THE DEVELOPMENT -

~~Designing~~ Secure and Usable Systems

Ivan Fléchais

A dissertation submitted in partial fulfilment
of the requirements for the degree of
Doctor of Philosophy
of the
University of London

Department of Computer Science
University College London

February 2005

ProQuest LLC
789 East Eisenhower Parkway
P.O. Box 1346
Ann Arbor, MI 48106-1346

Abstract

"People are the weakest link in the security chain" – Bruce Schneier

The aim of the thesis is to investigate the process of designing secure systems, and how designers can ensure that security mechanisms are usable and effective in practice. The research perspective is one of security as a socio-technical system.

A review of the literature of security design and Human Computer Interactions in Security (HCISec) reveals that most security design methods adopt either an organisational approach, or a technical focus. And whilst HCISec has identified the need to improve usability in computer security, most of the current research in this area is addressing the issue by improving user interfaces to security tools. Whilst this should help to reduce users' errors and workload, this approach does not address problems which arise from the difficulty of reconciling technical requirements and human factors. To date, little research has been applied to socio-technical approaches to secure system design methods. Both identifying successful socio-technical design approaches and gaining a better understanding of the issues surrounding their application is required to address this gap.

Appropriate and Effective Guidance for Information Security (AEGIS) is a socio-technical secure system development methodology developed for this purpose. It takes a risk-based approach to security design and focuses on recreating the contextual information surrounding the system in order to better inform security decisions, with the aim of making these decisions better suited to users' needs. AEGIS uses a graphical notation defined in the UML Meta-Object Facility to provide designers with a familiar and well-supported means of building models.

Grid applications were selected as the area in which to apply and validate AEGIS. Using the research methodology *Action Research*, AEGIS was applied to a total of four Grid case studies. This allowed in the first instance the evaluation and refinement of AEGIS on real-world systems. Through the use of the qualitative data analysis methodology *Grounded Theory*, the design session transcripts gathered from the Action Research application of AEGIS were then further analysed. The resulting analysis identified important factors affecting the design process – separated into categories of *responsibility*, *motivation*, *stakeholders* and *communication*. These categories were then assembled into a model informing the factors and issues that affect socio-technical secure system design. This model therefore provides a key theoretical insight into real-world issues and is a useful foundation for improving current practice and future socio-technical secure system design methodologies.

Acknowledgements

I would firstly like to thank BT and the EPSRC for providing the CASE studentship number 01305495 which funded this research. Very special thanks also go to my supervisor M. Angela Sasse, without whom none of this would have happened. I also wish to thank my second supervisor Stephen Hailes for his help and critical insight which improved this work in so many ways, and Cecilia Mascolo for her help with UML semantics.

Like most other significant pieces of work, this was only achieved with the support of colleagues, friends and family. So thanks to my UCL colleagues Sacha Brostoff, Dirk Weirich, Phillip Bonhard, John McCarthy and Jens Riegelsberger for their willingness to have coffee.

Whilst thanks are chronological, they are by no means preferential. Therefore I would particularly like to thank my parents for giving me these opportunities, and my wife Lorna for her unwavering belief and support. This thesis would and could not have happened without them.

Contents

List of Figures

List of Tables

1 Introduction

There has been a consistent increase in the number of computer security breaches reported by businesses. In 2004, 94% of UK businesses surveyed reported suffering at least one incident [45], compared to 44% in 2002 [44], and 42.6% of US businesses surveyed reported an increase in the total number of electronic crimes and network systems or data intrusions compared to 2003 [40]. The average UK reported cost of an incident in 2004 was £10,000 per incident, and although the true cost of computer security breaches is hard to ascertain (in some surveys 45% of respondents are unwilling or unable to quantify the losses attributed to security breaches [39]), computer security breaches in the UK were thought to *"continue to cost several billions of pounds"* [45]. Other figures for US businesses put the total annual cost of security incidents from $141,496,560 (based on estimates from 262 respondents out of 494) [39] to $666,000,000 (based on estimates from 338 respondents out of 500) [40]. Whilst an accurate idea of the true cost of computer security incidents is impossible to obtain from these surveys, it is clear that computer security still has many unresolved problems and issues that need to be addressed.

