Professional Access Control

by

Leigh Howard de la Motte, B.Comp.

A dissertation submitted to the
School of Computing
in partial fulfilment of the requirements for the degree of

Bachelor of Computing with Honours

University of Tasmania

November 2004

Declaration

I, Leigh de la Motte, declare that this thesis contains no material which has been accepted for the award of any other degree or diploma in any tertiary institution. To my knowledge and belief, this thesis contains no material previously published or written by another person except where due reference is made in the text of the thesis.

Leigh de la Motte

Abstract

Professional Access Control (PAC) is a self-administrating access control model for professional users which employs a peer review process and oversight by system administrators. It is characterised by the existence of ethical controls on the relationships between the users (those accessing data or granting access privileges to others) and data owners. Investigations revealed that the issue of availability was crucial to users in the hospital domain studied, and that to minimise the administrative burden on system administrators, the users needed to take some of the load. These factors led to the development of the new Trusted Access Control (TAC) model which gives users control. TAC is a fundamental access control model, complementary to the well-known Mandatory Access Control (MAC) and Discretionary Access Control (DAC) models. PAC uses TAC at its core and also incorporates Role Based Access Control (RBAC) and Provision Based Access Control (PBAC). This gives it the flexibility and user-friendliness necessary in the hospital environment, while still providing a high degree of data confidentiality and integrity protection. The required PAC functionality has been built into an Oracle package which can be used by new and existing applications, making it a viable access control solution for complex environments such as hospitals. When enabled workflow applications use the Oracle package, access control is automatically effected behind-the-scene, providing both usability benefits and reduced administrative burden.

Acknowledgements

Special thanks go to my Supervisor, Jacky Hartnett, for her patience and her ability to keep me on track. The encouragement she gave made the whole process much more pleasant. Her advice on security concepts and insightful comments regarding my work, were extremely important.

Peter Vamplew and Jacky Hartnett provided good support and advice in their roles as Honours Coordinators.

I would also like to thank all the health practitioners and administrators that have contributed ideas. Especially, thanks to Chris Showell and Justin Thurley, from the Department of Health and Human Services, for their time and input into this project. Their advice and comments greatly assisted the development of a practical solution.

Christian McGee and Greg Frith were very helpful with assisting me to set up Oracle. Andrew Spilling and Rick Smith provided great tech support during the year. My thanks also go to Claire Tomlim and Heather de la Motte for proof reading this document.

Finally, thank you to all my honours colleagues, who have always made themselves available to assist when asked. Particular thanks go to Tim Everts, Barry Pearn and Chris MacLeavy for their help with managing the idiosyncrasies of Microsoft Word, among other things.

Table of Contents

Chapter 7

Chapter 8

Appendices

List of Figures

List of Tables

Chapter 1

Introduction

Much research has been done into developing access control systems for health related environments. Even after all this work, no clear system has emerged that meets the requirements in an efficient way. The two main reasons for this are the obvious complexity of the health domain and the inherent difficulties in successfully implementing new systems in such a volatile and changeable environment. Access control systems are at the foundation of such networks. Thus, any investment that is made for incorporating new access control mechanisms is substantial, both in terms of finance and personnel training.

Many sophisticated and clever access control solutions have been developed. While many of them have shown promise, there has been a tendency to reject them on the basis that they are either too inflexible or hard to implement.

Occam's razor, which is attributed to the 14th century English Franciscan friar, William of Ockham, states that, 'When two viable explanations are offered for a phenomenon, the simplest full explanation is preferable.' (Wikipedia date unknown) Applying his principle to the complex domain of hospital access control would suggest that if a simple solution to the problem can be found which meets the requirements, then that solution will be good. With this in mind, the aim of this project was to try to find a simple, low-impact solution to the access control problem.

Threats against repositories of personal digital data are greatly increasing at present. Identity theft and the inappropriate use of personal data are problems which need to be addressed by organisations that hold such data. Privacy laws and regulations have already been put in place, and it seems inevitable that organisations will increasingly be required to adequately protect the data they hold. Access control is the fundamental means of ensuring such protection. As such, it is vital that organisations

put appropriate access control systems in place. To enable organisations to take this step, access control solutions need to be tailored so that they can efficiently meet the demands placed on them. They need to take the organisational environment and requirements into account in order to provide access control mechanisms that are efficient and usable.

This thesis focuses on the problem of finding a suitable access control solution which can be incorporated into public hospitals. It deals with the aspect of user authorisation and does not cover the aspect of user authentication. In the State of Tasmania all public hospitals are run by the State Government *Department of Health and Human Services* (DHHS). The solution proposed in this thesis is therefore focussed somewhat on meeting the access control requirements within DHHS controlled hospitals. The DHHS made it clear that a workable solution must guarantee availability of records to clinicians, to ensure they are never denied access to information necessary for clinical decision making. In addition, for the solution to be acceptable it could not introduce any additional work for users. While DHHS patient records are still largely paper-based, the current state of access control in the hospitals in question is that all clinicians have access to all the electronic hospital records of all patients. The solution seeks to provide a system which can easily be implemented and that gives access to users on a need-to-know basis.

Initially it was envisaged that a simple team-based concept could be implemented using Oracle. This simple concept was that each patient would have a treating team (users) and that only people on the team could access the patient's record. It soon became clear that in order to guarantee availability and to minimise the burden on system administrators, the users needed to take some of the administrative load. To meet the requirement that users not be given additional work, this had to be done in a way that utilised normal workflow operations. These requirements meant that the simple team-based concept would have to be expanded to allow people other than those on the team to assist in certain circumstances. The choice of using Oracle was made purely because the DHHS already uses Oracle. An Oracle-based implementation consequently had a better chance of being implemented in the State's public hospitals.

One of the positive factors in the hospital domain is the professional ethical environment that already exists. Clinicians are well aware of their duty to maintain patient confidentiality. Peer review processes and evidence-based practice are also now commonly used within the health system. The solution proposed here seeks to make use of these positive factors to provide a system which is highly usable and easily managed by the clinicians who will have to use it. This thesis promotes the view that in such environments, there is fundamentally no reason why informed users should be more incompetent or unethical in their management of access issues than system administrators. The solution should therefore allow system administrators to take a back seat as far as hands-on control is concerned. Their role in such a system would be one which entails role management, monitoring and auditing rather than direct control.

The first thought was to use an existing access control model to facilitate the solution. It was found that while a number of existing models showed potential, no one model was able to meet the access control requirements of the system. Most of the existing models, including the popular Role Based Access Control (RBAC) model, rely on specifying access privileges in advance. This can be a very complex task. The volatile nature of the team situation and the need to satisfy all possible access scenarios meant that such models were not suitable. The solution therefore needed to knit together the positive features of the existing models with new features tailored for handling volatile teams in the professional environment.

The solution proposed here suggests a new way of thinking be employed when dealing with such professional environments. It introduces the concept of user-controlled access systems, which are complimentary to the admin-controlled and owner-controlled access systems characterised by the well-known Mandatory Access Control (MAC) and Discretionary Access Control (DAC) models. The thesis goes on to detail a user-controlled access model called Trusted Access Control (TAC) and to show how it can be used as the foundation for a higher level model called Professional Access Control (PAC). It then explains how the PAC model was implemented in Oracle.

While the solution was aimed at meeting the requirements in the local domain, it is of such a general nature that it can be used in other hospital domains. There is also potential for it to be extended to other domain types where volatile teams are used. This thesis also raises some fundamental access control issues and proposes two new access control models. The fundamentals of the models are covered, but the technical details, such as the mathematics, are not.

Before discussing related access control work in the third chapter, detail is provided about the health and hospital domain in which the system is designed to work. The subsequent chapter outlines the method used in the model and implementation developments. The results of the project are laid out and discussed in the fifth chapter, while conclusions and further work are given in the final two chapters.

Chapter 2

Background

2.1 Patient Medical Records

Traditionally medical records have been stored on paper, with images such as X-Rays stored in hardcopy form in formats such as PACS. Records are increasingly being stored in digital formats, providing various storage, distribution and access advantages (Coiera 2003, pp. 118,119). Increased security requirements come with these advantages. The important consideration for hospital records is that users need to be restricted to seeing only relevant records, but they need to be able to access those records easily.

2.1.1 EHRs, Data ownership, Consent and Duty of Care

It is worth noting that much work is being done to try to come up with a national system for using and storing individual patient records, termed Electronic Health Records (EHRs) (Englebardt & Nelson 2002)

There is no general agreement globally on who actually owns medical records. For example, in the UK, it is considered that the patient owns their own record, whereas in the US, the practitioners are considered to own the record. Information privacy is increasingly becoming a legal consideration which institutions and practitioners must take into account. In the United States the Health Insurance Portability and Accountability Act (HIPAA) of 1996 requires that access to health information be tightly controlled. (Nanda 2004) (Reuters 2004)

In Australia, a formal position on ownership has not yet been reached, but Australia seems to be following the UK's patient ownership approach. In July 2000 The

National Electronic Health Records Taskforce submitted a comprehensive report to health ministers which proposed a uniform data protection regime across Australia with individual participation based on informed consent (NEHRT 2000). National privacy legislation which is applicable to health records has also been put into place (The Office of the Federal Privacy Commissioner 2004). The Commonwealth HealthConnect initiative (Czapski & Creevey 2003) (HealthConnect 2002) includes detailed guidelines with relation to patient consent. Patient consent is seen as important in order to gain consumer trust. (Win et al. 2003, p. 6)

While these aspects may seem irrelevant to the issue of access control in a hospital, it is worth noting that any legislation which specifies data ownership or consent requirements will affect access control requirements. Regardless of specific legal regulations, the bottom line is that a hospital collectively and practitioners individually, have a duty of care with respect to maintaining the confidentiality and integrity of patient records. Protection of a patient's records means only allowing individuals who have been granted permission by the patient to access their data.

2.1.2 Increasing Threats against Medical Records

When evaluating medical record security requirements and the cost-effectiveness of solutions, it is necessary to take a long term view of the potential threats that may come against a system. It is not sufficient just to assume that no-one is really interested in medical records. Schneier (2000, p. 66) emphasises that "medical information is about as personal as it gets". He notes genetic predisposition to disease, abortions and reproductive health, emotional health and psychiatric care, drug abuse, sexual behaviours, sexually transmitted diseases, HIV status, physical abuse as sensitive personal information. Patients have a right to expect that their medical records are adequately protected. This is increasingly important as the amount of digital data stored increases.

Medical records are already a target for attack. For example, an attacker broke into the University of Washington Medical Center and downloaded confidential information on thousands of patients (CSD 2001a). The report went on to say, that according to experts in the field, "medical records are a veritable treasure trove for those who would make the patients victims of identity theft." Another report from

the United States documented the recent hacking of 5,000 administrative patient files from one of the country's top hospitals (CSD 2000).

Identity theft is a high-growth area of crime (Schneier 2000). The collection of personal information is becoming a major industry in itself. In the United States, the Federal Trade Commission (FTC) identified 200 firms - 175 of them on the Web - that offer to collect personal financial information and then sell it to third parties, in violation of federal law (CSD 2001b).

It is worth noting that not all attacks necessarily come from the outside. Kropp & Gallaher (2001) reported that a significant number of organizations have experienced an insider security lapse, costing an average of about a quarter-million dollars per incident. There is also the possibility that evidence of medical errors can be erased by insiders if integrity controls are not present.

2.2 Hospital Environment

A hospital ward is a dynamic environment in which patients, visitors and staff constantly come and go. Staff members have many different roles – administrative, clinical and service related. Practitioners may work directly for the hospital, be in private practice, or be contracted to the hospital from another organisation. To add to the confusion, all these different categories of people are mingled together.

Conditions on different wards can vary significantly due to the nature of the ward. While hospital wide management practices exist, practices may also vary from ward to ward.

Hospital staff members are generally busy and tend to focus on the delivery of clinical care. They can often be frustrated by administrative tasks and generally do not consider IT security as an important or primary issue. This is evident when it is common practise to do such things as placing computer passwords on sticky notes attached to monitors.

2.2.1 Patient Movement

In the course of their treatment, patients are often moved between wards. It can be that they are transferred to a more appropriate ward according to their current condition, or that they temporarily go to a service unit, such as the X-Ray unit, for a short term procedure. While these movements occur, at any one time a particular patient is considered to be under the care of a particular ward.

2.2.2 Staff Movement

Each staff member is basically associated either with a particular ward (or wards) or a particular care unit. As such, some staff members are generally static in that they work a whole shift in one location, while others are not – they can care for patients in a number of different locations. There is also a degree of flexibility within the work structure in that staff can be moved from quieter to busy wards as needs are determined. As well as this there is the shift-based nature of the work to consider.

2.2.3 Openness to the Public

Many places in a hospital are open directly to the public. In other places certain visitors are allowed, maybe at specified times. There is little to stop an individual with intent from going virtually anywhere within the hospital.

2.3 The Professional Environment and Ethics

The American Heritage® Dictionary of the English Language (AHD 2000) describes the adjective professional as “characterized by or conforming to the technical or ethical standards of a profession”. The question is, “Should the existence of a professional environment be considered when designing an access control system?”

Spencer et al (2003, p. 4) investigated telephone, paging and information systems in a hospital emergency ward and concluded that a well-planned intervention design was required which takes into account organisational, professional and team factors. They point out that communication and information requirements of clinical teams need to be understood within the organisational and cultural context in which they

occur. This study shows that it is important to consider the professional environment.

Computer security is much more than just a technical IT access problem. The largest security weaknesses in IT systems are the people in the system. Schneier (2000, p. 255) goes as far as saying, "People often represent the weakest link in the security chain and are chronically responsible for the failure of security systems". Mandatory Access Control is often thought of as being foolproof – but what if the system administrator is corrupt or a system user has obtained access by using fraudulent identification? In fact, all system users, with the exception of hackers, initially gain system access by completing some human-to-human identification/ authorisation process. In many cases this stage can be the greatest security risk.

The vast majority of Health professionals pride themselves on their ethical practice. Being a professional entails conformity to regulations, professional codes of behaviour, and relevant organisational policies. Breaches of these standards can result in severe personal repercussions. The very reason why it is uncommon for system administrators to be corrupt is because they are professionals and there are consequences if they are found wanting. There is no reason why health professionals should behave more irresponsibly given that they are informed of their responsibilities and that the system supports appropriate security policies.

In fact, the hospital environment is one of high ethical standards where it is easy to facilitate peer review systems. It is, however, important that hospital policies and staff professional development includes appropriate IT security requirements (Webb et al. 2003).

2.3.1 System Admin vs Professional Decisions

With paper records it has been traditional for hospital administrators and health professionals to guard access to patient records. The patient record is generally left with the patient in the hospital and any access to it is normally in view of the patient or other staff members. In contrast, with computer systems, it has been traditional for the system administrator to have control over who accesses which records. The emergence of digital health records therefore creates a "professional control" issue.

Who should manage access to digital health records – the system administrator, the health professionals, or both?

There are two key issues here; what is meant by maintaining the security of patient records; and the importance that is associated with providing protection. Pfleeger (2000, p. 9) identifies the three primary qualities of data security as confidentiality (preventing unauthorized disclosure), integrity (preventing unauthorized modification), and availability (preventing denial of authorized access). IT security theorists and system administrators tend to place an emphasis on the aspects of confidentiality and integrity. For example, many access control models have been constructed which deal with confidentiality and integrity, but few if any, have dealt with availability. Health professionals, in contrast, place a high emphasis on having all the relevant information about a patient available, so that they can make informed clinical decisions.

The professional control issue can be compounded if conflict over the importance of availability versus confidentiality and integrity are not resolved. Health professionals naturally tend to see limiting availability to records as a threat, while system administrators naturally see a need to restrict confidentiality or integrity breaches as they reflect badly on their professionalism. There is obviously a need for policies which clearly spell out priorities in these areas. On balance, it seems that that the vast majority of patients are clearly more concerned about their health than the security of their records. Therefore, an access control model which makes availability a priority is preferable.

The ideal access control model would, it is suggested, give health professionals 100% guaranteed access to all relevant records while maximising confidentiality and integrity safeguards. This all needs to be done in a fashion that minimises implementation and running costs, and maximises system usability.

2.4 Implementation Requirements

Health budgets are stretched and new systems obviously need to be cost effective. Studies have identified usability as a key issue in the success of new IT systems

(Englebardt & Nelson 2002, pp. 322-324) (Lam 2003, p. 4). This aspect is magnified when users are already busy. Security is also dependent on usability. Schneier (Schneier 2000, pp. 260-262) states the most insecure system is the one that is not used and that such a system is often not used because it is just too irritating. He goes on to say that smart security designers know that users find security measures intrusive and that they will work around them whenever possible. It follows that a system which uses day to day workflow to enforce security, hiding it behind-the-scene, will be more successful that one that increases user workload.

Englebardt & Nelson point out that usability is multidimensional and includes:

- Ease of using an application;
- Ease of learning;
- Ease of remembering interaction methods;
- User satisfaction with system use;
- Efficiency of use;
- Error-free/error-forgiving interactions; and
- Seamless fit of an information system to the task(s) at hand.

Implementing a system can be a complex task that is more than just technical. Webb et al. (2003), Doolan (2003), and Parry & Tucker (2003) list key considerations as:

- Taking the perspective of the clinician into account;
- Developing competencies relevant to the application;
- Organising and implementing a training program to fit in with workforce constraints and the need to maintain client services;
- Policy Development;
- Consideration of the underlying processes and workflows in their entirety and within the relevant context;
- Developing and continuously updating a migration management plan that focuses on maintaining clinical productivity and quality;
- Building, monitoring and maintaining momentum, principally by engagement key members to “driver” changes.
- Encouraging innovation; and
- Establishing risk management processes.

These useability and implementation issues show the considerable advantage that a simple system can provide. Past experience of implementing a complex middleware solution (Hartnett 2002) has also highlighted the need for simplicity. To this end, the implementation developed uses an off-the-shelf product that is already in use in the target domain (Oracle is used by the DHHS). In consideration of the importance of taking the perspective of the clinicians into account and on providing user satisfaction, the solution provides 100% availability of patient records. In any case, a simple, usable, and efficient system is better than no system at all – which is effectively the case at present in the state's public hospitals.

2.5 Hospital Environment Pros and Cons

To conclude discussion of domain conditions, it is worth noting the major pros and cons that preliminary research and personal communications with health practitioners unearthed. On the positive side:

- A strong ethical environment exists;
- Computers are already used for everyday work;
- A team environment is established;
- Peer review principles are in place; and
- Computers are increasingly being used for medical communications.

On the negative side:

- Different procedures exist in different wards/units;
- The environment is rapidly changing (volatile);
- The domain is open to outsiders;
- There is generally a poor understanding of computer security issues;
- There is often a poor view of computer usefulness;
- Implementation cost tends to be a big problem; and
- There are many complex access control cases to consider.

2.6 Discussion

2.6.1 Who Makes Authorisation Decisions?

Consider the three basic types of people in the hospital environment in terms of the IT system. They are the System Administrator (sys admin), the User (health practitioner), and the Owner (the patient generally). Authorisation decisions are made by all three of these classes of people. For example, the patient can consent for a health care worker to access their record. A health care worker can, usually with the patient's consent, refer the patient to a specialist. The sys admin can authorise a user to perform certain accesses.

For the sake of efficiency it is better if the person making the authorisation decision enables the access in question. As the patient does not have any input into the system, someone has to implement the accesses which they authorise. The worker closest to the scene is the health practitioner currently dealing with the patient. Wherever it is reasonable for them to do so, they should implement the patient's decision. Similarly, when a fellow practitioner needs to be authorised to perform an access, the closest person to the scene is generally another user. For maximum efficiency they should be enabled to make the authorisation. Consequently, efficiency is maximised if the users can authorise each other to perform accesses. This, in most aspects, reflects how the paper-based record system operates. It is therefore reasonable to postulate that changing to a digital system that reflects the paper-based system will maximise usability and implementation efficiency. System administrators should be left to do the high level tasks rather than being dragged onto the wards to do the patient-user and user-user authorisation tasks.

Kropp & Gallaher (2001) state that, "The effectiveness of access control systems can be measured on two criteria: reliability of security and ease of administration." Administering authorisations at the point of the authorisation decision certainly provides the easiest solution, provided that the process is simple. It is therefore just an issue of whether or not a sufficient level of security can be achieved. It is suggested in this thesis that, with the existence of professional ethics in the workplace, this should be achievable. There will inevitably be a trade off between

usability and security. It is a matter of maximising the benefits to find the correct balance. Ideally, normal workflow mechanisms can be used to make the authorisations happen. For example, access for a clinician to a patient's record can be triggered by the patient's admission to a ward where the clinician is working.

2.6.2 Business Rules and Policies

In order to have an authorisation system that works efficiently it is important, for the reasons outlined previously in Section 2.3.1, that everyone works together towards a set goal. Thus, it is vital that hospital management take the responsibility for making appropriate policy decisions and ensuring that all staff members honour the policies developed. Policies must clearly define user responsibilities, detail steps to deal with security breaches, and provide mechanisms for policy reviews and upgrades.

As mentioned previously, system administrators should ideally be left to deal with the high level authorisation tasks. These would include overall user management, system auditing, breach detection and response. Administration should concentrate on efficient monitoring rather than hands-on authorisation tasks.

The requirement for guaranteed availability comes at the cost of having to allow emergency accesses where prior access privileges cannot be given in advance. The potential for abuse of the emergency access mechanism necessitates that breach detection and response mechanisms be built into the system. The volatility and unpredictability of the environment dictate that the response mechanisms must be flexible enough to deal with unforseen occurrences. It is therefore necessary that the solution not be prescriptive in nature or require complex privilege reassignments in reaction to every unforseen situation.

Rather, it must be seen that a part of the solution is to manipulate user behaviour through policy incentives and disincentives, and personnel management processes. An adequate security solution cannot be solely built into the access control system. The IT access control system must work in tandem with hospital policy to achieve the required security outcomes. The proposed solution recognises these facts.

2.6.3 Summary

Research into the hospital domain indicates that the solution should:

- Guarantee user access;
- Not increase user workload;
- Minimise the burden on system administrators;
- Be incorporated with normal workflow processes;
- Be flexible enough to cope with volatility and unforseen circumstances;
- Take advantage of the professional environment;
- Take the perspective of the clinician into account;
- Be easily implemented;
- Not be prescriptive in nature; and
- Work in tandem with hospital security policies.

Chapter 3

Related Work

This section deals with relevant practical and theoretical work in the area of access control. Before going on to look at individual models and principles, the basic access control terminology used throughout the thesis is introduced.

3.1 Basic Access Control Terminology

In order to contrast existing access control models and to introduce the new models in an understandable fashion, each model will be represented in a similar, related diagrammatic form. This section introduces the diagram format and briefly summarises basic access control principles.

There are three basic categories of people involved with a computer system – System Administrators, Users and Owners. These are represented by the three stickmen in Figure 1. In the remaining model figures circles represent individual players.

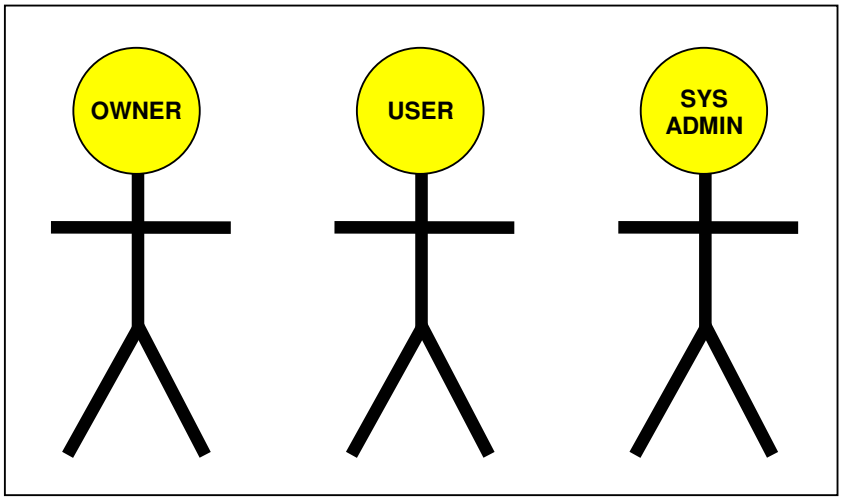

Figure 1: Player Categories

Figure 2 lays out the basic relationships between the players and the computer system. A system administrator manages the system. The system contains objects (for example, data), each of which has an owner. Users are players who seek to access the objects. System administrator, owner and user are the generic access control terms for the players. In a real domain the players are usually known by other terms. For example, in the hospital domain, system administrators may be known as IT services personnel, the owners as patients or clients, and the users as health practitioners or staff members.

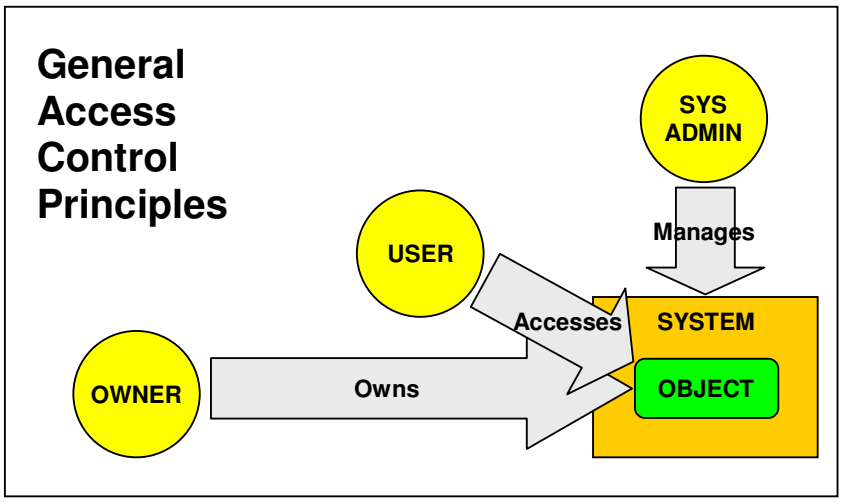

Figure 2: General Access Control Principles

3.2 Mandatory and Discretionary Access Control:

MAC and DAC are both fundamental access control models. They have been around for several decades and form the basis for most access control implementations.

MAC invests all the power in a central authority, usually known as the system administrator. As depicted in Figure 3, the sys admin determines who can access which objects. In most systems different access *privileges* are specified, such as READ, WRITE, APPEND… The sys admin assigns privileges to each user-object (often termed subject-object) relationship.

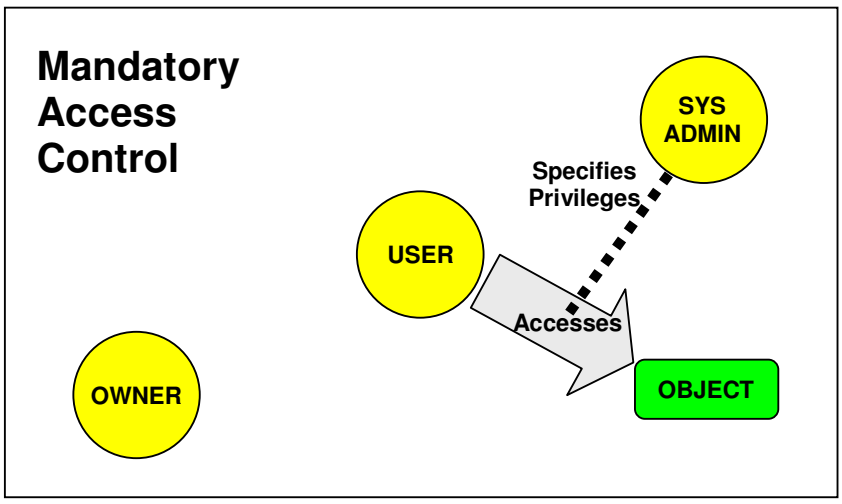

Figure 3: Mandatory Access Control

At the time of MAC's inception computer systems were much smaller, so it was a practical solution for system administrators to manage all the user object accesses directly. Even today many systems can be managed adequately with MAC. However, with the emergence of larger systems with more users and with systems becoming more volatile in terms of user movement and access change requirements, management is much more demanding. Many of the models described in the rest of this section are extensions to the MAC model, which are designed to cope with the increased administrative burden of large systems.

