Available at
www.ComputerScienceWeb.com
POWERED BY SCIENCE DIRECT

Information and Software Technology 45 (2003) 979–991

INFORMATION AND SOFTWARE TECHNOLOGY

www.elsevier.com/locate/infsof

Modelling access policies using roles in requirements engineering

Robert Crook*,[1], Darrel Ince, Bashar Nuseibeh

Security Requirements Group, Department of Computing, The Open University, Walton Hall, Milton Keynes MK7 6AA, UK

Received 1 December 2002; revised 3 March 2003; accepted 1 April 2003

Abstract

Pressures are increasing on organisations to take an early and more systematic approach to security. A key to enforcing security is to restrict access to valuable assets. We regard access policies as security requirements that specify such restrictions. Current requirements engineering methods are generally inadequate for eliciting and analysing these types of requirements, because they do not allow complex organisational structures and procedures that underlie policies to be represented adequately. This paper discusses roles and why they are important in the analysis of security. The paper relates roles to organisational theory and how they could be employed to define access policies. A framework is presented, based on these concepts, for analysing access policies.

Keywords: Access policies; Security requirements; Roles

1. Introduction

There is increasing recognition of the importance of requirements engineering (RE) in the formulation and analysis of security policies [1]. The term *policy* is broad but can be interpreted as a rule, which may be of a procedural or technical nature, and is formulated to satisfy a goal. In this paper we focus on policies for data access and how the rules by which data should be accessed can be expressed.

Access is controlled by allocating permissions to users, allowing them to access particular information only. In order to specify access control requirements, an understanding of organisational structures and procedures is needed [12]. While some RE methods incorporate notions of actors or agents, they do not explicitly allow relating actors to organisational groups or modelling of authority, both of which are keys to identifying responsibilities of users with respect to the information assets that are to be protected.

The ideas presented in this paper have emerged from a project within the Security Requirements Group at the Open University. The objectives of this work are to

1. Identify concepts and organisational principles that underlie access restrictions and authorisation procedures.
2. Develop an analytical framework based on the above concepts for specifying and analysing these types of requirements.
3. Explore how this framework could be integrated into existing RE methods such as goal-based analysis.
4. Validate the framework with a set of case studies, either using real life projects or analysing case studies from the literature.

In this paper, we report on the outcome of the first two objectives. The paper is organised as follows. In Section 2 we discuss the importance of roles for analysing security goals and policies, the weaknesses of current RE methods, and examine contributions from the policy and access control research community as to how roles can be used to formulate access policies. In Section 3 we relate this research to organisational theory, and identify categories of roles and how they may be derived from an organisational structure. In Section 4 we introduce a framework for specifying and analysing role-based access restrictions, which includes a graphical notation, and illustrate its use on a healthcare example. We conclude the paper with a short summary and a discussion of future work.

* Corresponding author. Tel.: +44-1908-274-066; fax: +44-1908-653-744.

E-mail addresses: robert.crook@t-online.de (R. Crook), d.c.ince@open.ac.uk (D. Ince), b.a.nuseibeh@open.ac.uk (B. Nuseibeh).

[1] Robert Crook is also an independent consultant.

0950-5849/03/$ - see front matter © 2003 Elsevier B.V. All rights reserved.
doi:10.1016/S0950-5849(03)00097-1

2. Roles in Security Requirements Engineering

2.1. Security goals

Information security is concerned with safeguarding information assets. The reason we wish to safeguard these information assets is to protect the individuals or organisations that depend on them. When formulating the requirements of a system, a set of functional requirements will arise. The functions that we define also bring with them dangers. In a banking application, the function to transfer money could be abused, and we would want to ensure that criminals do not transfer money out of a customer's account. At the requirements level we need to consider these threats and the goals that we define, in order to express the desire to prevent such untoward events. In particular, we want to ensure that the behaviour of users does not compromise security goals. There are many ways in which users can compromise security. Users can potentially corrupt data, delete it, access information they should not, and distribute it or perform actions that render the system unavailable. This can occur intentionally or unintentionally. In either case, the threats to the system need to be evaluated and measures decided upon to prevent these threats from occurring. The British Standards Institution [24] defines the top level goals for information security as maintaining confidentiality, integrity, and availability.

- *Confidentiality*—ensuring that information is only accessed to those who are authorised.
- *Integrity*—safeguarding the completeness and accuracy of the information and processing methods.
- *Availability*—ensuring authorised users have access to information when required.

In maintaining these goals it is important to restrict access to information [24]. Access control is an important component of a security policy, particularly with regard to confidentiality and integrity. Normally these restrictions along with other security requirements are dealt with late on in the software development cycle, often during the design phase. The consequences are that a rigorous analysis is not carried out as to who should be able to do what. One common problem, for example, is the employee who leaves an organisation but whose permissions are not removed. This exposes the organisation to an unnecessary risk and is typical of the sort of loopholes that occur if the analysis is insufficiently rigorous.

There is therefore a need to tackle access control earlier on in the development process during the RE phase to determine exactly what the access control requirements are. In the RE literature, although there are some important contributions with respect to the analysis of threats [25], evaluation of design alternatives to satisfy security goals [26], and checking the consistency of requirements with respect to a security policy [1], there is very little with regard to access policies. An exception to this is Fontaine [27], who explores how a goal model based on the goal-oriented approach KAOS can be mapped onto Ponder [21], a language for specifying access policies, which we review in more detail later on. What is important here is the principle of how operationalised goals can be translated into authorisation policies. The authors propose that the assignment of an agent to a goal maps on to an authorisation policy in Ponder. This simple definition, as we will see, however, is inadequate to accurately represent access restrictions, which are actually somewhat more complex and need to mirror organisational structures and procedures.

2.2. Access policies

The RE literature on this subject is somewhat wanting; until now the RE community has largely neglected the plethora of research on security policy.

Research on computer security policy has been ongoing since the 1970s. The Bell–LaPadula (BLP) model [3] has been particularly influential, and forms the basis of a family of multi-level security models, usually referred to as mandatory access control [8]. Discretionary access control on the other hand has been accepted as a less rigorous way of controlling access. Clark and Wilson [7] identified important principles for protecting the integrity of data in commercial organisations and in particular the separation of duties. Actions can be divided into smaller tasks, which must be carried out by different individuals, so preventing one individual alone from being able to defraud the system.

Research into access control has become focused increasingly around Role-Based Access Control (RBAC) [14]. The basic premise underlying RBAC is that in order to simplify administration, permissions are assigned to roles rather than users; a user gains permissions by being assigned a role. Roles are a way of defining positions in organisations, bundling responsibilities, or perhaps representing a qualification. Sandhu et al. [14] argue that RBAC is policy neutral, thereby allowing the enforcement of a variety of security constraints. A good example is the separation of duties, an important policy to ensure integrity. Separation of duties can be achieved by defining mutually exclusive roles, which have to be invoked for a collaborative task, in order to ensure that a sequence of tasks cannot be carried out by a single individual. Furthermore, RBAC can coexist with, or be used to support, mandatory access control policy [13] such as BLP or a discretionary access control policy [16], where users are assigned permissions individually. There is quite a significant variation in the interpretation of RBAC, differing in the level of sophistication that different models support. The authors illustrate this by presenting a family of models with varying levels of complexity. More recently, Sandhu et al. [15] have proposed the NIST RBAC model, which is an attempt to define a unified standard.

