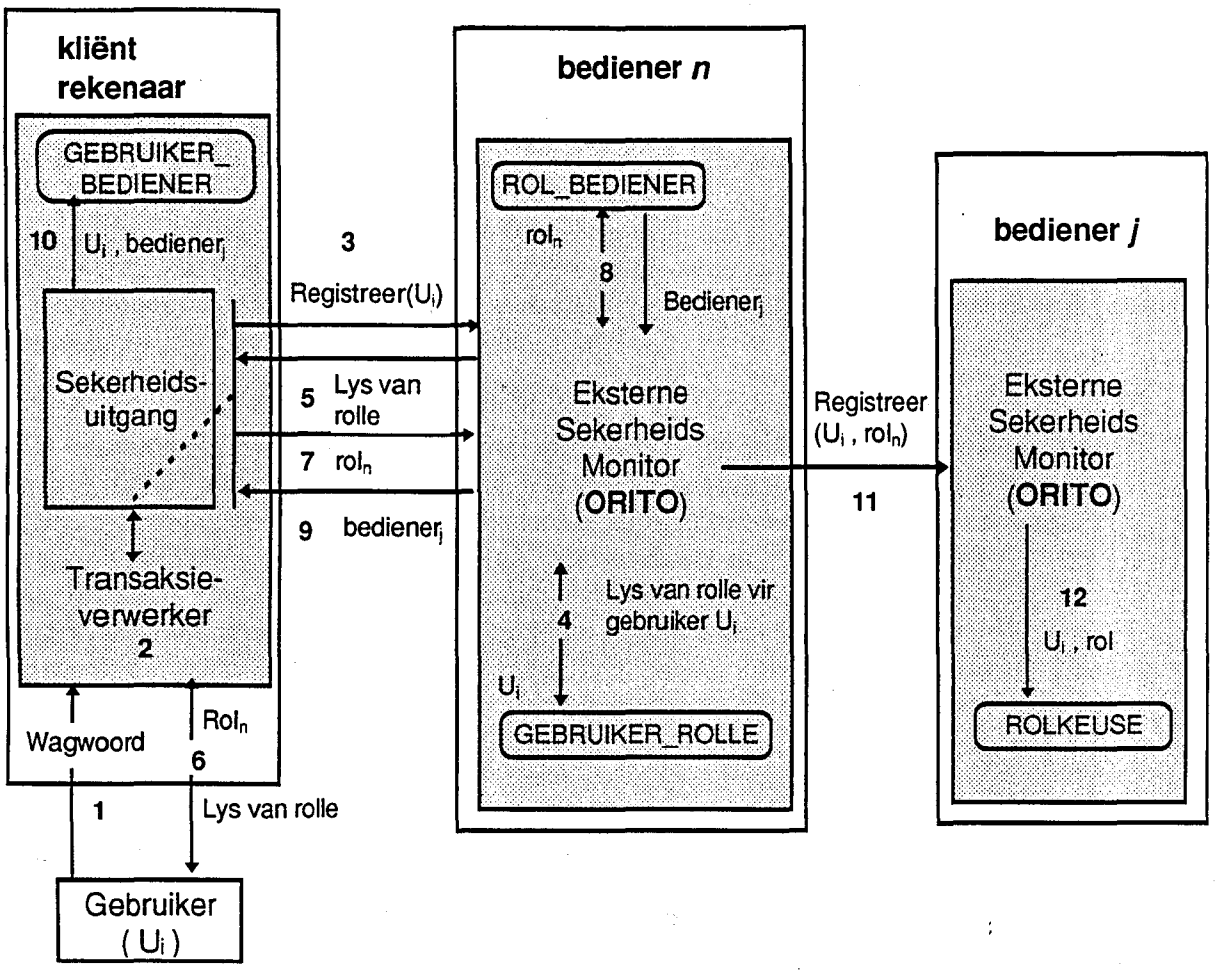


Onthou dat alle boodskappe wat gestuur word geënkripteer is. Elke bediener het dus ook 'n lys met bedieners en sleutels.

Bediener j ontvang die boodskap en opdateer sy ROLKEUSE tabel sodat die gebruiker se keuse later gebruik kan word.

Indien bediener n self die magtiging vir rol Klerk hanteer, is die boodskap van bediener n na bediener j natuurlik nie nodig nie en opdateer bediener n sy eie ROLKEUSE tabel.

Figuur 8.7 toon die stappe betrokke by die registrasie van 'n nuut aangetekende gebruiker. Ons aanvaar die kliënt rekenaar versoek 'n bediener om die gebruiker te registreer en hierdie bediener is nie dieselfde bediener wat uiteindelik verantwoordelik is vir die magtiging van transaksies vir die rol wat die gebruiker gekies het nie.



Figuur 8.7
Registrasie deur ORITO in 'n kliënt/bediener omgewing met meer as een bediener

8.3.5.1 Die stappe volgens die figuur is as volg:

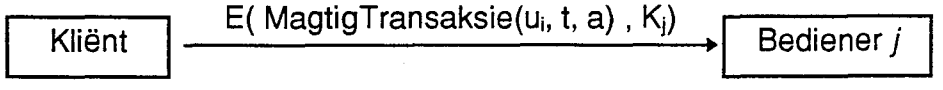
1. Die gebruiker U_i gee sy identifikasie en wagwoord aan die transaksieverwerker
2. Die transaksieverwerker verifieer gebruiker U_i.
3. Die transaksieverwerker se sekerheidsuitgang stuur die Registreer boodskap aan bediener n.
4. Bediener n onttrek die lys gemagtigde rolle vir gebruiker U_i uit sy GEBRUIKER_ROLLE tabel.
5. Bediener n stuur hierdie lys rolle aan die kliënt rekenaar
6. Die transaksieverwerker op die kliënt rekenaar vertoon die lys rolle aan die gebruiker en lees sy rolkeuse. Veronderstel die gebruiker kies rol_n.
7. Die transaksieverwerker stuur die gebruiker se rolkeuse terug aan bediener n.
8. Bediener n lees uit die ROLLE_BEDIENER tabel die naam van die bediener wat magtiging vir rol_n hanteer (sê bediener j).
9. Bediener n stuur die naam van die geleesde bediener (bediener j) aan die kliënt rekenaar.
10. Die kliënt rekenaar opdateer die GEBRUIKER_BEDIENER tabel.
11. Bediener n stuur aan bediener j die boodskap om gebruiker U_i te registreer as rol_n.
12. Bediener j opdateer sy ROLKEUSE tabel en die registrasie fase is voltooi.