While the ability to cope with more complexity has increased, the authorisation responsibility still rests with system administrators. This essentially means that a system administrator has to be involved with all accesses, specifying necessary privileges to users in advance of them making an access. In volatile situations such as those present in the hospital environment, it is often impossible to predict in advance which privileges are necessary. For example, in an emergency a doctor may need to access a record to check for the presence of adverse drug reactions or specific conditions. If the doctor has not already been granted access rights to the patient's record then there is a major problem. If allowances for such occurrences are made beforehand then many unnecessary access privileges have to be assigned, opening the system up to a greater degree of risk.

DAC, in contrast, gives the owner the ability to grant access permissions to users (see Figure 4). DAC and MAC can both be applied in a system (Pfleeger 2000, p. 290). If this is done MAC takes precedence. In many domains a DAC system can provide a useful degree of flexibility, lessening the demands on system administrators. There is however, a basic need for the owner to also be a user – so that they can physically grant the access permission. While it could be technically feasible to grant patients some form of ability to interact with the system in order to make a grant, it would seem impractical. The need for patients to be present, conscious, cooperative and capable can introduce obstacles that are difficult to overcome. Consequently, in the Australian health domain and similar domains, DAC seems of little use for managing user accesses.

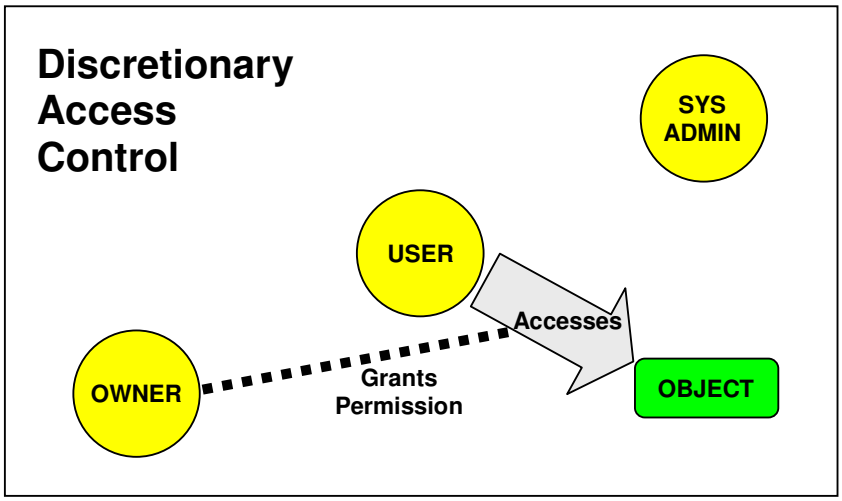

Figure 4: Discretionary Access Control

On the other hand, in cases where the object is owned by a user, DAC can be useful. If the American approach to the ownership of medical records were in place (where the practitioners own the records), DAC may prove useful. As this does not seem to be the way the Australian system is going, it is of little point considering DAC as a useful tool.

While it seems largely irrelevant to the domain considered here, there is a need to describe DAC. A third fundamental access control model, TAC, is proposed. It is important therefore, to be able to compare and contrast TAC with both MAC and DAC.

3.3 Role Based Access Control:

Role Based Access Control (RBAC) was formally described by Ferraiolo & Kuhn (1992) and has since evolved to a stage where it has been incorporated in many commercial systems, including Informix, Oracle and Sybase (Ramaswamy & Sandhu 1998). In February 2004, the National Institute of Standards and Technology (NIST) RBAC Model was adopted by the American National Standards Institute, International Committee for Information Technology Standards (ANSI/INCITS) as ANSI INCITS 359-2004 (NIST 2004).

RBAC is based on MAC and as such all authorisation powers reside in the system administrator. As shown in Figure 5, RBAC requires that access privileges are assigned to abstractions called 'roles', which are created by the system administrator. The system administrator specifies the privileges for the role and can allocate one or more roles to each user, as required. The administrative burden in RBAC, compared to MAC, is greatly reduced because privileges are not required for each subject-object relationship. The success of RBAC is mainly due to the fact that the concept of roles fits closely with business practice and that it has administrative advantages over older methods.

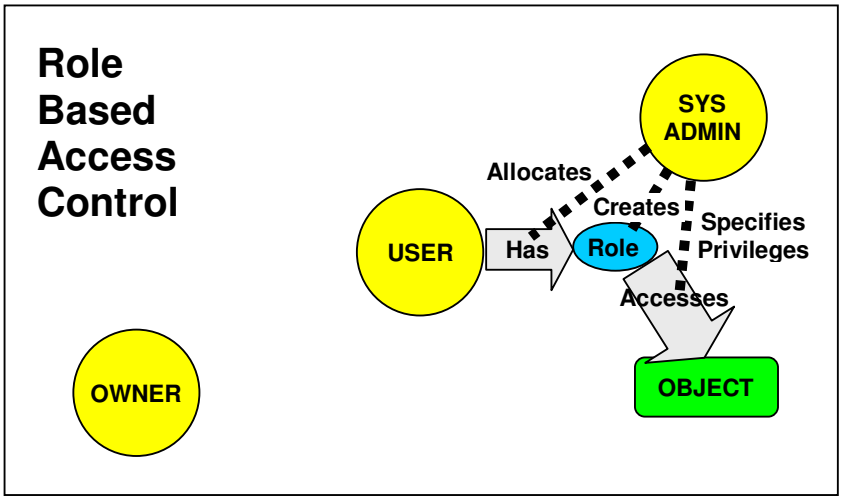

Figure 5: Role-Based Access Control

There is much research being conducted into extending RBAC to get it to provide additional functionality. Of particular relevance, Neumann and Strembeck (2003) have described how context constraints can be used to check predefined conditions to

determine whether an access request is to be allowed. Such things as time and location can be used. This idea has considerable merit when it comes to implementing a real time authorisation process. An interesting approach could actually be to use roles themselves as context constraints.

One of the security concerns relating to RBAC is the possibility that a user with multiple roles may be allowed to commit fraud or make errors due to the privileges of each role. Botha and Eloff (2001) show how the principle of separation of duties can be satisfied in an active access control environment, to overcome this concern. The term "active access control" relates to the ability of the system to make "just-in-time" authorisations which take the context into account.

In order to try to reduce the demands on system administrators when it comes to dealing with volatile situations, variations of RBAC can delegate authorisation tasks to certain users. While this is seen by many as a panacea to solve problems, it suffers from two major flaws. Firstly, delegation of administrative authorisation tasks to users sets a dangerous precedent, as their primary jobs mean that they may tend to make unnecessary authorisations for reasons of expediency. If the delegation of administrative responsibilities is done without the knowledge of the user there is added danger. The inherent positives of RBAC are due to the fact that the roles given to users reflect their workplace responsibilities and qualifications. Any RBAC implementation which assigns administrative responsibilities to non-administrative personal takes the concept of roles beyond its proper place! Secondly, and most vitally, the RBAC process still requires that authorisations are made prior to when they are needed. While this delegation process puts the authoriser closer to the point of decision making, the fact is that it is still not necessarily at the point of the decision making.

More generally, while RBAC is a good solution for easing the burdens on system administration, it still fails to be flexible enough to meet the demands of volatile environments, such as those present in hospitals. RBAC is however, a useful tool that can be used to restrict what users can access within the object or set of objects to be accessed. For example, clinical users could be restricted from accessing the contact details of patients or even patient names, if they wish to remain anonymous.

3.4 Team-based Access Control:

TeaM-based Access Control (TMAC) is a system that is built on top of RBAC and is fundamentally a MAC type model. It attempts to mould RBAC to be more flexible by using an abstraction for user groups called teams.

Figure 6 shows only the basic TMAC concept with the foundational RBAC role abstraction supplemented with team and context abstractions. Users can make RBAC accesses in the normal way. In addition, if they are part of a team and the context they are in is appropriate, then they can access the other objects. There is still a basic reliance on the system administrator to perform the relevant administrative activities, although these can be delegated in normal RBAC fashion.

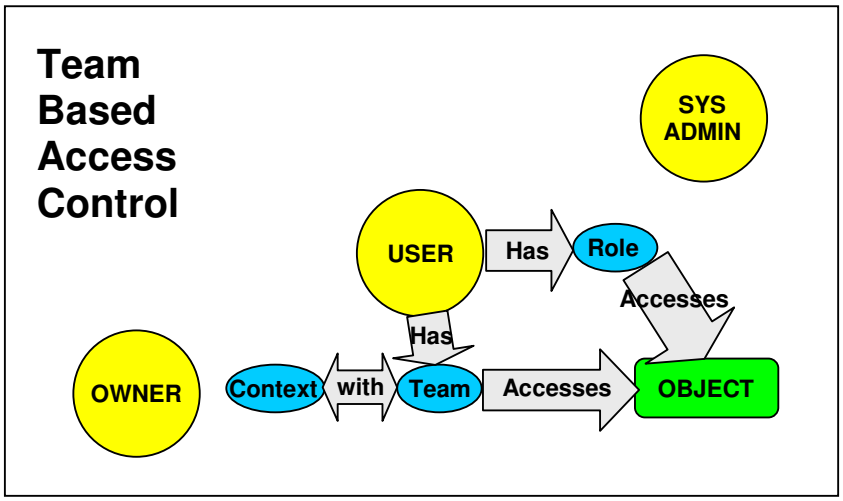

Figure 6: Team-Based Access Control

3.4.1 TMAC Versions

The various versions of TMAC, while suffering in a number of ways by their fundamental ties to RBAC and MAC, offer a number of useful insights. Three significant versions of TMAC are outlined in this section.

Thomas (1997) introduced the notion of TMAC and described how the advantages of RBAC could be coupled with the concept of restricting access only to relevant clients. This reflects the major benefit of using a team-based approach. He also indicated how the framework could be made to be self-administering by "trapping basic calls by the host information system to assign and de-assign team members" -

the purpose of this being to reduce the administrative overheads. Again, this is a worthy ambition. Unfortunately, the cost in using this approach is to turn users into quasi-system administrators.

Georgiadis et al (2001) extended the TMAC model proposed by Thomas to use contextual information, such as time and location. The idea of using context constraints is a useful concept that may enable a wide range of access control solutions. Georgiadis and his team developed a model called Context-based Team Based Access Control (C-TMAC) which uses authorisations that are activated at the time that they are needed. The group programmed a "view-based active access-control system", using the C-TMAC model, for use in healthcare environments (Georgiadis et al. 2002).

The most recent incantation of TMAC was TMAC 2004, by Alotaiby & Chen (2004). It is fairly similar to previous versions in nature, but does not specify team specific roles as the other versions do. Rather, it uses the already specified organisational roles. This seems like a good step as team roles appear to introduce a high level of complexity for no significant gain.

3.4.2 TMAC Deficiencies

While the TMAC variants has some good points, such as introducing the very idea of team based solutions and the use of constraints to aid active access, they still suffers from a number of drawbacks. There is a fundamental problem in their notion of what constitutes a team. TMAC teams are relatively inflexible entities, requiring someone with system admin privileges to change team composition in advance of any necessary action. A hospital clinical team would generally be made up of the staff on a ward - the whole team having access to all the patient records for the ward. This represents a many-to-many owner-user relationship (see Figure 7).

In contrast, by allocating a separate team for each patient, the *Team-per-Patient* approach uses one-to-many owner-user relationships. It provides a more fine grained security solution which cuts down the number of allowable accesses. In the small example in Figure 7 the number is reduced by a quarter. With a ward of 20 patients and 9 staff, with an average of 3 staff per team, the number is reduced by two thirds

(from 180 to 60). The team-per-patient approach is also more beneficial for workflow management purposes because it captures the particular individuals who are working with each patient.

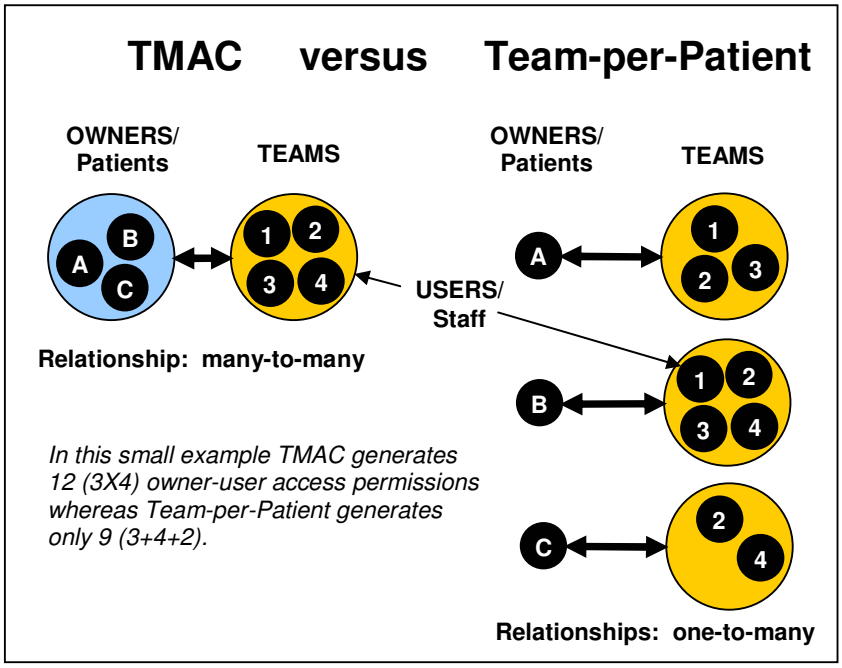

Figure 7: Team Concept Differences

The question of who adds and takes away users from teams is another concern. In order to have a flexible system, team changes need to be made quickly and efficiently. This cannot happen if the system administrators are the only ones controlling team composition. The only way that TMAC can get around this is to delegate these administrative tasks to team members, which then begs the question of whether the delegates are actually qualified to do such tasks. Would it be alright to give system administrators the right to perform clinical procedures? The concept of giving clinicians privileges designed for system administrators is just as bizarre! It may appear to solve the problems imposed by RBAC's prescriptive nature, but it is fundamentally flawed. Users must be treated as users and system administrators as system administrators! While the solution proposed in this thesis in fact gives users more control, it requires that system administrators monitor how they perform. It thus recognises the inherent qualifications and responsibilities of the players, a thing which, ironically, these TMAC role-based solutions have deviated from.

Another problem with TMAC is that it is still tied to the RBAC/MAC idea that authorisations must happen before they are required. The whole problem with this approach goes back to the fundamental security myth that confidentiality and integrity are always paramount. Availability seems not to be of concern when it is, in many cases, the primary thing required! As with the requirement that staff with system admin privileges are required to change team composition, the need for authorisations to be made in advance, introduces added inflexibility to team management procedures.

In conclusion, when looking from a distance, TMAC appears to offer a viable solution, but when the details are inspected, the fundamental floors in the approach are revealed. Role-based solutions are no panacea!

3.5 Task-Based Access Control:

Task-Based Access Control (TBAC) is another active access control model which has been proposed (Thomas & Sandhu 1997). It tackles access control from a workflow perspective and provides mechanisms for the management of permissions. While this concept has merit when considering the problem from the perspective of task management, this thesis is concerned with just access authorisation.

It was useful to look at TBAC because it shows that it is possible to build access control into workflow applications. If workflow applications can be used to trigger access control mechanisms, the administrative burden will be greatly reduced.

3.6 Organisation-Based Access Control:

Organisation Based Access Control (ORBAC) is another proposal which introduces some interesting concepts. ORBAC was proposed by Kalam et al. (2003). The main entity it defines is the *organisation*, one of which can represent an entire organisation, with others representing the units within it. It uses contexts to determine where organisations may grant role permissions to perform activities on views. This model is quite comprehensive and is of interest because it treats the problem of dealing with volatile situations on its merits without simple trying to

extend RBAC. It deals with the need for users other than system administrators to specify permissions. It is however a great deal more complex than the solution that is required.

3.7 Provisional Authorisation Models:

Provisional authorisation models (Bretan 2004) do not seem to be in the mainstream. Critically though, they introduce a mechanism which is ideal for dealing with emergency accesses where there is no time to grant permission for an access in the usual way. While other models simply accept or deny a request, provisional authorisations deal with the grey area between acceptance and denial. They allow a user to make an access subject to them taking some security action such as signing a statement (see Figure 8).

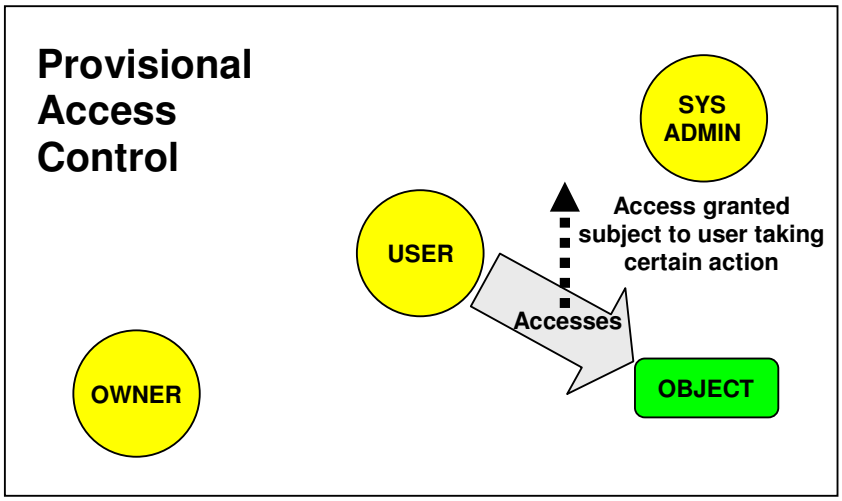

Figure 8: Provisional Access Control

Kudo (2002) proposed the Provision-Based Access Control model (PBAC). PBAC is a specific example of a provisional access control model (the general term). This model uses the "access control framework", an international architecture standard, as its basis. It also uses normal hierarchical authorisation structures. The relevance of this work is that it shows that a retrospective authorisation process can be built into a hierarchical authorisation system. That is, retrospective authorisation can coexist with context and role-based mechanisms.

3.8 Auditing:

The concept of auditing is an old one (Schneier 2000) (Stallings 2003). While it is a retrospective activity, its implications as an access management tool need to be considered. If users know that their accesses are audited and that breaches of organisational policy will indeed be observed and dealt with, then there is an element of "mind control" influenced over users. In some cases management may consider that this is all the access control that is necessary.

It is proposed in the solution provided to use auditing as a primary tool for handling accesses which are not made by team members. This is viable in ethical environments which are capable of delivering a high degree of user compliance. Provisional authorisations will be granted to such users subject to them providing (or agreeing to provide at a later time) an access reason or report. Systematic auditing of these reasons and reports provides the required retrospective system control. For most accesses that require it, the auditing will initially be done by colleagues, taking the load off system administrators. This type of auditing will be done as part of the daily workflow as a communication feature. As such, it will not generate any extra work

In general terms, RBAC type active access control solutions seek to delegate access permissions down through the role based hierarchy, from top to bottom. In contrast, the proposed system percolates auditing responsibilities from the bottom up, leaving only the most critical auditing tasks to the system administrators.

The key for enhancing security whilst using auditing is to place restrictions on the allowed number of non-standard accesses. While total system security is not guaranteed by such a technique, its effect in reducing the number of policy breaches when compared to a system that has minimal security should be substantial.

3.9 Clark-Wilson:

The Clark-Wilson model (Gollmann 1999, pp. 56-58) introduced the idea of "well-formed transactions". Under the model, users do not have direct access to stored data. Rather, they are given access to procedures which interact with the data. The Oracle implementation outlined in this thesis makes use of this concept. The ability to store procedures in a database and control access to the procedures provides a good security-based option. Oracle provides the ability to produce a total database package, with all the functional requirements and security built-in. The package can then be used as a foundation to produce new workflow applications or provide the necessary access control features to existing systems.

3.10 Middleware:

Middleware has been seen as a solution for adding access control and consent features to existing and new systems (Woodcock & Gillies 2003) (Hartnett 2002). While such systems have great potential for providing the appropriate security, they fall down in the area of complexity. They tend to be seen by organisational decision makers as being too complex. It could be that they are ahead of their time in that there is merely not enough demand for such sophisticated security solutions at this time.

The failure of such systems to be implemented successfully in real-world situations has been a major motivation for seeking a simpler solution such as the one proposed here. It is envisaged that organisations are more likely to accept solutions that employ well-known and tested off-the-shelf products such as Oracle.

The middleware solution, while not being suitable, does introduce the idea that the software needed for access control should be modular in nature and kept separate from workflow applications. When this idea is coupled with the ability of database products to store procedures, it can be seen that a single database toolkit can be designed to provide access control functionality.

3.11 Review of Existing Models

The main requirement for the system is to guarantee availability. In emergency situations in hospitals it is possible that clinicians will need immediate access to patient records without any other person being available to grant permission for the access. It is not suitable to cater for this rare occurrence by granting all clinicians such access by default. This means that any model which does not facilitate accesses without prior grant being made, is not suitable. Both MAC and DAC based models require such grants and are therefore not suitable as the basis for the volatile team-based model that is required.

The only models which do meet such requirements are the PBAC model and auditing. PBAC allows access subject to a further security action being taken by the user. While PBAC functionality can be included in the required model, PBAC alone does not solve the issues of restricting access by individuals to objects based on their qualifications and function within the organisation. It also does not provide an efficient way of easing the burden on system administrators.

Auditing is similar in nature to PBAC, but essentially requires personnel to continually be monitoring the system. The benefits and effects of auditing are well known. Of particular relevance is the deterrent factor that it provides to users who may be tempted to flout security requirements. PBAC provides the potential to deploy an active auditing mechanism which can do most of the day to day auditing tasks. If a way can be found to incorporate a peer-review process into such a mechanism, then the burden on system administrators can be substantially reduced.

While RBAC, being based on MAC, has been discounted as a mechanism for administrative grant assignments, it still can serve a useful purpose in restricting access for individuals based on their role within the organisation. This aspect of RBAC can therefore be employed in the required model. It is useful to note that the guaranteed availability requirement highlights the weakness of RBAC in being a useful tool to use for the granting of administrative privileges, particularly in volatile situations. In contrast, RBAC is good at restricting individual access based on role. It is therefore important to use RBAC for what it is good at, and not to view it as a panacea for all access control requirements.

TMAC based models are deficient due to their inflexibility in dealing with the management of team composition, and on the granularity of access control that they can provide. A one-to-many owner-user relationship was shown here to be more beneficial. The TMAC 2004 model did however show, as did ORBAC, that using organisation wide roles was the best way to proceed and that context constraints can be a useful tool.

TBAC and aspects of the TMAC models showed that it is possible to use normal workflow operations to trigger access control functions. Such functionality allows access control to take place behind-the-scene. This is beneficial in minimising the burden on system administrators.

It is interesting to note that one of the aims of many of the models based on RBAC, such as TMAC, is to come up with a "self-administering" model. What they are really attempting to do is to define a user-controlled access system based on MAC. Surely it is more appropriate to base a user-controlled system on a foundational access control model which specifies the scope of user control!

The middleware solution points the way to using stored procedures within a database to provide access control functionality. The Clark-Wilson model inturn provides a powerful way to control the confidentiality and integrity of stored data, by introducing the concept of well-formed transactions. Both these concepts are worthy of consideration.

In summary, all the models covered in this section provide useful concepts which have the potential to solve part of the problem. It is therefore a matter of determining an appropriate way of knitting all these positive features into a solution that is efficient and workable.

Chapter 4

Method

This chapter outlines the steps taken in the development and testing of the TAC and PAC models and the resulting Oracle implementation. Figure 9 shows the steps in a flow diagram. Each step is covered by its own section in this or the next chapter.

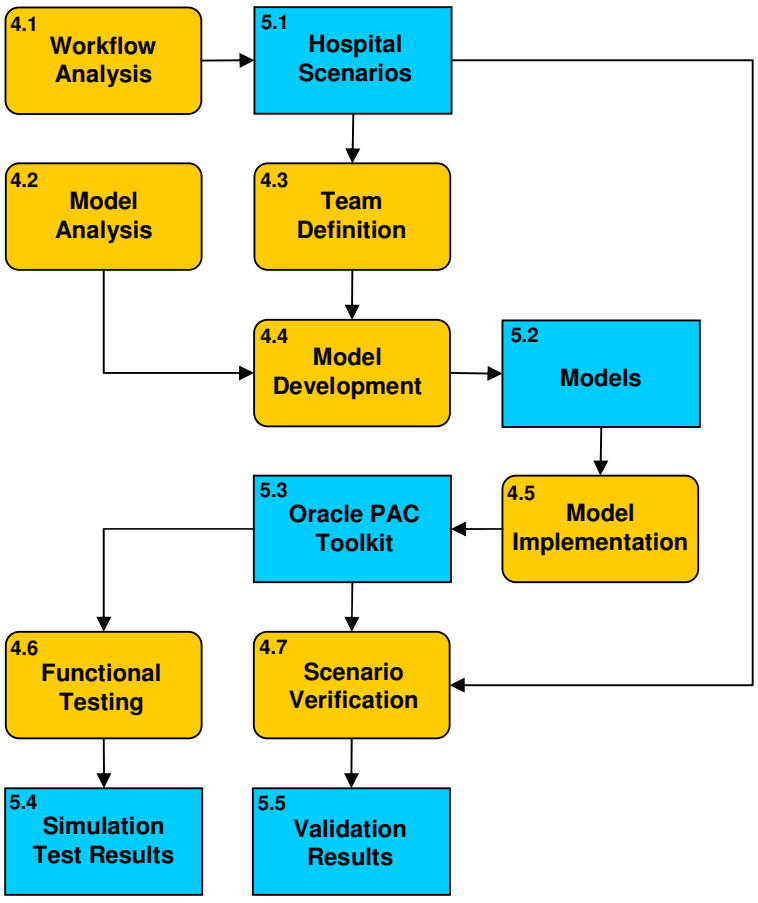

Figure 9: Method Flow Diagram

In Figure 9, the blue rounded boxes represent the tasks that were undertaken in the process. The orange rectangular boxes show the resulting items produced. The numbers in the top corners of the boxes show the section which covers the item.

4.1 Workflow Analysis

After preliminary research gave a broad understanding of the domain, it was then possible to establish the detailed requirements of a hospital access control system. This was done by interviewing health practitioners, administrators and IT managers. As well as getting a feel for practical implementation issues, the purpose of this was to establish as many relevant and distinct hospital requirements, in the form of scenarios, as possible.

From time to time during the remainder of the process other scenarios emerged. These were included with the initial set. The complete set of scenarios is shown in Section 5.1.

4.2 Model Analysis

This stage of the process involved researching related work in access control techniques and models. The initial purpose of the research was to find a suitable model to meet the domain requirements. As the research proceeded, it became evident that no single existing model fulfilled all the requirements. The purpose thus changed into one of extracting useful concepts from existing models that could be incorporated into a new model.

Chapter 3, on Related Work, described each of the models which provided useful information for this part of the process.

4.3 Team Definition

A team-based access control model requires the team concept to be defined. The scenarios produced by the workflow analysis and the team-based examples researched in the model analysis were analysed qualitatively to produce a suitable team concept for the hospital domain.

Section 5.2.1 describes the team concept that resulted from the stage of the process.

4.4 Model Development

Having extracted the requirements, the next step was to establish an appropriate access control model or mechanism. This was done by looking at established models and researching newer developments. It was found that no one model in itself was adequate to meet the requirements that were demanded. It seemed a new model needed to be developed that fitted the established constraints. The existing models were reviewed in order to establish whether any of their features would be useful. An outline of the relevant models considered was provided in the Related Works chapter.

