To appear at ACM SACMAT 2002, Monterey, CA

Observations on the Role Life-Cycle in the Context of Enterprise Security Management

Axel Kern & Martin Kuhlmann
Systor GmbH & Co. KG
Hermann-Heinrich-Gossen-Str. 3
50858 Cologne, Germany

{axel.kern|martin.kuhlmann}@systor.com

Andreas Schaad & Jonathan Moffett
Department of Computer Science
University of York
York, YO10 5DD, UK

{andreas|jdm}@cs.york.ac.uk

ABSTRACT

Roles are a powerful and policy neutral concept for facilitating distributed systems management and enforcing access control. Models which are now subject to becoming a standard have been proposed and much work on extensions to these models has been done over the last years as documented in the recent RBAC/SACMAT workshops. When looking at these extensions we can often observe that they concentrate on a particular stage in the life of a role.

We investigate how these extensions fit into a more general theoretical framework in order to give practitioners a starting point from which to develop role-based systems.

We believe that the life-cycle of a role could be seen as the basis for such a framework and we provide an initial discussion on such a role life-cycle, based on our experiences and observations in enterprise security management. We propose a life-cycle model that is based on an iterative-incremental process similar to those found in the area of software development.

General Terms

Security, Access Control, System Management

Keywords

Security Administration Manager (SAM), role life-cycle, role management, role engineering, security administration, distributed system management, enterprise user administration (EUA), process model, enterprise security management (ESM)

1 INTRODUCTION

Over the last few years, a number of large organisations have introduced role concepts for enterprise security management. These concepts rely on roles which are defined at the enterprise level. Our practical experience, as well as the recent research on roles as a system management concept, indicates that roles evolve according to a life-cycle model, similar to those found in the area of software engineering process models.

When a new role-based system is introduced within an organisation, we are rarely able to immediately identify and transform ‘natural’ organisational roles into roles as they are required by the system. *Analysis* of the organisational structures and further *design* of the identified roles allow us to transform such roles for use in the enterprise role repository and in the target systems. Once the roles are part of our system, role *management* activities comprise the daily administrative duties. As any information system is subject to a continuous organisational change process, role *maintenance* is another vital part of the life-cycle of a role.

Steady progress has been made over the last years in establishing a framework for roles in the general area of distributed systems management with particular progress in the area of role-based access control. However, no life-cycle model has yet been identified, nor have the theoretical and practical benefits that would arise out of such an approach been discussed.

2 MOTIVATION

We believe that our work on the life-cycle of roles provides researchers and practitioners alike with a better understanding of the technical and organisational dimensions of roles beyond the traditional notion of being an intermediate construct to relate users and permissions.

While the importance of roles for security management and access control has been recognised, research has so far neglected the use of rigorous software engineering techniques to support the development and management of role-based systems. This aspect is even more important when considering the complexity of the systems and environments in which roles are employed.

The motivation to initiate the discussion on a role life-cycle particularly includes:

- A role life-cycle provides a framework for identifying the connection between existing and future work on roles.
- A role life-cycle supports a structured process for the development of role-based systems.
- A role life-cycle supports the efficient employment of roles within the context of organisational change.
- A role life-cycle is the basis for role engineering.

The discussion on such a role life-cycle model is based on the following two main points:

- A role life-cycle must be sufficiently abstract to acknowledge a role as being a context-dependent construct with varying semantics.
- When establishing a role life-cycle the (re)-use of techniques from the discipline of software engineering needs to be investigated.

3 OUTLINE

Evidence for the existence of a role life-cycle comes from the research and industrial areas. We provide a review of selected theoretical work on roles (Section 4.1) which indicates certain stages of the life-cycle. This is followed by a description of the process model that we will later adopt for our purposes (Section 4.2). The industrial enterprise user administration tool SAM (Security Administration Manager) which uses roles as one concept for facilitating security management is then described (Section 4.3). This is followed by a more detailed discussion on enterprise roles (Section 4.4).

Before we identify an initial role life-cycle, we discuss how and why our practical experience with developing and configuring role-based systems influenced this paper (Section 5). We specifically address the complexity of real-world systems and present our proven implementation method (Section 5.1); industrial goals and challenges (Section 5.2); and the importance of structured role engineering (Section 5.3). We then discuss the stages of a simple role life-cycle model (Section 6) based on the process model described previously. Specific emphasis will be put on the life-cycle phase of role design and our practical experiences (Section 6.2), before we conclude our paper.

4 Related Work

4.1 The Role Life-Cycle

Although the concept of a role has been used by researchers [1], [2] and practitioners [3] for many years, the establishment of the RBAC96 model [4] and subsequent RBAC workshops provided an initial basis for the research area of access control models and technologies. It is within this context that some work has already indirectly described the stages of a role life-cycle.

A methodical approach to determine role rights from use cases is described in [6]. All use cases for a specific system are established and role rights are determined from this collection, thus following the principle of least privilege.