The NIST RBAC model is actually a sequence of models, with each subsequent model containing an increased set of capabilities. Three important characteristics are identified. The first is the basic ternary user role permission relationship. The idea is that permissions are assigned to roles rather than to users. The second key characteristic is a role hierarchy, which is relatively simple, whereby senior roles inherit the permissions of junior roles. For permissions that should not be inherited, Sandhu et al. propose an activity hierarchy in which senior roles only inherit permissions from junior roles if they have been activated. The final key characteristic is the principle of role constraints, which is important for enforcing separation of duties by, for example, defining two roles as mutually exclusive.

The idea of a single hierarchy based on inheritance, as proposed in the NIST model, has been called into question [12]. A manager does not necessarily inherit the roles of his juniors. For example, in a project, a manager does not necessarily possess the necessary competence to carry out specialised activities. Moffett [12] has derived a set of hierarchies based on organisational control principles. The three main principles are separation of duties, decentralisation and supervision, and review. In order to capture these characteristics, Moffett proposes three types of hierarchies.

- *The 'is-a' hierarchy based on generalisation*. It is often possible to identify common responsibilities amongst members of an organisation. These responsibilities can be bundled together to form a generalised role. An example of this in a hospital, say, would be to define a generalised role of health care provider, which provides permissions shared by both doctors and nurses. The role doctor is itself a generalised role for a physician or a surgeon. The inverse of generalisation is specialisation.
- *An activity hierarchy based on aggregation*. In this hierarchy, permissions are bundled together to form a collection of permissions, which are needed to carry out the various tasks that logically belong together from an organisational standpoint. For example, a sequence of tasks may be required to provide a specific customer service, such as booking a flight.
- *A supervision hierarchy based on the organisational hierarchy*. This hierarchy is what is normally considered to be the organisational hierarchy in that it represents the lines of supervision showing the seniority of the members of staff. This is the hierarchy that can also be used to differentiate permissions between junior and senior members of staff.

Bacon et al. [4] also demonstrate how roles based on function and seniority can be combined. In order to be assigned certain roles, a user must have been assigned other prerequisite roles, for example, a doctor can only be assigned the role of senior haematologist if the roles senior doctor and haematologist have already been assigned.

More recent work in this area is concerned with access control models referred to as active security models, which are aware of the context of an ongoing activity. Bertino et al. [2] describe how temporal constraints can be defined for roles, for example, when a role is activated for a shift, and then subsequently deactivated. In addition, an administrator can activate roles in an ad hoc way. Covington et al. [6] have explored applications for the home and suggest how environmental roles could be useful. Access can be permitted based on environmental factors, such as location or time of day. Georgiadis et al. [9] combine contextual information with team based access control. Team based roles identified by Thomas [17] are useful for collaborative working environments, where users are assigned to teams and get access to the team's resources. This idea can be combined with other contextual information, such as location or time intervals. Yao et al. [18] present an access control model (OASIS), whereby users can activate roles, provided they satisfy prerequisite conditions, such as having an appropriate qualification, assigned function, task competence, or environmental constraint.

Importantly, this research demonstrates that roles provide an effective basis for defining restrictions. However, the questions remain: where do roles come from, what are their relationships with one another, and how do we address these questions during RE?

2.3. Policy specification languages

We can specify policies at various levels of abstraction [32]. However we are only interested in those languages that are at a level of abstraction useful to an analyst or designer, rather than low level languages or definitions that enable us define such things as firewall configurations, or low level access control restrictions.

Earlier in this section we mentioned an important contribution by Fontaine [27] proposing a mapping between KAOS and Ponder. Ponder is one of a number of languages that have been developed by the security policy community for specifying access policies. These complement the research described above and often a notation is developed to specify access control models based on the concepts, and indeed, most policy specification languages are designed to be executed in a deployment model as a part of the security administration, in a distributed system.

Ponder [21] has been developed over a number of years at Imperial College. It is a declarative language containing constructs for specifying authorisation, obligation and delegation policies, whereby authorisation policies represent the actions that subjects are permitted to carry out on targets, obligation policies relate to actions that must be carried out on targets by subjects when a predefined event occurs, and delegation policies define the policies that a subject can delegate to other subjects. These are termed basic policies, additionally there are also meta and composite policies. The language also contains constructs to model the organisational structures. Objects can be grouped together in domains to represent the partition of objects in geographical locations, roles can also be defined relating to positions in an organisation requiring a common set of policies and roles, and relationships can grouped together in a management structure A collection of policies that are assigned to groups or roles are termed composite policies. Meta-policies enable the definition of complex policies with constraints defined using the Object Constraint Language. A typical meta-policy could model the separation of duties. In the future it is planned to include constructs to model interactions between roles, the basis of which has been the subject of a doctoral research project [22].

Ponder is the most comprehensive in terms of the features that it can model, and in this respect makes a good choice by Fontaine to integrate with KAOS. However, the Ponder language is designed for the deployment of security policies, and is hence a language designed to be interpreted and executed by a deployment environment. This makes it unsuitable for specifying at the requirements level, as it is difficult to read, being more akin to a programming language. The KAOS mapping is an important move in the right direction, the key here being the assignment of agents to goals. These capabilities that can be defined in KAOS are functionally based, with the possibility of defining an inheritance hierarchy of agents. This is very similar conceptually to the inheritance hierarchy proposed in the NIST-RBAC model, which we reviewed in the last section. It therefore shares the same fundamental weakness identified by Moffett [12], that it cannot adequately model the principles of organisational control.

Jajodia et al. [20] propose a logical authorisation specification language (ASL) for specifying security and an associated deployment model FAM. Bacon et al. [28] propose how RBAC policies for the OASIS environment can be specified in a pseudo-natural language. As with Ponder, these languages are at a level too low for requirements analysis.

A notable exception to these languages, is a graphical language proposed by Thomas and Sandhu [23]. This language is in fact targeted at policy requirements analysis. The language focuses on task authorisation procedures, for specifying task sequences and approval steps. It, however, does not include a way of assigning agents or roles, which means that although the procedural steps can be represented, it is not possible to define who should do what.

2.4. Concluding remarks

It is necessary to build a bridge between a requirements model and the complex access policy deployment environments such as Ponder or Oasis. However, we believe that the current RE frameworks such as a KAOS are inadequate to represent these policies, because, as we mentioned above, the simple inheritance hierarchy, with regard to the capabilities of agents, does not allow modelling organisation control.

3. On organisational structures and roles

There is no universally accepted definition of roles. As we have seen above, researchers have different views on what constitutes a role and how role hierarchies can be defined. One of the reasons for this is that a role is a conceptual notion. In this section we discuss different categories of roles, and how they can be derived from organisational structures as this forms the central pillar of our work.

3.1. Organisational structure

The organisational structure of an enterprise is designed by the top management of that enterprise and defines the lines of authority and the division of work. These are the two principle characteristics that determine the responsibilities[2] for each individual member of the organisation.

Organisations can take many forms ranging from very simple structures, such as are normally found in small start-up companies, to the complex divisionalised structures found in large multinational corporations and although no two organisations are exactly the same, neither are they truly unique. There are common aspects that enable us to categorise the structure, generalise about it and posit views about it.