It was clear from the scenarios that the main priority was to find a way to transfer the bulk of the authorisation decision making to the users. This had to be done in a way that made access control as fine grained as possible while providing both good security and guaranteed availability.

The initial concept was to use a team based approach with a team for each patient. Members of the team would have access to the patient's record. It was clear that a model was needed where the users within a team could bring other users into the team. This simply reflected the referral process used in the health system.

The scenarios established that patient admissions basically just allocated a patient to a particular unit or ward. It was therefore necessary to find an efficient way to initialise the patient's team on their admission. The concept of a default team for each ward was established.

The research into team based approaches confirmed that such an approach was feasible. After comparing the differences in the team concepts, it was felt that the approach of one team per patient was the most practical, due to its flexibility and its potential usefulness as an abstraction relevant to workflow processes.

With the team concept established it was evident that a way needed to be found to permit accesses to users who were outside the team. In considering this it was realised that there were various levels of relationship between the members on a team and those not on the team. For example, practitioners who worked alongside fellow

practitioners were far more likely to need to access the records of their colleague's patients than those with no such association. This reflects the team work that occurs between fellow workers of the same profession. Rather than put all such workers on the team by default, it was felt that it was more appropriate to get their fellow workers to put them onto the team and to also provide them with a simple non-member access method. In addition to this, a method was established for users with no such relationship to gain access.

RBAC provided a method of limiting what a particular user could see within a patient's record. It was felt that this was a worthwhile additional property to include. Once the inclusion of role-based restrictions were included, it was necessary to include a way for users to view things that they would not normally be allowed to see. This was necessary in order to keep in line with the established requirement of providing 100% availability.

In addition to RBAC and the team based concept, the research into existing models confirmed a number of other initial ideas. Particularly, the use of context variables was confirmed, as was the concept of providing some form of provisional access. Oracle's ability to allow "well-formed" transactions was also appealing.

Details of the resulting models developed are given in Section 5.2.

4.5 Model Implementation

The initial proposal for this project envisaged using Oracle. One reason for this was that, for the sake of simplicity and acceptance, it was thought best to use a well regarded off-the-shelf product. The State Government Health Department, the *Department of Health and Human Services* (DHHS), were already using Oracle within their system. The implementation would therefore have a better chance of being trialled in a clinical environment if Oracle were used.

Research into Oracles capabilities was performed in order to confirm that it had the functionality that was desired. (Hale et al. 2002) (Nanda 2004) (Theriault & Heney 1998) Some of the useful Oracle functionality that was revealed included - single

sign on, role based access control, virtual private networks (VPD), the Enterprise Manager interface and PL/SQL Procedures and Functions.

The Oracle research revealed PL/SQL or Java as the logical language choices. Further research into PL/SQL functionality led to its adoption as the language choice (Moore 2004) (Naude 2003) (Robinson 2003). There were a number of reasons for choosing it. Firstly, it could be stored within the Oracle database, making it portable as well as easy to install and use. Secondly, the research indicated that it could be made to run faster than Java. Thirdly, role-based access privileges could be applied to each PL/SQL procedure, thus enabling well-formed transactions.

A difficult choice had to be made regarding what form the implementation should take. It was decided to concentrate on verifying that the proposed model was workable by providing the required functionality in an Oracle package, rather than producing an implementation with its own tailor-made interface. The reason for this was that the workability of the model was the primary goal. Producing an application from the functionality provided in the Oracle package would be fairly trivial and would divert attention to the interface chosen. The interface would inevitably not match that which would be used by a particular organisation. The whole point was to highlight that any desired interface can be produced or an existing one used, not to highlight a particular interface design or a particular functionality that could be developed.

When Oracle is used in practice, database administrators can use a number of built-in Oracle tools to manage the database. One of these tools is the Oracle Enterprise Manager. This tool is particularly useful in dealing with the specification of access permissions and roles. It basically provides an interface with which to manage RBAC. As such it was basically not worth rebuilding RBAC type procedures into the implementation when an interface is already there to manage those aspects of the implementation. For example, the issue of how to deal with agency staff (casual short term contract staff) can be taken care of by simply specifying appropriate RBAC roles and privileges. This can be done entirely within Enterprise Manager.

The database design was completed and the necessary PL/SQL procedures and functions listed. Once the listing was complete work began on implementing the

design. The command line based SQL*Plus interface was used to program the database and the PL/SQL functions and procedures. In order to test the procedures and functions as they were loaded, a small amount of test data was placed in the database.

Once each procedure or function was completed it was tested briefly to check that the desired functionality was achieved. Print statements were placed within the procedures and functions to show the flow of control. In this way, it was possible to determine that each possible path taken within a procedure or function was checked.

4.6 Functional Testing

The implementation consists of a set of top-level PL/SQL procedures which perform the team management and auditing functionality required by the PAC model. Behind the set of procedures lie a collection of PL/SQL functions which perform operations for each other and the top-level procedures.

As mentioned above, each procedure and function went through a brief functionality test on completion. The next testing phase was undertaken once the implementation was complete. The purpose of this phase (the "simulation testing" phase) was to test the procedures and functions rigorously on a complete set of data.

The simulation testing stage involved creating a simulated hospital with wards, administrators, health practitioners and patients, and testing the model-based procedures. As all the functions are called by the procedures, it is sufficient to assume that they are performing correctly if the procedures work correctly, as long as all the possible variations of input and hospital states are tested. The goal then was to test each procedure in a way that took into account all the variations of input and hospital states.

The scale of the project and the time allocated to it did not allow any testing in a real situation with real people. In any case, this would have been difficult, given that the implementation is basically just a toolkit without an interface. Such testing would have required the users to enter procedures with parameters in a command line interface.

It was felt that testing using a simulated hospital would achieve more than just purely testing the functionality of the implementation. It would also give some insight into how the procedures could be used in hospital based applications and on how the database performs with a reasonably sized set of data. The other positive feature of the process was that the changing relationships between patients and users could be observed. Simulation testing also verified that the database was operating correctly because the overall view of the hospital could be checked rather than just localised changes.

The simulated hospital was set up with 24 patients and 24 users on 3 wards. Scripts for creating and loading the database were used for this purpose. The users consisted of administrators, nurses, RMOs, specialists, and allied health professionals, as well as supervisors for each category. In order to check the state of the hospital at any time, two test procedures were programmed. The output of the first gave the current state of the hospital, while that of the second showed the changes that have occurred since the last hospital state change output. Both procedures produce tables (see Section 5.4 and Appendix A for examples).

It is important to note that these tables were produced just for test purposes and that they were not generated from a stored access control matrix. The implementation calculates the category of the user-owner relationship at the time of each access, by checking the user's role and unit set with the roles on the owner's team and the owner's unit.

The tables provided an ideal way to get a "whole-of-hospital" view. The team composition could be seen for all patients and users at once. Furthermore, by viewing the whole hospital it was possible to test that no unforseen events occurred in relation to data that should not have been affected by a change. For example, it was important to know that when a patient was removed, that no other patients' data changed.

4.6.1 Testing Procedure

Testing was then performed by the following process:

1. The initial state of the hospital was output using the first test procedure;
2. A set of procedures (stored in a test script) was loaded which produced text showing the flow of control; and
3. The changes from the initial state were output using the second test procedure.
4. The changes were checked for correctness.

4.6.2 Testing Sequence

The following set of tests was performed:

TEST #1: LOGON PROCEDURE – SET UP
TEST #2: ADMITTING PATIENTS – SET UP
TEST #3: LOGON PROCEDURE
TEST #4: ADMITTING PATIENTS
TEST #5: RE-ADMITTING PATIENTS
TEST #6: REFERRING PATIENTS
TEST #7: TRANSFERRING PATIENTS
TEST #8: RELEASING PATIENTS
TEST #9: REMOVING PATIENTS
TEST #10: VIEWING PATIENT RECORDS
TEST #11: VIEWING HIDDEN PATIENT RECORDS
TEST #12: REQUESTS AND REPORTS
TEST #13: VIEWING REPORTS AND LATE REPORTS
TEST #14: PATIENT LISTINGS
TEST #15: LOGOFF PROCEDURE

The tests were done sequentially with the input state for each test being the output state of the previous test. The hospital was initially empty with no patients or users. This method simulated the real world process and enabled a complex hospital state to be generated.

Verification that the tests produced correct results were done by checking that the changes to the hospital state matched with the changes that the set of procedures was expected to produce. The flow of control was also checked to ensure that it was correct.

4.7 Scenario Verification

The process of developing the procedures stemmed from the original hospital scenarios. Consequently, the procedures collectively should meet all the requirements of the scenarios. As a double check to verify this, the scenarios were listed and the relevant procedures matched to them. It was necessary just to make sure that the required functionality was provided for each scenario.

Chapter 5

Results and Discussion

This chapter contains the results that were obtained in the various stages of the project. The chief results of the project were the access control models that were developed and the verification that Oracle could be used to implement a suitable access control solution. It is worth thinking of the other results as being supplementary, in that they either led to these developments or verified their completeness. Please refer back to the flow diagram in Figure 9 to see where each of the sections discussed in this chapter fit in with the overall picture.

5.1 Hospital Scenarios

Table 1 shows the scenarios that were collected in the workflow analysis stage of the project. The *Scenario Category* column is the name of the scenario group. The *Model Requirements* column shows the task that is required in terms of model functionality. The other columns are self-explanatory.

Scenario Category	Scenario Name	Scenario Description	Model Requirements
Patient Issues	Admission	Patient is assigned to a ward on admission.	Patient assigned ward and an initial team.
	Anonymity	Patient wants to remain anonymous.	None. Use admin procedure of assigning a dummy name.
	Change Carer	Patient requests someone else looks after them.	Replace one team member with a colleague.
	Transfer Patient	Patient is moved to another ward in an emergency.	Remove patient from ward and admit to another ward.
Staff Change Issues	Staff Break	Staff member goes on a break and colleagues cover for him.	Allow access to the record and require a report back to the staff member.

Scenario Category	Scenario Name	Scenario Description	Model Requirements
	Staff Sick	Staff member rings in sick.	None. Staff log-on when they arrive.
	Staff Leaves	Staff member becomes incapacitated (sickness/injury/family emergency) during a shift.	Supervisor or Admin removes patient from teams by logging them off.
	Staff Transfer	Staff member wants a colleague to take over the care of one of their patients.	Replace one team member with a colleague.
	Emergency	Staff member must treat a patient in an emergency.	Allow access by staff who are not members or colleagues and require a report to sys admin
	Referral	Staff member wants to call in an Allied Health Professional.	Add a staff member to the team.
	Staff Removal	Staff member has finished treating a patient.	Remove a staff member from a team.
Staff Information Issues	Help Colleague	Staff member helps another busy staff member by carrying out a task which requires they access the patient's record.	Allow access to the record and require a report back to the staff member.
	Show Team	Staff member wants to know the staff members that are treating a patient.	Display a list of team members.
	Contact Kin	Staff member needs to contact the next of kin of a patient (where they normally don't have access to the required information).	Allow a staff member to access all the information about a patient and require a report to sys admin.
	Show Messages	On returning from a break, a staff member wants to know of anything that happened to their patients, but not everyone who attended the patients is available.	Display a list of current messages for a staff member.
	Contact Team Member	Staff member wants to contact a carer who is not present to get them to attend to a patient or for advice.	Display a list of team members.
	List Patients	Staff member wants a list of their patients.	Display a list of patients for a given staff member.
Administ-rative Issues	Agency	An agency staff member is employed.	None. Registration by normal admin means before log-on (different role may be applied).

Scenario Category	Scenario Name	Scenario Description	Model Requirements
	Show Ward Patients	A supervisor wants to know how many patients there are on another ward.	Display a list of patients on a ward or just return a count. Application can secure this information.
	Check Team	Administration wants to know who was looking after a patient at some time in the past.	Sys admin accesses audit logs.
	No Log Off	Staff member forgets to log off.	Supervisor or Admin removes patient from teams by logging them off.
	Admin Override	An addition or removal needs to be performed and relevant people are not available to make the change.	Admin can add or remove team members.
Security Incidents	Alter Record	Staff member attempts to alter records to cover up a medical error.	None. System privileges should only allow staff to APPEND information, not to UPDATE it.
	Unwarrant-ed Access	Staff member abuses access privileges and views patient data for personal reasons.	System admin should be alerted of suspect access through a colleagues report or direct report.

Table 1: Hospital Scenarios

5.2 Models

In this section two new access control models, Trusted Access Control (TAC) and Professional Access Control (PAC) are described. TAC is seen as a foundational access control model akin to MAC and DAC. PAC is a higher level model which guarantees availability to users. It incorporates TAC, RBAC and PBAC.

5.2.1 The Trusted Access Control (TAC) Model

In Section 3.1 the three types of system players were identified; system administrators, owners and users. Two foundational access control models were also identified; mandatory access control and discretionary access control. Mandatory access control requires system administrators to authorise accesses, while discretionary access control requires owners to authorise accesses. The two

foundational access control models therefore specify that two of the three types of system players can grant accesses. Why only two of the three? Well, only because the third has not been defined.

The third foundational access control model – Trusted Access Control (TAC) is described here. It takes the remaining alternative of a user authorising access. Figure 10 shows the three foundational access control models.

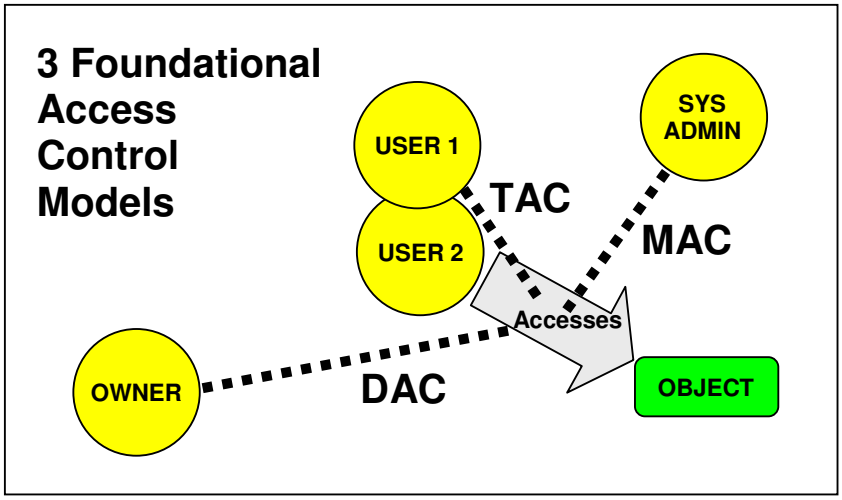

Figure 10: Foundational Access Control Models

Figure 10 shows that TAC allows User 1 to a authorise User 2 to access an object. This authorisation is conditional upon User 1 already holding the privilege of being able to access the object in question. The rule which defines TAC is, "A user with an access right can grant that access right to other users." In the health domain context, this amounts to the referral of a patient by one practitioner to another (as shown in Figure 11).

The model is named Trusted Access Control because the users are required to exhibit a high level of responsibility. In other words, they must be trusted. When the model is applied in practice, the level of "trust achieved" can be measured in terms of the rate of compliance. The required rate of compliance in a system will depend on such things as the cost of non-compliance.

TAC exists by its very definition. The nature of its definition clearly makes it a third foundational access control model. One could argue for or against its practicality or

on where it may be useful, but that is not relevant to the matter of whether it exists or whether it is foundational. The only possible argument against its definition is the possibility that it may have been defined before. The main possible argument being that it is just a subset of MAC or DAC.

5.2.1.1 The Independence of TAC

The case could be put that TAC is merely and extension of MAC and/or DAC. If User 1 is the first user to have access to the object apart from the Owner and the System Administrator, there may have previously been either a MAC or DAC authorisation to establish User 1's authorisation status (indicated by the dotted lines in Figure 11). Certainly TAC can work with MAC and DAC in this manner.

It is however, feasible that certain users or classes of users may be given this authorisation by some other specified security mechanism. For example, a consent mechanism may require that a user gains consent from an owner before authorisation is given. This could reflect the referral mechanism that exists within the health system. Alternatively, the system may automatically provide authorisation to certain users on the basic of some defined context constraints, such as role and location. In a further scenario, users may appoint themselves as the first user (User 1). The fact that these other mechanisms exist (See Figure 11) means that TAC is not just an extension of MAC or DAC.

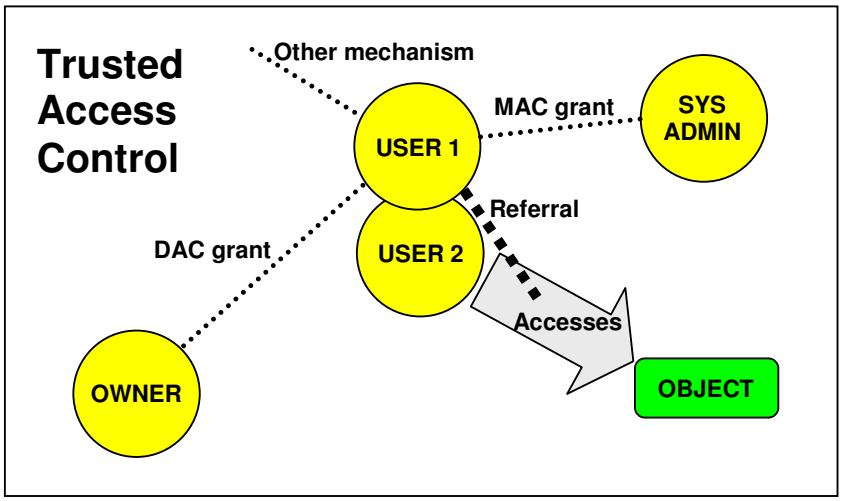

Figure 11: Trusted Access Control

TAC provides new types of well-formed transactions that are independent of MAC and DAC. TAC supplies the appropriate well-formed transactions necessary to facilitate user control. In the past, MAC and DAC styled well-formed transactions have been passed on to users, a situation which generally passes on privileges which are either too powerful or too restrictive.

Another point which illustrates the need for TAC is to understand that not all systems incorporate all three players. For example, in the hospital domain it is normal that the owners don't have system access. In this situation DAC cannot therefore be used while MAC and TAC can be. This implies that DAC cannot be used to produce TAC-type functionality in this case. Another example to consider is an ad hoc network which has no system administrator. Here MAC cannot be used to produce TAC-type functionality. A LAN gaming network is an example where there may be no owners or system administrators, only users. How do MAC and DAC function in this environment? Surely this is a situation built for TAC!

5.2.1.2 Defining Access Control System Types

It is important to consider the aspect of *control* in understanding the legitimacy of TAC. In MAC the Sys Admin controls access; in DAC the Owner controls access; and in TAC the users control access. In the same way that MAC overrides DAC (Pfleeger 2000, p. 290), MAC and DAC can override TAC.

When designing an access control model or mechanism for a particular domain it is important to determine the relationships between the players. Who is it that should have the ultimate *control* over *access*? The answer may often be the player who has to take the responsibility for, or bear the cost of, access breaches. For example, in a school situation it would most likely be appropriate for the system administrator to control access to the system. In a hospital it may be most appropriate for users (that is, health practitioners) to control access to the system. In the UK national health system it may be appropriate for owners to control access to their records. Control of access is not the only element in the control of a system. As such, while system administrators may not be given primary responsibility for access control, they have other important duties to perform.

If the system administrator does not have the primary access control responsibility, then what is their role? It can be to monitor access control to ensure that breaches do not occur and to take certain action under specified circumstances. For example, they may monitor the number of certain types of access by an individual. If the individual makes more than a specified number of such accesses within a specified period, the system administrator can either make a report to personnel managers or lock them out of the system.

In terms of control, MAC can therefore be defined as *admin-controlled*, DAC as *owner-controlled*, and TAC as *user-controlled*. Control can also be specified in terms of the primary, the secondary, and even the tertiary control method. For example a DAC and MAC system could be defined as "owner/admin-controlled" or a system that uses all three models as "user/owner/admin-controlled". The latter would mean that the primary method of control was TAC. TAC could be overridden by DAC (the secondary method), which could in turn be overridden by MAC (the tertiary method).

5.2.2 The Professional Access Control (PAC) Model

Professional Access Control (PAC) is a high level user/admin-controlled access control model which incorporates TAC, RBAC and PBAC. It is designed to be used in domains such as hospitals where the users are professionals. As such the users have a duty of care to the owners (their patients). The priorities of the model are to guarantee availability and to minimise administrative overheads.

5.2.2.1 A Useful Analogy

Consider a limousine with three occupants – the vehicle's owner, the owner's chauffeur, and the vehicle's mechanic. The limousine is analogous to a computer system; the vehicle owner to a patient, the mechanic to a system administrator, and the chauffeur to a clinician. The following question can be put. Who should drive the vehicle?

If the owner has a licence, they can certainly drive the vehicle, but they do not have the skill of the chauffeur. In the hospital domain, the patient can control access if

they indeed have access to the system, but they do not necessarily know what the appropriate clinical actions are for their current condition.

The mechanic can drive the vehicle and will no doubt ensure that it is driven in a way that does not excessively stress the vehicle mechanically, but they do not have the road sense of the chauffeur. While the system administrator can control access and will no doubt ensure that security measures protect the system objects, they do not have the clinical knowledge or ethical training to know what information should be shared with other clinicians.

The chauffeur is employed to be a driver. They know the best way to get the owner safely to the required destination. The clinician has a duty of care to the patient. They know the best way to treat the patient's condition.

While it may be logical for the chauffeur to drive most of the time, the owner may wish to go to some destinations alone and the mechanic may need to test the vehicle's performance from time to time. While it is logical for the clinicians to control access most of the time, patients have the right to deny certain accesses and system administrators need to monitor the system to ensure that security requirements are properly enforced.

PAC seeks to give clinicians primary access control in clinical hospital systems. It reduces the role of system administrators to system monitoring and breach control, while providing the option to easily add patient consent mechanisms which can allow patients to deny accesses in specific circumstances.

5.2.2.2 Team Definition

Before defining a team-based access control model like PAC, it is necessary to define the team concept. One of the major differences between the PAC model proposed here and other team-based models is the fundamental difference in team definition. The three main features of the PAC team concept are that:

1. Each patient has their own personal team;
2. There are no team specific roles – roles are organisation wide; and
3. Each team is supported by two layers of backup personnel.

The first of the preceding three points is self-explanatory, but it should be noted that team definitions in other models are often many-to-many in nature. That is, a team tends to represent a group of workers (for example, all staff on a hospital ward) who care for a group of clients (for example, all the patients on the ward). The PAC model, by contrast, is one-to-many, with each patient having their own group of carers. It would therefore be usual in the PAC model for staff members to be members of many teams, one for each patient that they care for.

PAC teams are initialised automatically as part of the process of admitting a patient to a ward. This is achieved by having a *default team* for each ward. Default teams are defined by specifying a *default role set* for each ward. Each default team is then generated by automatically placing all staff members with the specified default roles who are able to work on the ward, on the patient's team.

The second point relates to staff roles. PAC uses the same concept of roles as RBAC. This means that staff can have multiple roles and generally more than one staff member has a particular role. PAC also uses a location context constraint called a *unit*. Each staff member has what is called a *unit set*, which is the group of units on which the staff member is currently allocated to work. Figure 12 shows an example of four hospital staff, each with a role and a unit set (for example, Nurse, Ward A). All four staff members in the diagram are able to work on Ward A; all but the nurse can work on Ward B; and the administrator and the specialist can work on Ward C.

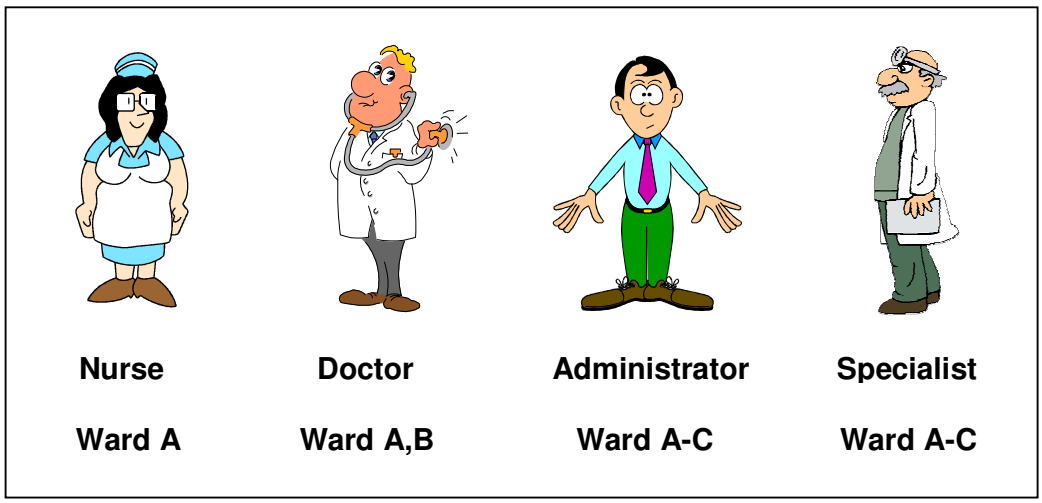

Figure 12: Staff Roles and Units

The most unique feature of the PAC team concept is outlined by the third point. Once a team is defined for a patient, there by definition exist two further groups of staff which may be called upon to care for the patient. These further groups are defined by the relationship they have with the members on the patient's team. Figure 13 shows the team group and the two supporting staff groups. Staff members on the team are defined as *members*. Staff members who share a role with any of the team members and are allocated to work on the patient's unit are defined as *colleagues*. The remaining staff members, those who are neither members nor colleagues, are defined as *associates*. From a particular nurse's point of view, they may be a member of 6 patient teams; a colleague of 20 other patient teams; and an associate of 200 further teams.

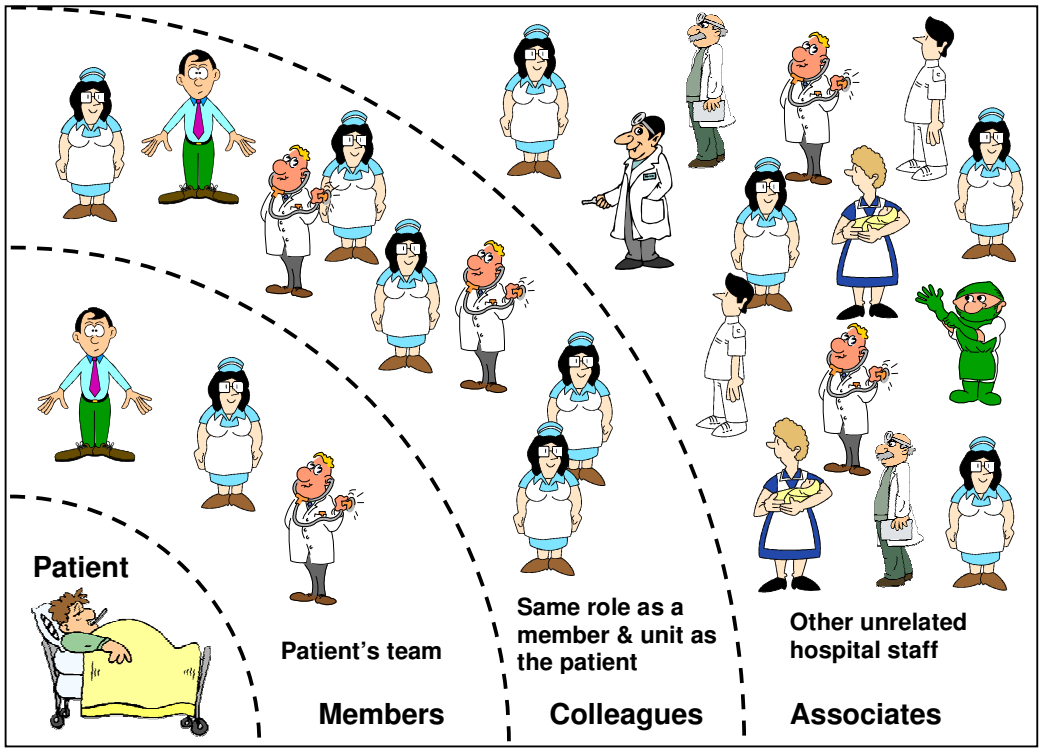

Figure 13: Team Staff Categories

PAC defines these three staff categories in order to allow different access control procedures to be used, depending on the closeness of a staff member to the patient in question. Access needs to be controlled on a need-to-know basis. Team members have the greatest need for access; colleagues may need access to help members in the normal course of work; and associates should only need access in exceptional

circumstances. Access control should therefore be tight for associates, moderate for colleagues, and easy for members.