In the past, computer security research has focussed on technical defences to safeguard systems. But it has become clear that technical measures are not enough:

"People are the weakest link in the security chain."

This statement by Bruce Scheiner [103] has been confirmed by reformed hacker Kevin Mitnick [81], who claims that the most effective and devastating means of attacking a system is through social engineering – an attack that targets authorised users of that system and attempts to trick, con or otherwise compel them to break security policy.
Recent research efforts to address human factors in security have concluded that security mechanisms are too difficult to use [123], and that most users do not maliciously break security policies [10, 101, 117], but do so as a consequence of bad design, complex requirements or an inadequate security culture. The focus of human-computer interaction in security (HCISec) research has been the improvement of user interfaces to security tools [50, 56, 73]. Whilst this is an important part of improving the overall usability of secure systems, and hence the effectiveness of the security, it is not enough.

Social engineering targets users who have access to the secure system, bypassing software and hardware countermeasures. Thus social engineering specifically targets the

social element of the *socio-technical system.* Such an attack is successful if there is a mismatch between the behaviour that software and hardware countermeasures require from the user and the actual behaviour of the user. For example, a user may decide to be helpful by sharing their password with someone masquerading as technical support staff carrying out updates. However, one of the requirements of password-based authentication mechanisms is that passwords should never be divulged. This difference between what users should do and what they actually do is a significant problem for computer security.

The selection of a secure system's software and hardware countermeasures occurs during the security design stage. This raises the question of whether security design methods address the need for getting users to behave in the way that the security mechanisms require. A mismatch between expected and actual user behaviour can occur as a result of either of two factors (this is discussed in more detail in section 2):

(1) Even though security is a socio-technical system that has both technical and human components, most current security design methods do not address human factors. This may result in design decisions that do not consider the needs and requirements of both social and technical aspects of the system.

(2) The security design method can address both technical and human factors, but is not applied by developers as intended. This can be the result of a variety of factors, such as for example, the method being incompatible with other design tools, the priority of functional needs over security design, or the design technique being difficult to use.

A large number of computer security design methods exist, such as formal development methods, checklists or risk management (see section 2). These design methods, however, tend to focus on hardware and software countermeasures and do not explicitly consider the needs of users. An equally large number of information systems design methods exists that approach security from a variety of different angles, such as information modelling, responsibility modelling or business process perspectives. Whilst these methods do address human issues in a secure organisation, the problem here is that they do not integrate very well into the design process of a software or hardware technical system. Both these types of security design methods fall under (1): design methods that do not accommodate both technical and social aspects of secure systems.

A few proposed design methods address (1) by recommending socio-technical approaches to secure system design [16, 64, 65, 69, 80]. These types of approach actively seek to incorporate both organisational and technical needs into the design method, thereby improving the overall system design.

Whilst some of these proposals have been tested practically and are reported to be successful, there has been little research into (2) – identifying significant real-world factors affecting the secure system design and how these are influenced or addressed by the proposals. Without a framework describing these factors and their significance during the practical application, it is difficult to know why or how the proposed solution is effective. There is therefore a need to identify the type and significance of the factors that affect the secure system design process in order to improve it, and inform future research efforts into secure system development.

Based on the identification of these problems (see section 2.8), a socio-technical secure system design process was developed called Appropriate and Effective Guidance for Information Security (AEGIS). This process provides a simple means of addressing both security and human factors whilst incorporating into a technical system design lifecycle. The empirical application of AEGIS is presented as validation of the process, and analysed to provide a framework of the factors and issues that surround practical socio-technical security design.

1.1 Definitions

Presented here are definitions of essential concepts that are used throughout this thesis. The detailed review of current literature which lead to the selection and formulation of these definitions can be found in section 2.3.

Computer security deals with the deterrence, avoidance, prevention, detection and reaction to events in and affecting a computer system that are undesirable to the owner of that system.

Information security consists of the concepts, techniques, technical measures, and administrative measures used to protect information assets from deliberate or inadvertent unauthorized acquisition, damage, disclosure, manipulation, modification, loss, or use. [79]

Given that information assets are inextricably linked to computer systems, and computer systems operate in and rely on larger social settings, the effective distinction

between the two concepts is minimal. In this thesis, the distinction between *information* and *computer* security refers to the difference in the research fields. Where the term security is used without any qualifier the notions of computer or information security can be used interchangeably.