An organisation is a composite structure made up of organisational units, which can also be divided into even smaller units. There are different ways in which individual positions in the organisation can be grouped together to form these units, depending on the needs of the organisation. In defining roles it is useful to understand how these groupings come about.

3.2. Organisational groupings

Mintzberg [11] identified the key characteristics that are used to form organisational groupings. Essentially there are two basic types of characteristics that are used to form organisational groups. The first is function; this is defined around the tasks that a group performs, and the second is market defined around responsibility for product, service, or customers.

Often, particularly in large organisations, several of these characteristics are used. The National Health Service in the UK is divided into regional health authorities, which in turn, are composed of hospitals to serve the different population centres, so that the authority and the hospitals are organised on a geographical basis. A hospital, however, is organised on a functional basis according to administration, medical

[2] We use the term responsibility to denote obligation and accountability.

specialities and supporting services. Similarly, retail banks have autonomous branches dispersed to serve local markets with a functional structure in each branch.

Another way in which an organisation can be structured according to multiple factors is through a matrix structure. In this case, each member of the organisation will belong to two groups. One group is responsible for the product or market, and the other has a functional responsibility. An example of this is in an engineering company undertaking projects. Each project consists of a multidisciplinary team of engineers, and each member of the team reports to the project manager, but there are also departments that carry responsibility for staff development and maintaining standards in the different engineering disciplines.

3.3. Organisational roles

Each member of the organisation has a set of responsibilities, which is the ultimate determinant of access privileges for applications. It is worth stepping back briefly to consider what organisational roles are, so that we can relate them to the groupings we defined above. In order to do this we need to use role theory.

Role theory is largely a sociological science focusing on how individuals interact with one another. Handy [10] describes the important concepts of role theory. The individual who is the centre of analysis is called the focal person, and the group of individuals with whom he interacts is called his role set. Thus, depending on the situation and the person with whom he is interacting, an individual will adopt a specific role, perhaps as a father, customer, friend, or advisor, and there are certain societal expectations of people in these respective roles and in the way they are supposed to behave. Some roles are occupationally defined, such as doctor or lawyer, and for which there are legal as well as cultural expectations.

In an organisation, the situation is similar in that members of this organisation in a particular position may have multiple roles such as manager, specialist, or subordinate depending on the task they are currently undertaking, and particularly with whom they are interacting. Thus, roles in an organisation, and the expectations of an individual adopting a role are defined by management, and relate to the responsibilities and tasks that have been assigned to that individual.

3.4. Role categories

It is clear we need to create a link between roles and assets, or more appropriately between sets of roles and assets, because, as we have seen, an individual will have several roles. Seniority is one of the important dimensions and the other is scope of work, for which Mintzberg has defined two groups of characteristics based on functions and markets. The characteristics seniority, function and market give us three groupings, which can be used as a basis for the identification of roles and are summarised as follows

- Roles based on supervision
  - Seniority
- Roles based on function
  - Qualification
  - Function
  - Work-process
- Roles based on market
  - Market/customer/client
  - Product/service
  - Location
  - Time

For each of these categories, there is often a potential for a hierarchy, perhaps reflecting hierarchical structures in the organisation. The characteristics of this hierarchy also need to be determined, for example, whether permissions are automatically inherited.

It is also necessary to establish the relationship between these roles and the access to information assets. Having established the key characteristics by which responsibilities are assigned, it is then a matter of identifying the criteria by which access is restricted. This needs to be done for each type of access or operation on each asset.

3.5. Roles based on seniority

Roles that are based on seniority are reflected in the hierarchical lines of authority. Supervision is a key coordination mechanism and seniority is the cornerstone of this mechanism. But what are the characteristics of a role based on seniority, and what is its relationship to other roles?

Mintzberg has identified three types of authority: line, staff, and functional authority. Staff and functional authority are very similar in nature in that they transcend organisational groups. Line authority is the most common form, and is the kind of authority found within a single organisational unit. This is what we focus on in this paper.

One of the key characteristics of seniority is span of control, i.e. which subordinates are under the control of a supervisory role. There will be subordinates who are directly controlled and those who are indirectly controlled through delegated authority. There are two properties that need to be considered: the first is the cardinality representing the span of control, and the second is the transitivity, which represents the extent to which control transcends levels.

An important issue is how the authority relationship between two users can be modelled. There are two ways that this can be done. The first is to link the relationship through the users, i.e. a user with a subordinate role is directly assigned to a supervisor. The alternative is to link the relationship through other roles representing situational factors, such as a project, a ward in a hospital, or a product line in a sales department. In the latter case, the line of authority in an organisational grouping is linked to the functional or market factor on which the group is based.

Subordinates may have more than one supervisor. There are several situations in which this can occur. One situation is when the subordinate answers to superiors with varying degrees of delegated authority. In the absence of the immediate superior, one of the subordinates could act in a temporary capacity as a supervisor to whom the others report. A second situation occurs in matrix organisations, where subordinates answer to two superiors: one with responsibility for assigning the tasks, and the other with a more indirect responsibility for the quality of work, standards, and personal development. In fact, in some organisations, which operate in this way, an individual may be assigned to more than one project. In such cases, the individual reports to a senior in the functional hierarchy and several project managers in a market-based hierarchy.

Delegation is also a key aspect of a supervisory structure [12]. Delegated authority is an inherent part of any hierarchy and there are several forms of this. Barka and Sandhu [5] identify the following characteristics

- *Permanence*. This determines whether authority is delegated on a temporary or permanent basis.
- *Totality*. This determines the extent to which all or only some permissions from a user or role are delegated.
- *Duration*. This determines the length of time for which a delegation is valid.
- *Delegation levels*. This determines the extent to which delegated permissions can be further delegated.

Each of these characteristics needs to be modelled in order to capture the principles of delegated authority as they are commonly practised in organisations.

3.6. Roles based on function

It is difficult to envisage a role structure for an organisation that does not include roles with functionally-based characteristics. Seniority alone is not normally sufficient to identify a position in an organisation effectively. The different ways in which roles based on functions that were listed in Section 3.4 are qualification, function, and work-process.

Roles based on qualifications or functions are similar in that they basically model the capability of an individual. In Section 3.5, the idea of a generalisation or 'is-a' hierarchy [12] was reviewed. In fact, a specialisation hierarchy would be a more appropriate term, as it reflects the principle of specialisation either through qualification or assignment to a function. More specialised roles inherit responsibilities from roles further up in the hierarchy.

Tasks that logically belong together can be grouped together to form work-process based roles. In Section 3.5, the idea of aggregating activities was presented by Moffett [12].

It is normal that a dependency exists between work-process based roles, the assigned qualification or function roles, and the level of seniority. In order to assign a work-process role, the user must have the appropriate qualification or function and seniority assigned to him.

3.7. Roles based on market

The roles based on functions and seniority determine the tasks that a user can carry out, but it is roles based on market characteristics that determine on which targets a user may carry out the tasks. There needs to be a correspondence with the asset affected. The role needs to link the role with the asset through a context, as we will demonstrate in Section 4. We therefore refer to them as contextually-based roles. As with roles based on functions, there is the potential for a hierarchy similar to the aggregation of activities. In Section 2.2 the concept of roles based on contextual information was reviewed [9], which has certain similarities; location is an example of this.