8.3.5.2 Registrasie fase deur van DCE gebruik te maak

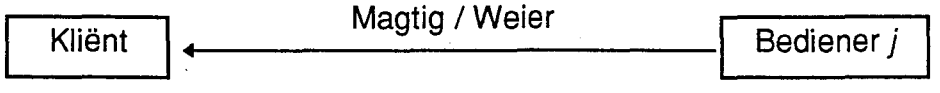
Let op dat die grootste verskil tussen die registrasie fase in die verspreide omgewing teenoor registrasie in die konvensionele een-bediener omgewing is dat elke kliënt moet weet watter bediener om te gebruik vir watter gebruiker van die transaksieverwerker. DCE van OSF bied onder andere 'n soortgelyke diens by wyse van sy gidsdienste [8, 18]. Indien die uitgebreide (verspreide) ORITO model dus geïmplementeer word in 'n omgewing waar DCE gebruik word kan die stappe wat hierbo beskryf is grootliks vereenvoudig word deur van DCE se gidsdienste gebruik te maak. CICS/6000 is 'n transaksieverwerker wat van DCE gebruik maak [11].

8.3.6 Die Magtiging fase in 'n verspreide stelsel

Die Magtiging fase in 'n verspreide stelsel met meer as een bediener lyk nie veel anders as in 'n stelsel met net een bediener nie. Die gebruiker versoek by die kliënt rekenaar transaksies wat deur 'n sekerheidsbediener gemagtig moet word. Vir elke transaksie wat gebruiker U_i versoek, lees die transaksieverwerker op die kliënt rekenaar in sy GEBRUIKER_BEDIENER tabel die naam van bediener wie gebruiker U_i se magtiging van transaksies moet hanteer. Gestel bediener j hanteer die magtiging van transaksies van gebruiker U_i. Nou word die MagtigTransaksie boodskap aan die bediener j tesame met die gebruikeridentifikasie, transaksienaam en transaksieomstandighede gestuur.



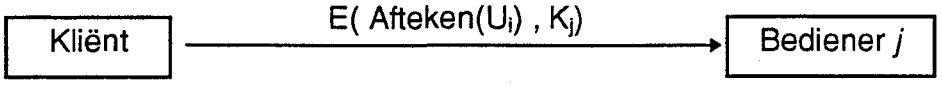
Bediener j ontvang die boodskap, lees in sy ROLKEUSE tabel aan watter rol gebruiker U_i tans gekoppel is. Hierna word die rol waaraan die gebruiker gekoppel is se inskrywing in die ROLLE tabel opgesoek. Die MagtigTransaksie lidfunksie van die rol se rolprofielobjek word geroep. Die resultaat wat die RPO terugstuur word net so aan die kliënt rekenaar aangestuur en die transaksie word slegs uitgevoer as dit 'n magtig resultaat is.



Let op dat die ROLLE tabel slegs inskrywings hoef te bevat van die rolle wat deur hierdie bediener hanteer word. Ook so, hoef bediener j slegs die rolprofielobjekte te bevat wat dit hanteer (ons maak in die volgende hoofstuk 'n verdere opmerking hieroor).

8.3.7 Die Aftekening fase in 'n verspreide stelsel

In die Aftekening fase teken 'n gebruiker af by 'n kliënt rekenaar en die Aftekening boodskap tesame met die gebruiker se identifikasie word aan die bediener, wat in die GEBRUIKER_BEDIENER tabel opgesoek is gestuur. Die ESM op die bediener opdateer die ROLKEUSE tabel om hierdie aftekening in die toestand van sy tabelle te weerspiëel.



In hierdie afdeling is gewys hoe die roltralie gebruik kan word om rolprofielobjekte tussen bedieners te versprei. Die afdeling het ook getoon watter stappe gevolg word om ORITO te implementeer in so 'n verspreide stelsel. In die res van hierdie hoofstuk gee ons aandag aan 'n paar losstaande aspekte van ORITO in 'n verspreide stelsel.

8.4 'n Objek-georiënteerde sekerheidsbediener

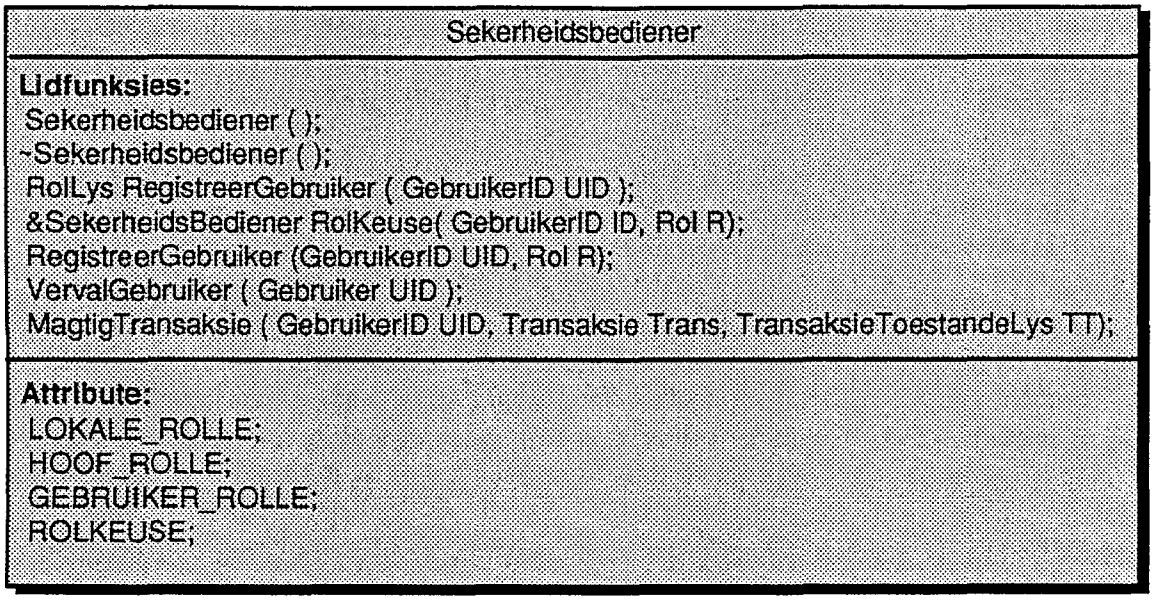
Tot dusver maak ons gebruik van 'n sekerheidsbediener waarvan slegs die rolprofiele objekte is. In hierdie gedeelte word gekyk hoe die sekerheidsbediener as een groot objek beskou kan word. Met ander woorde die sekerheidsbediener is nie 'n program met losstaande komponente soos tabelle nie - alles is deel van die sekerheidsbedienerobjek. Dit beteken dat wanneer 'n kliënt rekenaar 'n transaksie wil magtig, dit 'n lidfunksie van 'n bediener objek roep.