Usability is *"the ease with which a user can learn to operate, prepare inputs for, and interpret outputs of a system or component"* [66]

1.2 Research Problem

As mentioned above, traditional computer security focuses on technical elements, such as encryption algorithms, secure communication protocols or firewalls. In contrast to this, many approaches in the field of information systems address security from a non-technical perspective – i.e. looking at organisational processes and describing appropriate policies, measures and mechanisms for preventing potential security violations. Usability problems in secure systems [10, 29, 100, 101, 117, 118, 123] are not only the consequence of bad interface design, but also arise out of a mismatch between how technical systems are designed and how socio-technical systems operate in practice.

Whilst there has been some research into the socio-technical aspects of information system security [64, 65, 69, 80], there is still a lot of ground to cover in this field. Even though some of these methods have been empirically tested [64, 69], there has been no investigation into practical real-world factors and their effect on a socio-technical approach to secure system design. Knowledge of these factors is necessary to understand more about the environment, pressures and limitations surrounding socio-technical secure system design. This in turn can be used to inform future research and improve on existing security design methods.

Given these limitations, the research question addressed by this thesis can be framed as:
How can the design of usable and secure socio-technical systems be better *understood* and *supported*?
This thesis addresses the question by ***presenting* a method for *integrating technical and socio-technical aspects of secure system design*. As a result of empirical application, the method is *refined* and *evaluated*. In addition to this, the *real-world factors* that affected the development process are identified, analysed and presented in a *model of the factors affecting socio-technical secure system design*.**

For this purpose the research conducted will be presented in the thesis according to two themes:

1. The identification of issues in the development of secure technical systems. This will consist of:
   - **1.1.** A theoretical perspective of security design issues based on the literature.
   - **1.2.** An empirical perspective of security design issues based on the analysis of the case studies of AEGIS (developed in **(2.1)**), culminating in a model of the relevant factors.
2. The presentation and evaluation of a socio-technical design method for secure systems. This will consist of:
   - **2.1.** The formulation, based on **(1.1)**, of Appropriate and Effective Guidance for Information Security (AEGIS). AEGIS is a secure system design technique that actively adopts a socio-technical approach in order to assist developers in designing secure systems.
   - **2.2.** The practical application of AEGIS in order to evaluate the benefits and disadvantages of that process.
   - **2.3.** The refinement of AEGIS, based on the results of **(1.2)** and **(2.2)**.

1.3 Research Scope

This thesis does not seek to present a comparative evaluation of secure system design techniques. Instead, the evaluation of AEGIS is based on the findings of its application to four case studies, in which it was used to help designers identify security requirements for Grid applications.

The Action Research case studies in which AEGIS is applied also provide the opportunity for identifying and analysing significant real-world factors in a socio-technical secure design process. Whilst some of these factors may apply to more conventional secure system design approaches, this is not within the scope of this research.

Furthermore, as with any research affecting engineering or design methodologies, generalising beyond the immediate empirical basis is difficult. The research presented in this thesis seeks to reduce the problem of generalisation of the findings by presenting four different case studies in which the AEGIS was applied, and explain and highlight the differences and similarities between them. In addition to this, since all the case

studies are Grid projects, an analysis of the major features of Grids is given with an argumentation as to their significance and their relation to the results (see section 3.4).

1.4 Research Approach

There is a wide variety of research relevant to the problem field, ranging from software and security engineering, information systems and computer security or HCISec. Each discipline has approached this problem within its own disciplinary knowledge and methods. These efforts have produced software engineering methods that do not address issues of both security and usability, or information systems security methods that do not integrate into a software or hardware development lifecycle. Each of these research disciplines can contribute pertinent expert knowledge: software engineering addresses the state-of-the-art of system design and implementation, computer and information security provide security counter measures and design methods, and HCISec addresses human factors in security. However, taken in isolation, these approaches lack the necessary coverage for resolving the problem. As a result it is necessary to adopt a multidisciplinary and flexible research strategy.