3.8. Concluding remarks

Using organisational structure in this way—to define roles to form a basis of a security framework—has some significant advantages

- It provides a clear focus for analysts and users eliciting requirements, as organisational groupings and the lines of authority are relatively easy to identify.
- Users are able to relate more easily to the defined roles.
- Organisational procedures can be more readily translated into a security framework, because roles more accurately reflect the organisation.

4. An analytical role modelling framework

In this section, we present an analytical framework based on the ideas developed in Section 3. The hypothesis is that our framework can be used by analysts to model and analyse access control requirements. We demonstrate the application of the framework through the use of an example of access restrictions to patient records.

4.1. Framework

Fig. 1 shows the relationship between the key conceptual components of the framework in which the cardinality is shown between the different elements. In this framework there are two levels. The first is the meta-level. This includes the definitions of role types, asset categories, context types that enable us to define access policies. The second is the instance level; this includes instance

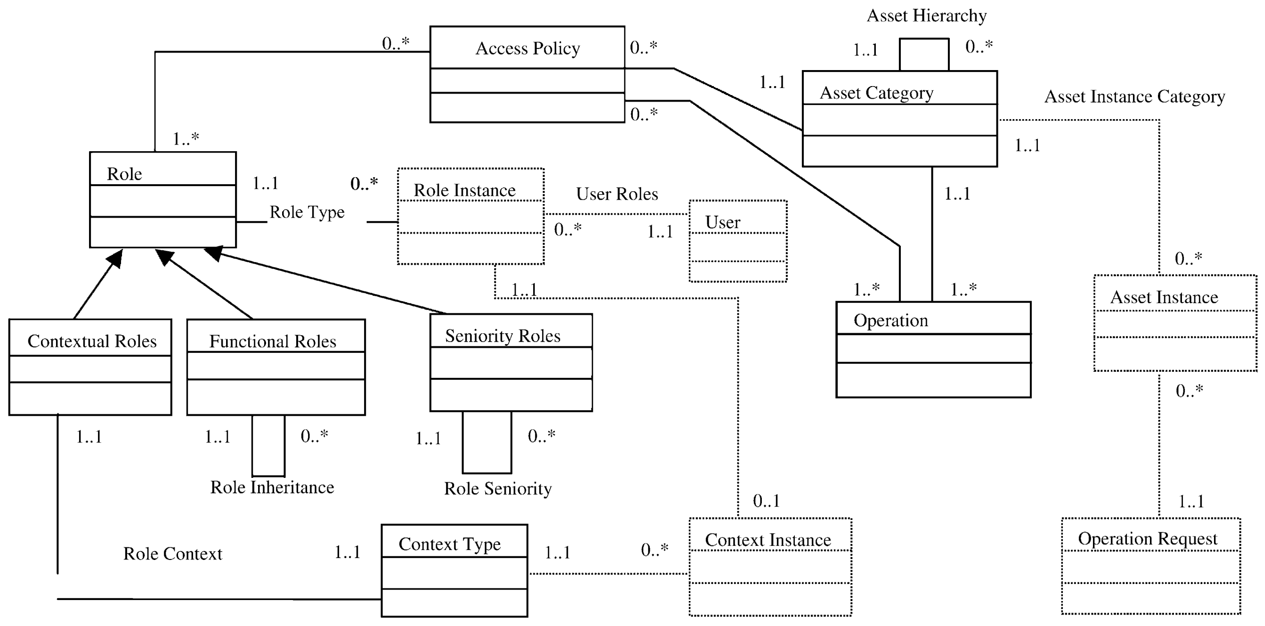

Fig. 1. The key components of our framework.

definitions of users, context instances, asset instances and role instances that enable us to validate policies through the definition of scenarios. In the diagram the instance level components are represented as dashed boxes.

In this framework we include three types of roles: functional, seniority and contextual. The contextual roles represent market-based roles that we described in Section 4. This is because these roles relate to information assets through a context. For example, in a regional branch of a retail bank, bank tellers only have access to accounts of customers of that branch. Therefore, the branch outlet represents a context, which has to be assigned both to the account and the bank teller in order for access to be granted.

An access policy is modelled as a ternary relationship between role sets, sets of operations and an asset category. A role set is required to model restrictions where a combination of market, functional and seniority based characteristics are a prerequisite for accessing an operation or a set of operations. The set of role sets can be considered as alternative conjunctions of roles, i.e. a user must satisfy at least one of the specified role sets. This enables us to allow users with different functions or responsibilities to access the same functions. The reason for including sets of operations in an access policy is that sometimes tasks may be bundled to form a logical unit of work. An access policy relates only to a single asset category, however, an asset hierarchy has been included so that through inheritance the policy can be applied to more than one category of assets.

The components at the meta-level in the framework, are as follows

Role. Role is a type representing either functional, seniority, or contextual roles.

Context type. Some policies require a context, such as an assignment to a patient, or a ward, in order for them to be resolved. We use a context type to define this in a policy definition.

Asset category. Information needs to be categorised, so that we can determine what policy is relevant. All secure entities must be assigned to a category.

Operation. This represents an operation that can be carried out on an asset.

The components at the instance level in the framework are as follows

User. When validating a policy through a scenario we need to define users, who we then assign roles in order to explore whether requests to execute operations on stored information are authorised or not.

Role instance. This is an instance of an assigned role. A role is instantiated before it is assigned, so that useful information about it can be maintained, primarily the context. If, for example, a role is based around a client of a consultant or a patient in a hospital, then a role instance modelling responsibility for the patient or client would be related to a context. Using a role instance enables us to track other information, such as who assigned the role, when was it assigned and when shall it expire, although in this particular model this has not been done.

Context instance. This represents the actual context, which can be used to resolve an authorisation request. A context instance could be a ward in a hospital, where a patient is located. By being assigned to the ward the nurse may access the patient's record.

Asset instance. This is the instantiation of an asset, which for example could be a specific patient record.

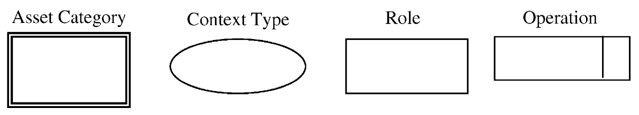

Fig. 2. Policy definitions.

Operation request. We use this to model the request that a user makes when he wishes to execute an operation.

A role–role relation is used to represent the role hierarchy. For functional roles, a hierarchy serves to model inheritance, whereas for seniority roles it models the lines of authority. A similar hierarchy has also been included for asset categories, rather similar to the generalisation hierarchy of a functional role model. At the top of the hierarchy are the most general asset categories and as we move down the hierarchy the categories become more specific. An example of a very basic asset type is static data. Static data is a term used in business process systems to describe data that rarely changes, such as names and addresses, account numbers and so on; another common category is that of transactional data. A more specific category could be financial transactions, and even more specific would be a debit.

The hierarchies enable us to define policies at the desired level of abstraction. For example, a general policy could be defined for a financial transaction, and this policy would be applicable to debit and credit transactions. *User Roles* is a relation that gives the role instances assigned to a user, and the role type relates the role to the role instance *Role Type*. The relation *Asset Instance Category* relates the asset category to an asset instance. *Asset Context* is a function that gives the context instances assigned to asset instances. In this model a policy containing a context role, such as patient, location or product group, is resolved by matching the context of a contextual role assigned to a user with one assigned to an asset instance. If a doctor wishes to access a patient record, then this needs to be associated with a patient for whom the doctor has responsibility. In this instance the context is the patient.