In [7] a UML based approach to Role Engineering is described. Role Engineering is seen as the discipline to develop RBAC components such as role hierarchies, permissions and their respective assignment, and constraints. The applicability of the Unified Modelling Language (UML) for these activities is described in the context of the Role Based Access Control Framework for Network Enterprises [8].

A process-oriented approach for role-finding is discussed in [9]. It follows a top-down approach deriving business roles from enterprise process models. This approach is described in the context of a more general case-study outlining the experiences of finding, implementing and changing roles in a large industrial organisation.

The need for supporting role-based access control management in rapidly changing heterogeneous environments is outlined in [10]. The implementation of a role-based access control maintenance system is described. This system is mostly target independent and thus allows for efficient adoption to new access control requirements as they result out of continuous organisational change.

Considering the concept of roles in distributed systems management, Lupu [5] describes the life-cycle of a role in the context of a state machine specification following a simple DORMANT / DISABLED / ENABLED / DELETED pattern. In this context the actual state depends on whether a manager subject has been assigned to that role and must thus not be confused with the RBAC96 session concept.

4.2 Software Life-Cycle and Process Models

It is important to understand that any software process model can only illustrate a certain perspective on a development process. This perspective could be technical, managerial or dependent on some other paradigm and for reasons of space we will only be able to present a limited review on process models.

Within the area of software engineering, a ‘life-cycle’ is an abstract description of the structured and methodical development and modification process of a piece of software [11]. The term ‘process model’ is often used synonymously and some of the most well-known representatives are the waterfall, V, and spiral model [12], [13]. More recently, new process models based on advances made in the area of object-oriented analysis and design have been developed and provide an integrated approach to software development [14].

Traditional approaches like the waterfall model consist of the following steps:

- Requirements Analysis
- Design
- Implementation
- Testing
- Maintenance

These steps are processed in a sequential order. The main assumption is that it is possible to define the requirements at the beginning of the project and that they will be more or less stable over development time. If requirements change or new ones occur, the development team has to go back to the analysis phase and perform the subsequent steps again. The same is true for errors, if an error is found the team has to go back to the step where its origin lies (e.g. design step for design errors which are found during testing).

One main drawback of traditional approaches to software development is that they aim at developing a ‘complete’ product. As a result, the development process will take a long time before the product is released. This viewpoint does not reflect the pressure of the market to deliver results as quickly as possible. It has also been realised that in many cases product requirements are only uncovered with the actual employment of the product itself. Furthermore, the requirements will normally change during development so that the assumption of a complete and stable requirements analysis at the beginning of the project becomes obsolete.

Modern software development therefore uses an incremental development process [15] (Figure 1). After an initial analysis a product model is defined which is as complete as possible. This basic model is then divided into several parts. For each of these parts the design, implementation and test phases are performed, resulting in a partial, but functionally complete product which can then be deployed. The advantages of this process are twofold:

- The software is already used in production with a partial functionality after a shorter time period, providing much earlier return on investment.
- The users of the product can provide real feedback which can already be integrated in the next development increments, after using the first versions of the product

After the software is in production the software process is supplemented by the maintenance phase thus leading to a complete software life-cycle.

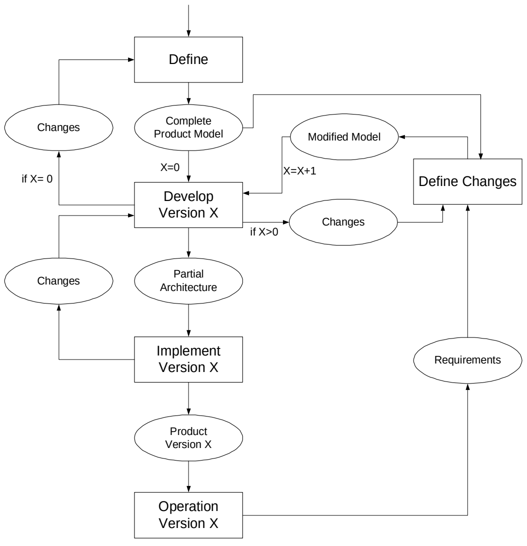

Figure 1: Incremental Process Model (as in [15])

4.3 The Security Administration Manager

The Security Administration Manager (SAM) is an Enterprise User Administration (EUA) product developed by Systor. It provides a central access rights repository and a unified management of users and resources in distributed heterogeneous systems. SAM specifically makes use of the role concept. The general working of SAM has been described in a previous paper [16] and we only provide a brief summary of the general context of commercial security administration and the role of a product like SAM.

The general notion of SAM is to manage different systems from a single point of control. In order to do so SAM is based on a “manager of managers” architecture. Interfaces to the specific systems and applications allow for consolidating information in a system-independent conceptual model.