Daar is dus een of meer sekerheidsbedieners waarop 'n sekerheidsbedienerobjek lê. Hierdie objek implementeer in wese ORITO en om die magtiging van 'n transaksie te doen, word die MagtigTransaksie lidfunksie van ander objekte (rolprofielobjekte) deur hierdie objek geroep.

Die bediener hoef dus nie meer 'n klomp tabelle aan te hou nie - die data wat voorheen in tabelle gestoor is, word nou in attribute van die bediener objek gehou. Dit is 'n beter benadering omdat die inhoud van die tabelle nou beskerm word deur die enkapsulasie eienskap van objek-georiënteerdheid - die waardes kan slegs verander word deur die bediener objek se lidfunksies te gebruik (wat 'n meer beheerde manier is).

Bogenoemde is 'n belangrike konsep. Voorheen is die tabelle heel moontlik as gewone lêers gestoor op die sekerheidsbediener. So 'n benadering laat baie ruimte vir ongemagtigde toegang tot die tabelle. Indien die tabelle deel is van die attribute van die sekerheidsbedienerobjek is toegang tot die tabelle slegs moontlik deur middel van die objek se lidfunksies. Laasgenoemde is 'n baie beter benadering uit 'n sekerheidsoogpunt.

Die sekerheidsbediener is nou 'n instansie van die sekerheidsbediener klas. Die sekerheidsbediener klas lyk soos in figuur 8.8.



Figuur 8.8
Die sekerheidsbediener-klas

Die kliënt rekenaar roep nou die RegistreerGebruiker lidfunksie van die bediener objek om 'n nuwe gebruiker te registreer. Let op die veelmorfisme in die RegistreerGebruiker lidfunksie - die funksie reageer anders afhangende van die parameters (RegistreerGebruiker(GebruikerID) word deur 'n kliënt rekenaar geroep, maar RegistreerGebruiker(GebruikerID, Rol) word deur 'n bediener geroep).

Die kliënt rekenaar roep ook nou die MagtigTransaksie en VervalGebruiker lidfunksies om 'n transaksie te magtig of 'n gebruiker se registrasie te laat verval.

Die tabelle LOKALE_ROLLE, HOOFROLLE, GEBRUIKER_ROLLE en HUIDIGE_STATUS is nou private attribute van die objek.

Die bediener objek doen dieselfde as wat die ESM gedoen het wat tot dusver bespreek is, behalwe dat alles geënkapsuleerd en objek-georiënteerd gedoen word. Dit is dus nie meer 'n program/proses wat die rolprofielobjekte roep vir magtiging nie, maar 'n ander (sekerheidsbediener) objek.

Elke kliënt rekenaar stoor dus in sy GEBRUIKER_BEDIENER tabel die unieke identifikasie van 'n sekerheidsbediener objek en roep lidfunksies van hierdie objek om rolgebaseerde inligtingsekerheid volgens die ORITO model te bewerkstellig.

8.4.1 Kommunikasie tussen objekte in 'n verspreide stelsel

Die sekerheidsbediener is nou ook 'n objek en die transaksieverwerker roep lidfunksies van hierdie objek. Hierdie objek lê egter op 'n bediener rekenaar op enige netwerk in die wêreld. Hoe kom die transaksieverwerker by die objek uit?

Voor ons die vraag beantwoord, let op dat die transaksieverwerker se sekerheidskomponent ook 'n objek kan wees. Ons het dus een objek wat 'n ander roep vir sy sekerheidsmagtiging en dié roep weer ander objekte (RPOs) om die magtiging te

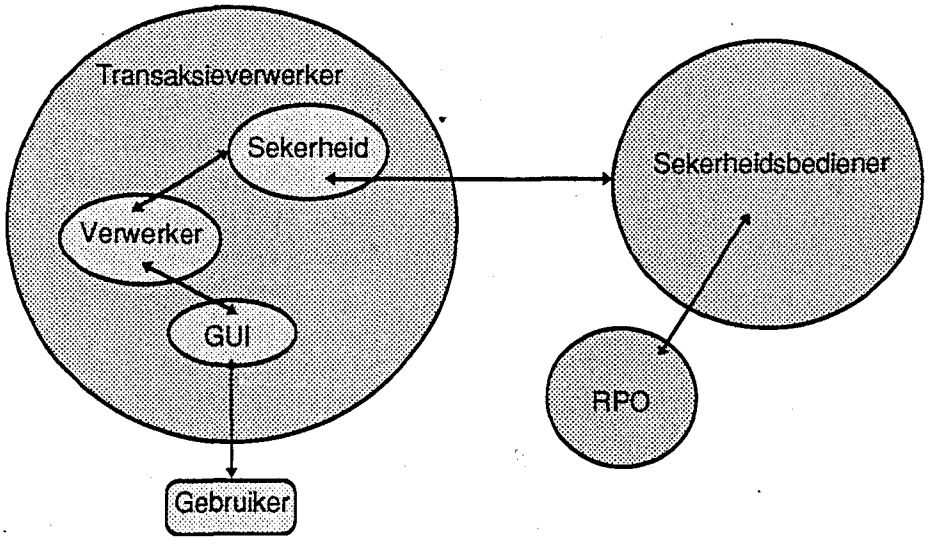
doen. Op een of ander manier moet die objekte van mekaar weet, weet hoe lyk die koppelvlakke en waar om hulle te kry. Hierdie en ander vrae word aangespreek deur die *Object Management Group* (OMG) [9].

Die OMG het 'n standaard ontwerp vir objekte wat so wil saamwerk in 'n verspreide omgewing (verspreide objekte). Die OMG stel 'n *Object Request Broker* (ORB) voor. Die ORB is die middelware wat die verwantskap tussen kliënt/bediener objekte daarstel [9 (p 699)].

CORBRA (*Common Object Request Broker Architecture*), 'n ORB argitektuur van die OMG, is 'n meganisme wat objekte deursigtelik toelaat om versoeke na, en resultate van, ander objekte wat lokaal of afgeleë is te verkry. Die huidige spesifikasie bevat 'n koppelvlak definisietaal wat gebruik word om objekte mee te definieer wat kommunikeer via die ORB.

Om al die verskillende objekte wat nou ter sprake is in die verspreide omgewing waar ORITO geïmplementeer word te laat saamwerk vereis ons dat die objekte volgens CORBRA gedefinieer word. Dit is veral belangrik wanneer ons in 'n oop omgewing werk soos vervolgens bespreek gaan word.

Figuur 8.9 toon al die objekte wat kan saamwerk wanneer 'n transaksie gemagtig word in objek-georiënteerde stelsel in 'n oop verspreide omgewing waar ORITO geïmplementeer word. Elke sirkel of ovaal is 'n objek in die stelsel.