In order to represent policy definitions, we use the graphical notation detailed in Fig. 2. We developed this notation ourselves for the purposes of demonstrating the framework. The reason we did this is because the graphical notations offered in existing RE methods and frameworks do not include the elements outlined above that we need to model. KAOS [19], GBRAMS [29] and Use Case methods only include simple definitions of actors or agents, which are often interpreted as roles. The *i** Framework [31] goes further in that roles are explicitly included, and there is a differentiation between position and task based roles; role instances are also defined. However, we want to differentiate further with contextual roles, and in addition include asset hierarchies, role hierarchies, contexts and explicit policy definitions. With regard to security policy languages that we reviewed in Section 2.3, the graphical notation proposed by Thomas and Sandhu [23] can only model task sequences and is therefore unsuitable. The other languages and in particular Ponder do include the right sorts of features, but are textual and difficult for non-technical specialists to understand. It is important to note, however, that the notation is secondary to the principles that we are demonstrating in terms of how roles can be used to define policies.

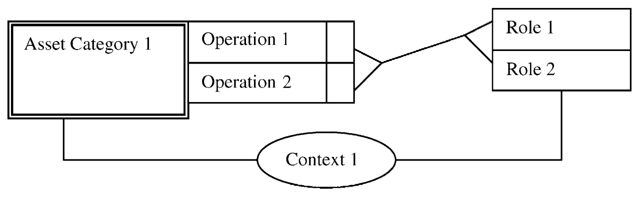

Fig. 3. The composition of policy diagrams.

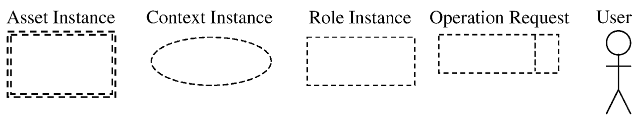

Fig. 4. The notation for policy scenario diagrams.

The components in Fig. 2 are used to compose policy diagrams. A policy is depicted as alternatives of conjunctions of roles, which are a prerequisite for a user to be assigned in order to be able to be executing operations on assets. A forked line joins roles with operations and indicates that the conjunction of roles at one end of the fork is necessary to execute either of the operations at the other end. In Fig. 1 this relationship between operations and roles is realised through the access policy. If a role is contextual, a line is shown between the role and the context type, and between the context type and the asset category. This is shown in Fig. 3.

In order to represent policy scenario diagrams we use the notation shown in Fig. 4.

Fig. 5 shows a policy scenario derived from the policy definition in Fig. 3 above. Role, asset and context instances are depicted, together with their type defined in the respective symbol. A user is allocated role instances indicated by the forked line between the user and the role instances. A context instance links a contextual role instance with an asset instance. In this case a request for *Operation 1* on asset instance *Asset1 Inst.* is authorised because of the policy defined above, but a request for *Operation 3* is rejected. *Operation 3* is an additional operation not defined in the policy.

4.2. An example

We now present a healthcare example to illustrate how the above framework can be used to analyse policies for accessing patient records. In this example there are two types of records: medical records, which contain diagnoses, observations, and treatment plans and are updated by medical practitioners; and nursing records, which record the treatment and observations from the nursing staff. A patient

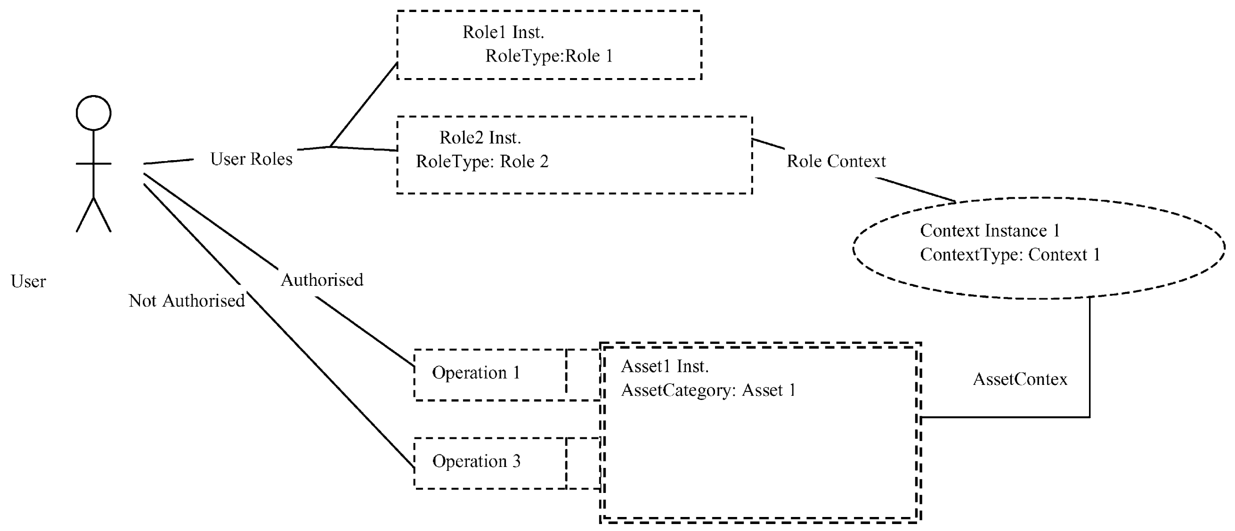

Fig. 5. A sample policy scenario.

is assigned to a consultant who is generally either a physician or a surgeon. The consultant needs read and update permissions for the patient's medical records, and read-only access for the nursing records. All nurses on the ward, where the patient is located, need read access to the medical records, and must be able to read and update the nursing record. As far as nursing care goes, the role set that is needed for read access includes a nursing qualification and an assignment to a ward. With regard to medical practitioners, access must be restricted to the consultant to whom the patient has been assigned.

This following role diagrams, shown as Fig. 6, represents the domain definitions that we need to enforce the policy. It includes the necessary role definitions. The functional roles *Surgeon* and *Physician* inherit permissions of the role *Medical Practitioner*. The seniority hierarchy for medical staff is partially represented with *Consultant* and *Registrar*. A registrar is a junior doctor who reports to a consultant. Access to a medical record by a registrar is achieved by delegation, but this has not been modelled here, because our framework cannot yet handle delegation policies. In this simplified example, nurses are represented by a single functional role of *Nurse*. The context role *Responsible For Patient* models the assignment of patient to a consultant, and *Ward Assignment* the assignment of a nurse to a ward.

Fig. 7 shows the asset categories. The two basic asset categories are *Medical Record* and *Nursing Record*. The asset categories *Treatment Plan* and *Diagnosis* are inherited from Medical Record, which is modelled in the mapping *Asset Hierarchy*, and therefore policies that apply to *Medical Record apply* also to these.

There are four policies depicted in Fig. 8. *Read Medical Record Policy* restricts read access to medical records to nurses that are assigned to the ward, where the patient has been stationed and the consultant that has responsibility for the patient. Similarly, the policy *Read Nursing Record Policy* restricts read access of the nursing records to the same groups. *Update Medical Record Policy* restricts the update of medical record to the consultant who has responsibility for the patient, while *Update Nursing Record Policy* restricts update access of nursing records to nurses assigned to the ward, where the patient is located.