The administration of users and their access rights in the IT environment of medium and large companies is a complex and expensive task. Most companies operate a large number of applications running on several different operating systems. According to Gartner Group, the number and variety of platforms continues to grow in most enterprises. As most applications and platforms have their own administration product, this results in the need for increased platform-specific know-how. When these systems are connected to SAM, administrators work only with the SAM environment. This not only consolidates the administration work but also reduces the need for in-depth knowledge about all underlying systems.

To further reduce the EUA cost, most enterprises want to automate the administration. The most accurate information about its employees can often be found in the human resources database. Extracted information like employee number, organisational unit, location or job description can be used to automatically add and delete users as well as update their roles. If a new employee starts with the company or changes his job within the company, this information is transferred directly from the human resources database to SAM, which automatically transforms the information to role assignments and makes the corresponding updates in the connected target systems. In addition, when an employee leaves the company, all accounts and access rights of this employee are automatically deleted, thus greatly reducing security risks (See also figure 2).

A prerequisite for this automation is the usage of roles that correspond to organisational structures, job descriptions etc. (See section 6).

4.4 Enterprise Roles

The IT environment of large enterprises consists of a variety of platforms and applications. These include a number of operating systems (e.g. OS/390, Windows NT/2000 and UNIX), databases (e.g. Oracle), SAP’s R/3 and a large number of individual business applications. Most of these components have a built-in security component (like Windows NT or Oracle) or are secured by an external security product (like RACF for OS/390). The mechanisms used by these security components are quite different. Some of them have already introduced roles.

A typical enterprise user must have access to a variety of applications on different platforms. However, current existing role concepts are mainly application and operating system specific, causing overheads in the administration. For example, when accessing an application, a user can have parallel sessions at different layers such as Windows NT, OS/390 and a database, requiring access rights for all of them. These rights have to be administered separately in the participating systems as currently no common RBAC support is implemented. There are currently some developments to broaden the scope of operating system security. Operating systems like Netware and Windows 2000 allow administration of other systems and applications using their directories. However, those approaches are restricted to only a subset of the systems operated by a typical enterprise; especially the support of mainframe and midrange systems is normally very restricted.

Because of the variety of different systems on the market and their dynamic behaviour this extended administration support will not be comprehensive enough in the near future. The EUA product SAM helps to mitigate the situation by introducing *Enterprise Roles*. In the SAM role concept of Enterprise Roles, roles span over more than one target system.

Enterprise Roles in SAM consist of permissions in one or more target systems. These permissions are specific to the target system and can be of various natures (See example in figure 3).

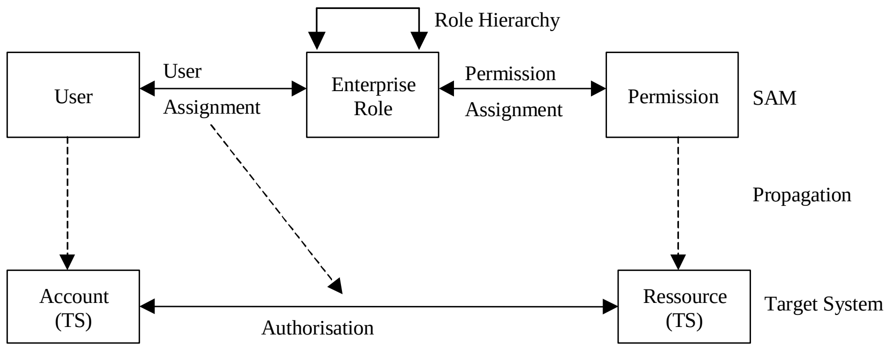

Figure 2: Enterprise RBAC Model (ERBAC)

There may be groups of authorisations like a group in UNIX; an authorisation to a specific resource like the access right to a shared resource in Windows NT; the read permission of a data set in RACF; or even a role like in Oracle. Enterprise Roles can also contain other enterprise roles thus inheriting their permissions. In contrast to the administered target systems, SAM – like the other EUA products – does not actively take part in authentication and authorisation of a user. Instead, it represents a layer above the existing security systems and controls them.

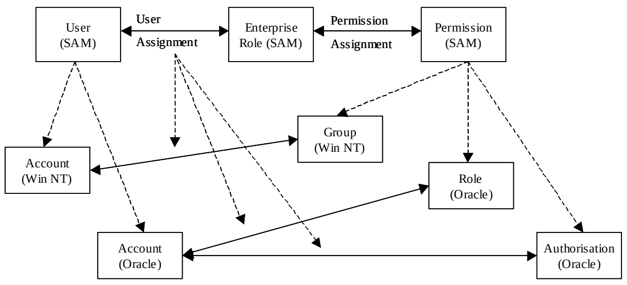

Fig. 3: Enterprise Role Example

Figure 2 shows the resulting model which we call Enterprise RBAC model (ERBAC). It is based on RBAC96 [4] and the NIST role standard draft [17]. The basic ERBAC model is analogous to core RBAC. Enterprise roles collect all permissions which are needed to perform a specific role. Users are assigned to these roles. The difference between ERBAC and the RBAC96 lies in the notion of sessions. EUA tools like SAM administer all systems of an enterprise but do not control the actual sessions of users. Therefore sessions can not be part of ERBAC. Instead, the definitions in SAM are propagated to the administered systems which we call "target system" (TS). The Enterprise user definition in SAM leads to the creation of user accounts in the TS. A permission can be any authorisation (called operation in core RBAC) to a resource in one of the underlying TS. The definition of a permission or the assignment of a permission to a role does not lead to any update in the TS. Only when a role is assigned to a user, the permissions of the role are propagated and the accounts of the user get the associated permissions in the respective TS.