Figuur 8.9
Interaksie tussen objekte in verspreide ORITO omgewing

Die enigste interaksies is nou tussen die gebruiker en die stelsel en tussen objekte, alle ander data word verkry deur boodskappe tussen objekte te stuur (dit is nou as ons 'n objek-georiënteerde transaksieverwerker gebruik).

8.5 ORITO in 'n oop kliënt/bediener omgewing

Deesdae is dit amper 'n vereiste in meeste groot stelsels dat rekenaars met verkillende argitekture en verkillende bedryfstelsels inligting uitruil. Ons kan ORITO ook aanpas om te funksioneer in sulke oop stelsels.

Wanneer ORITO geïmplementeer word in 'n oop kliënt/bediener omgewing is die enigste verskil dat die formaat waarin sekerheidsversoeke opgestel is nie noodwendig dieselfde is vir elke bediener of kliënt in die stelsel nie. Dit is dus nodig dat voordat 'n sekerheidsversoek of resultaat tussen kliënt en bediener gestuur word, die versoek of resultaat omgeskakel moet word na 'n formaat verstaanbaar vir die rekenaar wat die boodskap ontvang.

Die bedieners en kliënte in die stelsel kan aangepas word om hierdie omskakelings te doen. Ons hoef dus nie veel te wysig aan ons implementering soos in die vorige afdeling bespreek is nie.

Let egter op dat as ons die omgewing heeltemal objek-georiënteerd maak, dit beteken die transaksieverwerker, die transaksieverweker se sekerheidskomponent, die sekerheidsbediener en die rolprofielobjekte almal **objekte** is, dan kan ons die versoenbaarheidsprobleem oplos deur te vereis dat al die objekte in die stelsel gedefinieer word volgens die CORBRA argitektuur. Hiervolgens sal dit dan moontlik wees vir die objekte om met mekaar te kommunikeer maak nie saak op watter stelselargitektuur of bedryfstelsel hulle lê nie.

8.6 Slot

In hoofstuk 6 het ons die model vir objek-georiënteerde, rolgebaseerde inligtingsekerheid in 'n transaksieverwerking omgewing geformuleer. In hierdie hoofstuk is die model uitgebrei om te funksioneer as 'n model vir inligtingsekereid in 'n kliënt/bediener omgewing. Daar is getoon dat indien die model die uitvoer van transaksies magtig dan kan die model geïmplementeer word as 'n bediener in die stelsel, naamlik 'n sekerheidsbediener.

Die hoofstuk het die model nog verder uitgebrei en getoon dat ORITO kan gebruik maak van die voordele van verspreide stelsels deur die werk wat ORITO doen (die magtiging van transaksies) te versprei tussen bediener-rekenaars. 'n Belangrike gebruik van die roltralie wat in hoofstuk 6 gedefinieer is kom hier na vore. Die roltralie word gebruik deur die roltralie op te deel in kleiner subtralies en die ooreenkomstige rolprofielobjekte van rolle in een subtralie te versprei na een bediener. ORITO word dan 'n model vir objek-georiënteerde, rolgebaseerde verspreide inligtingsekerheid in 'n kliënt/bediener transaksieverwerking omgewing. Let daarop dat ORITO is nie 'n model vir inligtingsekerheid in verspreide omgewings nie, dit is 'n model vir inligtingsekerheid in kliënt/bediener omgewings en kan meer doeltreffend funksioneer deur van meer as een bediener gebruik te maak (met ander woorde deur van verspreide verwerking gebruik te maak).

In die volgende hoofstuk gaan ons kyk hoe ons die model effens kan aanpas om 'n model te skep wat nie alleen meer doeltreffend is nie, maar ook meer betroubaar. Ons poog om 'n model daar te stel waar 'n bediener te enige tyd aan of afgeskakel kan word

en die ander bedieners die werkslading oorneem of van hulle werkslading oordra na die nuwe bediener. Laasgenoemde word intyds gedoen sonder dat die stelsel hoef af te gaan, maar meer hieroor in die volgende hoofstuk.

9. 'n Meer doeltreffende en meer betroubare verspreide inligtingsekerheidstelsel

9.1 Inleiding

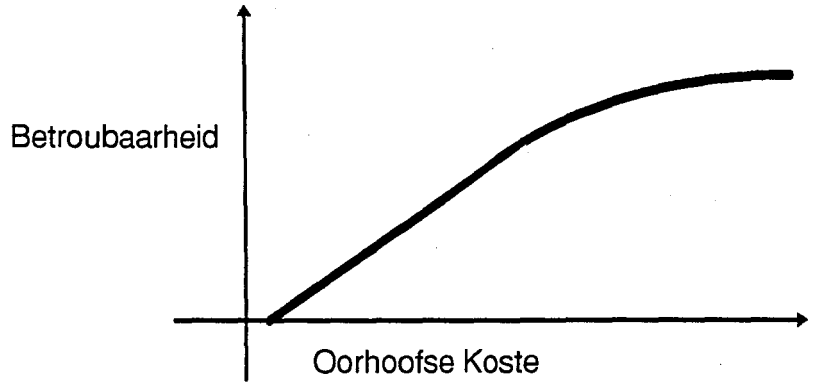
Die verspreide inligtingsekerheidstelsel wat in die vorige hoofstuk bespreek is het die voordeel dat bottelnekke by 'n bediener vermy word deurdat die werkslading oor meer as een bediener versprei word. So 'n verspreide stelsel is ook meer doeltreffend deurdat vertikale skaleerbaarheid moontlik is. Dit is maklik om die werksvermoë van die stelsel uit te brei deur bedieners by te voeg en sekere van die rolprofielobjekte na die nuwe bediener te versprei. Ons het al die opmerking gemaak dat in 'n stelsel waar slegs een sekerheidsbediener gebruik word, dit moontlik is dat die hele stelsel tot stilstand kom indien die sekerheidsbediener buite werking is. In die verspreide stelsel wat bespreek is, word hierdie risiko verlaag deurdat indien 'n bediener buite werking raak, slegs die rolle vir wie hierdie bediener magtiging doen beïnvloed word.

Is dit dalk moontlik om die verspreide stelsel so aan te pas dat wanneer 'n bediener buite werking raak, die ander bedieners tydelik sy werk kan oorneem? In hierdie hoofstuk bespreek ons hierdie benadering. Daar word ook gewys op die koste in terme van instandhouding en administrasie van so 'n meer doeltreffende en meer betroubare stelsel.