4.3. Policy scenario analysis

We now present two scenarios that we can use to validate the policy. Not all possible asset types and roles that we defined above are used in the following scenario analysis due to space limitations.

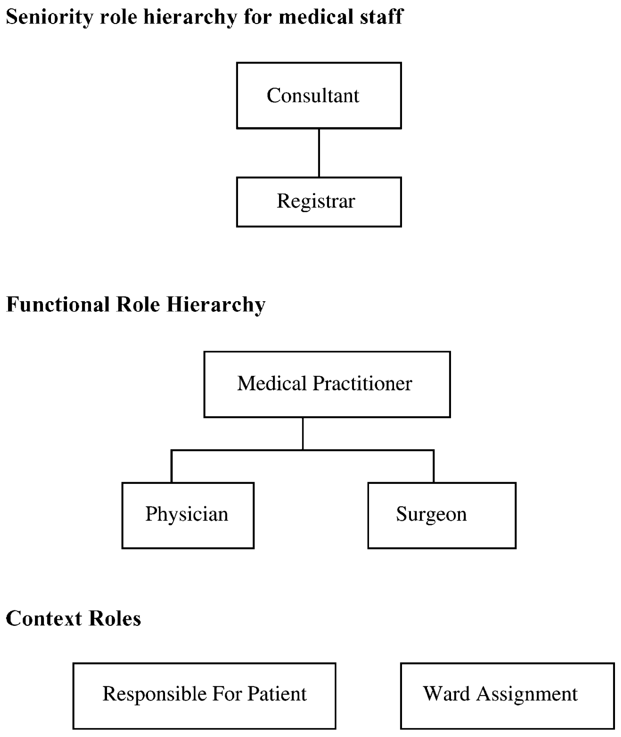

Fig. 6. Domain definitions for the example.

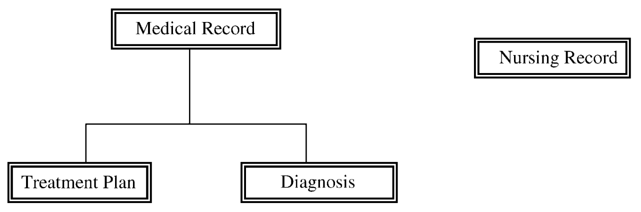

Fig. 7. The asset hierarchy for the example.

In the first scenario *John Smith* is a consultant physician who has responsibility for the patient *Richard Cargill*. Through the relation *User Roles*, he is therefore assigned the role instances *Physician Inst.* and *Responsible for Patient Inst.*, which are of role types *Physician* and *Responsible for Patient*, respectively, defined by the relation *Role Type*. The role instance *Responsible for Patient* is linked to the patient context instance *Richard Cargill* through the relation *Role Context*. The asset instance *Medical Record Cargill* is linked to the patient of *Richard Cargill* through the relation *Asset Context*. These assignments mean that *John Smith* has the prerequisite role assignments to satisfy policies *Read Medical Record Policy*, *Update Medical Record Policy*, and *Read Nursing Record Policy*, so that he can read medical and nursing records as well as update medical records associated with the patient *Richard Cargill*. He cannot, however, update the nursing records. Fig. 9 shows this scenario.

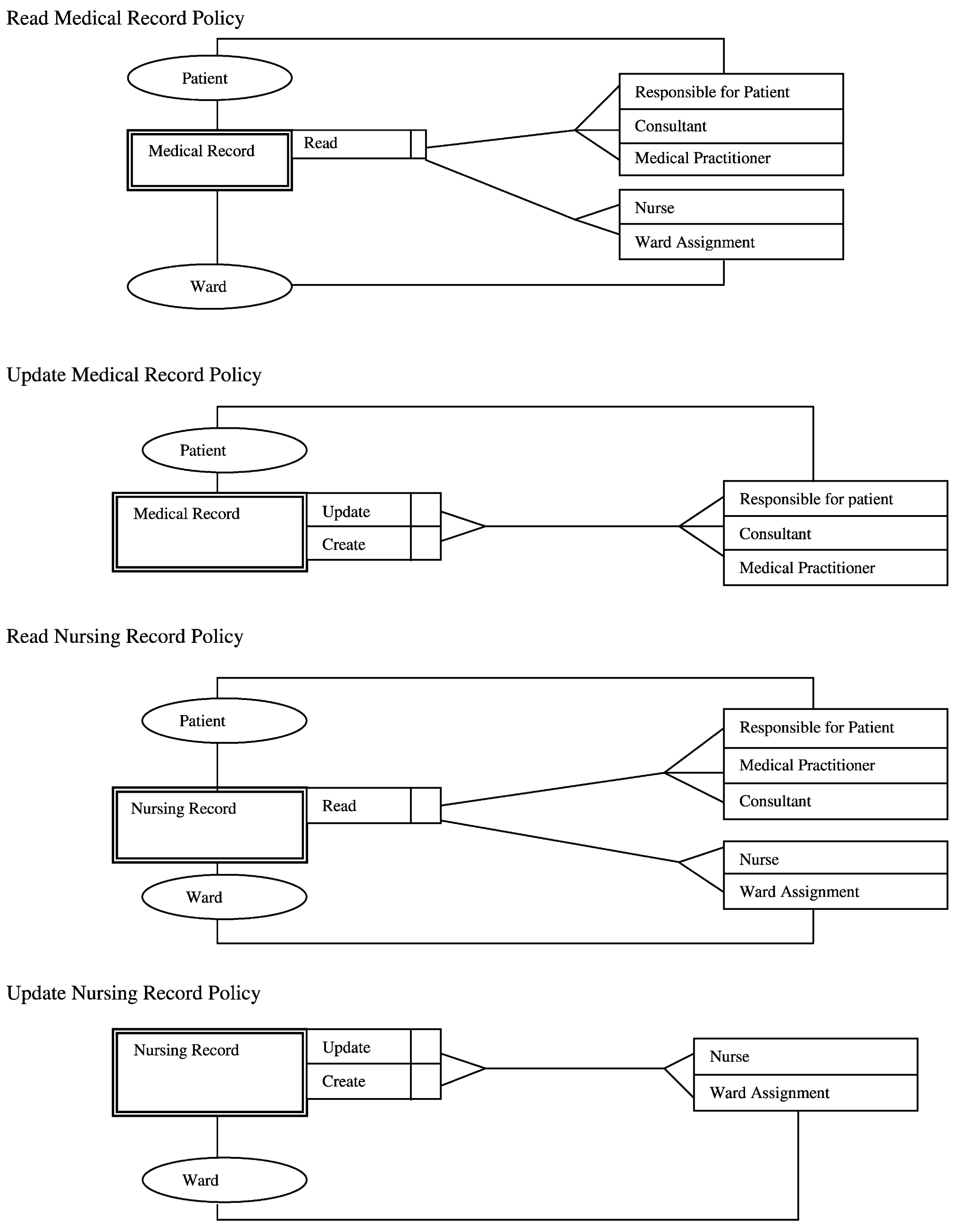

Fig. 8. Four policies from the example.