In addition to the core RBAC features, a general role hierarchy [4] is supported. Enterprise roles can be assigned to other roles in a directed acyclic graph. Child roles inherit all permissions from their parent roles (including all permissions these roles inherit). A user who is assigned to a child role thus gets all permissions assigned to this role plus all permissions which this role inherits from its ancestors. Role hierarchies allow structuring roles in an easy manner and reduce redundancy. This leads to a smaller number of roles to be defined in an enterprise and less administrative effort.

Of course, it is possible to enhance ERBAC by defining constraints (e.g. separation of duty) but this topic can not be addressed in the scope of this paper.

5 Enterprise Role Implementation in the "Real World"

As shown in section 4.1, the notion of roles and their advantages are now quite well understood in the research community. An increasing number of new applications are building its access control mechanism on roles. The overall usage of roles, however, is not as widespread as it might be assumed. Based on our experience during the development of the Security Administration Manager (SAM) and its implementation within some large enterprises, we emphasise that defining and implementing roles is by no means trivial and calls for the rigorous usage of a role engineering process.

5.1 Implementation Method

In this section we briefly describe a proven method for the implementation of enterprise roles in organisations which employ from 10,000 to more than 100,000 users.

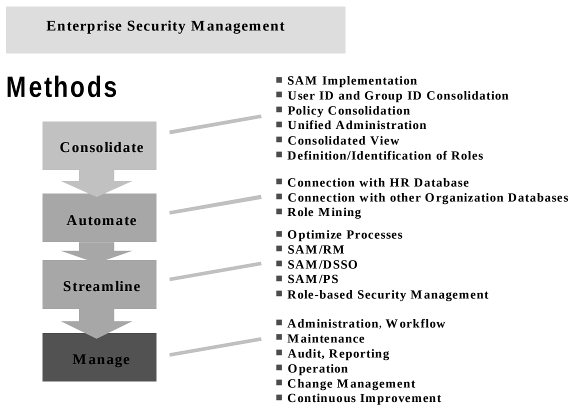

Figure 4: Implementation Method

We relate this method to the role life cycle and divide the implementation into four steps: Consolidation, automation, streamlining and management (Figure 4).

One element of the consolidation step is *role analysis* which is discussed in section 6.1. Parallel to that, the role administration software is implemented and the basic consolidation tasks are carried out: the user and group IDs in the different systems which will be connected with the enterprise roles are harmonised where necessary, the security policies (e.g. password policies) are consolidated, the administration is unified. The auditor gets a consolidated view of the organisation's access rights.

The second phase is dedicated to automation. This involves *role design* (see section 6.2) and the connection of the administration tool with the HR database and other organisational databases.

The streamlining step comprises the roll-out of the role-based administration in the organisation and the possible introduction of additional tools for the optimisation of administration, like a password synchronisation tool, a single-sign-on tool or an administration workflow system.

The last management phase stands for the operation, management and maintenance of the introduced system. This includes the establishment of regular reporting and auditing structures as well as the implementation of change management processes for software and the data. Consequently, the *role management* and *maintenance* steps of the role life cycle belong here (see sections 6.3 and 6.4).

To further illustrate the dimensions and types of organisations our method was applied in, we provide more figures on the number of roles and users of some of the customers of Systor in the following table (Figure 5). These numbers are consistent with the numbers given in [3].

Organisation Type	Figures
Insurance company A	17000 users 120 roles
Bank A	30000 users (user has on average 10 roles) 400 roles
Bank B	31000 users 450 roles (1 role is assigned to 70 users on average)

Figure 5: User/Role Figures

5.2 Goals and Challenges

The main goals that companies want to achieve in security administration are enhanced security at a low cost, while coping with the challenges of an increasingly complex IT environment. A high level of security is important to prevent possible losses through fraud and unauthorised disclosure of confidential information. Internal audits often encounter severe security weaknesses which oblige the IT security department to take appropriate countermeasures.

An increasingly dynamic economy is an additional challenge within this context. The most important issues here are the rapid growth of companies, large-scale mergers, the ubiquity of the internet and continuous organisational change:

- As most companies intensify their business of the internet, they are dealing with very large numbers of users. A large insurance company, for example, estimates the number of customers who will have access to its IT resources via the internet during the next three years at some 20 million. Of course, such numbers cannot be administered manually.
- During the past years and most likely also in the near future a lot of mergers take place between large enterprises. Such mergers normally involve the integration of both different IT infrastructures and different organisational structures, thus increasing their complexity and their user population.