In order to address research themes **(1.2)** and **(2.2)**, empirical evidence of the practical application of a security design process is required. From a logistical standpoint, however, empirical security research is difficult and hampered by the fact that few organisations or projects are willing to open their systems up to scrutiny – generally citing security concerns as the reason. Therefore, it is necessary to adopt a pragmatic research strategy, and given the nature of the research, a qualitative and exploratory research approach was selected as a means of gathering and interpreting the empirical data. *Action Research* and *Grounded Theory* [113] were chosen as a means of addressing the research problem (see section 3).

Action Research [19, 77, 84] describes a research strategy in which the researcher is actively involved with the research material, as opposed to being a simple observer. Originally coined by Kurt Lewin in the 1940's, Action Research is a structured research approach that *"identifies a question to investigate, develops an action plan, implements the plan, collects data, and reflects the findings of the investigation."* [71]

It has been argued that Action Research is ideal for studying information systems methods in a practical setting [20, 21], although only a few studies on secure methods have been published [69, 105, 112].

In this thesis, *Action Research* provides the framework in which qualitative information is gathered about the application of a socio-technical secure design process, and to evaluate the proposed process. *Grounded Theory* is used as a further analytical methodology to formulate the key factors that affect the design of secure systems, in order to inform and understand the process of designing security.

Grounded Theory [78] is *"an inductive theory discovery methodology that allows the researcher to develop a theoretical account of the general features of a topic while simultaneously grounding the account in empirical observations or data."* Grounded Theory is a qualitative theory building methodology, which ensures the validity of its results by continually comparing the output to the gathered data.

In the following description of the research approach, the relevant research themes are highlighted in **(bold brackets)**. It also should be noted that the thesis research approach follows the five Action Research steps [70, 114]:

A) identifying a research question (diagnosing),
B) developing an action plan (action planning),
C) implementing the plan (action taking),
D) gathering and analysing the data (evaluating),
E) reflecting on the findings of the investigation (specifying learning).

These steps are highlighted in the following in *[square italic brackets]*, and the relevant research themes are highlighted in **(bold brackets)**.

After *[A]* reviewing the available literature surrounding the different fields of research, and identifying research gaps and problems currently affecting the process of designing security **(1.1)**, *[B]* a socio-technical secure design process was proposed **(2.1)** named "Appropriate and Effective Guidance for Information Security" (AEGIS). This design process was then *[C]* applied to a total of four real-world case studies in a series of documented workshops. The analysis of this qualitative data using *[D]* Grounded Theory [113] provided a structured means of exploring the practical real-world factors and issues that affect a secure design process **(1.2)**. The results from the cases studies, *[E]* also provided a keen insight into strengths and weaknesses of AEGIS **(2.2)**, and together with the Grounded Theory analysis of the factors influencing the secure design process served as a basis for refining AEGIS **(2.3)**.

1.5 Contributions

There is an extensive amount of research surrounding security, in the various fields of software engineering, information security, computer security and HCISec. Despite this wealth of activity, there has been little research in the field of socio-technical approaches to secure system design. Whilst a number of approaches have been proposed, none of them incorporate insights from all four research areas. In addition, whilst these approaches have yielded positive results from practical application, none have undertaken research into identifying the specific factors that affect the design of a socio-technical security system. Without this theoretical framework, it is impossible to know which elements of these existing approaches are responsible for positive outcomes and which elements can be improved on further.

The research presented in this thesis describes a socio-technical secure design process that actively seeks to reconcile software engineering, information security, computer security and human factors.

As a means of validating this process, it is empirically applied to four real-world projects. In addition to validating the process, the empirical data is also analysed in order to uncover the significant real-world factors that affect socio-technical security design. This serves to inform a theoretical understanding explaining what factors the process addresses and therefore giving an insight into why the process is successful.

The contributions of this thesis are therefore summarised as follows:

1. The socio-technical secure system design process AEGIS (in chapters 3 and 7).
2. An evaluation of the AEGIS design process through empirical research (in chapters 4, 5 and 6).
3. An analysis and model of the real-world factors that affect the socio-technical process of developing secure systems (in chapters 4, 5 and 7).

1.6 Thesis Structure

Chapter 2 presents a review of the relevant research literature from the disciplines of software engineering, information security, computer security and HCISec.

Chapter 3 describes the research methodology and the basics of the socio-technical secure system design process AEGIS.