This team model allows PAC to be extremely flexible. It can easily cater for situations where the team structure varies from ward to ward in a hospital, or even for different supervisors using different approaches on the same ward, or different approaches to be taken on a per patient basis! A supervisor can make everyone on the ward a member of all patient teams, in one extreme. Conversely, at the other extreme, they can restrict access down to having only one carer for a patient. The approach used for a patient can be changed at any time. This flexibility is in stark contrast to the team structures used in TMAC models, and makes PAC-based systems highly usable.

5.2.2.3 Model Definition

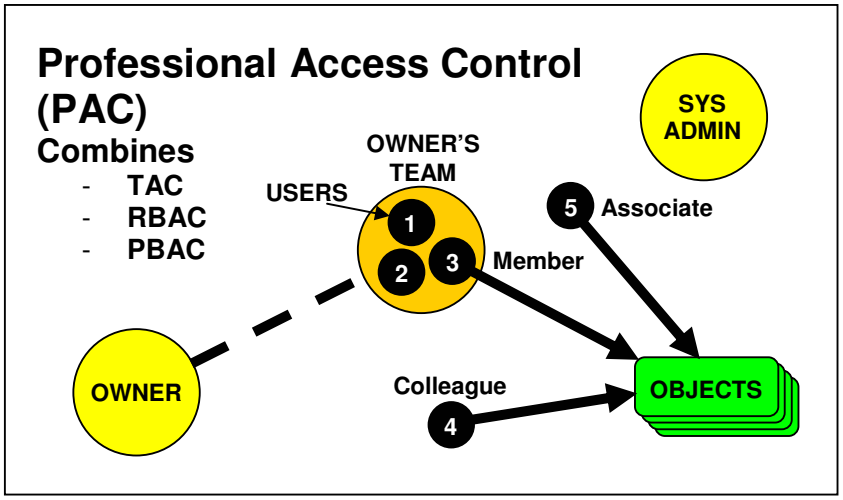

Figure 14: Professional Access Control

Figure 14 outlines the main concepts of the model. The objects in the diagram represent the collection of one or more objects owned by the owner. In the hospital domain the objects would represent different parts of the patient's record.

The model denotes three types of users; team Members, Colleagues and Associates. The patient's team is made up of the staff members who are currently treating the patient. As mentioned previously, the team model used in the solution proposed specifies a unique team for each patient. The dashed line in the diagram represents

the professional relationship between the patient and the team with the implication that the users have a duty of care to the patient.

PAC incorporates TAC, RBAC and PBAC. TAC is the method used to add additional members to the team. RBAC roles are used in connection with a location context constraint to automatically initialise team membership on patient admission, as well as to determine the colleague user status. RBAC is also used to determine which parts of the patient's record is available to the accessing user. PBAC is used as the mechanism to deal with accesses by colleagues and associates.

Being a user/admin-controlled model, PAC primary control of accesses is placed with the system users. The most common way for a user to get access to a patient's record is for them to be placed on a patient's team, which gives them direct access. If this is not practicable, a user can still get access, but the access has to be checked at a later time to determine its legitimacy. If the user is a colleague the access check is performed by a team member. If the user is an associate, the access check is performed by the system administrator.

The system administrator has only a secondary access control role. The role essentially involves monitoring accesses and checking the more unusual accesses.

The actual mechanism to deal with system breaches is beyond the scope of this project. It will however depend on the hospital security policy. The policy will specify what access actions are considered to be breaches of the duty of care, and the actions that are to be taken in response to any breach. Whether the system administrator has any power in this can also be specified. It may be desirable that disciplinary power rest with the clinical supervisors, in which case the system administrator would merely report breaches to the clinical supervisors.

5.2.2.4 PAC Access Types

PAC specifies four types of access, the first three of which are based on whether the user making the access is a team Member, a Colleague or an Associate. The four types are defined as *Direct Access* for Members, *Monitored Access* for Colleagues, *Emergency Access* for Associates, and *Critical Access* for all staff members.

Before detailing each of the four access types, it is important to realise that only the primary access type, Direct Access, is intended to be used for the vast majority of accesses. In other words, the normal way of gaining access to a patient's record is to be made a member of the patient treating team. The remaining three access types exist essentially just to guarantee access in situations where team membership was not organised beforehand. Consequently, when reporting is mentioned as part of the secondary access types, it is envisaged that few such reports will be necessary. The main reason for this is that it is normally much quicker and easier for a clinician to become a team member than to follow the reporting process. This subtlety, which encourages users to become team members, cannot be overstated.

The process that takes place when any access is attempted is as follows:

1. The user attempts to access the patient's record.
2. The system checks the database to determine the relationship between the user and the patient. That is, are they a member, a colleague, or an associate?
3. If they are not a member they are given the option of requesting membership.
4. Access is granted to the user unless they have been barred by the system administrator.
5. Relevant reports are dealt with for non-members.

Members have *Direct Access* (see Figure 15). This means that they are not hindered when accessing the patient's record. If a user is not a member (user 3 in the diagram) they may seek to gain team membership from one of the existing team members (users 1 and 2). Membership can thus be achieved using TAC. The TAC action is basically the same as a referral in the health system. That is, one health practitioner refers the patient to another health practitioner. The items that are accessible to the user are determined by the user's role. In the diagram, the green objects are accessible while the red ones are not. RBAC is applied to achieve this.

While PAC recognises the usefulness of RBAC for restricting access based of the user's role, it has nothing to do with the RBAC approach of delegating system admin privileges to users. TAC is used for passing on access rights.

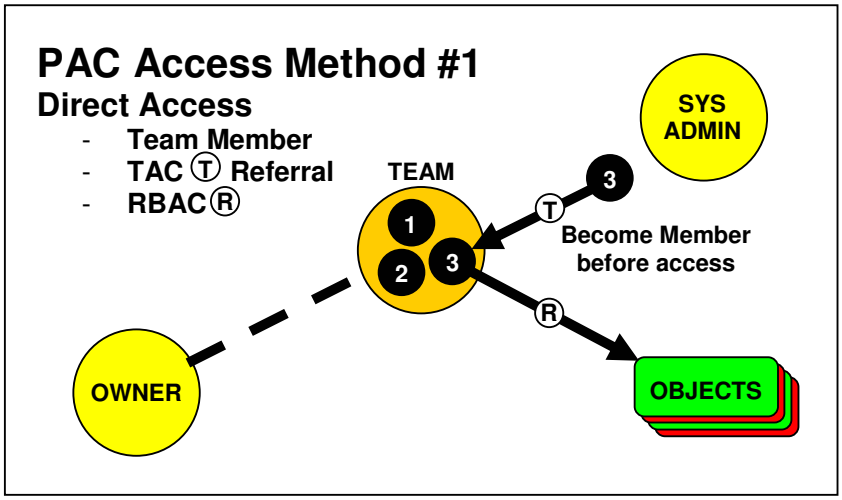

Figure 15: PAC Access Method #1

Colleagues are defined as users who have a relationship with one or more members of the team. The relationship is that defined as being that they share identical context constraints of user role and location. Role is an organisation wide variable which is the same as an RBAC role. Location is the set of units/wards where the user is permitted to work at the time. For example, a midwife may be permitted to work in the labour ward and the postnatal ward. Her colleagues would therefore be any midwife working on either of the two wards.

Colleagues from time to time, for various workflow reasons need to access the patient's record (see Figure 16). For example, if a nurse goes on a break without enabling direct access for the nurse who is covering for them. PAC allows such colleague accesses to be made, but under the provision that they are peer-reviewed by the related member of the team. This is done by requiring the colleague (user 4 in the diagram) to enter a reason for the access, which is communicated to the member (user 3) for their retrospective approval or disapproval. If the member feels that the access is unwarranted it is reported to the system administrator who has to determine what action to take. Colleagues therefore have what is termed *Monitored Access*.

Such communication and monitoring procedures are all part of the normal workflow process in hospitals which employ best practice. They are not extra work which has to be performed. Mechanisms for such reporting can be as simple as making a choice from a drop down menu, which is hardly time consuming. Evidence-based practice requires that users be able to justify their actions, which of course include their accesses to patient's records. It is also preferable that colleagues monitor such accesses because they understand the clinical requirement in the situation. A system administrator would find it extremely difficult to be able to make such judgements without any help from clinicians.

As mentioned previously, a colleague relationship is defined as practitioners who have the same role and who can work on the same ward. In the diagram the colleague relationship between users 3 and 4 is marked as "=context"; context being a general term for variables such as unit and role. This implies that the context is checked in order to validate the relationship.

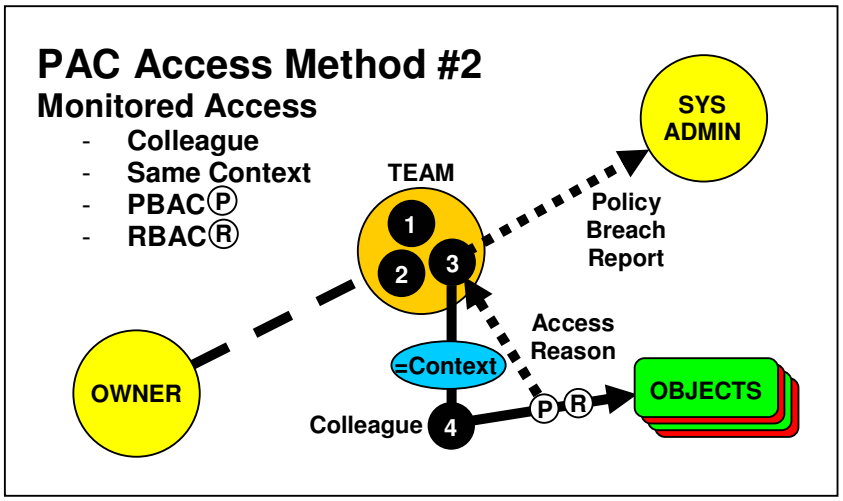

Figure 16: PAC Access Method #2

Associates are system users who are neither team members nor colleagues. As such, associates would rarely have a need to access the patient's record. Doing so would be seen as a potential breach of their duty of care. Accesses by associates are thus termed *Emergency Accesses* (see Figure 17). As such, any access made by an associate (user 5 in the diagram) is reported to the system administrator who is responsible for checking the legality of the access and taking the required punitive

action if required. An access of this type may be required when a doctor responds to an emergency and there is no-one available to facilitate a more direct access.

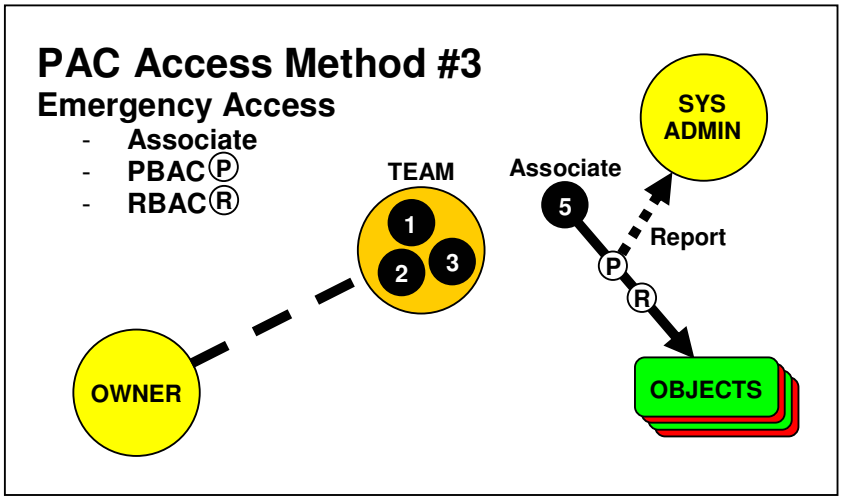

Figure 17: PAC Access Method #3

A fourth type of access is required in order to meet the goal of 100% availability. This type of access, *Critical Access*, should rarely need to be activated. Users are normally restricted by RBAC as to what they can access within a patient's record. There are possible situations where a user may require access to elements normally beyond that allowed by their role. Such accesses (see Figure 18) can be made by all users, but they are reported to the system administrator in the same way as emergency accesses.

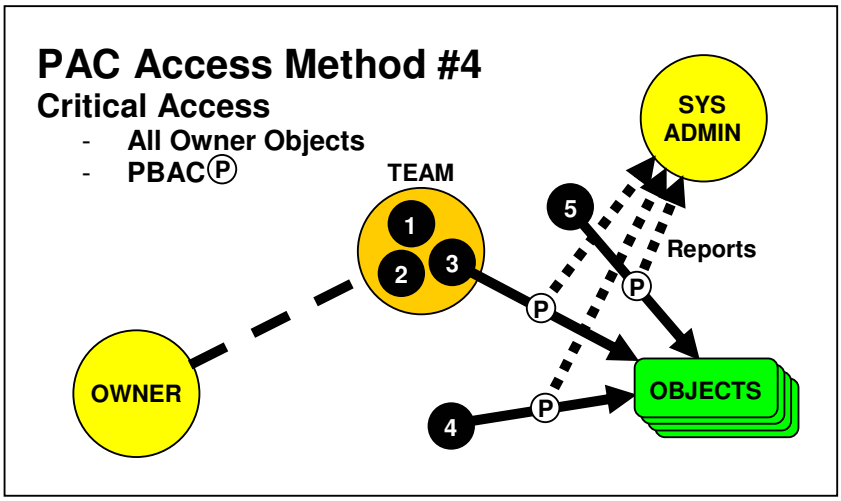

Figure 18: PAC Access Method #4

5.2.2.5 PAC Administration

Effectiveness of any PAC implementation depends greatly on how well it is administered. Any system which relies on auditing only works efficiently if the results that auditing provides are acted upon conclusively. In order to provide conclusive administration, organisational (hospital) policy must be in place which clearly specifies what constitutes both correct and incorrect behaviour in relation to access control. Predictable actions then must be taken in response to any possible or reported breach. For example, it is imperative that all emergency and critical accesses are followed up to determine their legitimacy. If it is found that a policy breach has occurred, it must be dealt with openly and definitively. Everyone involved must be aware that administration is vigilant and repercussions will definitely follow unacceptable actions.

With this type of administration in place the rewards of PAC can be reaped. Vigilant administration will inevitably lead to fewer breaches, less breach reports, and a lower administrative burden. Inaction is likely to have the opposite effect.

Perhaps the most vital aspect of PAC is that it provides subtle dissuasion to users to avoid the higher level access methods. PAC implementations should provide the means to make it easier for users to join a team than to write a report. In time users should seek to join the team in preference to using a higher level access method. One of the beauties of the model is that user report requirements and follow-up techniques can be modified in order to provide more disincentives for making high level accesses if the number of such reports is unnecessarily high. If too many critical accesses are being made, it may be a signal that role privileges need to be modified. By monitoring the proportions of access types used, system administrators are given feedback as to how the system is performing. The system does not suffer from the problem of mistakes or access failures being unreported, as prescriptive systems such as RBAC do.

The operation of the peer review process for monitored accesses is important. If the review process is working properly the number of actual breaches should be low and the number of breach reports should closely match this number. The situation should ideally be that all actual breaches are reported and no reports are forwarded when

breaches have not occurred. In order to get as close as possible to this ideal, policies regarding peer review and breach reports need to be finely tuned. Non-reporting of actual breaches must be dissuaded by strong policy in regard to security responsibilities. In other words policy must inspire clinicians to be vigilant in ensuring that colleagues do not breach the confidentiality and integrity requirements of patients that are in their care. Clinicians' duty of care to their patients should override tolerance of unreasonable accesses by colleagues. To achieve this, policy must be open and clear. Failure to report known breaches must be dealt with severely.

In summary, proper administration of a PAC implementation should ensure that breaches are rare and administrative overheads are low. A proactive approach is necessary in order to achieve this. Sufficient administrative resources need to be devoted to this in the period following initial system induction. PAC provides a highly flexible foundation which can be moulded with good organisational policy in order to achieve efficiency in administration while maximising the security of records.

5.3 Oracle PAC Toolkit

The result of implementing the PAC model in Oracle was the *Oracle PAC Toolkit*. As shown in Figure 19, the toolkit is made up of two components – a set of database tables and a set of PL/SQL procedures and functions. All the PAC functionality is provided in top level PL/SQL procedures. Lower level functions are also contained in the toolkit. These are used by the top level procedures to perform required tasks. The data necessary for access control is stored in the database tables. While system administrators may choose to manipulate the tables directly, or through the Oracle Enterprise Manager, applications which use the toolkit only need to access the top level procedures. The top level procedures do all the necessary database manipulation and can be protected by RBAC. The tables are thus protected from applications and their users by Clark-Wilson style well-formed transactions (see Section 3.9).

Both new and existing applications can use the toolkit. Simple calls to the top level procedures in the toolkit are all that is required. It should be noted that the applications in question are workflow applications. Examples of such applications in the hospital domain would be applications that admit patients or allow doctors to refer patients to specialists. The access control features can be built into these applications using the toolkit. The access control mechanisms thus work behind-the-scene and require no additional user input, with the possible exception of the reporting needed for non-standard accesses.

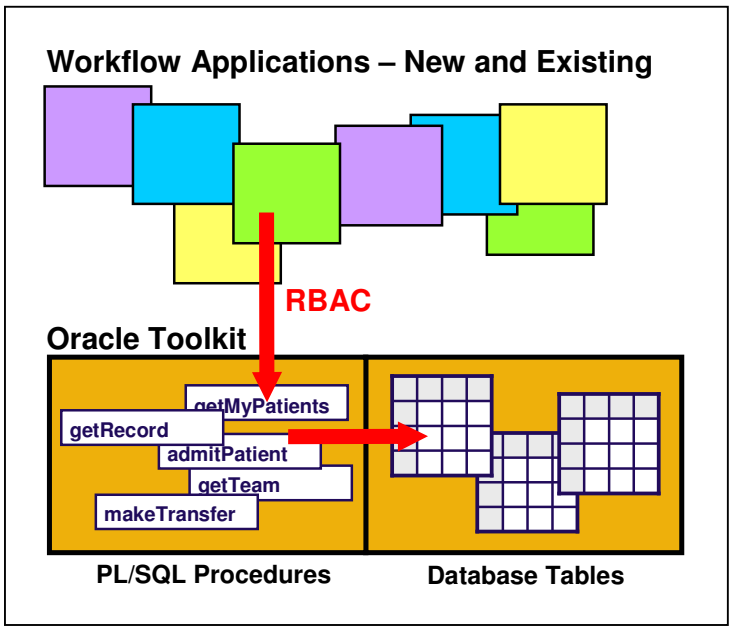

Figure 19: Applications using the Oracle PAC Toolkit

The purpose of producing the implementation was twofold. Firstly, to verify that the PAC model can be used as the basis for a workable access control mechanism and secondly, to show that Oracle can be used as the database management system. The functionality required by the PAC model was represented in the implementation as top level PL/SQL procedures. If it could be shown that these procedures produced the correct functionality then it would indicate that PAC can lead to a workable solution and that Oracle can be used. The simulation testing phase therefore concentrated on ensuring that the top level procedures produced the expected results.

5.4 Simulation Test Results

Appendix A shows the simulation test results for each of the 15 test stages. These results have been abbreviated from the full results for the sake of brevity. The full results include procedure and function print statements which indicate the flow of control. These statements were used in the troubleshooting stage of the implementation design and are not particularly relevant in ascertaining the validity of the simulation test results.

The results contained in Appendix A are based on the simulated hospital outlined in Section 4.6. The first of the two test procedure, named `showHospital`, was used to produce tables showing the current patient-staff relationships and ward locations. Table 2 is an example of such a table. The word EXEC in front of the procedure name means that the procedure has been executed, and the results are shown on the following lines. Note again, that these tables do not represent an access control matrix.

```
>EXEC showHospital;
```

	User	Amy	Bec	Col	Don	Eli	Fay	Gai	Hue	Ian	Jan	Kim	Lee	May	Nic	Oto	Pat	Qui	Ray	Sue	Tim	Uma	Vic	Wym	Xia
	MRole	Ad	AdS	RN	RN	RN	RN	RN	RN	RN	RNS	RNS	RNS	Dr	Dr	DrS	Ph	PhS	SW	SWS	S1	S1S	S2	S2S
	WdSet	1	7	1	1	1	3	2	2	4	4	7	2	6	3	7	7	7	7	7	7	7	6	7
Patient	CurWd	1	1	0	1	1	0	2	2	0	4	1	0	4	1	0	4	1	0	4	1	0	4	2	0
Adams	1	2	1	3	2	2	3	3	3	3	3	1	3	3	2	3	1	3	3	3	3	3	3	3	3
Barns	1	2	1	3	2	2	3	3	3	3	3	1	3	3	2	3	1	3	3	3	3	3	3	3	3
Corry	1	2	1	3	2	2	3	3	3	3	3	1	3	3	2	3	1	3	3	3	3	3	3	3	3
Deane	1	2	1	3	2	2	3	3	3	3	3	1	3	3	2	3	1	3	3	3	3	3	3	3	3
Erlik	1	2	1	3	2	2	3	3	3	3	3	1	3	3	2	3	1	3	3	3	3	3	3	3	3
Freud	1	2	1	3	2	2	3	3	3	3	3	1	3	3	2	3	1	3	3	3	3	3	3	3	3
Gaile	1	2	1	3	2	2	3	3	3	3	3	1	3	3	2	3	1	3	3	3	3	3	3	3	3
Hoode	1	2	1	3	2	2	3	3	3	3	3	1	3	3	2	3	1	3	3	3	3	3	3	3	3
Irwin	1	2	1	3	2	2	3	3	3	3	3	1	3	3	2	3	1	3	3	3	3	3	3	3	3
Johns	1	2	1	3	2	2	3	3	3	3	3	1	3	3	2	3	1	3	3	3	3	3	3	3	3
Kenny	2	3	1	3	3	3	3	2	2	3	3	1	3	1	3	3	3	3	3	3	3	3	3	3	3
Leach	2	3	1	3	3	3	3	2	2	3	3	1	3	1	3	3	3	3	3	3	3	3	3	3	3
Mayes	2	3	1	3	3	3	3	2	2	3	3	1	3	1	3	3	3	3	3	3	3	3	3	3	3
Newmn	2	3	1	3	3	3	3	2	2	3	3	1	3	1	3	3	3	3	3	3	3	3	3	3	3
Oxley	2	3	1	3	3	3	3	2	2	3	3	1	3	1	3	3	3	3	3	3	3	3	3	3	3
Paget	2	3	1	3	3	3	3	2	2	3	3	1	3	1	3	3	3	3	3	3	3	3	3	3	3
Quinn	4	3	1	3	3	3	3	3	3	3	2	1	3	1	3	3	3	3	3	3	3	3	1	3	3
Rosse	4	3	1	3	3	3	3	3	3	3	2	1	3	1	3	3	3	3	3	3	3	3	1	3	3
Stott	4	3	1	3	3	3	3	3	3	3	2	1	3	1	3	3	3	3	3	3	3	3	1	3	3
Thoms	4	3	1	3	3	3	3	3	3	3	2	1	3	1	3	3	3	3	3	3	3	3	1	3	3
Unwin	4	3	1	3	3	3	3	3	3	3	2	1	3	1	3	3	3	3	3	3	3	3	1	3	3
Voite	4	3	1	3	3	3	3	3	3	3	2	1	3	1	3	3	3	3	3	3	3	3	1	3	3
White	0	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3
Xiang	0	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3

Table 2: Example of the Current Hospital State Table

The names of the patients are contained in the left column.

The ward that they are on is shown in the next column (0 for No Ward, 1 for Ward A, 2 for Ward B, and 4 for Ward C). Note that No Ward means the patient has not yet been placed on a ward.

The names of the staff members are contained on the top row.

The second row shows their main role (Ad=Admin, +S=Supervisor, RN=Registered Nurse, Dr=RMO, Ph=Pharmacist, SW=Social Worker, S1/S2=Specialists)

The third row shows the staff ward set, which is represented as the sum of the numbers of the wards on which they can work (0 for No Ward, 1 for Ward A, 2 for Ward B, and 4 for Ward C). For example, a '6' indicate that the staff member can work on Ward B or Ward C. These numbers are preloaded into the database and do not change during the simulation testing. Note that No Ward means the staff member is not currently working.

The fourth row shows the current ward that the staff member is on (0 for No Ward, 1 for Ward A, 2 for Ward B, and 4 for Ward C).

The numbers in the remaining section of the table (the large rectangle at the bottom right) represent the individual relationship status for each patient-staff pair (1=Member, 2=Associate, 3=Associate). For example, a '2' on the row of patient 'Irwin' in the column 'Don' indicated that Don is is a colleague of an Irwin team member.

The next section in the each test, *PROCEDURE TESTS*, shows the procedures to be executed in the test. This is followed by the table, - the *Changes table* (see Table 3 example), which shows the changes that have occurred since the start of the test.

```
>EXEC showHospital_CHANGES;
```

	User	Amy	Bec	Col	Don	Eli	Fay	Gai	Hue	Ian	Jan	Kim	Lee	May	Nic	Oto	Pat	Qui	Ray	Sue	Tim	Uma	Vic	Wym	Xia
	MRole	Ad	AdS	RN	RN	RN	RN	RN	RN	RN	RN	RNS	RNS	RNS	Dr	Dr	DrS	Ph	PhS	SW	SWS	S1	S1S	S2	S2S
	WdSet	1	7	1	1	1	3	2	2	4	4	7	2	6	3	7	7	7	7	7	7	7	7	6	7
Patient	CurWd	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
Adams	1	2	1	-	2	2	-	-	-	-	-	1	-	-	2	-	1	-	-	-	-	-	-	-	-
Barns	1	2	1	-	2	2	-	-	-	-	-	1	-	-	2	-	1	-	-	-	-	-	-	-	-
Corry	1	2	1	-	2	2	-	-	-	-	-	1	-	-	2	-	1	-	-	-	-	-	-	-	-
Deane	1	2	1	-	2	2	-	-	-	-	-	1	-	-	2	-	1	-	-	-	-	-	-	-	-
Erlik	1	2	1	-	2	2	-	-	-	-	-	1	-	-	2	-	1	-	-	-	-	-	-	-	-
Freud	1	2	1	-	2	2	-	-	-	-	-	1	-	-	2	-	1	-	-	-	-	-	-	-	-
Gaile	1	2	1	-	2	2	-	-	-	-	-	1	-	-	2	-	1	-	-	-	-	-	-	-	-
Hoode	1	2	1	-	2	2	-	-	-	-	-	1	-	-	2	-	1	-	-	-	-	-	-	-	-
Irwin	1	2	1	-	2	2	-	-	-	-	-	1	-	-	2	-	1	-	-	-	-	-	-	-	-
Johns	1	2	1	-	2	2	-	-	-	-	-	1	-	-	2	-	1	-	-	-	-	-	-	-	-
Kenny	2	-	1	-	-	-	-	2	2	-	-	1	-	1	-	-	-	-	-	-	-	-	-	-	-
Leach	2	-	1	-	-	-	-	2	2	-	-	1	-	1	-	-	-	-	-	-	-	-	-	-	-
Mayes	2	-	1	-	-	-	-	2	2	-	-	1	-	1	-	-	-	-	-	-	-	-	-	-	-
Newmn	2	-	1	-	-	-	-	2	2	-	-	1	-	1	-	-	-	-	-	-	-	-	-	-	-
Oxley	2	-	1	-	-	-	-	2	2	-	-	1	-	1	-	-	-	-	-	-	-	-	-	-	-
Paget	2	-	1	-	-	-	-	2	2	-	-	1	-	1	-	-	-	-	-	-	-	-	-	-	-
Quinn	4	-	1	-	-	-	-	-	-	-	2	1	-	1	-	-	-	-	-	-	-	-	1	-	-
Rosse	4	-	1	-	-	-	-	-	-	-	2	1	-	1	-	-	-	-	-	-	-	-	1	-	-
Stott	4	-	1	-	-	-	-	-	-	-	2	1	-	1	-	-	-	-	-	-	-	-	1	-	-
Thoms	4	-	1	-	-	-	-	-	-	-	2	1	-	1	-	-	-	-	-	-	-	-	1	-	-
Unwin	4	-	1	-	-	-	-	-	-	-	2	1	-	1	-	-	-	-	-	-	-	-	1	-	-
Voite	4	-	1	-	-	-	-	-	-	-	2	1	-	1	-	-	-	-	-	-	-	-	1	-	-
White	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
Xiang	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-

Table 3: Example of the Changes Table

The second of the two test procedures, the `showHospital_CHANGES` procedure, is executed to generate this table. The table is essentially the same as the showHospital table, but only the values that have changes in the 'CurWd' and relationships sections are shown as integers ('-'s indicate that the value has not changed).