However, the primary goal of any public sector enterprise today is to reduce its operational costs. Thus, in order to enhance the security level while dealing with the aforementioned challenges, the productivity of security management must be increased dramatically. We see a rigorous role engineering process, as described in the next sections, as a good way to reach this goal.

5.3 Role Engineering

We believe that an efficient enterprise security management is based on:

- the usage of enterprise roles,
- automation of enterprise user administration (EUA) using an appropriate tool,
- a well-structured role engineering process.

The concept of a role helps to bridge the gap between the technical and the business side of security administration. The specialists in the IT department can group permissions with roles which represent all the authorisations that are needed for a specific job or are connected to an organisational unit. Given this abstraction, the actual role assignment to users can be done by business people who understand much better which person performs which job. It also dramatically reduces the need for system specific know-how and allows the decentralisation of user administration to the business units thus reducing the need (and cost) for a large central administration group. Role-based access control management software takes over the task of transforming business language (i.e. roles) into technical language (i.e. access rights of accounts to IT resources).

As pointed out in the previous section (5.1), the best way to increase the productivity of security administration is automation, e.g. by using data from the HR database. Prerequisites for this automation are:

- The data of the HR database must be accurate and maintained on time. Users who are not employees (e.g. consultants) will probably not be defined in the HR system. Thus, additional databases need to be accessed for obtaining the relevant information.
- Enterprise-wide "business roles" must be defined and implemented which can be unequivocally mapped to jobs, organisational units etc.
- An EUA tool like SAM must be implemented so that the roles can be propagated to all target systems.

An automatic process can then extract all new and leaving users, as well as changes of job or organisation, from the HR or other databases, compute the corresponding assignments or de-assignments of roles to/from the corresponding users, and communicate this information to SAM which in turn updates the actual target systems.

However, roles do not appear out of nowhere. To obtain roles which fit the business requirements of the enterprise and support easier administration, a thorough analysis and design of the role structure is required before implementing roles. This has to be done in a well-structured role engineering process which must be based on the *life-cycle* of a role. Such a life-cycle is similar to well-known iterative-incremental software engineering processes.

6 The Role Life-Cycle

The life-cycle of a role provides an abstract description of the structured, methodical development, modification and maintenance of roles in role-based systems.

The role life-cycle as we perceive it, is based on our previous literature analysis and the practical work with roles during the implementation of the Security Administration Manager (SAM) in large enterprises.

The structure of our role life-cycle is shown in figure 6. We identified the four stages of:

- role analysis,
- role design,
- role management and
- role maintenance.

The stages are described in detail in the following sections. The arrows between the process steps indicate an iterative process model that allows us to move forwards and backwards from any stage.

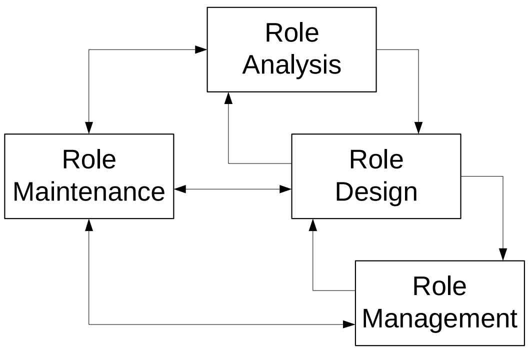

Figure 6: The Role Life-Cycle

According to the iterative-incremental development process as described in section 4.2, these steps are not performed at once in a sequential order. Instead, after having defined the basic role structure, analysis and design will be performed incrementally in cycles for different areas (e.g. a specific organisational unit). Normally one area will be designated for a prototypical cycle which is used to show the validity of the design so that corrections can be made before the complete deployment of roles in the whole enterprise. Afterwards, additional development cycles will be performed for the other areas. This iterative-incremental process has the following advantages:

- The design is validated early after implementation in a small part of the enterprise so that in case of problems it is not necessary to redesign the complete enterprise role structure.
- Having implemented roles for one area, these can already be used productively so that easier administration and automation lead to a return on investment at an early stage in the role development process.

Although the role life-cycle we describe is partly based on our experience with SAM, it is not dependant on a specific product or system environment, but can be used for any role development process.