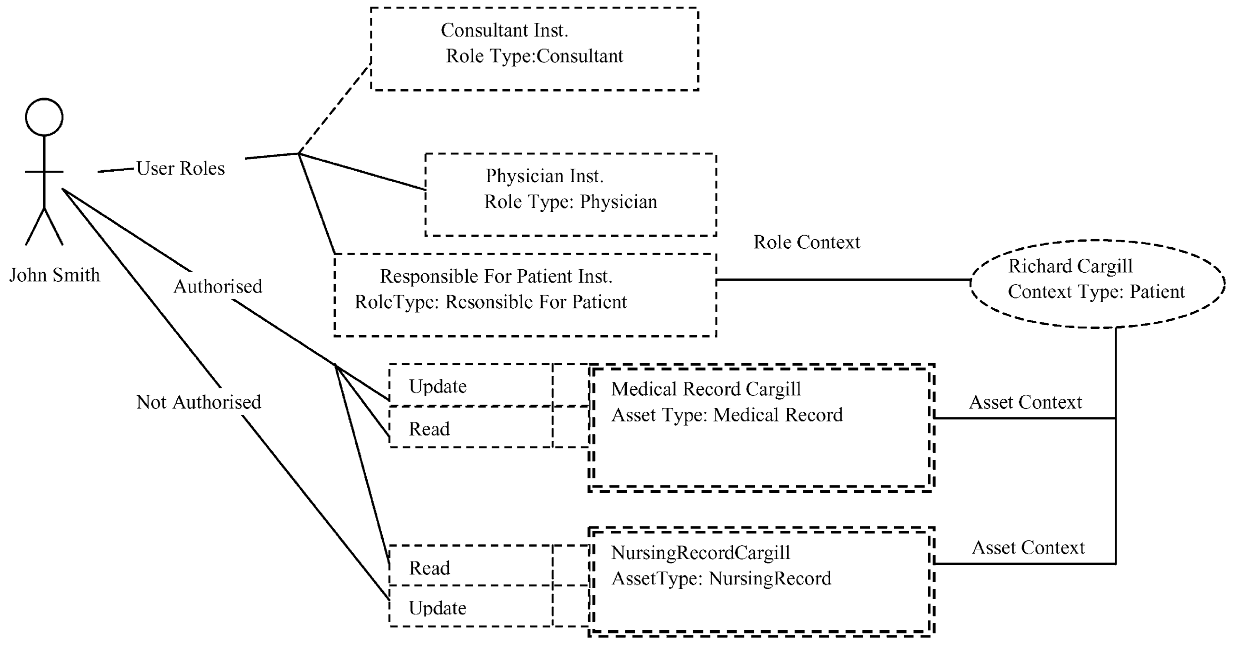

Fig. 9. The first scenario.

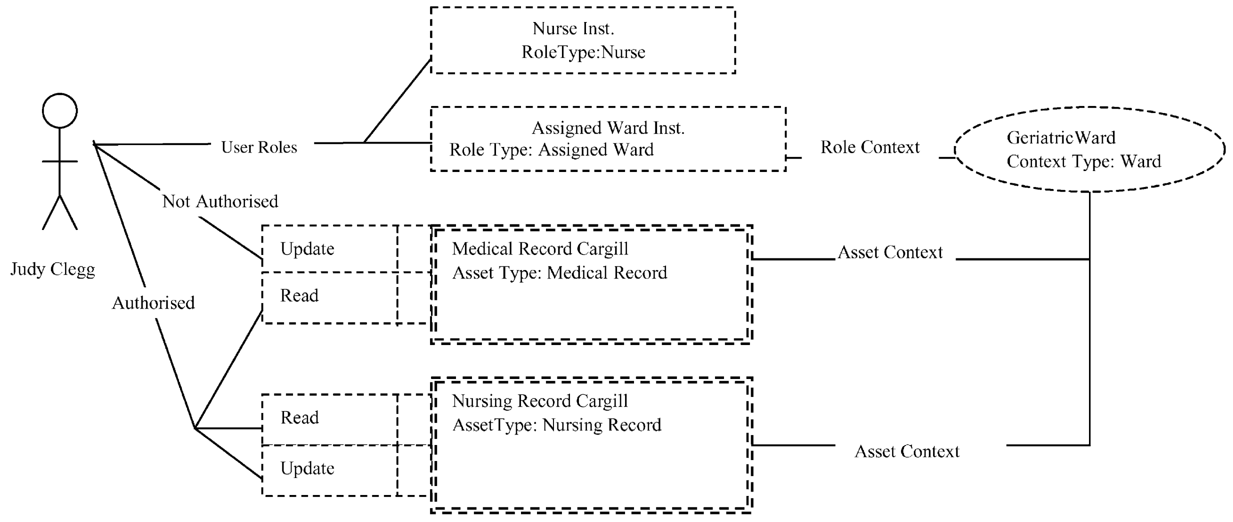

Fig. 10. The second scenario.

In the second scenario *Judy Clegg* is a nurse assigned to the ward *Geriatric Ward*, where the patient *Richard Cargill* has been stationed. In the relation *User Roles* she is therefore assigned the roles instances *Nurse Inst.* and *Assigned Ward Inst.*, which are of role types *Nurse* and *Assigned Ward*, respectively. The role instance *Assigned Ward Inst.* is linked to the ward context instance *Geriatric Ward*, which in turn is mapped to the patient record *Medical Record Cargill* in *Asset Context*. *Judy Clegg* therefore has the prerequisite role assignments to satisfy the policies *Update Nursing Record*, *Read Medical Record Policy*, *Read Nursing Record Policy* and *Update Nursing Record Policy*. This is shown as Fig. 10.

5. Conclusion

In this paper, we have explored the importance of roles in defining security goals and policies and how they can be used to represent organisational requirements that underpin access restrictions. We highlighted the different and sometimes contradictory viewpoints of security policy researchers with regard to the definition of roles, what they represent, and how they relate to one another. In order to resolve these conflicting viewpoints, we examined the problem from the perspective of the organisational structure, applying principles identified by Mintzberg [11] for structuring organisations to derive categories of roles. We then used this as a basis for an analytical framework, through which access policies can be defined and validated using scenarios. Finally, we demonstrated the use of the framework through an example in the healthcare domain.

In terms of defining access policies the RE literature is distinctly lacking. Most contributions demonstrate how existing RE methods can be used to define security goals and identify threats. Security policy research has demonstrated the importance of the organisational context in defining policies. Models that researchers in this area have developed are quite sophisticated and although focused on design and deployment, give us pointers as to how we need to deal with policies at the requirements level. Most RE methods have simple definitions of agents or actors, which are then assigned to goals or actions, however, the organisational context is somewhat more complex requiring a richer representation; roles provide us a way of capturing these complexities. The *i** Framework goes further than other RE methods by including the concept of roles explicitly, but the use of this framework in developing policies has not yet been explored.

We have demonstrated an alternative approach, focusing on the use of roles in defining access policies, with the objective of providing a bridge between existing RE methods and access control models, however, we have not yet demonstrated how the concepts in our framework can be integrated into current RE methods. Developing policies can only be done within the context of a functional or goal model.

There is still much that needs to be done to extend the work in this paper. The example given in this paper does not demonstrate how the role assignments and access requests can be integrated with the core business functions. Other issues that remain to be explored include delegation, authorisation procedures, and role constraints that can be used to enforce the separation of duties.