In order to check that each procedure produced the correct result and was performing properly, it was necessary to check that the changes in this table were correct. This was done manually. The results from each test are detailed and discussed in Appendix B. A summary of the results is shown in Table 4.

In Table 4 the *Procedures Executed* column shows the top level procedures executed as part of the test. The *No. Exe* column shows how many times each of the procedures was executed. The last column shows whether the result yielded was correct.

The 15 simulation tests confirmed that the top level procedures functioned correctly in all of the diverse manners envisaged. This shows that all the functionality required to design an application using the PAC model can be implemented within an Oracle package.

5.5 Validation Results

As a final double-check, the initial hospital scenarios were analysed to ensure that the functionality that they required was contained within the Oracle package. This was done qualitatively, by matching the required functionality specified in Table 1 with the PL/SQL procedure that provides that functionality. Table 5 displays the results of this cross-matching process.

It should be noted at this point that a number of the scenarios, such as those associated with Critical accesses, are satisfied by RBAC features. In Oracle, RBAC is best managed using the Enterprise Manager. The implementation therefore does not need to include procedures which manage these RBAC features. Also, many long term auditing features could be supplied with the package. However, only provisional access auditing features are of concern here.

Test No.	Test Name	Test Description	Procedures Executed	No. Exe	Correct Function
1	Logon 1	Logon to empty hospital	`logon`	16	Yes
2	Admission 1	Admission of initial patients	`admitPatient`	24	Yes
3	Logon 2	Logon to simulate a shift change	`logon`	8	Yes
4	Admission 2	Admitting new patients to wards	`admitPatient`	2	Yes
5	Re-admission	Moving patients to new wards	`admitPatient`	8	Yes
6	Referral	Referring a patient to a clinician	`makeReferral`	25	Yes
7	Transferral	Transferring a patient to another clinician	`makeTransfer`	5	Yes
8	Release	Clinicians drop patients	`releasePatient`	5	Yes
9	Removal	Supervisors remove staff from teams	`removePatient`	6	Yes
10	View Record	Member, colleague & associate accesses	`getRecord`	3	Yes
11	View Hidden	Getting access with reason/report submission	`getHiddenRecord`	7	Yes

12	Requests	Users requesting entry to team	`seekReferral`
			`seekTransfer`
			`selfReferral`
				1
				1
				3
					Yes

13	View Reports	Viewing user and admin reports	`showUserReports`
			`showAdminReports`
			`showDueAdminReports`
				4
				2
				2
					Yes

14	Patient Lists	Printing different patient groupings	`showAllPatients`
			`showWardPatients`
			`showMyPatients`
			`showMyWardPatients`
			`showGroupPatients`
			`showTeam`
				1
				1
				1
				1
				1
				1
					Yes

15	Logoff	Logging off users	`logoff`	5	Yes

Table 4: Simulation Test Results

Scenario Category	Scenario Name	Model Requirements	Top Level Procedure Name
Patient Issues	Admission	Patient assigned ward and an initial team.	`admitPatient`
	Anonymity	Patient wants to remain anonymous.	`Not implemented. Apply Hospital Policy`
	Change Carer	Replace one team member with a colleague.	`makeTransfer`
	Transfer Patient	Remove patient from ward and admit to another ward.	`admitPatient`
Staff Change Issues	Staff Break	Allow access to the record and require a report back to the staff member.	`getHiddenRecord`
	Staff Sick	Staff member rings in sick.	Functionality not required - covered by logon procedure
	Staff Leaves	Supervisor Admin removes patient from teams by logging them off.	`removePatient`
	Staff Transfer	Replace one team member with a colleague.	`makeTransfer`
	Emergency	Allow access by staff who are not members or colleagues and require a report to sys admin	`getHiddenRecord`
	Referral	Add a staff member to the team.	`makeReferral`
	Staff Removal	Remove a staff member from a team.	`makeTransfer releasePatient`
Staff Information Issues	Help Colleague	Allow access to the record and require a report back to the staff member.	`getHiddenRecord`
	Show Team	Display a list of team members.	`showTeam`
	Contact Kin	Allow a staff member to access all the information about a patient and require a report to sys admin.	Beyond implementation scope. To be implemented within RBAC using a temporary role allocation.
	Show Messages	Display a list of current messages for a staff member.	`showUserReports`
	Contact Team Member	Display a list of team members.	`showTeam`
	List Patients	Display a list of patients for a given staff member.	`showMyPatients`

Scenario Category	Scenario Name	Model Requirements	Top Level Procedure Name
Administ-rative Issues	Agency	An agency staff member is employed.	Beyond implementation scope. Can be implemented within RBAC.
	Show Ward Patients	Display patients on a ward or just return a count. Application can secure this information.	`showWardPatients`
	Check Team	Administration wants to know who was looking after a patient in the past.	Beyond implementation scope. Implement by archiving database entries.
	No Log Off	Supervisor or Admin removes patient by logging them off.	`logoff`
	Admin Override	Admin can add or remove team members.	`makeReferral` `removePatient`
Security Incidents	Alter Record	Staff member attempts to alter records to cover up a medical error.	Beyond implementation scope. Can be implemented within RBAC.
	Unwarrant-ed Access	Alert System admin of suspect access through a colleagues' report or direct report.	`submitAdminReport` `showAdminReports`

Table 5: Scenario Functionality Checking

It is clear that all the scenarios can be satisfied using the procedures provided in the Oracle PAC Toolkit, appropriate hospital policies, or Oracle RBAC features. The Oracle package therefore provides all the functionality necessary to meet the requirements of both the PAC model and the hospital workplace.

The list of scenarios is also a worthwhile tool for ongoing work. Experience showed that new, generally rare scenarios arose from time to time. The list provided a way to check whether a similar scenario already existed which covered a new scenario. The list also provides an ongoing record of what has been previously covered.

In addition to meeting the access control requirements, the package provides added functionality which can be used to advantage by workflow applications. It provides concise listings of many different patient and staff groups (see Test#14) which can be used to facilitate communications between clinicians, as well as for patient and staff management. Its team-based functionality can be used as a tool to manage the allocation of clinicians to patients. It also provides the means for communicating access reasons between clinicians working with the same patient.

Chapter 6

Conclusion

Initial studies into the health and hospital domains highlighted that for a system to be usable, access control mechanisms must guarantee the availability of patient information to practitioners. Health professionals need guaranteed access to patient records in order to ensure that inappropriate clinical decisions are not made due to access limitations. Such limitations can be caused by inflexible access control mechanisms. It is also vital that threats to the confidentiality and integrity of patient data be taken seriously. Security systems in the hospital domain should therefore guarantee availability to health professionals while at the same time maximising data protection.

It was found that there were no existing access control models which could suitably guarantee availability in a volatile environment. This was found to be because the models, with the exception of provisional access control and auditing, tend to be fundamentally either MAC or DAC in nature. As such, they provide control to either system administrators or data owners, never users. A third fundamental access control model, named Trusted Access Control (TAC) was therefore proposed, which allows users to have control over object accesses. TAC is suitable for use in domains where users exhibit a high level of responsibility. In the same way that MAC and DAC based systems can be combined, TAC can be incorporated as either the primary, secondary or tertiary access control mechanism in a given system.

It was noted that in designing access control systems, the fundamental question to ask at the beginning is, 'Who is to be in control of the system?' Until now systems have been either admin-controlled or owner-controlled. TAC introduces the concept of a user-controlled system. At a technical level, it supplies the appropriate well-formed transactions necessary to facilitate user control. In the past, systems have relied upon admin and owner styled well-formed transactions being passed on to

users, a situation which generally passes on privileges which are either too powerful or too restrictive.

It was argued that in highly ethic environments, where there are adequate professional incentives to induce proper behaviour, there is fundamentally no reason why informed users should not behave just as ethically as system administrators. MAC based systems rely on the incorruptibility of system administrators. Why should professionals, such as doctors, behave in a different way when confidentiality and integrity of patient data has long been a fundamental concept to them? In view of this, a new high level access control model, named Professional Access Control (PAC) has been proposed.

PAC is a user/admin-controlled model which combines TAC, Role Based Access Control (RBAC) and Provision Based Access Control (PBAC). RBAC is a MAC based model which uses the abstraction of user roles to facilitate easy administration. PBAC is a provisional access control model which allows users who cannot be granted direct access to system objects, to be granted access based on them performing some security based procedure.

PAC uses a team-based approach in a similar way to Team Based Access Control (TMAC) type models, but with a fundamentally different concept of what a team is. A PAC type team is very flexible in nature. Different team structures can easily exist on different wards, and even within the same ward! This is because every data owner (for example, a hospital patient) has their own team. On admission to a ward (represented by a location context variable) a patient is automatically assigned a default team unique to the ward. The default team is composed essentially of any staff member who is working on the patient's ward and who has a role which is specified in the default role set for the ward. Once admitted, a patient's team can be changed quickly and easily to meet any situation that may arise.

TAC is used by PAC to facilitate the addition of further members to the team. This is akin to a referral in the health system. Team membership can also be varied by supervisory users and system administrators. Members of a patient's team have direct access to the patient's record, but the amount of information they see is

governed by RBAC according to their role. The vast majority of record accesses are of this primary type.

In order to guarantee availability to all system users, while maximising confidentiality and integrity protection, PAC employs a novel approach. It specifies a number of secondary access levels, which are based on the closeness of the relationship between the staff member and the patient. Colleagues of team members are allowed access to a patient's record subject to them giving a reason to a team member (applying PBAC). A peer-review mechanism therefore provides management over most non-direct accesses. Other staff members who are not related to the patient's team can access the patient's record in a similar way, but such accesses are reported directly to the system administrator. Furthermore, any user can gain total access to the patient's record, bypassing RBAC, subject to them making a report to the system administrator.

PAC therefore provides guaranteed access to all users while maintaining a tight and efficient management system which maximises confidentiality and integrity protection. The peer-review control mechanism can be built in to the normal workflow as a way of enhancing clinical communications, whilst it maintains security behind-the-scene. The PAC team concept also provides a useful abstraction which can be utilised by workflow applications for a variety of reasons.

PAC provides primary control to the health practitioners, thus alleviating their dependence on system administrators to solve their access problems. As such it takes a great load off the system administrators, leaving them just the basic RBAC management, access monitoring tasks, and high level potential security breaches. The flexibility of user supervisory functions means that the reliance on the presence of individual administrative users in workplace situations is minimised.

While PAC is a new model, it applies fundamental and proven techniques from existing access control models, combining them in a highly efficient manner. The system was implemented as an Oracle package, called the Oracle PAC Toolkit, which can be used by both existing legacy applications and new applications. The package combines the required database table structure and a set of controlling PL/SQL procedures, which can easily be used by these applications. Features

inherent in Oracle, such as its RBAC functionality and Clark-Wilson style well-formed transactions are incorporated into the design. The bulk of management tasks can be performed in the Oracle Enterprise Manager.

PAC overcomes the inflexibilities inherent in other models by providing provision based access. This removes the need to always have to supply access privileges to users before they are required. Such prescriptive systems fundamentally fall down because they cannot provide availability when faced with unusual emergency situations. The flexibility of PAC means that there is no situation, no matter how bizarre, where access can be denied. Even in such extreme circumstances PAC can maintain an appropriate level of protection.

When compared to existing access control models, PAC is simple and easy to understand. The fact that a working version of it has been implemented successfully within the period of this project is testimony to this. From a user's point of view, all they really need to know is that they should be on a patient's team to get access to the patient's record. If they are not, then they can still get access, but they will have to state the reason why they need it. Simple interfaces can easily be put in place to guide them through the access process.

PAC was designed to meet the demands of the volatile hospital environment which is one of the most extreme environment in which an access control mechanism has to perform. The fact that it is able to perform in this domain, together with the fact that it offers the possibility of user-controlled access management systems that are inherently user-friendly, means that it is a feasible alternative in many other domains. Slight modifications and extensions can easily be made which increase security and limit availability, giving it the ability to mould itself to the requirements of these alternate domains.

Chapter 7

Further Work

There are a lot of potential directions in which this work can proceed. There is much theoretical work to be done on TAC to define its detail, scope and relevance. Research on how the three foundational models of MAC, DAC and TAC can work together is warranted. On a practical level, work on PAC to enable it to be incorporated in real systems, could prove fruitful.

Both TAC and PAC need to be analysed more closely and developed into formal access control models. The possibility that they could provide the framework for a general purpose access control model could also be investigated. It is envisaged that such a general purpose model will contain features which can be adjusted according to the control requirements of the domain.

The development of TAC gives users the ability to control accesses to owner objects. This having been achieved, it should be a relatively simple step to implement owner consent mechanisms into a TAC based model. The possibility of incorporating DAC and TAC mechanisms to achieve this also warrants investigation. In order to facilitate such consent mechanisms, the issue of data ownership needs to be clarified. The development of a suitable data ownership model would assist with this.

The next practical step in regard to PAC is to develop a working prototype for testing with real users. This needs to incorporate user-friendly interfaces which closely reflect normal workflow characteristics in the environment where the prototype is to be tested. The prototype should also include a suitable authentication mechanism.

Once a PAC based implementation has been fine-tuned, a management application can be added to facilitate system administration. The management application could be either a stand alone application or an add-on to Oracle Enterprise Manager. There

are many possible auditing and management features that could be added, as well as alarms which alert administrators to possible breaches.

It would be useful to investigate the possibility of incorporating TAC and PAC features at the operating system level. Alternatively, approaches which use systems other than Oracle could be trialled. An XML based storage system could prove to be useful.

Another step that can be taken once a PAC system has been fine-tuned is to generalise it to work in a variety of domains. The main idea here is to develop ways of making trade-offs between total availability and data protection. For example, total availability could be restricted to certain roles in the system. Other roles would have tighter controls placed on them.

This thesis has concentrated on access to records. A more general approach would entail using the system for task management purposes. Such a system would allow users to perform tasks and to pass on task permissions in much the same way as access privileges.

Chapter 8

References

AHD 2000, *The American Heritage Dictionary of the English Language, Fourth Edition*, Houghton Mifflin Company. viewed 1st October, 2004, <http://dictionary.reference.com/search?q=professional>

Alotaiby, F. T. and Chen, J. X. 2004, 'A Model for Team-based Access Control (TMAC 2004)', *International Conference on Information Technology: Coding and Computing (ITCC'04)*, IEEE, Las Vegas, Nevada, USA

Botha, R. A. and Eloff, J. H. P. 2001, 'Separation of duties for access control enforcement in workflow environments.' *IBM Systems Journal* vol. 40 iss. 3: pp. 666-682.

Bretan, A. 2004, *Authorization Models in Medical Information Systems*. viewed 2nd August, 2004, <http://www.cse.fau.edu/~security/public/AuthModelsMedInfSystems.ppt>

Coiera, E. 2003, *Guide to Health Infomatics*, Arnold, London.

CSD 2000, 'Stopping security breaches.' *Computer Security Digest* vol. 18 iss. 9: p. 1.

CSD 2001a, 'Medical records vulnerable to ID theft.' *Computer Security Digest* vol. 19 iss. 1: p. 3.

CSD 2001b, 'Privacy Issues.' *Computer Security Digest* vol. 19 iss.2: 3.

Czapski, M. and Creevey, A. 2003, 'Old Problem – New Solutions: HealthConnect System Architecture for Information Delivery', *HIC 2003 RACGP 12CC Combined Conferences*, Darling Harbour, Sydney Australia

Doolan, D. 2003, 'Unintended Consequences of Clinical Information Systems', *HIC 2003 RACGP 12CC Combined Conferences*, Darling Harbour, Sydney Australia

Englebardt, S. P. and Nelson, R. 2002, *Health Care Informatics: An Interdisciplinary Approach*, Mosby, Inc., St. Louis, Missouri.

Ferraiolo, D. and Kuhn, R. 1992, 'Role-Based Access Control', *15th National Computer Security Conference*

Georgiadis, C. K., Mavridis, I., Pangalos, G. and Thomas, R. K. 2001, 'Flexible Team-Based Access Control Using Contexts', *SACMAT '01*, ACM, Chantilly, Virginia, USA, pp. 21-27

Georgiadis, C. K., Mavridis, I. K. and Pangalos, G. I. 2002, 'Programming a view-based active access-control system for healthcare environments.' *Health Informatics Journal (2002)*: pp. 191-198.

Gollmann, D. 1999, *Computer Security*, John Wiley & Sons Ltd, Chichester, West Sussex, England.

Hale, L., Gilchrist, G., Ho, M.-H., Hwa, M., Iyer, S., Jacobs, A. L., Kethana, L., Koyfman, A., Le, V., Lewis, N., Narasinghanallur, J., Philips, A. and Turlapati, R. 2002, *Oracle Advanced Security: Administrator's Guide*, Oracle.

Hartnett, J. 2002, Research into the Implementation of Electronic Consent for the use of Patient Identifiable Health Data, University of Tasmania - School of Computing.

HealthConnect 2002, Consent and Electronic Health Records - A Discussion Paper.

Kalam, A. A. E., Baida, R. E., Balbiani, P., Benferhat, S., Cuppens, F., Deswarte, Y., Miege, A., Saurel, C. and Trouessin, G. 2003, 'Organisation based access control', *4th International IEEE Workshop on Policies for Distributed Systems and Networks*, IEEE, Lake Como, Italy, pp. 120-131

Kropp, B. and Gallaher, M. 2001, *Reach Out and ID Someone*. viewed 30th March 2004, <http://infosecuritymag.techtarget.com/articles/april01/cover.shtml>

Kudo, M. 2002, 'PBAC: Provision-based access control model.' *International Journal of Information Security* vol. 1 iss. 2: pp. 116-130.

Lam, S. 2003, 'Building and operating an integrated e-prescribing system in an Australian hospital', *HIC 2003 RACGP 12CC Combined Conferences,* Darling Harbour, Sydney Australia

Moore, D. 2004, *Java vc. PL/SQL: Where do I put the SQL?*, dbazine.com. viewed 25th August, 2004, <http://www.dbazine.com/moore2.shtml>

Nanda, A. 2004, *Keeping Information Private with VPD*, Oracle Magazine, March/April 2004**:** pp. 81-84.

Naude, F. 2003, *Oracle PL/SQL FAQ*, OracleFAQ. viewed 25th August, 2004, <http://www.orafaq.com/faqplsql.htm#ORJAVA>

NEHRT 2000, *A Health Information Network for Australia*, Department of Health and Aged Care.

Neumann, G. and Strembeck, M. 2003, 'An Approach to Engineer and Enforce Context Constraints in an RBAC Environment', *SACMAT '03*, ACM, Como, Italy, pp. 65-79

NIST 2004, *Role Based Access Control*. viewed 6th October, 2004, <http://csrc.nist.gov/rbac/>

Parry, K. and Tucker, J. 2003, 'Teaching Old Dogs New Tricks', *HIC 2003 RACGP 12CC Combined Conferences*, Darling Harbour, Sydney Australia

Pfleeger, C. P. 2000, *Security in Computing*, Prentice Hall PTR, Upper Saddle River, New Jersey.

Ramaswamy, C. and Sandhu, R. 1998, 'Role-Based Access Control Features in Commercial Database Management Systems', *21st National Information Systems Security Conference*, Crystal City, Virginia, USA

Reuters 2004, *Online Security: Who's Liable?*. viewed 30th March, <http://www.wired.com/news/privacy/0,1848,62843,00.html>

Robinson, G. 2003, *Real World Microsoft Access Database Protection and Security*, Apress. viewed 25th August, 2004, <http://www.developer.com/db/article.php/10920_3308841_1>

Schneier, B. 2000, *Secrets and Lies*, John Wiley & Sons, Inc., New York.

Spencer, R., Logan, P. and Coiera, E. 2003, 'Socio-technical factors surrounding practices related to telephone, paging and information system use in an Emergency Department', *HIC 2003 RACGP 12CC Combined Conferences,* Darling Harbour, Sydney Australia

Stallings, W. 2003, *Network Security Essentials*, Pearson Education, Inc., Upper Saddle River, New Jersey.

The Office of the Federal Privacy Commissioner 2004, *Privacy Information for Health*. viewed 2nd September 2004, <http://www.privacy.gov.au/health/index.html>

Theriault, M. and Heney, W. 1998, *Oracle Security*, O'Reilly & Associates, Inc., Sebastopol, California.

Thomas, R. K. 1997, 'Team-based Access Control (TMAC): A Primitive for Applying Role-based Access Controls in Collaborative Environments', *RBAC '97*, ACM, Fairfax Va USA, pp. 13-19

Thomas, R. K. and Sandhu, R. S. 1997, 'Task-based Authorisation Controls (TBAC): A Family of Models for Active and Enterprise-oriented Authorisation Management', *IFIP WG11.3 Workshop on Database Security*, Chapman & Hall, Lake Tahoe, California, USA

Webb, D., Wise, M. and Boote, A. 2003, 'Supporting end users to make IT happen for Community Health', *HIC 2003 RACGP 12CC Combined Conferences*, Darling Harbour, Sydney Australia

Wikipedia date unknown, *Occam's Razor*. viewed 13th Oct 2004, <http://en.wikipedia.org/wiki/Occam's_Razor>

Win, K. T., Croll, P. and Cooper, J. 2003, 'Privacy, Confidentiality and Consent of Electronic Health Record Systems', *HIC 2003 RACGP 12CC Combined Conferences*, Darling Harbour, Sydney Australia

Woodcock, D. and Gillies, I. 2003, 'Generic middleware as a new paradigm for providing a single user interface to multiple disparate web-based clinical applications', *HIC 2003 RACGP 12CC Combined Conferences*, Darling Harbour, Sydney Australia

Appendix A – Simulation Test Results

This appendix contains an abridged version of the simulation test results. For the sake of brevity, the print statements showing the flow of control between PL/SQL procedures and functions have been omitted.

Overall notes containing greater detail about the test implementation, the testing process, individual tests, and test results are contained in Appendix B.

```
>--------------------------------------
>--TEST #1: LOGON PROCEDURE - SET UP --
>--------------------------------------
>
>--Test Description: Logon a number of users and check current ward is correct.
>
>EXEC showHospital;
```

	User	Amy	Bec	Col	Don	Eli	Fay	Gai	Hue	Ian	Jan	Kim	Lee	May	Nic	Oto	Pat	Qui	Ray	Sue	Tim	Uma	Vic	Wym	Xia
	MRole	Ad	AdS	RN	RN	RN	RN	RN	RN	RN	RN	RNS	RNS	RNS	Dr	Dr	DrS	Ph	PhS	SW	SWS	S1	S1S	S2	S2S
	WdSet	1	7	1	1	1	3	2	2	4	4	7	2	6	3	7	7	7	7	7	7	7	7	6	7
Patient	CurWd	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Adams	0	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3
Barns	0	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3
Corry	0	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3
Deane	0	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3
Erlik	0	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3
Freud	0	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3
Gaile	0	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3
Hoode	0	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3
Irwin	0	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3
Johns	0	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3
Kenny	0	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3
Leach	0	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3
Mayes	0	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3
Newmn	0	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3
Oxley	0	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3
Paget	0	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3
Quinn	0	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3
Rosse	0	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3
Stott	0	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3
Thoms	0	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3
Unwin	0	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3
Voite	0	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3
White	0	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3
Xiang	0	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3

```
>--PROCEDURE TEST(S):--
>
>EXEC logon('Amy','Ward A');
>EXEC logon('Bec','Ward A');
>EXEC logon('Don','Ward A');
>EXEC logon('Eli','Ward A');
>EXEC logon('Gai','Ward B');
>EXEC logon('Hue','Ward B');
>EXEC logon('Jan','Ward C');
>EXEC logon('Kim','Ward A');
>EXEC logon('May','Ward C');
>EXEC logon('Nic','Ward A');
>EXEC logon('Pat','Ward C');
>EXEC logon('Qui','Ward A');
>EXEC logon('Sue','Ward C');
>EXEC logon('Tim','Ward A');
>EXEC logon('Vic','Ward C');
>EXEC logon('Wym','Ward B');

>EXEC showHospital_CHANGES;
```

	User	Amy	Bec	Col	Don	Eli	Fay	Gai	Hue	Ian	Jan	Kim	Lee	May	Nic	Oto	Pat	Qui	Ray	Sue	Tim	Uma	Vic	Wym	Xia
	MRole	Ad	AdS	RN	RN	RN	RN	RN	RN	RN	RN	RNS	RNS	RNS	Dr	Dr	DrS	Ph	PhS	SW	SWS	S1	S1S	S2	S2S
	WdSet	1	7	1	1	1	3	2	2	4	4	7	2	6	3	7	7	7	7	7	7	7	7	6	7
Patient	CurWd	1	1	-	1	1	-	2	2	-	4	1	-	4	1	-	4	1	-	4	1	-	4	2	-
Adams	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
Barns	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
Corry	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
Deane	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
Erlik	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
Freud	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
Gaile	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
Hoode	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
Irwin	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
Johns	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
Kenny	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
Leach	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
Mayes	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
Newmn	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
Oxley	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
Paget	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
Quinn	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
Rosse	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
Stott	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
Thoms	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
Unwin	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
Voite	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
White	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
Xiang	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-

```
>-----------------------------------------
>--TEST #2: ADMITTING PATIENTS - SET UP --
>-----------------------------------------
>
>--Test Description: Admit all the patients and check the current ward, the default role setting and other
>--                  relationships are correct.
>
>EXEC showHospital;
```