6.1 Role Analysis

Role analysis is the activity of identifying roles as they occur within the target domain in which the system will be placed. During the role identification process, several points are essential to achieve a robust and practical role model:

- The basic prerequisite for enterprise roles is the possibility to describe jobs and tasks which require access rights to IT resources by formal criteria, using structural information about the organisation.
- The structure of the role hierarchy reflects ”natural” organisational structures of the enterprise. Examples from large implementations are a cost centre structure, a departmental structure or a geographic structure. The chosen structures should be relatively constant over time. The consequence of frequent structural changes would be increased role management effort.
- The role analysis process should ideally be a mixed bottom-up and top-down approach. A top-down approach was described in [9]. A bottom-up approach means that, starting from the total set of permissions which exist in the organisation, one tries to find clusters of permissions which represent roles. While a top-down approach ignores existing permissions, a pure bottom-up approach would not take organisational structures into account.
- Complexity has to be balanced. Administrators have to be able to master the system. Too high complexity can result in lower security due to a lack of control. On the other hand, the role model has to be granular enough to ensure that all users get access to a set of resources which fits as exactly as possible to their position in the organisation. However, a compromise is sometimes unavoidable in which the principle of least privilege is broken: there are large implementations where the set of permissions granted by the user’s standard role is usually bigger than the smallest set of permissions that would be sufficient for job execution. If, for example, a bottom-up role finding process is applied, permission clusters will rarely map exactly 100% of the permissions of a group of users with the same or a similar task profile.

Role analysis is performed by domain specialists with explicit knowledge about the organisational aspects of a role as well as by the system engineers.

6.2 Role Design

While role analysis is mainly targeted at acquiring knowledge about the current organisational context of a role, role design has to convert this knowledge into concepts that can be used by the later system. This process includes the mapping of roles and design of roles for later administration.

6.2.1 Mapping Roles

For the implementation of a role model, the organisational structures which reflect the roles have to be mapped onto the syntax and semantics required by the enterprise role administration tool.

We take the most common approach in which the role hierarchy is a directed acyclic graph. Accordingly, the mapping process can be divided into the following steps:

- Identification of the organisational structure(s) which (parts of) the role model should represent. The ground work for this should have been done during role analysis.
- Definition of an (injective) mapping from each structure to the role model which maps the organisational structure onto roles and role-role relationships.
- Definition of a user-role relation with the help of user attributes attached to the user, for example in a digital certificate or in the HR database of the organisation. Such user attributes could be the user's location or job code. If for example each branch location of the organisation is represented by a role, then the rule "a user is connected with a role if the user's location corresponds with the location that the role represents" can be part of the user-role relation definition. By this rule, an employee in Munich is connected with the "Munich Branch" role and a Hamburg employee is connected with the "Hamburg Branch" role. This definition of a user-role relation is essential for automated administration of user-role relations.
- Definition of a role-permission relation using permission attributes as described above.
- Specification of an algorithm which transforms all organisational changes to changes in the role model, having the above mentioned mapping as an invariant. This step is required for automated administration; the algorithm provides the description for the automated processing. If, for example, an algorithm for changes in the role-permission relation as a result of organisational changes cannot be given, then this part cannot be automated.

6.2.2 Designing Roles for Administration

The following situations can be observed in practical implementations when roles are designed for later administration.

1. It may be desired that parts of the role model are administered manually, and parts are administered automatically (See figure 7).

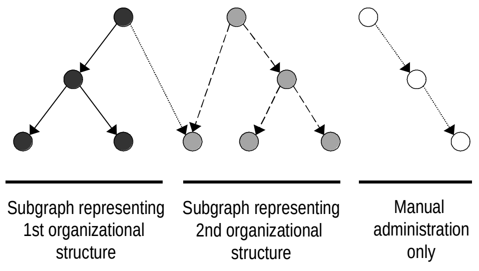
Subgraph representing 1st organizational structure
Subgraph representing 2nd organizational structure
Manual administration only

Figure 7: Manual/Automatic Administration

2. Several organisational structures are part of the role model and are mapped as disjoint subgraphs of the role graph (See figures 8 and 9).

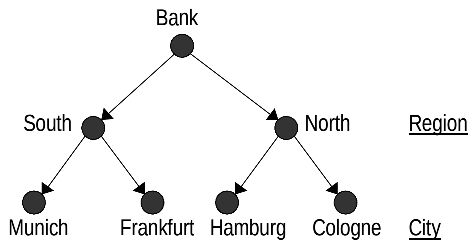

Figure 8: Admin. according to geograph. structure

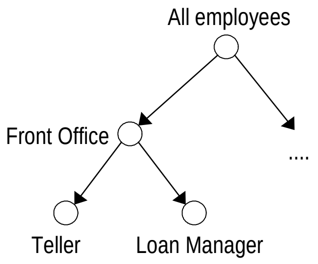

Figure 9: Admin. according to org. structure

3. Information from several organisational structures is needed to define a single role. Example: The access rights of a user depend on the location (i.e. geographic structure) and on his job function; a teller in Munich is allowed to use teller applications to access accounts from Munich customers and is allowed to have access to folders on the Munich branch file server.