Checking the consistency of security requirements is also crucial. Role and asset hierarchies offer a way of defining policies at different levels of abstraction but may also introduce inconsistency, particularly when access rights arise indirectly through the inheritance structure that are not foreseen.

Finally, the link between the requirements model and the implementation model needs to be investigated, when implementation options range from custom solutions to using standard access control models. It would be useful to investigate how policies, specified using our framework, could be mapped onto a language, which could be deployed, such as Ponder or ASL. This would help us to improve the process of translating access policies at the requirements level into a deployable access control model.

Acknowledgements

We would like to acknowledge the advice and feedback to our colleagues in the Security Requirements Group at The Open University, particularly Jonathan Moffett for his comments and views on earlier drafts of this paper. Thanks also to Pat and John Crook, former Clinical Director of Wrightington Hospital, Wigan, and Leigh NHS Trust, for insights into the procedures for accessing patient records. This paper is a revised and extended version of Ref. [30].

References

[1] A.I. Antón, J.B. Earp, Strategies for Developing Policies and Requirements for Secure Electronic Commerce Systems, Recent Advances in Secure and Private E-Commerce, Kluwer Acedemic Publishers, Dordecht, 2001.
[2] E. Bertino, P.A. Bonatti, E. Ferrari, TRBAC: a temporal role-based access control model, Proceedings of the 5th ACM Workship on Role-based Access Control, July 2000, pp. 21–30.
[3] D. Bell, L.J. La Padula, Secure Computer Systems: a Mathematical Model, MITRE Technical Report 2547, vol. II, 1973.
[4] J. Bacon, M. Lloyd, K. Moody, Translating Role-based Access Control within Context, Proceedings of International Workshop Policies for Distributed Systems and Networks (Policy 2001), Bristol, January 2001, LNCS, Springer-Verlag, pp. 107–119.
[5] E. Barka, R. Sandhu, A framework for role based delegation model, Proceedings of 23rd National Information Systems Security Conference, Baltimore, October 16–19 2000, pp. 101–114.
[6] M.J. Covington, W. Long, S. Srinivasan, A.K. Dev, M. Ahamad, G.D. Abowd, Securing context-aware applications using environment roles, Proceedings of the 6th ACM Symposium on Access Control Models and Technologies, May 2001, pp. 10–20.
[7] D. Clark, D. Wilson, A comparison of commercial and military computer security policies, Proceedings of the IEEE Symposium on Security and Privacy (1987) 184–194.
[8] Department of Defense Trusted Computer System Evaluation Criteria Dod 5200-28-Std, 1985.
[9] C.K. Georgiadis, I. Mavridis, G. Pangalos, R.K. Thomas, Flexible team-based access control using contexts, Proceedings of the 6th ACM Symposium on Access Control Models and Technologies, May 2001, pp. 21–27.
[10] C. Handy, Understanding Organizations, Penguin, Harmondsworth, 1985.
[11] H. Mintzberg, Structure in Fives: Designing effective organisations, Prentice Hall, Englewood Cliffs, NJ, 1992.
[12] J.D. Moffett, Control principles and role hierarchies, Proceedings of the 3rd ACM Symposium on Access Control Models and Technologies, October 1998, pp. 63–69.
[13] M. Nyanchama, S.L. Osborn, Modeling mandatory access control in role-based security systems, in: D.L. Spooner, S.A. Demurjian, J.E. Dobson (Eds.), Proceedings of the IFIP WG 11.3 9th Annual Working Conference on Database Security, Chapman and Hall, London, 1995, pp. 129–144.
[14] R. Sandhu, E. Coyne, H. Feinstaein, C. Youmann, Role-based access control models, IEEE Computer 29 (2) (1996) 38–47.
[15] R. Sandhu, D. Ferraiolo, R. Kuhn, The NIST model for role-based access control: towards a unified standard, Proceedings of the 5th ACN Workshop on Role-Based Access Control (RBAC-00), Berlin Germany, July 26–27 (2000) 47–64.

[16] R. Sandhu, Q. Munawer, How to do discretionary access control using roles, Proceedings of the 9th ACM Symposium on Access Control Models and Technologies, October 1998, pp. 47–54.

[17] R.K. Thomas, Team-based access control a primitive for applying role-based access controls in collaborative environments, Proceedings of the 2nd ACM Workshop on Role-Based Access Control, Fairfax, USA, 1997.

[18] W. Yao, K. Moody, J. Bacon, A Model of OASIS role-based access control and its support for active security, SACMAT'01, Chantilly Virginia, USA, 2001.

[19] A. Dardenne, A. Lamsweerde, Goal-Directed Requirements Acquisition, Science of Computer Programming, vol. 20, 1993.

[20] S. Jajodia, P. Samarati, V.S. Sabrahmanian, A logical language for expressing authorisations, Proceedings of the IEEE Symposium on Research in Security and Privacy, Oakland CA, May 1997, pp. 31–42.

[21] N. Damianou, N. Dulay, E. Lupu, M. Sloman, Ponder A Language for specifying Management and Security Policies for Distributed Systems, Imperial College Research Report DoC2001, January, 2001.

[22] E. Lupu, A Role-Based Framework for Distributed Management Systems, PhD Thesis, Imperial College of Science Technology and Medicine, Department of Computing, 1998.

[23] R.K. Thomas, Conceptual foundations for a Model of Task-Based Authorizations, IEEE proceedings on Computer Security Foundations Workshop VII, CSFW 7 (1994) 66–79.

[24] BS799-1: Information Security Management—Part 1: Code of Practice for Information Security, British Standards Institution, London, 1999.

[25] E. Yu, L. Liu, Modelling Trust in the *i** Strategic Actors Framework, Proceedings of the 3rd Workshop on Deception, Fraud and Trust in Agent Societies, Barcelona, Spain, 2000.

[26] L. Chung, Dealing with security requirements during the development of information systems, in: C. Rolland, F. Bodart, C. Cauvet (Eds.), Proceedings of CaiSE'93, 5th International Conferene on Advanced Information Systems Engineering, Paris, France, Springer-Verlag, Berlin, 1993, pp. 234–251.

[27] P.-J. Fontaine, Goal Oriented Elaboration of Security Requirements, Project Dissertation, Université Catholique de Louvain, Belgium, 2001.

[28] M. Bacon, K. Lloyd, K. Moody, Translating Role-Based Access Control within Context, Proceedings International Workshop Policy2001, Policies for Distributed Systems and Networks, Bristol, January 2001, Springer-Verlag, Berlin, 2001, pp. 107–119, LNCS.

[29] A. Antón, Goal Identification and Refinement in the Specification of Software-Based Information Systems, PhD Thesis, Georgia Institute of Technology, June 1997.

[30] R. Crook, D. Ince, B. Nuseibeh, Towards an Analytical Role Modelling Framework for Security Requirements, Proceedings of 8th International Workshop on Requirements Engineering: Foundation for Software Quality (REFSQ-02), Essen, Germany, September 9–10, 2002.

[31] E. Yu, A Framework for Organizational Modeling, PhD Thesis, Department of Computer Science, University of Toronto, 1995.

[32] N.C. Damianou, A Policy Framework for Management of Distributed Systems, PhD Thesis, Chapter 2, Imperial College of Science, Technology and Medicine, Department of Computing, London, 2002.