	User	Amy	Bec	Col	Don	Eli	Fay	Gai	Hue	Ian	Jan	Kim	Lee	May	Nic	Oto	Pat	Qui	Ray	Sue	Tim	Uma	Vic	Wym	Xia
	MRole	Ad	AdS	RN	RN	RN	RN	RN	RN	RN	RN	RNS	RNS	RNS	Dr	Dr	DrS	Ph	PhS	SW	SWS	S1	S1S	S2	S2S
	WdSet	1	7	1	1	1	3	2	2	4	4	7	2	6	3	7	7	7	7	7	7	7	7	6	7
Patient	CurWd	1	1	0	1	1	0	2	2	0	4	1	0	4	1	0	4	1	0	4	1	0	4	2	0
Adams	0	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3
Barns	0	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3
Corry	0	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3
Deane	0	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3
Erlik	0	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3
Freud	0	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3
Gaile	0	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3
Hoode	0	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3
Irwin	0	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3
Johns	0	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3
Kenny	0	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3
Leach	0	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3
Mayes	0	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3
Newmn	0	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3
Oxley	0	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3
Paget	0	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3
Quinn	0	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3
Rosse	0	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3
Stott	0	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3
Thoms	0	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3
Unwin	0	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3
Voite	0	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3
White	0	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3
Xiang	0	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3

```
>--PROCEDURE TEST(S):--
>
>EXEC admitPatient('Adams','Ward A');
>EXEC admitPatient('Barns','Ward A');
>EXEC admitPatient('Corry','Ward A');
>EXEC admitPatient('Deane','Ward A');
>EXEC admitPatient('Erlik','Ward A');
>EXEC admitPatient('Freud','Ward A');
>EXEC admitPatient('Gaile','Ward A');
>EXEC admitPatient('Hoode','Ward A');
>EXEC admitPatient('Irwin','Ward A');
>EXEC admitPatient('Johns','Ward A');
>EXEC admitPatient('Kenny','Ward B');
>EXEC admitPatient('Leach','Ward B');
>EXEC admitPatient('Mayes','Ward B');
>EXEC admitPatient('Newmn','Ward B');
>EXEC admitPatient('Oxley','Ward B');
>EXEC admitPatient('Paget','Ward B');
>EXEC admitPatient('Quinn','Ward C');
>EXEC admitPatient('Rosse','Ward C');
>EXEC admitPatient('Stott','Ward C');
>EXEC admitPatient('Thoms','Ward C');
>EXEC admitPatient('Unwin','Ward C');
>EXEC admitPatient('Voite','Ward C');
>EXEC admitPatient('White','NoWard');
>EXEC admitPatient('Xiang','NoWard');

>EXEC showHospital_CHANGES;
```

	User	Amy	Bec	Col	Don	Eli	Fay	Gai	Hue	Ian	Jan	Kim	Lee	May	Nic	Oto	Pat	Qui	Ray	Sue	Tim	Uma	Vic	Wym	Xia
	MRole	Ad	AdS	RN	RN	RN	RN	RN	RN	RN	RN	RNS	RNS	RNS	Dr	Dr	DrS	Ph	PhS	SW	SWS	S1	S1S	S2	S2S
	WdSet	1	7	1	1	1	3	2	2	4	4	7	2	6	3	7	7	7	7	7	7	7	7	6	7
Patient	CurWd	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
Adams	1	2	1	-	2	2	-	-	-	-	-	1	-	-	2	-	1	-	-	-	-	-	-	-	-
Barns	1	2	1	-	2	2	-	-	-	-	-	1	-	-	2	-	1	-	-	-	-	-	-	-	-
Corry	1	2	1	-	2	2	-	-	-	-	-	1	-	-	2	-	1	-	-	-	-	-	-	-	-
Deane	1	2	1	-	2	2	-	-	-	-	-	1	-	-	2	-	1	-	-	-	-	-	-	-	-
Erlik	1	2	1	-	2	2	-	-	-	-	-	1	-	-	2	-	1	-	-	-	-	-	-	-	-
Freud	1	2	1	-	2	2	-	-	-	-	-	1	-	-	2	-	1	-	-	-	-	-	-	-	-
Gaile	1	2	1	-	2	2	-	-	-	-	-	1	-	-	2	-	1	-	-	-	-	-	-	-	-
Hoode	1	2	1	-	2	2	-	-	-	-	-	1	-	-	2	-	1	-	-	-	-	-	-	-	-
Irwin	1	2	1	-	2	2	-	-	-	-	-	1	-	-	2	-	1	-	-	-	-	-	-	-	-
Johns	1	2	1	-	2	2	-	-	-	-	-	1	-	-	2	-	1	-	-	-	-	-	-	-	-
Kenny	2	-	1	-	-	-	-	2	2	-	-	1	-	1	-	-	-	-	-	-	-	-	-	-	-
Leach	2	-	1	-	-	-	-	2	2	-	-	1	-	1	-	-	-	-	-	-	-	-	-	-	-
Mayes	2	-	1	-	-	-	-	2	2	-	-	1	-	1	-	-	-	-	-	-	-	-	-	-	-
Newmn	2	-	1	-	-	-	-	2	2	-	-	1	-	1	-	-	-	-	-	-	-	-	-	-	-
Oxley	2	-	1	-	-	-	-	2	2	-	-	1	-	1	-	-	-	-	-	-	-	-	-	-	-
Paget	2	-	1	-	-	-	-	2	2	-	-	1	-	1	-	-	-	-	-	-	-	-	-	-	-
Quinn	4	-	1	-	-	-	-	-	-	-	2	1	-	1	-	-	-	-	-	-	-	-	1	-	-
Rosse	4	-	1	-	-	-	-	-	-	-	2	1	-	1	-	-	-	-	-	-	-	-	1	-	-
Stott	4	-	1	-	-	-	-	-	-	-	2	1	-	1	-	-	-	-	-	-	-	-	1	-	-
Thoms	4	-	1	-	-	-	-	-	-	-	2	1	-	1	-	-	-	-	-	-	-	-	1	-	-
Unwin	4	-	1	-	-	-	-	-	-	-	2	1	-	1	-	-	-	-	-	-	-	-	1	-	-
Voite	4	-	1	-	-	-	-	-	-	-	2	1	-	1	-	-	-	-	-	-	-	-	1	-	-
White	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
Xiang	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-

```
>------------------------------
>--TEST #3: LOGON PROCEDURE --
>------------------------------
>
>--Test Description: Logon a number of users and check the current ward and relationships are correct.
>
>EXEC showHospital;
```

	User	Amy	Bec	Col	Don	Eli	Fay	Gai	Hue	Ian	Jan	Kim	Lee	May	Nic	Oto	Pat	Qui	Ray	Sue	Tim	Uma	Vic	Wym	Xia
	MRole	Ad	AdS	RN	RN	RN	RN	RN	RN	RN	RN	RNS	RNS	RNS	Dr	Dr	DrS	Ph	PhS	SW	SWS	S1	S1S	S2	S2S
	WdSet	1	7	1	1	1	3	2	2	4	4	7	2	6	3	7	7	7	7	7	7	7	7	6	7
Patient	CurWd	1	1	0	1	1	0	2	2	0	4	1	0	4	1	0	4	1	0	4	1	0	4	2	0
Adams	1	2	1	3	2	2	3	3	3	3	3	1	3	3	2	3	1	3	3	3	3	3	3	3	3
Barns	1	2	1	3	2	2	3	3	3	3	3	1	3	3	2	3	1	3	3	3	3	3	3	3	3
Corry	1	2	1	3	2	2	3	3	3	3	3	1	3	3	2	3	1	3	3	3	3	3	3	3	3
Deane	1	2	1	3	2	2	3	3	3	3	3	1	3	3	2	3	1	3	3	3	3	3	3	3	3
Erlik	1	2	1	3	2	2	3	3	3	3	3	1	3	3	2	3	1	3	3	3	3	3	3	3	3
Freud	1	2	1	3	2	2	3	3	3	3	3	1	3	3	2	3	1	3	3	3	3	3	3	3	3
Gaile	1	2	1	3	2	2	3	3	3	3	3	1	3	3	2	3	1	3	3	3	3	3	3	3	3
Hoode	1	2	1	3	2	2	3	3	3	3	3	1	3	3	2	3	1	3	3	3	3	3	3	3	3
Irwin	1	2	1	3	2	2	3	3	3	3	3	1	3	3	2	3	1	3	3	3	3	3	3	3	3
Johns	1	2	1	3	2	2	3	3	3	3	3	1	3	3	2	3	1	3	3	3	3	3	3	3	3
Kenny	2	3	1	3	3	3	3	2	2	3	3	1	3	1	3	3	3	3	3	3	3	3	3	3	3
Leach	2	3	1	3	3	3	3	2	2	3	3	1	3	1	3	3	3	3	3	3	3	3	3	3	3
Mayes	2	3	1	3	3	3	3	2	2	3	3	1	3	1	3	3	3	3	3	3	3	3	3	3	3
Newmn	2	3	1	3	3	3	3	2	2	3	3	1	3	1	3	3	3	3	3	3	3	3	3	3	3
Oxley	2	3	1	3	3	3	3	2	2	3	3	1	3	1	3	3	3	3	3	3	3	3	3	3	3
Paget	2	3	1	3	3	3	3	2	2	3	3	1	3	1	3	3	3	3	3	3	3	3	3	3	3
Quinn	4	3	1	3	3	3	3	3	3	3	2	1	3	1	3	3	3	3	3	3	3	3	1	3	3
Rosse	4	3	1	3	3	3	3	3	3	3	2	1	3	1	3	3	3	3	3	3	3	3	1	3	3
Stott	4	3	1	3	3	3	3	3	3	3	2	1	3	1	3	3	3	3	3	3	3	3	1	3	3
Thoms	4	3	1	3	3	3	3	3	3	3	2	1	3	1	3	3	3	3	3	3	3	3	1	3	3
Unwin	4	3	1	3	3	3	3	3	3	3	2	1	3	1	3	3	3	3	3	3	3	3	1	3	3
Voite	4	3	1	3	3	3	3	3	3	3	2	1	3	1	3	3	3	3	3	3	3	3	1	3	3
White	0	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3
Xiang	0	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3

```
>--PROCEDURE TEST(S):--
>
>EXEC logon('Col','Ward A');
>EXEC logon('Fay','Ward A');
>EXEC logon('Ian','Ward C');
>EXEC logon('Lee','Ward B');
>EXEC logon('Oto','Ward B');
>EXEC logon('Ray','Ward B');
>EXEC logon('Uma','Ward B');
>EXEC logon('Xia','Ward C');

>EXEC showHospital_CHANGES;
```

	User	Amy	Bec	Col	Don	Eli	Fay	Gai	Hue	Ian	Jan	Kim	Lee	May	Nic	Oto	Pat	Qui	Ray	Sue	Tim	Uma	Vic	Wym	Xia
	MRole	Ad	AdS	RN	RN	RN	RN	RN	RN	RN	RN	RNS	RNS	RNS	Dr	Dr	DrS	Ph	PhS	SW	SWS	S1	S1S	S2	S2S
	WdSet	1	7	1	1	1	3	2	2	4	4	7	2	6	3	7	7	7	7	7	7	7	7	6	7
Patient	CurWd	-	-	1	-	-	1	-	-	4	-	-	2	-	-	2	-	-	2	-	-	2	-	-	4
Adams	-	-	-	2	-	-	2	-	-	-	-	-	-	-	-	2	-	-	-	-	-	-	-	-	-
Barns	-	-	-	2	-	-	2	-	-	-	-	-	-	-	-	2	-	-	-	-	-	-	-	-	-
Corry	-	-	-	2	-	-	2	-	-	-	-	-	-	-	-	2	-	-	-	-	-	-	-	-	-
Deane	-	-	-	2	-	-	2	-	-	-	-	-	-	-	-	2	-	-	-	-	-	-	-	-	-
Erlik	-	-	-	2	-	-	2	-	-	-	-	-	-	-	-	2	-	-	-	-	-	-	-	-	-
Freud	-	-	-	2	-	-	2	-	-	-	-	-	-	-	-	2	-	-	-	-	-	-	-	-	-
Gaile	-	-	-	2	-	-	2	-	-	-	-	-	-	-	-	2	-	-	-	-	-	-	-	-	-
Hoode	-	-	-	2	-	-	2	-	-	-	-	-	-	-	-	2	-	-	-	-	-	-	-	-	-
Irwin	-	-	-	2	-	-	2	-	-	-	-	-	-	-	-	2	-	-	-	-	-	-	-	-	-
Johns	-	-	-	2	-	-	2	-	-	-	-	-	-	-	-	2	-	-	-	-	-	-	-	-	-
Kenny	-	-	-	-	-	-	2	-	-	-	-	-	2	-	-	-	-	-	-	-	-	-	-	-	-
Leach	-	-	-	-	-	-	2	-	-	-	-	-	2	-	-	-	-	-	-	-	-	-	-	-	-
Mayes	-	-	-	-	-	-	2	-	-	-	-	-	2	-	-	-	-	-	-	-	-	-	-	-	-
Newmn	-	-	-	-	-	-	2	-	-	-	-	-	2	-	-	-	-	-	-	-	-	-	-	-	-
Oxley	-	-	-	-	-	-	2	-	-	-	-	-	2	-	-	-	-	-	-	-	-	-	-	-	-
Paget	-	-	-	-	-	-	2	-	-	-	-	-	2	-	-	-	-	-	-	-	-	-	-	-	-
Quinn	-	-	-	-	-	-	-	-	-	2	-	-	-	-	-	-	-	-	-	-	-	2	-	-	-
Rosse	-	-	-	-	-	-	-	-	-	2	-	-	-	-	-	-	-	-	-	-	-	2	-	-	-
Stott	-	-	-	-	-	-	-	-	-	2	-	-	-	-	-	-	-	-	-	-	-	2	-	-	-
Thoms	-	-	-	-	-	-	-	-	-	2	-	-	-	-	-	-	-	-	-	-	-	2	-	-	-
Unwin	-	-	-	-	-	-	-	-	-	2	-	-	-	-	-	-	-	-	-	-	-	2	-	-	-
Voite	-	-	-	-	-	-	-	-	-	2	-	-	-	-	-	-	-	-	-	-	-	2	-	-	-
White	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
Xiang	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-

```
>--------------------------------
>--TEST #4: ADMITTING PATIENTS --
>--------------------------------
>
>--Test Description: Admit the patients and check the current ward, the default role setting and other
>--                  relationships are correct.
>
>EXEC showHospital;
```

	User	Amy	Bec	Col	Don	Eli	Fay	Gai	Hue	Ian	Jan	Kim	Lee	May	Nic	Oto	Pat	Qui	Ray	Sue	Tim	Uma	Vic	Wym	Xia
	MRole	Ad	AdS	RN	RN	RN	RN	RN	RN	RN	RN	RNS	RNS	RNS	Dr	Dr	DrS	Ph	PhS	SW	SWS	S1	S1S	S2	S2S
	WdSet	1	7	1	1	1	3	2	2	4	4	7	2	6	3	7	7	7	7	7	7	7	7	6	7
Patient	CurWd	1	1	1	1	1	1	2	2	4	4	1	2	4	1	2	4	1	2	4	1	2	4	2	4
Adams	1	2	1	2	2	2	2	3	3	3	3	1	3	3	2	2	1	3	3	3	3	3	3	3	3
Barns	1	2	1	2	2	2	2	3	3	3	3	1	3	3	2	2	1	3	3	3	3	3	3	3	3
Corry	1	2	1	2	2	2	2	3	3	3	3	1	3	3	2	2	1	3	3	3	3	3	3	3	3
Deane	1	2	1	2	2	2	2	3	3	3	3	1	3	3	2	2	1	3	3	3	3	3	3	3	3
Erlik	1	2	1	2	2	2	2	3	3	3	3	1	3	3	2	2	1	3	3	3	3	3	3	3	3
Freud	1	2	1	2	2	2	2	3	3	3	3	1	3	3	2	2	1	3	3	3	3	3	3	3	3
Gaile	1	2	1	2	2	2	2	3	3	3	3	1	3	3	2	2	1	3	3	3	3	3	3	3	3
Hoode	1	2	1	2	2	2	2	3	3	3	3	1	3	3	2	2	1	3	3	3	3	3	3	3	3
Irwin	1	2	1	2	2	2	2	3	3	3	3	1	3	3	2	2	1	3	3	3	3	3	3	3	3
Johns	1	2	1	2	2	2	2	3	3	3	3	1	3	3	2	2	1	3	3	3	3	3	3	3	3
Kenny	2	3	1	3	3	3	2	2	2	3	3	1	2	1	3	3	3	3	3	3	3	3	3	3	3
Leach	2	3	1	3	3	3	2	2	2	3	3	1	2	1	3	3	3	3	3	3	3	3	3	3	3
Mayes	2	3	1	3	3	3	2	2	2	3	3	1	2	1	3	3	3	3	3	3	3	3	3	3	3
Newmn	2	3	1	3	3	3	2	2	2	3	3	1	2	1	3	3	3	3	3	3	3	3	3	3	3
Oxley	2	3	1	3	3	3	2	2	2	3	3	1	2	1	3	3	3	3	3	3	3	3	3	3	3
Paget	2	3	1	3	3	3	2	2	2	3	3	1	2	1	3	3	3	3	3	3	3	3	3	3	3
Quinn	4	3	1	3	3	3	3	3	3	2	2	1	3	1	3	3	3	3	3	3	3	2	1	3	3
Rosse	4	3	1	3	3	3	3	3	3	2	2	1	3	1	3	3	3	3	3	3	3	2	1	3	3
Stott	4	3	1	3	3	3	3	3	3	2	2	1	3	1	3	3	3	3	3	3	3	2	1	3	3
Thoms	4	3	1	3	3	3	3	3	3	2	2	1	3	1	3	3	3	3	3	3	3	2	1	3	3
Unwin	4	3	1	3	3	3	3	3	3	2	2	1	3	1	3	3	3	3	3	3	3	2	1	3	3
Voite	4	3	1	3	3	3	3	3	3	2	2	1	3	1	3	3	3	3	3	3	3	2	1	3	3
White	0	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3
Xiang	0	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3

```
>--PROCEDURE TEST(S):--
>
>EXEC admitPatient('White','Ward B');
>EXEC admitPatient('Xiang','Ward C');

>EXEC showHospital_CHANGES;
```

	User	Amy	Bec	Col	Don	Eli	Fay	Gai	Hue	Ian	Jan	Kim	Lee	May	Nic	Oto	Pat	Qui	Ray	Sue	Tim	Uma	Vic	Wym	Xia
	MRole	Ad	AdS	RN	RN	RN	RN	RN	RN	RN	RN	RNS	RNS	RNS	Dr	Dr	DrS	Ph	PhS	SW	SWS	S1	S1S	S2	S2S
	WdSet	1	7	1	1	1	3	2	2	4	4	7	2	6	3	7	7	7	7	7	7	7	7	6	7
Patient	CurWd	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
Adams	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
Barns	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
Corry	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
Deane	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
Erlik	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
Freud	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
Gaile	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
Hoode	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
Irwin	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
Johns	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
Kenny	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
Leach	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
Mayes	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
Newmn	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
Oxley	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
Paget	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
Quinn	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
Rosse	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
Stott	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
Thoms	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
Unwin	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
Voite	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
White	2	-	1	-	-	-	2	2	2	-	-	1	1	1	-	-	-	-	-	-	-	-	-	-	-
Xiang	4	-	1	-	-	-	-	-	-	2	2	1	-	1	-	-	-	-	-	-	-	2	1	-	-

```
>-----------------------------------
>--TEST #5: RE-ADMITTING PATIENTS --
>-----------------------------------
>
>--Test Description: Admit/Transfer the patients to new wards and check the current ward, the default role
>--                  setting and other relationships are correct.
>
>EXEC showHospital;
```

	User	Amy	Bec	Col	Don	Eli	Fay	Gai	Hue	Ian	Jan	Kim	Lee	May	Nic	Oto	Pat	Qui	Ray	Sue	Tim	Uma	Vic	Wym	Xia
	MRole	Ad	AdS	RN	RN	RN	RN	RN	RN	RN	RN	RNS	RNS	RNS	Dr	Dr	DrS	Ph	PhS	SW	SWS	S1	S1S	S2	S2S
	WdSet	1	7	1	1	1	3	2	2	4	4	7	2	6	3	7	7	7	7	7	7	7	7	6	7
Patient	CurWd	1	1	1	1	1	1	2	2	4	4	1	2	4	1	2	4	1	2	4	1	2	4	2	4
Adams	1	2	1	2	2	2	2	3	3	3	3	1	3	3	2	2	1	3	3	3	3	3	3	3	3
Barns	1	2	1	2	2	2	2	3	3	3	3	1	3	3	2	2	1	3	3	3	3	3	3	3	3
Corry	1	2	1	2	2	2	2	3	3	3	3	1	3	3	2	2	1	3	3	3	3	3	3	3	3
Deane	1	2	1	2	2	2	2	3	3	3	3	1	3	3	2	2	1	3	3	3	3	3	3	3	3
Erlik	1	2	1	2	2	2	2	3	3	3	3	1	3	3	2	2	1	3	3	3	3	3	3	3	3
Freud	1	2	1	2	2	2	2	3	3	3	3	1	3	3	2	2	1	3	3	3	3	3	3	3	3
Gaile	1	2	1	2	2	2	2	3	3	3	3	1	3	3	2	2	1	3	3	3	3	3	3	3	3
Hoode	1	2	1	2	2	2	2	3	3	3	3	1	3	3	2	2	1	3	3	3	3	3	3	3	3
Irwin	1	2	1	2	2	2	2	3	3	3	3	1	3	3	2	2	1	3	3	3	3	3	3	3	3
Johns	1	2	1	2	2	2	2	3	3	3	3	1	3	3	2	2	1	3	3	3	3	3	3	3	3
Kenny	2	3	1	3	3	3	2	2	2	3	3	1	2	1	3	3	3	3	3	3	3	3	3	3	3
Leach	2	3	1	3	3	3	2	2	2	3	3	1	2	1	3	3	3	3	3	3	3	3	3	3	3
Mayes	2	3	1	3	3	3	2	2	2	3	3	1	2	1	3	3	3	3	3	3	3	3	3	3	3
Newmn	2	3	1	3	3	3	2	2	2	3	3	1	2	1	3	3	3	3	3	3	3	3	3	3	3
Oxley	2	3	1	3	3	3	2	2	2	3	3	1	2	1	3	3	3	3	3	3	3	3	3	3	3
Paget	2	3	1	3	3	3	2	2	2	3	3	1	2	1	3	3	3	3	3	3	3	3	3	3	3
Quinn	4	3	1	3	3	3	3	3	3	2	2	1	3	1	3	3	3	3	3	3	3	2	1	3	3
Rosse	4	3	1	3	3	3	3	3	3	2	2	1	3	1	3	3	3	3	3	3	3	2	1	3	3
Stott	4	3	1	3	3	3	3	3	3	2	2	1	3	1	3	3	3	3	3	3	3	2	1	3	3
Thoms	4	3	1	3	3	3	3	3	3	2	2	1	3	1	3	3	3	3	3	3	3	2	1	3	3
Unwin	4	3	1	3	3	3	3	3	3	2	2	1	3	1	3	3	3	3	3	3	3	2	1	3	3
Voite	4	3	1	3	3	3	3	3	3	2	2	1	3	1	3	3	3	3	3	3	3	2	1	3	3
White	2	3	1	3	3	3	2	2	2	3	3	1	1	1	3	3	3	3	3	3	3	3	3	3	3
Xiang	4	3	1	3	3	3	3	3	3	2	2	1	3	1	3	3	3	3	3	3	3	2	1	3	3

```
>--PROCEDURE TEST(S):--
>
>EXEC admitPatient('Adams','Ward B');
>EXEC admitPatient('Kenny','Ward C');
>EXEC admitPatient('Quinn','Ward A');

>EXEC showHospital_CHANGES;
```

	User	Amy	Bec	Col	Don	Eli	Fay	Gai	Hue	Ian	Jan	Kim	Lee	May	Nic	Oto	Pat	Qui	Ray	Sue	Tim	Uma	Vic	Wym	Xia
	MRole	Ad	AdS	RN	RN	RN	RN	RN	RN	RN	RN	RNS	RNS	RNS	Dr	Dr	DrS	Ph	PhS	SW	SWS	S1	S1S	S2	S2S
	WdSet	1	7	1	1	1	3	2	2	4	4	7	2	6	3	7	7	7	7	7	7	7	7	6	7
Patient	CurWd	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
Adams	2	3	-	3	3	3	-	2	2	-	-	-	1	1	3	3	3	-	-	-	-	-	-	-	-
Barns	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
Corry	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
Deane	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
Erlik	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
Freud	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
Gaile	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
Hoode	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
Irwin	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
Johns	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
Kenny	4	-	-	-	-	-	3	3	3	2	2	-	3	-	-	-	-	-	-	-	-	2	1	-	-
Leach	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
Mayes	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
Newmn	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
Oxley	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
Paget	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
Quinn	1	2	-	2	2	2	2	-	-	3	3	-	-	3	2	2	1	-	-	-	-	-	-	-	-
Rosse	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
Stott	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
Thoms	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
Unwin	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
Voite	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
White	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
Xiang	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-

```
>--------------------------------
>--TEST #6: REFERRING PATIENTS --
>--------------------------------
>
>--Test Description: Make referrals to other users and check relationships.
>--                  Test SYS referral.  Test incorrect referrer (for 'Oxley' & 'Stott').
>
>EXEC showHospital;
```

	User	Amy	Bec	Col	Don	Eli	Fay	Gai	Hue	Ian	Jan	Kim	Lee	May	Nic	Oto	Pat	Qui	Ray	Sue	Tim	Uma	Vic	Wym	Xia
	MRole	Ad	AdS	RN	RN	RN	RN	RN	RN	RN	RN	RNS	RNS	RNS	Dr	Dr	DrS	Ph	PhS	SW	SWS	S1	S1S	S2	S2S
	WdSet	1	7	1	1	1	3	2	2	4	4	7	2	6	3	7	7	7	7	7	7	7	7	6	7
Patient	CurWd	1	1	1	1	1	1	2	2	4	4	1	2	4	1	2	4	1	2	4	1	2	4	2	4
Adams	2	3	1	3	3	3	2	2	2	3	3	1	1	1	3	3	3	3	3	3	3	3	3	3	3
Barns	1	2	1	2	2	2	2	3	3	3	3	1	3	3	2	2	1	3	3	3	3	3	3	3	3
Corry	1	2	1	2	2	2	2	3	3	3	3	1	3	3	2	2	1	3	3	3	3	3	3	3	3
Deane	1	2	1	2	2	2	2	3	3	3	3	1	3	3	2	2	1	3	3	3	3	3	3	3	3
Erlik	1	2	1	2	2	2	2	3	3	3	3	1	3	3	2	2	1	3	3	3	3	3	3	3	3
Freud	1	2	1	2	2	2	2	3	3	3	3	1	3	3	2	2	1	3	3	3	3	3	3	3	3
Gaile	1	2	1	2	2	2	2	3	3	3	3	1	3	3	2	2	1	3	3	3	3	3	3	3	3
Hoode	1	2	1	2	2	2	2	3	3	3	3	1	3	3	2	2	1	3	3	3	3	3	3	3	3
Irwin	1	2	1	2	2	2	2	3	3	3	3	1	3	3	2	2	1	3	3	3	3	3	3	3	3
Johns	1	2	1	2	2	2	2	3	3	3	3	1	3	3	2	2	1	3	3	3	3	3	3	3	3
Kenny	4	3	1	3	3	3	3	3	3	2	2	1	3	1	3	3	3	3	3	3	3	2	1	3	3
Leach	2	3	1	3	3	3	2	2	2	3	3	1	2	1	3	3	3	3	3	3	3	3	3	3	3
Mayes	2	3	1	3	3	3	2	2	2	3	3	1	2	1	3	3	3	3	3	3	3	3	3	3	3
Newmn	2	3	1	3	3	3	2	2	2	3	3	1	2	1	3	3	3	3	3	3	3	3	3	3	3
Oxley	2	3	1	3	3	3	2	2	2	3	3	1	2	1	3	3	3	3	3	3	3	3	3	3	3
Paget	2	3	1	3	3	3	2	2	2	3	3	1	2	1	3	3	3	3	3	3	3	3	3	3	3
Quinn	1	2	1	2	2	2	2	3	3	3	3	1	3	3	2	2	1	3	3	3	3	2	1	3	3
Rosse	4	3	1	3	3	3	3	3	3	2	2	1	3	1	3	3	3	3	3	3	3	2	1	3	3
Stott	4	3	1	3	3	3	3	3	3	2	2	1	3	1	3	3	3	3	3	3	3	2	1	3	3
Thoms	4	3	1	3	3	3	3	3	3	2	2	1	3	1	3	3	3	3	3	3	3	2	1	3	3
Unwin	4	3	1	3	3	3	3	3	3	2	2	1	3	1	3	3	3	3	3	3	3	2	1	3	3
Voite	4	3	1	3	3	3	3	3	3	2	2	1	3	1	3	3	3	3	3	3	3	2	1	3	3
White	2	3	1	3	3	3	2	2	2	3	3	1	1	1	3	3	3	3	3	3	3	3	3	3	3
Xiang	4	3	1	3	3	3	3	3	3	2	2	1	3	1	3	3	3	3	3	3	3	2	1	3	3

```
>--PROCEDURE TEST(S):--
>
>EXEC makeReferral('SYS', 'Fay', 'Adams');          >EXEC makeReferral('May', 'Gai', 'Mayes');
>EXEC makeReferral('Bec', 'Col', 'Barns');          >EXEC makeReferral('Kim', 'Hue', 'Newmn');
>EXEC makeReferral('Bec', 'Don', 'Corry');          >EXEC makeReferral('Ian', 'Qui', 'Oxley');
>EXEC makeReferral('Kim', 'Eli', 'Deane');          >EXEC makeReferral('Kim', 'Ray', 'Paget');
>EXEC makeReferral('Kim', 'Fay', 'Erlik');          >EXEC makeReferral('Pat', 'Tim', 'Quinn');
>EXEC makeReferral('Kim', 'Col', 'Freud');          >EXEC makeReferral('Vic', 'Ian', 'Rosse');
>EXEC makeReferral('Kim', 'Don', 'Gaile');          >EXEC makeReferral('Ian', 'Jan', 'Stott');
>EXEC makeReferral('Bec', 'Amy', 'Hoode');          >EXEC makeReferral('Vic', 'Sue', 'Thoms');
>EXEC makeReferral('Amy', 'Eli', 'Hoode');          >EXEC makeReferral('Kim', 'Uma', 'Unwin');
>EXEC makeReferral('Pat', 'Nic', 'Irwin');          >EXEC makeReferral('Vic', 'Xia', 'Voite');
>EXEC makeReferral('Pat', 'Oto', 'Johns');          >EXEC makeReferral('Lee', 'Nic', 'White');
>EXEC makeReferral('Bec', 'Gai', 'Kenny');          >EXEC makeReferral('Vic', 'Wym', 'Xiang');
>EXEC makeReferral('May', 'Hue', 'Leach');

>EXEC showHospital_CHANGES;
```