It is easy to deal with the first situation. The parts of the model which lie "outside" the mapping(s) mentioned above (i.e. the parts which are not controlled by the transformation algorithm) can be administered manually. The procedure for the second situation is straightforward as well, here each subgraph is treated separately in terms of automation. In the third situation, some problems arise for which we will give solution approaches. For a better illustration, let us take the example of a geographic structure and a job function structure:

If GEO denotes the set of roles in the geographic structure, JOB denotes the set of roles in the job function structure, then we can derive a set of total roles R as a subset of the direct product GEO $\times$ JOB of the two sets.The following problems arise:

- The result is usually a large number of roles (especially if each structure is large or if there are more than two structures).
- It has to be defined how the job function graph and the geographic structure graphs are merged to build a role graph.

For the construction of a merged graph, standard methods from graph theory could be used. However, the resulting graph will be very complex, unless one of the structures is very small. A practical example with one of the structures being small is the merge of a job function graph with a "staff – chief" distinction on the lowest level (See figure 10):

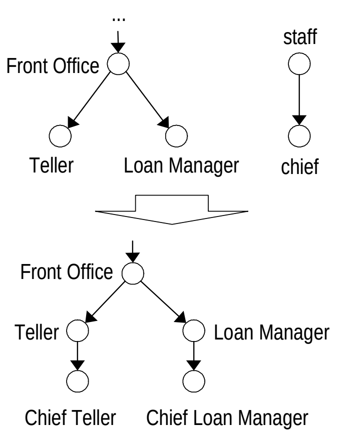

Figure 10: Merge of job functions

Another approach which has proven to be successful avoids increasing complexity. Only one structure is chosen as the role structure and the other structures are used as constraints. As an example we take the job function structure as the role graph and use the geographic limitations as constraints by adding the geographic limitation of a user as an attribute to the user-role relationship. In the figure below, we see that Charles works sometimes in Hamburg City as a teller and sometimes (let us assume in a substitute function) as a loan manager in the Hamburg North branch of a bank (See figure 11).

The disadvantages of this approach are:

- No possibility to assign authorisations to the objects of the "dropped" structures.
- Relations between the objects of the "dropped" structures (i.e. edges of the graphs) are lost.

One special situation is covered by the "Team based access control (TBAC)" model [18]. In TBAC, a situation is described where, in terms of our example, a user carries out the same job in each location.

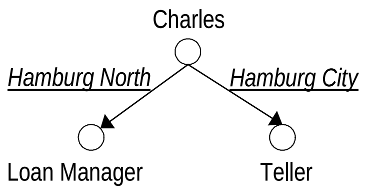

Figure 11: geographic limitations as constraints

Similar to role analysis, role design is performed by the relevant domain specialists, by the system engineers and the later administrators.

6.3 Role Management

By role management we understand the routine role administration within an organisation. Role management builds on an existing role model and requires the role design phase as described above as a prerequisite.

To give a precise definition, we consider as role management the following operations:

1. Changes in the role model as a result of organisational changes, as described by an algorithm that has already been specified in the design phase.
2. Creation or deletion of a user or a permission.
3. Connection/disconnection of a user and a role according to the definition of the user-role relationship in the role design phase.
4. Connection/disconnection of a role and a permission according to the definition of the role-permission relationship in the role design phase.

In practical implementations, the following role operations are of special importance:

a) Split of one role into two roles. This happens if for example a complex assignment is split into two jobs which are carried out by two different persons, or if one cost centre is split into two cost centres.

b) The inverse operation of merging two roles into one role.

These operations can be described easily by standard graph theoretic means.

Thus, role management comprises all activities which are carried out within the designed role concept. Role management leaves the concept untouched. Activities that require a change to the concept relate to role maintenance (Section 6.4).

Role management is performed solely by the system administrators.

6.4 Role Maintenance

Organisations are subject to a continuous change process. The reasons for this change vary and some examples would be mergers, acquisitions or business process re-engineering activities. A role structure which was designed and implemented is unlikely to stay unchanged for a long time.

The correct functioning of information systems often relies on the exact matching of the system's conceptual model with the existing organisational structure. So in the case of an accounting system, departments and projects must be identified. In a similar manner a material planning system will have to reflect the basic steps of a production process.

In the worst case, a system will not be adjustable to the organisational change. These kind of systems are often custom-built and it is not feasible to change their conceptual model.

In the case of enterprise-wide access control administration systems, the use of RBAC models provides a high flexibility and allows for easier response to organisational changes. The adoption of organisational changes which we will discuss as "role maintenance" in this section is the only way to reach an optimal exploitation of the advantages of RBAC.

By Role Maintenance we understand changes in the chosen role concept. Such changes occur if, for example, the geographic structure of an organisation which is used as part of the role hierarchy is at a certain point of time not regarded as useful for this purpose any more.

Role Maintenance activities comprise changes in the mapping of organisational structures to roles and changes in the definition of user-role and role-permission relationships.

At least a partial redesign of the role concept is necessary. But there are considerable differences to the initial role design phase:

1. Roles do not have to be created from the scratch; the existing role concept can be used and a migration strategy to the new role concept may be developed.
2. Parts of the existing role concept may stay untouched.