9.2 Meer doeltreffendheid en betroubaarheid het 'n prys

In meeste stelsels is dit so dat 'n toename in doeltreffendheid en betroubaarheid gewoonlik 'n koste het. Ons maak hier opmerkings oor hoe om die stelsel wat ons tot dusver bespreek het meer betroubaar en meer doeltreffend te maak maar let op dat hierdie metodes almal 'n toename in die hoeveelheid administrasie en meer oorhoofse kostes by die instandhouding van die stelsel veroorsaak. Of die toename in betroubaarheid of doeltreffendheid die moeite werd is hang af van die tipe stelsel, die hoeveelheid bedieners, die impak van bedieners wat buite werking is ens. Ons maak deurgaans in die hoofstuk opmerkings oor die koste van die stelsel wat ons voorstel.

Figuur 9.1 toon die verwantskap tussen betroubaarheid en oorhoofse koste in 'n stelsel.



Figuur 9.1
Verwantskap tussen betroubaarheid en oorhoofse koste in 'n stelsel

Die verwantskap in figuur 9.1 is voor die hand liggend. Soos wat betroubaarheid toeneem, neem oorhoofse koste ook toe. As motivering neem 'n stelsel met slegs een bediener: indien die bediener faal is geen dienste beskikbaar nie (lae betroubaarheid), maar dit is relatief maklik om die bediener in stand te hou (lae koste). 'n Stelsel met meer as een bediener kan steeds dienste verskaf wanneer slegs een bediener faal (hoër betroubaarheid) maar dit is moeiliker om die bedieners in stand te hou want sekere inligting moet op elke bediener gedupliseer en op datum gehou word (meer koste). Daar is natuurlik ook 'n punt van optimale betroubaarheid en daarom plat die grafiek af.

9.3 'n Meer betroubare inligtingsekerheidstelsel

In die vorige hoofstuk is ORITO aangepas om te funksioneer in 'n kliënt/bediener omgewing en benut die voordele van verspreide verwerking. Deurdat rolprofielobjekte tussen bedieners versprei word, is die stelsel meer doeltreffend deurdat die werkslading versprei word tussen bedieners. Die stelsel is reeds in 'n mate meer betroubaar omrede 'n bediener wat buite werking is, net 'n sekere aantal rolle beïnvloed en alle ander rolle kan voortgaan met die uitvoer van transaksies. In hierdie gedeelte kyk ons hoe ORITO aangepas kan word sodat dit 'n inligtingsekerheidstelsel is wat nog meer betroubaar is. Trouens, ons wil 'n verspreide inligtingsekerheidstelsel daarstel wat sal sorg dat solank ten minste een bediener funksioneer, kan alle rolle voortgaan met die uitvoer van transaksies.

9.3.1 Aanpassing van ORITO vir meer betroubaarheid

Gestel 'n sekerheidsbediener raak tydelik buite werking, hoe moet ons ORITO aanpas sodat die ander sekerheidsbedieners sy werk kan oorneem?

9.3.2 Enige bediener moet 'n gebruiker kan registreer

Die eerste probleem wat kan ontstaan is dat die bediener wat buite werking is nie meer die registrasie van nuut aangetekende gebruikers kan doen nie. Onthou dat elke kliënt rekenaar 'n voorkeurlys het van bedieners vir registrasie van nuwe gebruikers. Indien 'n kliënt dus geen respons van 'n bediener kry nie, word die volgende bediener in die voorkeurlys gebruik (noem hierdie bediener k). Bediener k hanteer die registrasie en lees uiteindelik die rol wat die nuwe gebruiker gekies het by die kliënt rekenaar. Indien die bediener wat verantwoordelik is vir die magtiging van hierdie rol (gestel dit is

bediener n) buite werking is, is dit nou bediener k se taak om die magtiging oor te vat. Die gebruiker word nou by bediener k i.p.v. bediener n geregistreer as 'n *vreemde* gebruiker. Aan die kliënt rekenaar word die boodskap gestuur dat bediener k die magtiging hanteer.

Tabel 9.1 toon hoe die ROLKEUSE tabel uitgebrei word om ook aan te dui by watter bediener 'n gebruiker eintlik hoort (vir bediener k is Mary in die voorbeeld 'n vreemde gebruiker).

ROLKEUSE		
Gebruiker Identifikasie	Huidige Rol	Bediener
John	Klerk	Bediener k
Mary	Rekeninge_Klerk	*Bediener n*
Sarah	Bankbestuurder	Bediener k

vreemde gebruiker vir bediener k

Tabel 9.1
Voorbeeld van die nuwe ROLKEUSE tabel vir bediener k.

9.3.3 Elke bediener moet al die rolle en rolprofielobjekte stoor

Wanneer 'n vreemde gebruiker dus 'n transaksie versoek op 'n kliënt rekenaar en 'n bediener moet hierdie magtiging hanteer, het die bediener al die inligting van die rol tesame met die rol se rolprofielobjek nodig. Elke bediener sal dus die volledige roltralie in 'n HOOF_ROLLE tabel moet hê asook die rolprofielobjek van elke rol. Elke bediener het nou ook 'n lokale subtralie (in 'n LOKALE_ROLLE tabel) om te weet watter rolle dit eintlik voor verantwoordelik is. Let op dat elke LOKALE_ROLLE tabel is 'n deelversameling van die HOOF_ROLLE tabel.

Bogenoemde veroorsaak 'n paar probleme. Wanneer 'n rol by sy bediener verander word, moet die veranderinge by die bediener se LOKALE_ROLLE tabel asook by elke bediener in die stelsel se HOOF_ROLLE tabel aangebring word. Verder moet die veranderinge aan die objek se datawaardes aangebring word en die nuwe RPO moet aan elke bediener gestuur word.

Hierdie beperking verhoog die oorhoofse koste van die stelsel. Neem die eenvoudige geval waar 'n rolprofielobjek een transaksie bykry. Dit sal beteken dat elke kind-rol van die RPO asook elke kind se kind verder af in die roltralie moet hierdie transaksie bykry. Die probleem is nou dat hierdie opdatering op elke bediener gedoen kan word en dit kan lank neem in 'n verspreide stelsel. Weer eens is dit 'n oorhoofse koste wat in sommige stelsels (bv. missie kritiese stelsels) die moeite werd sal wees en by ander nie.

Die oorhoofse koste kan deels verlaag word deur te vereis dat slegs een rugsteun bediener al die rolle se magtiging kan hanteer en slegs hierdie bediener word gebruik wanneer 'n ander bediener faal. Let weer op dat ons nou oorhoofse koste verlaag (slegs twee bedieners se rolle moet opgedateer word wanneer 'n rol verander) maar betroubaarheid afneem wanneer meer as een bediener afgaan kan hierdie rugsteun

bediener oorlaai word met versoeke en wanneer dit self af is, is daar geen rugsteun bediener beskikbaar sou nog 'n bediener faal nie.