Chapter 4 describes the first case study in which AEGIS was applied. A Grounded Theory analysis and model is presented within which provides a detailed look at practical factors that affect the secure system design process.

Chapter 5 describes the second case study, on a project which investigates the use of patient records for medical research. The Grounded Theory analysis builds on the initial model and provides a different insight into the secure system design process, and a discussion of the differences with the first case study.

Chapter 6 describes the remaining two case studies in which AEGIS was applied. In these the AEGIS process was initially taught to graduate students and they in turn applied it to two different projects. The analysis of this data provides validation of the Grounded Theory model, as well as providing more objective information about the benefits and disadvantages of AEGIS.

Chapter 7 presents the final story of the factors that affect the secure design process, discusses and reviews the significance of the case studies and presents the finalised version of AEGIS based on these results.

2 Literature Review

2.1 Introduction

In this chapter, an overview of the historical developments leading to the need for computer security is presented. This is followed by a detailed examination of how four different fields of research address computer security:

1. software engineering,
2. computer security design,
3. information systems security,
4. human-computer interaction in security (HCISec).

From the review of the current state of the art of computer security as seen from four distinct perspectives, the main weakness remains the need for accommodating human factors in the design of a secure technical system. Software engineering and computer security design approaches tend to adopt an objectivist and regulatory approach (see section 2.6 for more detail) to secure system design, which is at odds with the findings from the fields of HCISec and information systems security that indicate that real-world security is affected by social and somewhat subjective considerations.

Based on this, it is concluded that a secure design method must reconcile both social and technical aspects of the system with the goal of assuring that desirable interactions are actually carried out and undesirable interactions are prevented, detected, reacted to, deterred or avoided.

2.2 Background

Taken from [36, 122], a brief overview of the history of computer security is presented here.

Security, and more specifically the Allied War Machine, was at the heart of the development of the digital computer during the 1940s. The need to break Axis encryption codes galvanised research into creating one of the key inventions of the 20th century.

Computers evolved from these wartime origins, becoming both smaller and more powerful, yet despite technical advances such as transistors, they still followed exactly the same principles as their vacuum tube driven ancestors. The possibility of mechanistically manipulating vast quantities of data, which made computers the ideal tool for analysing and breaking codes – also known as cryptanalysis – were also found

to be useful in many domains other than military, mostly academic such as mathematics and engineering, but also economic such as banking. With the growing capabilities of computer hardware came the possibility for increasing the complexity of software, allowing the number-crunching capabilities of computers to be harnessed for less directly mathematical endeavours.

The miniaturisation of the computer and a declining cost led away from the centralised mainframe architecture and towards dispersed workstation architectures. This led to novel problems, such as the need to share data distributed across a variety of different machines, or the need to share particular peripherals, such as printers. These problems were resolved through the use of computer networks, not only allowing different workstations to communicate, but supporting social communication programs such as email.

Since then, the usage of computer networks has increased exponentially. The evolution of ARPANET into the Internet, combined with technologies supporting social interactions, such as the World Wide Web, email, newsgroups and forums, has led to an unprecedented popular interest in computers and the Internet. With such a growing user base, the demand for conducting more and more transactions (business or social) online has also been increasing.

It is this combination of high usage and increasing reliance on technology for business that makes the need for computer security more pressing. The continuing escalation in the number of security incidents and breaches, from as few as 252 in 1990 to 82,094 in 2002 and 137,529 in 2003 [1] illustrates this growing need for practical computer security.

2.3 Definitions

2.3.1 Stakeholders

The systems theory concept of stakeholders is introduced here. A definition of a stakeholder is *"a person such as an employee, customer or citizen who is involved with an organization, society, etc. and therefore has responsibilities towards it and an interest in its success"* [4]. With regards to computer and software systems, stakeholders include for example users, developers, administrators, owners, security experts and any other party that holds a stake in the system. It is widely accepted, particularly in the field of Human-Computer Interaction (HCI), that stakeholders have a vital role to play in the design of a software or hardware system.