	User	Amy	Bec	Col	Don	Eli	Fay	Gai	Hue	Ian	Jan	Kim	Lee	May	Nic	Oto	Pat	Qui	Ray	Sue	Tim	Uma	Vic	Wym	Xia
	MRole	Ad	AdS	RN	RN	RN	RN	RN	RN	RN	RN	RNS	RNS	RNS	Dr	Dr	DrS	Ph	PhS	SW	SWS	S1	S1S	S2	S2S
	WdSet	1	7	1	1	1	3	2	2	4	4	7	2	6	3	7	7	7	7	7	7	7	7	6	7
Patient	CurWd	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
Adams	-	-	-	-	-	-	1	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
Barns	-	-	-	1	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
Corry	-	-	-	-	1	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
Deane	-	-	-	-	-	1	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
Erlik	-	-	-	-	-	-	1	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
Freud	-	-	-	1	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
Gaile	-	-	-	-	1	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
Hoode	-	1	-	-	-	1	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
Irwin	-	-	-	-	-	-	-	-	-	-	-	-	-	-	1	-	-	-	-	-	-	-	-	-	-
Johns	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	1	-	-	-	-	-	-	-	-	-
Kenny	-	-	-	-	-	-	-	1	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
Leach	-	-	-	-	-	-	-	-	1	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
Mayes	-	-	-	-	-	-	-	1	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
Newmn	-	-	-	-	-	-	-	-	1	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
Oxley	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
Paget	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	2	1	-	-	-	-	-	-
Quinn	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	2	1	-	-	-	-
Rosse	-	-	-	-	-	-	-	-	-	1	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
Stott	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
Thoms	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	1	2	-	-	-	-
Unwin	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	1	-	-	-
Voite	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	2	1	
White	-	-	-	-	-	-	-	-	-	-	-	-	-	-	1	2	2	-	-	-	-	-	-	-	-
Xiang	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	1	2	

```
>-----------------------------------
>--TEST #7: TRANSFERRING PATIENTS --
>-----------------------------------
>
>--Test Description: Make Transferrals to other users and check relationships.
>--                  Test incorrect transferrer(for 'Oxley' & 'Stott').
>
>EXEC showHospital;
```

	User	Amy	Bec	Col	Don	Eli	Fay	Gai	Hue	Ian	Jan	Kim	Lee	May	Nic	Oto	Pat	Qui	Ray	Sue	Tim	Uma	Vic	Wym	Xia
	MRole	Ad	AdS	RN	RN	RN	RN	RN	RN	RN	RN	RNS	RNS	RNS	Dr	Dr	DrS	Ph	PhS	SW	SWS	S1	S1S	S2	S2S
	WdSet	1	7	1	1	1	3	2	2	4	4	7	2	6	3	7	7	7	7	7	7	7	7	6	7
Patient	CurWd	1	1	1	1	1	1	2	2	4	4	1	2	4	1	2	4	1	2	4	1	2	4	2	4
Adams	2	3	1	3	3	3	1	2	2	3	3	1	1	1	3	3	3	3	3	3	3	3	3	3	3
Barns	1	2	1	1	2	2	2	3	3	3	3	1	3	3	2	2	1	3	3	3	3	3	3	3	3
Corry	1	2	1	2	1	2	2	3	3	3	3	1	3	3	2	2	1	3	3	3	3	3	3	3	3
Deane	1	2	1	2	2	1	2	3	3	3	3	1	3	3	2	2	1	3	3	3	3	3	3	3	3
Erlik	1	2	1	2	2	2	1	3	3	3	3	1	3	3	2	2	1	3	3	3	3	3	3	3	3
Freud	1	2	1	1	2	2	2	3	3	3	3	1	3	3	2	2	1	3	3	3	3	3	3	3	3
Gaile	1	2	1	2	1	2	2	3	3	3	3	1	3	3	2	2	1	3	3	3	3	3	3	3	3
Hoode	1	1	1	2	2	1	2	3	3	3	3	1	3	3	2	2	1	3	3	3	3	3	3	3	3
Irwin	1	2	1	2	2	2	2	3	3	3	3	1	3	3	1	2	1	3	3	3	3	3	3	3	3
Johns	1	2	1	2	2	2	2	3	3	3	3	1	3	3	2	1	1	3	3	3	3	3	3	3	3
Kenny	4	3	1	3	3	3	3	1	3	2	2	1	3	1	3	3	3	3	3	3	3	2	1	3	3
Leach	2	3	1	3	3	3	2	2	1	3	3	1	2	1	3	3	3	3	3	3	3	3	3	3	3
Mayes	2	3	1	3	3	3	2	1	2	3	3	1	2	1	3	3	3	3	3	3	3	3	3	3	3
Newmn	2	3	1	3	3	3	2	2	1	3	3	1	2	1	3	3	3	3	3	3	3	3	3	3	3
Oxley	2	3	1	3	3	3	2	2	2	3	3	1	2	1	3	3	3	3	3	3	3	3	3	3	3
Paget	2	3	1	3	3	3	2	2	2	3	3	1	2	1	3	3	3	2	1	3	3	3	3	3	3
Quinn	1	2	1	2	2	2	2	3	3	3	3	1	3	3	2	2	1	3	3	2	1	2	1	3	3
Rosse	4	3	1	3	3	3	3	3	3	1	2	1	3	1	3	3	3	3	3	3	3	2	1	3	3
Stott	4	3	1	3	3	3	3	3	3	2	2	1	3	1	3	3	3	3	3	3	3	2	1	3	3
Thoms	4	3	1	3	3	3	3	3	3	2	2	1	3	1	3	3	3	3	3	1	2	2	1	3	3
Unwin	4	3	1	3	3	3	3	3	3	2	2	1	3	1	3	3	3	3	3	3	3	1	1	3	3
Voite	4	3	1	3	3	3	3	3	3	2	2	1	3	1	3	3	3	3	3	3	3	2	1	2	1
White	2	3	1	3	3	3	2	2	2	3	3	1	1	1	1	2	2	3	3	3	3	3	3	3	3
Xiang	4	3	1	3	3	3	3	3	3	2	2	1	3	1	3	3	3	3	3	3	3	2	1	1	2

```
>--PROCEDURE TEST(S):--
>
>EXEC makeTransfer('Col', 'Don', 'Barns');
>EXEC makeTransfer('May', 'Lee', 'Leach');
>EXEC makeTransfer('Ian', 'Hue', 'Oxley');
>EXEC makeTransfer('Vic', 'Uma', 'Rosse');
>EXEC makeTransfer('Ian', 'Jan', 'Stott');

>EXEC showHospital_CHANGES;
```

	User	Amy	Bec	Col	Don	Eli	Fay	Gai	Hue	Ian	Jan	Kim	Lee	May	Nic	Oto	Pat	Qui	Ray	Sue	Tim	Uma	Vic	Wym	Xia
	MRole	Ad	AdS	RN	RN	RN	RN	RN	RN	RN	RN	RNS	RNS	RNS	Dr	Dr	DrS	Ph	PhS	SW	SWS	S1	S1S	S2	S2S
	WdSet	1	7	1	1	1	3	2	2	4	4	7	2	6	3	7	7	7	7	7	7	7	7	6	7
Patient	CurWd	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
Adams	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
Barns	-	-	-	2	1	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
Corry	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
Deane	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
Erlik	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
Freud	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
Gaile	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
Hoode	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
Irwin	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
Johns	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
Kenny	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
Leach	-	-	-	-	-	-	-	-	-	-	-	-	1	2	-	-	-	-	-	-	-	-	-	-	-
Mayes	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
Newmn	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
Oxley	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
Paget	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
Quinn	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
Rosse	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	1	2	-	-
Stott	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
Thoms	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
Unwin	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
Voite	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
White	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
Xiang	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-

```
>--------------------------------
>--TEST #8: RELEASING PATIENTS --
>--------------------------------
>
>--Test Description: Users remove themselves from patient teams - check relationships.
>--                  Test incorrect release (for 'Oxley' & 'Stott').
>
>EXEC showHospital;
```

	User	Amy	Bec	Col	Don	Eli	Fay	Gai	Hue	Ian	Jan	Kim	Lee	May	Nic	Oto	Pat	Qui	Ray	Sue	Tim	Uma	Vic	Wym	Xia
	MRole	Ad	AdS	RN	RN	RN	RN	RN	RN	RN	RN	RNS	RNS	RNS	Dr	Dr	DrS	Ph	PhS	SW	SWS	S1	S1S	S2	S2S
	WdSet	1	7	1	1	1	3	2	2	4	4	7	2	6	3	7	7	7	7	7	7	7	7	6	7
Patient	CurWd	1	1	1	1	1	1	2	2	4	4	1	2	4	1	2	4	1	2	4	1	2	4	2	4
Adams	2	3	1	3	3	3	1	2	2	3	3	1	1	1	3	3	3	3	3	3	3	3	3	3	3
Barns	1	2	1	2	1	2	2	3	3	3	3	1	3	3	2	2	1	3	3	3	3	3	3	3	3
Corry	1	2	1	2	1	2	2	3	3	3	3	1	3	3	2	2	1	3	3	3	3	3	3	3	3
Deane	1	2	1	2	2	1	2	3	3	3	3	1	3	3	2	2	1	3	3	3	3	3	3	3	3
Erlik	1	2	1	2	2	2	1	3	3	3	3	1	3	3	2	2	1	3	3	3	3	3	3	3	3
Freud	1	2	1	1	2	2	2	3	3	3	3	1	3	3	2	2	1	3	3	3	3	3	3	3	3
Gaile	1	2	1	2	1	2	2	3	3	3	3	1	3	3	2	2	1	3	3	3	3	3	3	3	3
Hoode	1	1	1	2	2	1	2	3	3	3	3	1	3	3	2	2	1	3	3	3	3	3	3	3	3
Irwin	1	2	1	2	2	2	2	3	3	3	3	1	3	3	1	2	1	3	3	3	3	3	3	3	3
Johns	1	2	1	2	2	2	2	3	3	3	3	1	3	3	2	1	1	3	3	3	3	3	3	3	3
Kenny	4	3	1	3	3	3	3	1	3	2	2	1	3	1	3	3	3	3	3	3	3	2	1	3	3
Leach	2	3	1	3	3	3	2	2	1	3	3	1	1	2	3	3	3	3	3	3	3	3	3	3	3
Mayes	2	3	1	3	3	3	2	1	2	3	3	1	2	1	3	3	3	3	3	3	3	3	3	3	3
Newmn	2	3	1	3	3	3	2	2	1	3	3	1	2	1	3	3	3	3	3	3	3	3	3	3	3
Oxley	2	3	1	3	3	3	2	2	2	3	3	1	2	1	3	3	3	3	3	3	3	3	3	3	3
Paget	2	3	1	3	3	3	2	2	2	3	3	1	2	1	3	3	3	2	1	3	3	3	3	3	3
Quinn	1	2	1	2	2	2	2	3	3	3	3	1	3	3	2	2	1	3	3	2	1	2	1	3	3
Rosse	4	3	1	3	3	3	3	3	3	1	2	1	3	1	3	3	3	3	3	3	3	1	2	3	3
Stott	4	3	1	3	3	3	3	3	3	2	2	1	3	1	3	3	3	3	3	3	3	2	1	3	3
Thoms	4	3	1	3	3	3	3	3	3	2	2	1	3	1	3	3	3	3	3	1	2	2	1	3	3
Unwin	4	3	1	3	3	3	3	3	3	2	2	1	3	1	3	3	3	3	3	3	3	1	1	3	3
Voite	4	3	1	3	3	3	3	3	3	2	2	1	3	1	3	3	3	3	3	3	3	2	1	2	1
White	2	3	1	3	3	3	2	2	2	3	3	1	1	1	1	2	2	3	3	3	3	3	3	3	3
Xiang	4	3	1	3	3	3	3	3	3	2	2	1	3	1	3	3	3	3	3	3	3	2	1	1	2

```
>--PROCEDURE TEST(S):--
>
>EXEC releasePatient('Don', 'Barns');
>EXEC releasePatient('Lee', 'Leach');
>EXEC releasePatient('Ian', 'Oxley');
>EXEC releasePatient('Uma', 'Rosse');
>EXEC releasePatient('Ian', 'Stott');

>EXEC showHospital_CHANGES;
```

	User	Amy	Bec	Col	Don	Eli	Fay	Gai	Hue	Ian	Jan	Kim	Lee	May	Nic	Oto	Pat	Qui	Ray	Sue	Tim	Uma	Vic	Wym	Xia
	MRole	Ad	AdS	RN	RN	RN	RN	RN	RN	RN	RN	RNS	RNS	RNS	Dr	Dr	DrS	Ph	PhS	SW	SWS	S1	S1S	S2	S2S
	WdSet	1	7	1	1	1	3	2	2	4	4	7	2	6	3	7	7	7	7	7	7	7	7	6	7
Patient	CurWd	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
Adams	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
Barns	-	-	-	-	2	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
Corry	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
Deane	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
Erlik	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
Freud	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
Gaile	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
Hoode	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
Irwin	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
Johns	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
Kenny	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
Leach	-	-	-	-	-	-	-	-	-	-	-	-	2	-	-	-	-	-	-	-	-	-	-	-	-
Mayes	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
Newmn	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
Oxley	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
Paget	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
Quinn	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
Rosse	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	3	3	-	-
Stott	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
Thoms	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
Unwin	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
Voite	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
White	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
Xiang	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-

```
>--------------------------------
>--TEST #9: REMOVING PATIENTS --
>--------------------------------
>
>--Test Description: Supervisors/SysAdmin remove users from patient teams - check relationships.
>--                  Test SYS removal.  Test incorrect removal(for 'Oxley' & 'Erlik').
>
>EXEC showHospital;
```

	User	Amy	Bec	Col	Don	Eli	Fay	Gai	Hue	Ian	Jan	Kim	Lee	May	Nic	Oto	Pat	Qui	Ray	Sue	Tim	Uma	Vic	Wym	Xia
	MRole	Ad	AdS	RN	RN	RN	RN	RN	RN	RN	RN	RNS	RNS	RNS	Dr	Dr	DrS	Ph	PhS	SW	SWS	S1	S1S	S2	S2S
	WdSet	1	7	1	1	1	3	2	2	4	4	7	2	6	3	7	7	7	7	7	7	7	7	6	7
Patient	CurWd	1	1	1	1	1	1	2	2	4	4	1	2	4	1	2	4	1	2	4	1	2	4	2	4
Adams	2	3	1	3	3	3	1	2	2	3	3	1	1	1	3	3	3	3	3	3	3	3	3	3	3
Barns	1	2	1	2	2	2	2	3	3	3	3	1	3	3	2	2	1	3	3	3	3	3	3	3	3
Corry	1	2	1	2	1	2	2	3	3	3	3	1	3	3	2	2	1	3	3	3	3	3	3	3	3
Deane	1	2	1	2	2	1	2	3	3	3	3	1	3	3	2	2	1	3	3	3	3	3	3	3	3
Erlik	1	2	1	2	2	2	1	3	3	3	3	1	3	3	2	2	1	3	3	3	3	3	3	3	3
Freud	1	2	1	1	2	2	2	3	3	3	3	1	3	3	2	2	1	3	3	3	3	3	3	3	3
Gaile	1	2	1	2	1	2	2	3	3	3	3	1	3	3	2	2	1	3	3	3	3	3	3	3	3
Hoode	1	1	1	2	2	1	2	3	3	3	3	1	3	3	2	2	1	3	3	3	3	3	3	3	3
Irwin	1	2	1	2	2	2	2	3	3	3	3	1	3	3	1	2	1	3	3	3	3	3	3	3	3
Johns	1	2	1	2	2	2	2	3	3	3	3	1	3	3	2	1	1	3	3	3	3	3	3	3	3
Kenny	4	3	1	3	3	3	3	1	3	2	2	1	3	1	3	3	3	3	3	3	3	2	1	3	3
Leach	2	3	1	3	3	3	2	2	1	3	3	1	2	2	3	3	3	3	3	3	3	3	3	3	3
Mayes	2	3	1	3	3	3	2	1	2	3	3	1	2	1	3	3	3	3	3	3	3	3	3	3	3
Newmn	2	3	1	3	3	3	2	2	1	3	3	1	2	1	3	3	3	3	3	3	3	3	3	3	3
Oxley	2	3	1	3	3	3	2	2	2	3	3	1	2	1	3	3	3	3	3	3	3	3	3	3	3
Paget	2	3	1	3	3	3	2	2	2	3	3	1	2	1	3	3	3	2	1	3	3	3	3	3	3
Quinn	1	2	1	2	2	2	2	3	3	3	3	1	3	3	2	2	1	3	3	2	1	2	1	3	3
Rosse	4	3	1	3	3	3	3	3	3	1	2	1	3	1	3	3	3	3	3	3	3	3	3	3	3
Stott	4	3	1	3	3	3	3	3	3	2	2	1	3	1	3	3	3	3	3	3	3	2	1	3	3
Thoms	4	3	1	3	3	3	3	3	3	2	2	1	3	1	3	3	3	3	3	1	2	2	1	3	3
Unwin	4	3	1	3	3	3	3	3	3	2	2	1	3	1	3	3	3	3	3	3	3	1	1	3	3
Voite	4	3	1	3	3	3	3	3	3	2	2	1	3	1	3	3	3	3	3	3	3	2	1	2	1
White	2	3	1	3	3	3	2	2	2	3	3	1	1	1	1	2	2	3	3	3	3	3	3	3	3
Xiang	4	3	1	3	3	3	3	3	3	2	2	1	3	1	3	3	3	3	3	3	3	2	1	1	2

```
>--PROCEDURE TEST(S):--
>
>EXEC removePatient('SYS', 'Pat', 'Deane');
>EXEC removePatient('Fay', 'Pat', 'Erlik');
>EXEC removePatient('Kim', 'Col', 'Freud');
>EXEC removePatient('Bec', 'Kim', 'Leach');
>EXEC removePatient('Lee', 'Kim', 'Oxley');
>EXEC removePatient('May', 'Ian', 'Rosse');

>EXEC showHospital_CHANGES;
```

	User	Amy	Bec	Col	Don	Eli	Fay	Gai	Hue	Ian	Jan	Kim	Lee	May	Nic	Oto	Pat	Qui	Ray	Sue	Tim	Uma	Vic	Wym	Xia
	MRole	Ad	AdS	RN	RN	RN	RN	RN	RN	RN	RN	RNS	RNS	RNS	Dr	Dr	DrS	Ph	PhS	SW	SWS	S1	S1S	S2	S2S
	WdSet	1	7	1	1	1	3	2	2	4	4	7	2	6	3	7	7	7	7	7	7	7	7	6	7
Patient	CurWd	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
Adams	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
Barns	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
Corry	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
Deane	-	-	-	-	-	-	-	-	-	-	-	-	-	-	3	3	3	-	-	-	-	-	-	-	-
Erlik	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
Freud	-	-	-	2	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
Gaile	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
Hoode	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
Irwin	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
Johns	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
Kenny	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
Leach	-	-	-	-	-	-	-	-	-	-	-	2	-	-	-	-	-	-	-	-	-	-	-	-	-
Mayes	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
Newmn	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
Oxley	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
Paget	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
Quinn	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
Rosse	-	-	-	-	-	-	-	-	-	2	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
Stott	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
Thoms	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
Unwin	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
Voite	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
White	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
Xiang	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-

```
>--------------------------------------
>--TEST #10: VIEWING PATIENT RECORDS --
>--------------------------------------
>
>--Test Description: Test that appropriate categories are chosen when access is sought.
>
>--PROCEDURE TEST(S):--
>
>EXEC getRecord('Eli', 'Deane');
getRecord:              You =Eli have accessed the record of Deane

>EXEC getRecord('Fay', 'Deane');
getRecord:              You =Fay must enter an access reason to access the record of Deane

>EXEC getRecord('Gai', 'Deane');
getRecord:              You =Gai must enter an access report now or later to access the record of Deane

>EXEC showHospital;
```

	User	Amy	Bec	Col	Don	Eli	Fay	Gai	Hue	Ian	Jan	Kim	Lee	May	Nic	Oto	Pat	Qui	Ray	Sue	Tim	Uma	Vic	Wym	Xia
	MRole	Ad	AdS	RN	RN	RN	RN	RN	RN	RN	RN	RNS	RNS	RNS	Dr	Dr	DrS	Ph	PhS	SW	SWS	S1	S1S	S2	S2S
	WdSet	1	7	1	1	1	3	2	2	4	4	7	2	6	3	7	7	7	7	7	7	7	7	6	7
Patient	CurWd	1	1	1	1	1	1	2	2	4	4	1	2	4	1	2	4	1	2	4	1	2	4	2	4
Adams	2	3	1	3	3	3	1	2	2	3	3	1	1	1	3	3	3	3	3	3	3	3	3	3	3
Barns	1	2	1	2	2	2	2	3	3	3	3	1	3	3	2	2	1	3	3	3	3	3	3	3	3
Corry	1	2	1	2	1	2	2	3	3	3	3	1	3	3	2	2	1	3	3	3	3	3	3	3	3
Deane	1	2	1	2	2	1	2	3	3	3	3	1	3	3	3	3	3	3	3	3	3	3	3	3	3
Erlik	1	2	1	2	2	2	1	3	3	3	3	1	3	3	2	2	1	3	3	3	3	3	3	3	3
Freud	1	2	1	2	2	2	2	3	3	3	3	1	3	3	2	2	1	3	3	3	3	3	3	3	3
Gaile	1	2	1	2	1	2	2	3	3	3	3	1	3	3	2	2	1	3	3	3	3	3	3	3	3
Hoode	1	1	1	2	2	1	2	3	3	3	3	1	3	3	2	2	1	3	3	3	3	3	3	3	3
Irwin	1	2	1	2	2	2	2	3	3	3	3	1	3	3	1	2	1	3	3	3	3	3	3	3	3
Johns	1	2	1	2	2	2	2	3	3	3	3	1	3	3	2	1	1	3	3	3	3	3	3	3	3
Kenny	4	3	1	3	3	3	3	1	3	2	2	1	3	1	3	3	3	3	3	3	3	2	1	3	3
Leach	2	3	1	3	3	3	2	2	1	3	3	2	2	2	3	3	3	3	3	3	3	3	3	3	3
Mayes	2	3	1	3	3	3	2	1	2	3	3	1	2	1	3	3	3	3	3	3	3	3	3	3	3
Newmn	2	3	1	3	3	3	2	2	1	3	3	1	2	1	3	3	3	3	3	3	3	3	3	3	3
Oxley	2	3	1	3	3	3	2	2	2	3	3	1	2	1	3	3	3	3	3	3	3	3	3	3	3
Paget	2	3	1	3	3	3	2	2	2	3	3	1	2	1	3	3	3	2	1	3	3	3	3	3	3
Quinn	1	2	1	2	2	2	2	3	3	3	3	1	3	3	2	2	1	3	3	2	1	2	1	3	3
Rosse	4	3	1	3	3	3	3	3	3	2	2	1	3	1	3	3	3	3	3	3	3	3	3	3	3
Stott	4	3	1	3	3	3	3	3	3	2	2	1	3	1	3	3	3	3	3	3	3	2	1	3	3
Thoms	4	3	1	3	3	3	3	3	3	2	2	1	3	1	3	3	3	3	3	1	2	2	1	3	3
Unwin	4	3	1	3	3	3	3	3	3	2	2	1	3	1	3	3	3	3	3	3	3	1	1	3	3
Voite	4	3	1	3	3	3	3	3	3	2	2	1	3	1	3	3	3	3	3	3	3	2	1	2	1
White	2	3	1	3	3	3	2	2	2	3	3	1	1	1	1	2	2	3	3	3	3	3	3	3	3
Xiang	4	3	1	3	3	3	3	3	3	2	2	1	3	1	3	3	3	3	3	3	3	2	1	1	2

```
>-----------------------------------------------
>--TEST #11: VIEWING HIDDEN PATIENT RECORDS --
>-----------------------------------------------
>
>--Test Description: Test that reporting procedures are activated when text is provided or not provided.
>
>--PROCEDURE TEST(S):--
>
>EXEC getHiddenRecord('Col', 'Corry', 'Gave medication');
getRecord:              You =Col have accessed the record of Corry

>EXEC getHiddenRecord('Eli', 'Corry', 'Checked history');
getRecord:              You =Eli have accessed the record of Corry

>EXEC getHiddenRecord('Fay', 'Corry', 'Checked medication');
getRecord:              You =Fay have accessed the record of Corry

>EXEC getHiddenRecord('Qui', 'Corry', 'Checked for overdose');
getRecord:              You =Qui have accessed the record of Corry

>EXEC getHiddenRecord('Ray', 'Corry', 'Report Later');
getRecord:              You =Ray have accessed the record of Corry

>EXEC getHiddenRecord('Sue', 'Corry', 'Checked for violence');
getRecord:              You =Sue have accessed the record of Corry

>EXEC getHiddenRecord('Tim', 'Corry', 'Report Later');
getRecord:              You =Tim have accessed the record of Corry
```

```
>-----------------------------------
>--TEST #12: REQUESTS AND REPORTS --
>-----------------------------------
>
>--Test Description: Make referral and transferral requests.
>
>EXEC showHospital;
```