9.3.4 Sekere gebruikers moet geherregistreer word

Vir gebruikers wat reeds by bediener n geregistreer is wanneer dit buite werking raak, beteken dit dat hulle weer by 'n ander bediener geregistreer moet word. Hierdie herregistrasie word gedoen sonder dat die gebruiker daarvan kennis neem. Die stappe is dieselfde as om 'n nuut aangetekende gebruiker te registreer behalwe dat die gebruiker nie weer sy rol hoef te kies nie.

9.3.5 Die kliënt rekenaar moet ook weet watter rolkeuse 'n gebruiker gemaak het

Hierdie nuwe stelsel vereis natuurlik ook dat wanneer 'n kliënt rekenaar in sy GEBRUIKER_BEDIENER tabel aanteken watter bediener om vir watter gebruiker te gebruik, moet dit ook die gebruiker se huidige rol stoor sodat wanneer die bediender nie meer beskikbaar is nie, die gebruiker geheregistreer kan word sonder dat die gebruiker weer 'n rol moet kies. Tabel 9.2 toon die nuwe GEBRUIKER_BEDIENER tabel wat elke kliënt rekenaar onderhou.

GEBRUIKER_BEDIENER		
Gebruiker identifikasie	Bediener	Rol
U_k	Bediener *l*	Klerk
Ui	Bediener *j*	Personeelbestuurder

Tabel 9.2
Voorbeeld van 'n GEBRUIKER_BEDIENER tabel.

9.3.6 'n Bediener moet weer sy werk kan terug kry

Wat gebeur indien 'n bediener wat buite werking was, weer in werking tree? Op daardie stadium het dit heel moontlik al sy gebruikers verloor en word hulle transaksies deur ander bedieners gemagtig. Vir hierdie bediener om sy gebruikers weer terug te kry stuur dit 'n boodskap aan elke bediener in die stelsel om aan te dui dat dit buite werking was maar nou weer in werking getree het. Elke bediener wat hierdie boodskap ontvang kyk in sy ROLKEUSE tabel of van sy vreemde gebruikers eintlik hoort by die bediener wat nou weer in werking getree het, indien wel word hulle bloot net uit die tabel geskrap. Wat is die effek hiervan? Omdat so 'n gebruiker nou in effek by geen bediener geregistreer is nie, sal die kliënt rekenaar geen respons kry wanneer dit 'n magtiging versoek nie. Indien dit gebeur, sal die kliënt rekenaar die gebruiker probeer herregistreer wat sal veroorsaak dat die gebruiker by die korrekte bediener geregistreer word.

9.3.7 Gebruikers se registrasie verval outomaties indien 'n bediener buite werking raak

Gebruikers wat nooit transaksies versoek het terwyl hulle sekerheidsbediener buite werking was, se registrasie sal ook nie meer geldig wees by die bediener nie. Sodra hulle weer 'n transaksie versoek, word hulle geherregistreer.

Ten laaste, let op dat indien 'n gebruiker 'n aftekening versoek stuur na 'n bediener wat buite werking is, word die versoek nie hanteer nie, maar dit is ook nie nodig nie, want wanneer die bediener weer in werking tree is sy ROLKEUSE tabel leeg (die registrasie verval outomaties).

9.4 Slot

In hierdie hoofstuk is gewys dat dit moontlik is om ORITO so aan te pas dat dit optimale betroubaarheid verskaf. Meeste stelsels, insluitend ORITO, se oorhoofse koste verhoog soos wat doeltreffendheid toeneem. In ORITO raak die oorhoofse koste hoog omdat dit nodig is om al die rolprofielobjekte in die stelsel op elke bediener te dupliseer. Dit is 'n probleem omdat 'n verandering in 'n rolprofielobjek vereis dat die verandering by elke bediener in die stelsel gedoen word.

Die inligtingsekerheidstelsel wat in hierdie hoofstuk voorgestel is, sal dus slegs aanvaarbaar wees in 'n stelsel waar die oorhoofse koste die moeite werd is. Daar bestaan toepassings waar dit die moeite werd is om 'n hoë oorhoofse koste te hê net so lank as wat die magtiging van transaksies vir die stelsel kan voortgaan. Indien die magtiging van transaksies *missie krities* is vir 'n organisasie, soos 'n bank, kan betroubaarheid belangriker word as 'n lae oorhoofse koste.

In die vorige hoofstuk is ORITO aangepas vir 'n verspreide omgewing en in hierdie hoofstuk is getoon hoe ORITO aangepas kan word vir meer doeltreffendheid en meer betroubaarheid. Daar is vele ander voordelige aanpassings en uitbreidings wat aan ORITO gedoen kan word. Hierdie hoofstuk sluit egter die uitbreidings van ORITO in hierdie verhandeling af maar in die volgende hoofstuk word moontlike toekomstige uitbreidings en navorsingsareas genoem. Hoofstuk 10 gee ook 'n evaluering en opsomming van die navorsing wat in hierdie verhandeling gedokumenteer is.

10. Samevatting en moontlike toekomstige navorsing

10.1 Inleiding

In hierdie hoofstuk word 'n samevatting gegee van die verhandeling. Daar word gekyk na die belangrikste resultate van die navorsing en die voordele van die model wat in die verhandeling geformuleer is word samevattend beskryf. Die bydra van die navorsing tot die rekenaarwetenskap vakkennis uit die skrywer se oogpunt word ook bespreek.

Die navorsing wat in hierdie verhandeling gedokumenteer is skep ruimte vir toekomstige navorsing en uitbreiding van ORITO. Die hoofstuk gee 'n paar moontlike onderwerpe wat verder nagevors en uitgebrei kan word.

10.2 Samevatting

10.2.1 Doel van verhandeling

Hierdie verhandeling dokumenteer die navorsing wat nodig was om 'n nuwe model vir inligtingsekerheid te ontwikkel. Die model word die model vir objek-georiënteerde, rolgebaseerde inligtingsekerheid in transaksieverwerker omgewings (ORITO) genoem.

In deel 1 van die verhandeling word die navorsing gedokumenteer wat dien as agtergrond vr die model. Deel 2 dokumenteer die model asook die uitbreidings wat nodig is om die model aan te pas vir 'n model vir magtiging in kliënt/bediener omgewings. Die model word ook in deel 2 aangepas om te funksioneer as 'n verspreide inligtingsekerheidsmodel wat verhoogde betroubaarheid en doeltreffendheid bied.