2.3.1.1 Participation in System Design

Considering the seminal research of Enid Mumford and her colleagues [83, 85] in participative system design, the inclusion of users and other stakeholders in design is not a new concept. Indeed, it can go back as far as ancient Greece, where democracy arose as a means of making decisions. More recent researchers in this field were *"contributors to the human relations' movement in industry and they include famous names from the United States as Mayo, McGregor, and Likert. The Tavistock Institute in England had a major influence on organizational participation from the 1950's onwards as did Norwegian social scientists such Thorsrud and Herbst."* [83]

The intrinsic notion of participation implies the involvement of more than one party, and as such the following definition is given for participation:

"(...) a process in which two or more parties influence each other in making plans, policies or decisions. It is restricted to decisions that have future effects on all those making the decisions or on those represented by them" [83].

It is frequently argued that participation in information system design is very important to the success of a system [18, 83, 85, 97, 115, 120]. Studies [35, 68, 120] of participative design practices, such as the ETHICS method (Effective Technical and Human Implementation of Computer-based Systems) [83], suggest that - whilst user involvement does not guarantee a successful system design - the use of a participative approach does foster a climate that is conducive to successful development, and can also lead to more pragmatic designs [68].

The key elements of participative approaches revolve around representing the relevant viewpoints of different parties in such a manner as to achieve a consensus. The degree and type of stakeholder involvement can vary, ranging from consultative (where stakeholders are asked for their views) to representative (where selected stakeholders represent the views of a wider group within the design group) to consensual (where every stakeholders is involved in making a design decision).

As a consequence of this, some of the problems of participative design revolve around getting groups of people to communicate and agree. As such, conflicts of interest, poor communication, a lack of trust or rapidly changing goals can all be serious problems (although these may not be exclusive to participative approaches). Furthermore, since different stakeholder groups will have different interests, it is only natural that conflicts will arise during decision-making. Resolution of these conflicts is a central aspect of participative approaches and centres heavily on negotiation and reconciliation. It is

through this negotiation that the system design better accommodates the needs of its stakeholders.

2.3.2 Information and Computer Security

The most widely held definition of *information security* is described as a set of properties that must be upheld. Commonly referred to as the *CIA of security*, the BS7799/ISO17799 [28] standard describes information security as the protection of information for:

1. **Confidentiality**: protecting sensitive information from unauthorised disclosure or intelligible interception.
2. **Integrity**: safeguarding the accuracy and completeness of information and computer software.
3. **Availability**: ensuring that information and vital services are available to users when required.

Computer security can be defined as the technological and managerial procedures applied to computer systems to ensure the availability, integrity and confidentiality of information managed by the computer system [6].

From this definition, it can be noted that computer security is the application of information security to a particular domain, namely that of computing. Given that the application of information security usually involves computers and networks, the two terms are occasionally used interchangeably. In the literature, however, the field of information security (see section 2.6) has a much greater tradition of taking an organisational approach to security compared to computer security, which tends to focus on technical issues. As mentioned previously, the term *security* is used in this thesis to refer both to information and computer security.

Gollman [60], however, rightly argues that the CIA definitions are incomplete in that they are only aspects of access control and put their emphasis exclusively on the prevention of undesirable events. He proposes other desirable properties, such as:

- **Accountability**: audit information must be selectively kept and protected so that actions affecting security can be traced back to the responsible party.
- **Dependability**: the property of a computer system such that reliance can justifiably be placed on the service it delivers.

He defines security as relating to the protection of assets. Protective measures can be roughly classified into prevention, detection and reaction [60]:

- **Prevention**: Measures that avert damage to an asset.
- **Detection**: Measures that afford the knowledge of when, how and who has damaged an asset.
- **Reaction**: Measures that stop ongoing damage and recover from damage to an asset.

Additional categories of avoidance and deterrence should also be included in this list [53]:

- **Avoidance**: Measures that discontinue the possibility of a given threat, or transfer liability to a third party.
- **Deterrence**: Measures that discourage the abuse of and damage to an asset.

Although deterrence could be subsumed under the notion of prevention, it is useful to distinguish that deterrence is both aimed only at people (i.e. the source of attacks) and is understood to be fallible (i.e. it does not prevent an attack, it merely discourages an attacker from engaging in one). A number of authors [50, 54, 103, 104] have pointed out that the assumption that there is a "silver bullet" – i.e. that protective measures must be absolutely perfect in order to be of any use for security - is a common misconception. This assumption can still be seen in attitudes and statements from experts, for instance:

"Firewalls can be effective only if all traffic must go through them to get from the outside of the protected network and vice versa" [24].