	User	Amy	Bec	Col	Don	Eli	Fay	Gai	Hue	Ian	Jan	Kim	Lee	May	Nic	Oto	Pat	Qui	Ray	Sue	Tim	Uma	Vic	Wym	Xia
	MRole	Ad	AdS	RN	RN	RN	RN	RN	RN	RN	RN	RNS	RNS	RNS	Dr	Dr	DrS	Ph	PhS	SW	SWS	S1	S1S	S2	S2S
	WdSet	1	7	1	1	1	3	2	2	4	4	7	2	6	3	7	7	7	7	7	7	7	7	6	7
Patient	CurWd	1	1	1	1	1	1	2	2	4	4	1	2	4	1	2	4	1	2	4	1	2	4	2	4
Adams	2	3	1	3	3	3	1	2	2	3	3	1	1	1	3	3	3	3	3	3	3	3	3	3	3
Barns	1	2	1	2	2	2	2	3	3	3	3	1	3	3	2	2	1	3	3	3	3	3	3	3	3
Corry	1	2	1	2	1	2	2	3	3	3	3	1	3	3	2	2	1	3	3	3	3	3	3	3	3
Deane	1	2	1	2	2	1	2	3	3	3	3	1	3	3	3	3	3	3	3	3	3	3	3	3	3
Erlik	1	2	1	2	2	2	1	3	3	3	3	1	3	3	2	2	1	3	3	3	3	3	3	3	3
Freud	1	2	1	2	2	2	2	3	3	3	3	1	3	3	2	2	1	3	3	3	3	3	3	3	3
Gaile	1	2	1	2	1	2	2	3	3	3	3	1	3	3	2	2	1	3	3	3	3	3	3	3	3
Hoode	1	1	1	2	2	1	2	3	3	3	3	1	3	3	2	2	1	3	3	3	3	3	3	3	3
Irwin	1	2	1	2	2	2	2	3	3	3	3	1	3	3	1	2	1	3	3	3	3	3	3	3	3
Johns	1	2	1	2	2	2	2	3	3	3	3	1	3	3	2	1	1	3	3	3	3	3	3	3	3
Kenny	4	3	1	3	3	3	3	1	3	2	2	1	3	1	3	3	3	3	3	3	3	2	1	3	3
Leach	2	3	1	3	3	3	2	2	1	3	3	2	2	2	3	3	3	3	3	3	3	3	3	3	3
Mayes	2	3	1	3	3	3	2	1	2	3	3	1	2	1	3	3	3	3	3	3	3	3	3	3	3
Newmn	2	3	1	3	3	3	2	2	1	3	3	1	2	1	3	3	3	3	3	3	3	3	3	3	3
Oxley	2	3	1	3	3	3	2	2	2	3	3	1	2	1	3	3	3	3	3	3	3	3	3	3	3
Paget	2	3	1	3	3	3	2	2	2	3	3	1	2	1	3	3	3	2	1	3	3	3	3	3	3
Quinn	1	2	1	2	2	2	2	3	3	3	3	1	3	3	2	2	1	3	3	2	1	2	1	3	3
Rosse	4	3	1	3	3	3	3	3	3	2	2	1	3	1	3	3	3	3	3	3	3	3	3	3	3
Stott	4	3	1	3	3	3	3	3	3	2	2	1	3	1	3	3	3	3	3	3	3	2	1	3	3
Thoms	4	3	1	3	3	3	3	3	3	2	2	1	3	1	3	3	3	3	3	1	2	2	1	3	3
Unwin	4	3	1	3	3	3	3	3	3	2	2	1	3	1	3	3	3	3	3	3	3	1	1	3	3
Voite	4	3	1	3	3	3	3	3	3	2	2	1	3	1	3	3	3	3	3	3	3	2	1	2	1
White	2	3	1	3	3	3	2	2	2	3	3	1	1	1	1	2	2	3	3	3	3	3	3	3	3
Xiang	4	3	1	3	3	3	3	3	3	2	2	1	3	1	3	3	3	3	3	3	3	2	1	1	2

```
>--PROCEDURE TEST(S):--
>
>EXEC seekReferral('Wym', 'Corry');
>EXEC seekTransferral('Col', 'Corry');
>EXEC selfreferral('Pat', 'Mayes');
>EXEC selfreferral('Oto', 'Newmn');
>EXEC selfreferral('Nic', 'Newmn');

>EXEC showHospital_CHANGES;
```

	User	Amy	Bec	Col	Don	Eli	Fay	Gai	Hue	Ian	Jan	Kim	Lee	May	Nic	Oto	Pat	Qui	Ray	Sue	Tim	Uma	Vic	Wym	Xia
	MRole	Ad	AdS	RN	RN	RN	RN	RN	RN	RN	RN	RNS	RNS	RNS	Dr	Dr	DrS	Ph	PhS	SW	SWS	S1	S1S	S2	S2S
	WdSet	1	7	1	1	1	3	2	2	4	4	7	2	6	3	7	7	7	7	7	7	7	7	6	7
Patient	CurWd	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
Adams	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
Barns	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
Corry	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
Deane	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
Erlik	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
Freud	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
Gaile	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
Hoode	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
Irwin	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
Johns	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
Kenny	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
Leach	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
Mayes	-	-	-	-	-	-	-	-	-	-	-	-	-	-	2	2	1	-	-	-	-	-	-	-	-
Newmn	-	-	-	-	-	-	-	-	-	-	-	-	-	-	2	1	2	-	-	-	-	-	-	-	-
Oxley	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
Paget	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
Quinn	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
Rosse	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
Stott	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
Thoms	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
Unwin	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
Voite	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
White	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
Xiang	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-

```
>-----------------------------------------------
>--TEST #13: VIEWING REPORTS AND LATE REPORTS --
>-----------------------------------------------
>
>--Test Description: Test that reasons and reports are available to users and admin.
>--                  Test late report submission.
>
>--PROCEDURE TEST(S):--
>
>EXEC showUserReports('Bec');
#11: Note from Wym(Spec2) re Corry: Wym requests referral for Corry

>EXEC showUserReports('Don');
#1: Note from Col(RN) re Corry: Gave medication
#3: Note from Eli(RN) re Corry: Checked history
#5: Note from Fay(RN) re Corry: Checked medication
#12: Note from Wym(Spec2) re Corry: Wym requests referral for Corry
#15: Note from Col(RN) re Corry: Col requests transferral for Corry

>EXEC showUserReports('Kim');
#2: Note from Col(RN) re Corry: Gave medication
#4: Note from Eli(RN) re Corry: Checked history
#6: Note from Fay(RN) re Corry: Checked medication
#13: Note from Wym(Spec2) re Corry: Wym requests referral for Corry
#16: Note from Col(RN) re Corry: Col requests transferral for Corry

>EXEC showUserReports('Pat');
#14: Note from Wym(Spec2) re Corry: Wym requests referral for Corry

>EXEC showAdminReports;
#7: Report type 3 from Qui(Pharm) re Corry: Checked for overdose
#9: Report type 3 from Sue(Soc) re Corry: Checked for violence

>EXEC showDueAdminReports;
#8: Report type 3 from Ray(PharmS) re Corry due.
#10: Report type 3 from Tim(SocS) re Corry due.

>EXEC submitLateReport(8, 'Ray', 'Corry', 'Emergency assistance');
submitLateReport:        Submit late report =#8 from Ray

>EXEC showAdminReports;
#7: Report type 3 from Qui(Pharm) re Corry: Checked for overdose
#8: Report type 3 from Ray(PharmS) re Corry: Emergency assistance
#9: Report type 3 from Sue(Soc) re Corry: Checked for violence

>EXEC showDueAdminReports;
#10: Report type 3 from Tim(SocS) re Corry due.


PL/SQL procedure successfully completed.
```

```
>-------------------------------
>--TEST #14: PATIENT LISTINGS --
>-------------------------------
>
>--Test Description: Test that lists are complete and correct.
>
>EXEC showHospital;
```

	User	Amy	Bec	Col	Don	Eli	Fay	Gai	Hue	Ian	Jan	Kim	Lee	May	Nic	Oto	Pat	Qui	Ray	Sue	Tim	Uma	Vic	Wym	Xia
	MRole	Ad	AdS	RN	RN	RN	RN	RN	RN	RN	RN	RNS	RNS	RNS	Dr	Dr	DrS	Ph	PhS	SW	SWS	S1	S1S	S2	S2S
	WdSet	1	7	1	1	1	3	2	2	4	4	7	2	6	3	7	7	7	7	7	7	7	7	6	7
Patient	CurWd	1	1	1	1	1	1	2	2	4	4	1	2	4	1	2	4	1	2	4	1	2	4	2	4
Adams	2	3	1	3	3	3	1	2	2	3	3	1	1	1	3	3	3	3	3	3	3	3	3	3	3
Barns	1	2	1	2	2	2	2	3	3	3	3	1	3	3	2	2	1	3	3	3	3	3	3	3	3
Corry	1	2	1	2	1	2	2	3	3	3	3	1	3	3	2	2	1	3	3	3	3	3	3	3	3
Deane	1	2	1	2	2	1	2	3	3	3	3	1	3	3	3	3	3	3	3	3	3	3	3	3	3
Erlik	1	2	1	2	2	2	1	3	3	3	3	1	3	3	2	2	1	3	3	3	3	3	3	3	3
Freud	1	2	1	2	2	2	2	3	3	3	3	1	3	3	2	2	1	3	3	3	3	3	3	3	3
Gaile	1	2	1	2	1	2	2	3	3	3	3	1	3	3	2	2	1	3	3	3	3	3	3	3	3
Hoode	1	1	1	2	2	1	2	3	3	3	3	1	3	3	2	2	1	3	3	3	3	3	3	3	3
Irwin	1	2	1	2	2	2	2	3	3	3	3	1	3	3	1	2	1	3	3	3	3	3	3	3	3
Johns	1	2	1	2	2	2	2	3	3	3	3	1	3	3	2	1	1	3	3	3	3	3	3	3	3
Kenny	4	3	1	3	3	3	3	1	3	2	2	1	3	1	3	3	3	3	3	3	3	2	1	3	3
Leach	2	3	1	3	3	3	2	2	1	3	3	2	2	2	3	3	3	3	3	3	3	3	3	3	3
Mayes	2	3	1	3	3	3	2	1	2	3	3	1	2	1	2	2	1	3	3	3	3	3	3	3	3
Newmn	2	3	1	3	3	3	2	2	1	3	3	1	2	1	2	1	2	3	3	3	3	3	3	3	3
Oxley	2	3	1	3	3	3	2	2	2	3	3	1	2	1	3	3	3	3	3	3	3	3	3	3	3
Paget	2	3	1	3	3	3	2	2	2	3	3	1	2	1	3	3	3	2	1	3	3	3	3	3	3
Quinn	1	2	1	2	2	2	2	3	3	3	3	1	3	3	2	2	1	3	3	2	1	2	1	3	3
Rosse	4	3	1	3	3	3	3	3	3	2	2	1	3	1	3	3	3	3	3	3	3	3	3	3	3
Stott	4	3	1	3	3	3	3	3	3	2	2	1	3	1	3	3	3	3	3	3	3	2	1	3	3
Thoms	4	3	1	3	3	3	3	3	3	2	2	1	3	1	3	3	3	3	3	1	2	2	1	3	3
Unwin	4	3	1	3	3	3	3	3	3	2	2	1	3	1	3	3	3	3	3	3	3	1	1	3	3
Voite	4	3	1	3	3	3	3	3	3	2	2	1	3	1	3	3	3	3	3	3	3	2	1	2	1
White	2	3	1	3	3	3	2	2	2	3	3	1	1	1	1	2	2	3	3	3	3	3	3	3	3
Xiang	4	3	1	3	3	3	3	3	3	2	2	1	3	1	3	3	3	3	3	3	3	2	1	1	2

```
>--PROCEDURE TEST(S):--
>
>EXEC showAllPatients;
Adams
Barns
Corry
Deane
Erlik
Freud
Gaile
Hoode
Irwin
Johns
Kenny
Leach
Mayes
Newmn
Oxley
Paget
Quinn
Rosse
Stott
Thoms
Unwin
Voite
White
Xiang

>EXEC showWardPatients('Nic');
Barns
Corry
Deane
Erlik
Freud
Gaile
Hoode
Irwin
Johns
Quinn

>EXEC showMyPatients('Nic');
Irwin
White

>EXEC showMyWardPatients('Nic');
Irwin

>EXEC showMyGroupPatients('Nic');
Barns
Corry
Erlik
Freud
Gaile
Hoode
Irwin
Johns
Mayes
Newmn
Quinn
White

>EXEC showTeam('Kenny');
Bec
Gai
Kim
May
Vic
```

```
>--------------------------------
>--TEST #15: LOGOFF PROCEDURE --
>--------------------------------
>
>--Test Description: Test that logoff procedure produces correct ward and relationships.
>--                  Test persistent users.
>
>EXEC showHospital;
```

	User	Amy	Bec	Col	Don	Eli	Fay	Gai	Hue	Ian	Jan	Kim	Lee	May	Nic	Oto	Pat	Qui	Ray	Sue	Tim	Uma	Vic	Wym	Xia
	MRole	Ad	AdS	RN	RN	RN	RN	RN	RN	RN	RN	RNS	RNS	RNS	Dr	Dr	DrS	Ph	PhS	SW	SWS	S1	S1S	S2	S2S
	WdSet	1	7	1	1	1	3	2	2	4	4	7	2	6	3	7	7	7	7	7	7	7	7	6	7
Patient	CurWd	1	1	1	1	1	1	2	2	4	4	1	2	4	1	2	4	1	2	4	1	2	4	2	4
Adams	2	3	1	3	3	3	1	2	2	3	3	1	1	1	3	3	3	3	3	3	3	3	3	3	3
Barns	1	2	1	2	2	2	2	3	3	3	3	1	3	3	2	2	1	3	3	3	3	3	3	3	3
Corry	1	2	1	2	1	2	2	3	3	3	3	1	3	3	2	2	1	3	3	3	3	3	3	3	3
Deane	1	2	1	2	2	1	2	3	3	3	3	1	3	3	3	3	3	3	3	3	3	3	3	3	3
Erlik	1	2	1	2	2	2	1	3	3	3	3	1	3	3	2	2	1	3	3	3	3	3	3	3	3
Freud	1	2	1	2	2	2	2	3	3	3	3	1	3	3	2	2	1	3	3	3	3	3	3	3	3
Gaile	1	2	1	2	1	2	2	3	3	3	3	1	3	3	2	2	1	3	3	3	3	3	3	3	3
Hoode	1	1	1	2	2	1	2	3	3	3	3	1	3	3	2	2	1	3	3	3	3	3	3	3	3
Irwin	1	2	1	2	2	2	2	3	3	3	3	1	3	3	1	2	1	3	3	3	3	3	3	3	3
Johns	1	2	1	2	2	2	2	3	3	3	3	1	3	3	2	1	1	3	3	3	3	3	3	3	3
Kenny	4	3	1	3	3	3	3	1	3	2	2	1	3	1	3	3	3	3	3	3	3	2	1	3	3
Leach	2	3	1	3	3	3	2	2	1	3	3	2	2	2	3	3	3	3	3	3	3	3	3	3	3
Mayes	2	3	1	3	3	3	2	1	2	3	3	1	2	1	2	2	1	3	3	3	3	3	3	3	3
Newmn	2	3	1	3	3	3	2	2	1	3	3	1	2	1	2	1	2	3	3	3	3	3	3	3	3
Oxley	2	3	1	3	3	3	2	2	2	3	3	1	2	1	3	3	3	3	3	3	3	3	3	3	3
Paget	2	3	1	3	3	3	2	2	2	3	3	1	2	1	3	3	3	2	1	3	3	3	3	3	3
Quinn	1	2	1	2	2	2	2	3	3	3	3	1	3	3	2	2	1	3	3	2	1	2	1	3	3
Rosse	4	3	1	3	3	3	3	3	3	2	2	1	3	1	3	3	3	3	3	3	3	3	3	3	3
Stott	4	3	1	3	3	3	3	3	3	2	2	1	3	1	3	3	3	3	3	3	3	2	1	3	3
Thoms	4	3	1	3	3	3	3	3	3	2	2	1	3	1	3	3	3	3	3	1	2	2	1	3	3
Unwin	4	3	1	3	3	3	3	3	3	2	2	1	3	1	3	3	3	3	3	3	3	1	1	3	3
Voite	4	3	1	3	3	3	3	3	3	2	2	1	3	1	3	3	3	3	3	3	3	2	1	2	1
White	2	3	1	3	3	3	2	2	2	3	3	1	1	1	1	2	2	3	3	3	3	3	3	3	3
Xiang	4	3	1	3	3	3	3	3	3	2	2	1	3	1	3	3	3	3	3	3	3	2	1	1	2

```
>--PROCEDURE TEST(S):--
>
>EXEC logoff('Amy');
>EXEC logoff('Eli');
>EXEC logoff('Hue');
>EXEC logoff('Tim');
>EXEC logoff('Wym');

>EXEC showHospital_CHANGES;
```

	User	Amy	Bec	Col	Don	Eli	Fay	Gai	Hue	Ian	Jan	Kim	Lee	May	Nic	Oto	Pat	Qui	Ray	Sue	Tim	Uma	Vic	Wym	Xia
	MRole	Ad	AdS	RN	RN	RN	RN	RN	RN	RN	RN	RNS	RNS	RNS	Dr	Dr	DrS	Ph	PhS	SW	SWS	S1	S1S	S2	S2S
	WdSet	1	7	1	1	1	3	2	2	4	4	7	2	6	3	7	7	7	7	7	7	7	7	6	7
Patient	CurWd	0	-	-	-	0	-	-	0	-	-	-	-	-	-	-	-	-	-	-	0	-	-	0	-
Adams	-	-	-	-	-	-	-	-	3	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
Barns	-	3	-	-	-	3	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
Corry	-	3	-	-	-	3	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
Deane	-	3	-	-	-	3	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
Erlik	-	3	-	-	-	3	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
Freud	-	3	-	-	-	3	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
Gaile	-	3	-	-	-	3	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
Hoode	-	3	-	-	-	3	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
Irwin	-	3	-	-	-	3	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
Johns	-	3	-	-	-	3	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
Kenny	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
Leach	-	-	-	-	-	-	3	3	3	-	-	3	3	3	-	-	-	-	-	-	-	-	-	-	-
Mayes	-	-	-	-	-	-	-	-	3	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
Newmn	-	-	-	-	-	-	-	-	3	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
Oxley	-	-	-	-	-	-	-	-	3	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
Paget	-	-	-	-	-	-	-	-	3	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
Quinn	-	3	-	-	-	3	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
Rosse	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
Stott	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
Thoms	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	3	-	-	-	-
Unwin	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
Voite	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	3	-
White	-	-	-	-	-	-	-	-	3	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
Xiang	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-

Appendix B – Simulation Test Notes

Running the Tests

The tests were performed by running the test script from the SQL*Plus command line interface. The tests utilised the Oracle database and PL/SQL function and procedure set that was previously loaded in the same way from the database creation and loading scripts. In order to access the Oracle database, users were set up using Oracle Enterprise Manager. The tests were performed by accessing the database as the 'SYS' (SYS_DBA) user.

Hospital Simulation Details

The simulated hospital contained three wards Ward A, Ward B and Ward C. The numbers 1, 2 and 4 were assigned respectively to represent each of these wards. The ward set of each of the 24 staff members was represented by adding the appropriate ward numbers together. The 24 patients were each only ever assigned to a single ward.

Each staff member was assigned a role. Any staff member with a supervisory role was given the power to remove members from patient teams. Staff members with the roles of Pharmacist, Social Worker, or Specialists, were made persistent users, which meant that they were not to be removed from teams when the patient was assigned to another ward or when they were logged out. The Doctor default role on Ward B was denoted unassigned, which meant that it was not assigned to individual doctors on the patient's admission. Rather, doctors on that ward had to assign themselves to a patient team. They could only do this if there was no other doctor already on the team.

PL/SQL Procedures and Functions

The 15 tests were performed using the top level PL/SQL procedures in the set. Altogether there were in fact 85 procedures and functions in the set. By running the

top level procedures, the lower procedures and functions were called as necessary. The fact that the top level procedures performed properly indicated that the lower level procedures and functions were also doing their job.

The Tests

The following sections detail each of the individual tests. While most of the test execution lines were designed to perform a specific task that was intended to happen, a number test execution lines were run to test that thing which weren't allowed to happen were actually disallowed. For example, the self-referral of Nic in Test#12 did not proceed because Oto had already been assigned to the same patient.

TEST #1: LOGON PROCEDURE – SET UP

The tests begin with no staff or patients allocated to the current hospital. In terms of the staff, that means that they are not currently on the job, but they are employed. In the case of the patients, they have not yet been admitted to any of the wards in the simulation. The first two tests essentially just initialise the hospital into a working state.

The logon procedure allocates a staff member to a ward. This ward represents the ward that they are currently on. They may move to any ward specified by their ward set. Once logged on, a staff member can be allocated patients.

Each logon is achieved by executing the logon procedure for the staff member (for example, `EXEC logon('Amy','Ward A');`). The results shown in the Changes table indicate that each of the staff members have been logged on to the appropriate wards. This is indicated by the numbers that appear in the CurWd row.

TEST #2: ADMITTING PATIENTS – SET UP

This test initialises the patients into wards. It tests the patient admission procedure (for example, `EXEC admitPatient('Adams','Ward A');`). The CurWd column in the Changes table shows that each patient has been placed on the correct ward. On admission each patient is allocated the default ward team. The team for each patient is made up of the staff members with a '1' relationship to the patient. A

‘2’ relationship shows colleagues of the team members. Analysis of this table shows that the default teams have been assigned correctly and that the ward set of the staff members is taken into account correctly. No unexpected allocations were made.

TEST #3: LOGON PROCEDURE

This test retests the logon procedure with patients and staff already in the hospital. The Change table results show the correct results have been achieved. The current ward appears correctly as does the colleague relationships to patients who have previously been allocated the default team.

TEST #4: ADMITTING PATIENTS

Again, this test retests the patient admission procedure. Note that patients ‘White’ and ‘Liang’ were initially not allocated a ward. Now they have been. Again, the Changes table shows the expected default team allocations.

TEST #5: RE-ADMITTING PATIENTS

This test is aimed at testing the ability of the system to transfer a patient from the ward that they are currently on to another ward. The results should show four things; firstly, the removal of all initial team members with non-persistent roles; secondly, the retention of team members with persistent roles; thirdly, the addition of new default team members from the new ward; and fourthly the correct handling of Colleague relationships.

The Changes table shows that all these things occur correctly. It should be noted that some staff members, such as ‘Bec’ in the case of ‘Kenny’ still remain on the team (their ‘1’ relationship hasn’t changed). This is not because they hold persistent roles, but because they work on both wards and are on the default team of the ward no which the patient has been transferred to.

Also, the presence of a ‘3’ in the changes table indicates that the staff member has either been removed from the team or that the member who was their colleague has been removed form the team.

TEST #6: REFERRING PATIENTS

In terms of the TAC and PAC models, referring patients means that a current team member, or the system administrator, adds a new member to the patient's team. In this test one referral was made for each patient (for example, `EXEC makeReferral('Bec', 'Col','Barns');`). The name of both the referrer and the referee are specified, as well as the patient's name. Again, the Changes table shows that the changes brought about by the procedures are correct. The administrative task of the 'SYS' user making a referral shows that this supervisory facility works correctly.

TEST #7: TRANSFERRING PATIENTS

The transferral of patients can only be done between staff members with the same role. It is essentially for cases where one staff member takes over the care of one of their colleagues' patients. This procedure should only execute when there is a legitimate colleague relationship between the staff members. The current carer should be removed from the team and the new one added where the relationship is legitimate. The results again are correct. The transfers involving 'Oxley' and 'Stott' do not proceed because the colleague relationship does not exist between the referrer and the referee.

TEST #8: RELEASING PATIENTS

Releasing a patient is analogous to a practitioner removing the patient from their current patients because their treatment of the patient has either been completed or terminated. The tests involved here were fairly straightforward. A couple of the test case involved trying to release patients where the remover was not already on the team. These two cases, 'Oxley' and 'Stott', were again dealt with correctly, as were the simple cases.

TEST #9: REMOVING PATIENTS

Removing patients is an administrative task. The system should allow users who either have a supervisory role, or system administrators, to make such removals. The patient removal procedure (for example, `EXEC removePatient('Kim','Col',`

`'Freud');` requires the specification of the names both the person making the removal and the person to be removed, as well as the patient's name. In practice, applications making use of this procedure should probably ensure that records are kept of such removals and that staff members being removed are notified by the system of their removal.

TEST #10: VIEWING PATIENT RECORDS

This is the central procedure in the package. All attempts to access records use this procedure. The job of the procedure is to check the relationship status that currently exists between the patient and the staff member and to administer the appropriate type of access process. The access processes either give direct access or require some reason for access or access report to be given. In the simulation there was no need to represent actual patient records, only to put in place the access control mechanism. As such, the results of executing the record retrieval procedure (for example, `EXEC getRecord('Eli','Deane');`) are given as text descriptions of how the procedure deals with the access attempt. All the procedure executions gave the anticipated results.

TEST #11: VIEWING HIDDEN PATIENT RECORDS

This procedure is used for accesses that require a textual reason or report to be submitted in order to gain access. The text is therefore included as a parameter. Reports to the system administrator are required for emergency and Critical accesses. It is envisaged that these types of accesses will often be made in situations where there is no time for the staff member making the access to immediately write the report. There is therefore the opportunity for the staff member to submit the report later. In these cases the text "Report Later", is supplied by the application and the procedure deals with it by noting that a report is due. The execution of the procedures in all cases granted access to the user, as was required. The procedures in this test produce database entries, which are checked in Test#14.

TEST #12: REQUESTS AND REPORTS

This test is on the feature of the system which allows one user to make a request to the members on a team to join the team. It also test the self-referral aspect of the

system, where unfilled roles in a team that are denoted as unassigned may be filled by staff members of the correct role assigning themselves to the task. The `seekReferral` and `seekTransferral` procedures produce database entries, which are checked in Test#14. The `selfReferral` procedure actually changes the relationship state. The Changes table shows that the self-referrals proceeded in two out of the three cases. In the third case the self-referral did not occur because the position in the team had already been filled by the second self-referral. This reflected correct operation of the system.

TEST #13: VIEWING REPORTS AND LATE REPORTS

This test is aimed at ensuring that the reporting procedures did in fact produce the expected reports. In the cases of accesses by colleagues the reasons should have been added to the messages of appropriate team members. In the cases involving Emergency and Critical accesses the reports should have been added to the system administration messages. System administration messages should also deal correctly with reports that are to be filed at a later time. That is, reports that are due should be noted and when the reports are submitted they should no longer appear as being due.

The test required that the reports that should have been submitted during the previous two tests were checked. All were found to be in order. In addition to this, a late report (#8) was submitted. It can be seen that after its submission the lists of admin reports are properly amended.

TEST #14: PATIENT LISTINGS

The procedures that produce various lists of patients are not essentially required by the PAC model, but they do provide useful a considerably useful tool that can be utilised by hospital applications. The procedures allow concise patient lists to be produced for workflow purposes. Patient team users can also be listed.

The test firstly output the current hospital state table. Examples of each of the five listing procedures were then executed. The lists produced were in agreement with the current hospital state. *WardPatients* refer to patients that are on the same ward as the specified staff member is currently on. *MyPatients* are all the patients for which the given staff member is on the patient's team. *MyWardPatients* are all the patients

for whom the given staff member is on the patient's team, who are also on the same ward as the staff member. *MyGroupPatients* are all patients for whom the specified staff member is either on the patient's team or is the colleague of a staff member on the team (a '1' or '2' relationship).

TEST #15: LOGOFF PROCEDURE

The logoff procedure removes the specified user from all teams except where the user has a persistent role. Colleague relationships are affected in the usual way by such removals. Logoff also sets the current ward of the user to '0'. Each of the executions of the logoff procedure (for example, `EXEC logoff('Tim');`) produced the expected results.

Appendix C – Database Scripts and Test Data

The attached CD contains the following files:

- PAC_CREATE.sql — Creates the tables and PL/SQL
- PAC_DROP.sql — Clears the tables and PL/SQL
- PAC_LOAD.sql — Loads the simulation test data
- PAC_RELOAD.sql — Reinitialises the simulation test data
- PAC_TEST.sql — Contains the 15 simulation tests & useful SQL*Plus Commands
- PAC_Test_Results.txt — Contains the full simulation test results