10.2.2 Navorsingsvelde

Om bogenoemde model en die uitbreidings daarvan te kon formuleer was dit nodig om 'n studie van die konsepte en studierigtings wat in die model gebruik word te doen. In deel 1 van die verhandeling is hierdie navorsing gedokumenteer. Daar is hoofsaaklik navorsing gedoen in die volgende rekenaarwetenskap studievelde: inligtingsekerheid, objek-georiënteerde programmering en ontwerp, kliënt/bediener omgewings, verspreide verwerking en transaksieverwerking. Omdat ORITO rolgebaseerde inligtingsekerheid gebruik en die verwantskap tussen rolle met traliegrafieke voorstel was dit ook nodig om navorsing te doen in grafiekteorie, 'n wiskundige studieveld.

10.2.3 Evaluering (voordele) van ORITO

ORITO, 'n model vir inligtingsekerheid, bring 'n paar studievelde (soos hierbo bespreek is) bymekaar om sodoende 'n sterk model vir inligtingsekerheid daar te stel deur die voordele van hierdie studievelde te kombineer. In sommige opsigte gee die kombinasie van party van die studievelde nog meer voordele as wanneer hulle apart funksioneer.

Ten eerste is ORITO 'n rolgebaseerde inligtingsekerheidsmodel. Die voordele van rolgebaseerde inligtingsekerheid is in hoofstuk 2 bespreek en hoofsaaklik bied dit makliker bestuur van inligtingsekerheid. Wanneer rolgebaseerde inligtingsekerheid gebruik word, is daar gewoonlik minder kundigheid nodig vir die bestuur van inligtingsekerheid in 'n organisasie.

ORITO implementeer rolgebaseerde inligtingsekerheid op 'n objek-georiënteerde wyse. Deur dit te doen word die voordele van objek-georiënteerdheid benut soos in hoofstuk 3 bespreek is. Een van die grootste voordele is dat nuwe rolle geskep kan word uit bestaande rolle deur oorerwing toe te pas op rolprofielobjekte en sodoende 'n rol te skep waarvan die voorregte die kombinasie van ouer-rolle se voorregte is. ORITO maak dit vir bestuur makliker om rolle te skep en aan gebruikers toe te ken deurdat die verwantskap tussen rolle met 'n traliegrafiek voorgestel word. Die gebruik van traliegrafieke het ook ander voordele wat in hoofstuk 4 beskryf is.

Enige model is gewoonlik baie teoreties en dit is moeilik om die gebruike en voordele van so 'n teoretiese model te beskryf. Om hierdie probleem te oorkom is ORITO deurgaans bespreek uit die oogpunt van transaksieverwerking (hoofstuk 5). ORITO is dus 'n model vir die magtiging van die uitvoer van transaksies. Hierdie benadering maak dit maklik om die voordele van ORITO te beskryf deur te verwys na transaksies as die hulpbron wat beskerm word deur ORITO.

In hoofstuk 6 is ORITO formeel beskryf. ORITO is 'n model vir inligtingsekerheid (magtig die uitvoer van transaksies) en maak die bestuur van inligtingsekerheid aansienlik makliker.

Met die vinnige groei van datanetwerke is dit nodig om ORITO uit te brei na 'n verspreide omgewing. In hoofstuk 7 word getoon hoe ORITO die uitvoer van transaksies kan doen in 'n kliënt/bediener omgewing. ORITO word geïmplementeer as 'n diens wat 'n sekerheidsbediener lewer.

Deur ORITO op 'n verspreide wyse te implementeer is dit moontlik om doeltreffendheid en betroubaarheid (ten opsigte van inligtingsekerheid) te verhoog. In hoofstuk 8 is getoon hoe ORITO inligtingsekerheid verpreid doen deur rolprofielobjekte te versprei tussen bediener rekenaars.

Ten laaste is in hoofstuk 9 getoon hoe ORITO nog meer betroubaarheid en doeltreffendheid kan bied deurdat bedienders mekaar se sekerheidsdienste tydelik kan oorneem wanneer een of meer bedieners buite werking raak.

10.3 Selfevaluering

In hierdie gedeelte word 'n evaluering van die navorsing deur die outeur gegee. Die bydra, uit die skrywer se oogpunt, van die verhandeling tot die vakkennis word bespreek.

Die belangrikste bydra van hierdie verhandeling en die navorsing daar agter is dat 'n nuwe model vir magtiging in 'n verspreide kliënt/bediener omgewing ontwikkel is. Die

model is uniek in 'n paar opsigte wat vervolgens bespreek word. Hierdie model vergemaklik die bestuur van inligtingsekerheid.

Die model implementeer rol-gebaseerde inligtingsekerheid op 'n objek-georiënteerde wyse as 'n eksterne sekerheidsmonitor in 'n transaksieverwerking omgewing. In die verhandeling word getoon hoe rolle as objekte geïmplementeer kan word. Die klasse waaruit uit rolobjekte geskep word is ontwerp en geformuleer.

Die model wys hoe nuwe rolle geskep kan word uit bestaande rolle deur oorerwing op rolobjekte toe te pas. In [1] word rolverwantskappe met 'n traliegrafiek voorgestel. In hierdie studie is gewys hoe so 'n traliegrafiek gebruik kan word om die rolobjekte in 'n stelsel te versprei tussen bedieners in 'n kliënt/bediener omgewing. Deur laasgenoemde te doen is ORITO 'n model vir verspreide inligtingsekerheid. Verspreide inligtingsekerheid word nie so algemeen in die literatuur bespreek nie. Daar word baie klem geplaas op inligtingsekerheid in verspreide omgewings maar min aandag word geskenk aan die idee om die inligtingsekerheid self verspreid te hanteer. Hierdie verhandeling het by wyse van die nuwe model, ORITO, getoon hoe inligtingsekerheid self verspreid in 'n kliënt/bediener omgewing gedoen kan word. Die voordele van so 'n benadering is in hoofstuk 8 bespreek.

Die navorsing se toepaslikheid word in die verhandeling getoon deur te wys dat ORITO geïmplementeer kan word as 'n eksterne sekerheidsmonitor vir 'n transaksieverwerker. Die model is dus nie net teoreties nie maar kan werklik prakties gebruik word.

Die navorsing het reeds 'n bydrae gelewer tot die vakkennis by wyse van 'n aanbieding van die model deur die outeur by 'n internasionale konferensie (IT Sicherheit '95, Oostenryk, September 1995) asook 'n artikel wat vir aanbieding by die IFIP/SEC '96 konferensie oorweeg word.

Opsommend bied die navorsing 'n nuwe model vir inligtingsekerheid wat die bestuur van inligtingsekerheid aansienlik vergemaklik.

10.4 Toekomstige navorsing

Die resultate wat bereik is met die navorsing wat in hierdie verhandeling gedokumenteer is, laat ruimte vir verdere navorsing. Dit word vervolgens bespreek.