Depending on the kind of required role maintenance activities, domain experts, system engineers and administrators will be involved.

7 Conclusion

In the course of this paper we have discussed the evolution of roles in role-based systems and provided an initial outline of an abstract iterative-incremental life-cycle model. This discussion was based on existing research on roles and our practical experience with the development and customisation of a specific role-based system called SAM. We focused on providing a more realistic picture of implementing enterprise roles before discussing the stages of the proposed role life-cycle in detail.

We believe that the life-cycle of a role is an important concept that must be further investigated in order to:

a) categorise existing work and identify open issues in the area of role-based research.

b) give practitioners a framework for role-based systems development and maintenance.

Our work is by no means complete and we could only provide an initial motivation for, and coarse description of a role life-cycle model, mainly adapting existing work from the area of software process models. What is needed now is to continue work on a more complete role life cycle model, probably looking at research in the area of software process modelling [19]. In order to be able to validate such a future model we must establish a set of design criteria, which should be ideally supported by a more formal technique such as provided by the E3 language [20].

8 Acknowledgements

The UK Engineering and Physical Sciences Research Council has provided funding for Andreas Schaad under EPSRC Scholarship number 99311141.

9 REFERENCES

[1] Biddle B. and E. Thomas, Role Theory: Concepts and Research. New York: Robert E. Krieger Publishing Company, 1979.

[2] Ferraiolo D. and R. Kuhn, "Role-Based Access Control." presented at 15th NCSC National Computer Security Conference, Baltimore, 1992.

[3] Schaad A., J. Moffett, and J. Jacob, "The access control system of a European bank - a case study." presented at Sixth ACM Symposium on Access Control Models and Technologies (SACMAT), Chantilly, VA, USA, 2001.

[4] Sandhu R., E. Coyne, H. Feinstein, and C. Youman, "Role-based access control models." *IEEE Computer,* vol. 29, no. 2, pp. 38-47, 1996.

[5] Lupu E., "A Role-Based Framework for Distributed Systems Management." PhD Thesis: *Department of Computing*. London, Imperial College, 1998.

[6] Fernandez E. and J. Hawkins, "Determining role rights from use cases." presented at Second ACM Workshop on Role-Based Access Control, Fairfax, VA, USA, 1997.

[7] Epstein P., Sandhu, R., "Towards a UML based approach to role engineering." presented at Fourth ACM Workshop on Role-Based Access Control, Fairfax, VA, USA, 1999.

[8] Thomsen D., D. O'Brian, and J. Bogle, "Role Based Access Control for Network Enterprises." presented at 14th Annual Computer Security Applications Conference, Gaithersburg, MD, USA, 1998.

[9] Roeckle H., G. Schimpf, and R. Weidinger, "Process-Oriented Approach for Role-Finding to Implement Role-Based Security Administration in a Large Industrial Organisation." presented at Fifth ACM Workshop on Role-Based Access Control, Berlin, Germany, 2000.

[10] Moenkeberg A. and R. Rakete, "Three for One: Role-Based Access Control Management in Rapidly Changing Heterogeneous Environments." presented at Fifth ACM Workshop on Role-Based Access Control, Berlin, Germany, 2000.

[11] McDermid J., *Software Engineer's Reference Book*: Butterworth Heinemann, 1990.

[12] Pressman R., *Software Engineering: A Practitioner's Approach*: McGrawHill, 2000.

[13] Sommerville I., *Software Engineering*: Addison-Wesley, 2001.

[14] Booch G., I. Jacobson, and J. Rumbaugh, *The Unified Software Development Process*: Addison Wesley, 1998.

[15] Balzert H., “Lehrbuch der Software-Technik II”, Spektrum Akademischer Verlag, Heidelberg/Berlin, 1998.

[16] Awischus R., "Role based access control with the security administration manager (SAM)." presented at the Second ACM Workshop on Role-Based Access Control, Fairfax, USA, 1997.

[17] Ferraiolo D., R. Sandhu, S. Gavrila, R. Kuhn and R. Chandramouli, "Proposed NIST standard for role-based access control", ACM Transactions on Information and System Security (TISSEC), vol. 4, no. 3, 2001.

[18] Georgiadis C., I. Mavridis, G. Pangalos and R. Thomas, “Flexible Team-Based Access Control Using Contexts” presented at Sixth ACM Symposium on Access Control Models and Technologies (SACMAT), Chantilly, VA, USA, 2001.

[19] Finkelstein A., J. Kramer, and B. Nuseibeh, *Software Process Modelling and Technology*. Taunton, UK: Research Studies Press Ltd., 1994.

[20] Jaccheri M., G. Picco, and P. Lago, "Eliciting Software Process Models with the E3 Language." *ACM Transactions on Software Engineering and Methodology*, vol. 7, pp. 368-410, 1998.