10.4.1 'n Metodiek vir die implementering van ORITO

Dit kan nuttig wees om 'n metodiek vir die implementering van ORITO in 'n werklike situasie te doen. Die aspekte wat in ag geneem moet word voordat ORITO geïmplementeer kan word kan bestudeer word en 'n stapsgewyse metode vir die implementering van ORITO kan opgestel word.

10.4.2 Tegniese oorwegings by implementering van ORITO

Tot dusver het ons nie aandag gegee oor hoe die objekte wat rolprofiele voorstel en hanteer (die magtiging ens.) geimplementeer word op die bediener rekenaar nie. Ons verwys dus na die objek in die stelsel wat 'n rol voorstel en gebruik die rolprofielobjek se lidfunksies sonder om bekommerd te wees oor hoe hierdie rolprofiel geïmplemteer is nie (dalk as 'n werklike kode objek in die geheue ruimte van 'n program/proses op

die bediener, dalk bloot 'n aantal attribute wat in beskermde teks lêers gestoor is, ens.). In verdere navorsing kan die moontlike opsies vir die implementasie van rolprofielobjekte in 'n stelsel bespreek word.

10.4.3 Implementering van ORITO

Om die voordele en beperkings van ORITO nog verder te bestudeer en moontlike verdere aanpassings te doen kan dit nuttig wees om die model te implementeer in verspreide 'n rekenaarstelsel.

CICS/6000 en CICS for OS/2 bied albei die opsie om 'n eksterne program te gebruik vir die magtiging van die uitvoer van transaksies [8, 10]. Hierdie transaksieverwerkers benut ook reeds verspreide verwerking en ORITO sal moontlik verspreid geïmplementeer kan word in sulke omgewings.

Let op dat 'n rolgebaseerde inligtingsekerheidstelsel reeds as deel van hierdie en vorige navorsing geïmplementeer is. Die implementasie is gedoen op 'n CICS/6000 transaksieverwerker. Objek-georiënteerheid of verspreide verwerking is egter nog nie geïmplementeer nie.

10.4.4 Verspreiding van rolprofielobjekte sonder verhoging in koste

Die verspreiding van objekte in 'n rekenaarnetwerk word op die oomblik nagevors deur verskeie navorsers en instansies [8, 10, 17]. Soos in hoofstuk 9 genoem is, kan die verspreiding van rolprofielobjekte betroubaarheid verhoog maar instandhouding van die bedieners raak ook moeiliker. Die verspreiding van rolprofielobjekte sonder beduidende toename in koste kan nagevors word.

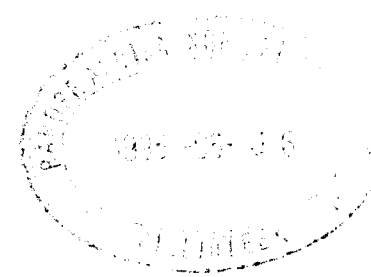
10.4.5 Vergelyking en kombinering van ORITO met ander modelle

Die vergelyking van ORITO met ander modelle vir inligtingsekerheid (soos byvoorbeeld MoRP [2]) asook die kombinering van ORITO met ander modelle (soos byvoorbeeld PCM [3]) kan 'n interessante studie wees.

10.5 Slot

Hierdie hoofstuk sluit die verhandeling af met 'n samevatting van die verhandeling en 'n evaluering van die model. Daar is gewys op moontlike toekomstige navorsing wat gedoen kan word as 'n voorsetting van die navorsing wat in hierdie verhandeling gedokumenteer is.

Ten laaste, die model wat in hierdie verhandeling geformuleer is bied hoop vir die sekerheidsbestuur-probleem waarmee baie organisies op die oomblik worstel. Indien ORITO geïmplementeer en verder nagevors word kan dit baie probleme in inligtingsekerheid, veral die bestuur daarvan, aanspreek.



Bronnelys

[1] F. Rabitti, E.Bertino, W.Kim en D. Woelk,
A Model of Authorization for Next-Generation Database systems, ACM Transactions on Database Systems, Vol 16, Nr. 1, (1991), 88 - 131.

[2] S.H.von Solms en I. van der Merwe,
The Management of computer security profiles using a role-oriented approach, Computers & Security, 13 (1994), 673-680.

[3] W.H. Boshoff en S.H. von Solms,
A Path Context Model for addressing security in potentially non-secure environments, Computers & Security, 8 (5) (1989).

[4] R. Ahuja, T. Magnanti, J. Orlin,
Network Flows , Theory, Algorithms and Applications, Prentice Hall, 1993, 24 - 27 en 73 - 75

[5] M. Nyanchama en S.L. Osborn,
Access Rights Administration in Role-Based Security Systems, Database Security VIII Status and Prospects, Proceedings of the IFIP WG11.3 Working Conference on Database Security, 1994, 37 - 56.

[6] M. Nyanchama en S.L. Osborn,
Modeling Mandatory Access Control in Role-Based Security Systems, Database Security IX Status and Prospects, Proceedings of the IFIP WG11.3 Working Conference on Database Security, 1995, 143 - 159.

[7] L.G. Lawrence,
The Role of Roles, Computers & Security, 12 (1993), 15-21.

[8] I. M. Symonds,
Security in Distributed and Client/Server Systems - A Management View, Computers & Security, 13 (1994), 473- 480.

[9] R Orfali, D Harkey,
Client/Server Survival Guide with OS/2, Van Nostrand Reinhold, An International Thomson Publishing Company, 1994 .

[10] M Ozsu, P Valdurez,
Principles of Distributed Database Systems, Prentice Hall International Editions, 1991, 494 - 514.

[11] IBM UK Laboratories Information Development,
IBM AIX CICS/6000 Customization and Operation release 1, International Business Machines Corporation, 1993.

[12] C. P. Pfleeger,
Security in Computing, Prentice Hall International Editions, 1989, 89 - 103.

[13] G, Chartrand en O. R. Oelermann,
Applied and Algorithmic Graph Theory, McGraw-Hill International Editions, 1993, 99 - 128.

[14] P. Coad en E. Yourdon,
Object-oriented analysis, Prentice-Hall Inc., 1991, 79 - 93.

[15] R Sandhu,
Call for participation, First ACM Workshop on Role-based Access Control, ACM Workshop call for participation, 1995, 1-2.

[16] IBM International Technical Support Center,
CICS/6000 and AIX OLTP T3 Workshop, An IBM Workshop document, 1993.

[17] V Varadharajan en S Black,
Multilevel Security in a Distributed Object-Oriented System, Computers & Security, 10 (1991), 51-68.

[18] M. D. Millikin,
DCE: Building the Distributed Future, BYTE magazine, 1994.

[19] N. Barkakati,
The Waite Group's Turbo C++ Bible, SAMS publishing, 1990, 29 - 71