Thisimplies that unless they satisfy the given condition of *"all traffic must go through them"*, firewalls are completely ineffective, as opposed to having a diminished effectiveness.

Parker [89] also argues that the CIA definitions are incomplete and inaccurate. He asserts, for example, that the definition of integrity is incorrect in that it contains a reference to accuracy, and that availability is circularly defined as being available. He further states that the definitions only hold insofar as they apply to actions under the control of the owner. They do not cover third-party events such as interception, repudiation – the ability of a third party to deny a past interaction, or misrepresentation. Parker proposes new definitions of security which should be rated in terms of:

- **Availability**: usability of information for a purpose.
- **Utility**: usefulness of information for a purpose.
- **Integrity**: completeness, wholeness and readability of information and quality being unchanged from a previous state.

- **Authenticity**: validity, conformance and genuineness of information.
- **Confidentiality**: limited observation and disclosure of knowledge.
- **Possession**: the holding, control and ability to use information.

The difficulty with these new definitions is that they involve highly subjective notions, such as usefulness, genuineness, or readability. Whilst the process of designing and building security undeniably involves subjective assessments, the definition of security concepts should not be open to debate. It is more important to discuss the need for and extent to which these concepts are necessary in a given system.

Yet other experts prefer to describe security as ideals to be achieved. Ross Anderson [15], for example, describes the field of security engineering as *"building systems to remain dependable in the face of malice, error or mischance. As a discipline, it focuses on the tools, processes, and methods needed to design, implement, and test complete systems, and to adapt existing systems as their environment evolves."*

Already apparent from the variety of definitions arising from the need to qualify security, it is clear that apart from being complex, the mandate of computer security can be narrow or large, depending on the exponent.

For the purposes of the following, a system represents the combination of technical, managerial and human components working together for the accomplishment of specified goals:

Security deals with the deterrence, avoidance, prevention, detection and reaction to events in a system that are undesirable to the owner of that system.

This definition is useful in that it distinguishes between:

1. How security works – deterrence, avoidance, prevention, detection and reaction.
2. What security applies to – undesirable events in a system.
3. Who requires security – the owner of the system.

In addition, the definition of a system to include both technical and human components reflects the need for security to address socio-technical issues, as well as technical and organisational concerns.

2.3.3 Human-Computer Interaction

The research discipline of Human-Computer Interaction (HCI) is now well-established, with multiple conferences and publications dedicated to the subject. Central to HCI is the notion of usability. Usability is defined as *"...the ease with which a user can learn*

to operate, prepare inputs for, and interpret outputs of a system or component." [66]. Given this definition, a significant portion of HCI is concerned with user interfaces to software systems.

While usability and user interface design can be said to be looking at the issues of making a given interaction easier, HCI design has come to develop a broader picture, which includes identifying and resolving conflicting goals in a socio-technical system consisting of the stakeholders and their activities. The design process must not only consider the characteristics of the immediate user, but their goals when interacting with the system, and the physical, social and cultural context in which that interaction takes place. A *goal* differs from a *task* because the goal of a system is its *raison d'être*, whereas a task is a process whereby goals are achieved. Successfully identifying the most effective and efficient tasks for achieving specific goals is a key notion in HCI design.

Also central to the discipline of HCI is the concept of human factors and how these affect and shape how people react to computer systems. Human factors typically refer to the intrinsic properties of people, such as short-term memory, visual acuity, physical dexterity, etc. These properties can strongly influence the design of a system, most visibly at the interface level, but also at a more fundamental level such as the underlying model of operation of the system. For example, problems can arise if users' mental models of the operation of the system differ from its real operating model (see [99] for a more complete discussion of mental models and HCI).

Many design methods exist for achieving technical systems that accommodate human factors, and reviewing these goes beyond the scope of this thesis. Instead the guiding principles of Contextual Design [23] are presented here as a sample HCI design methodology, which is used later to inform the AEGIS design methodology.

2.3.3.1 Contextual Design

The core principle of Contextual Design is the idea that a good design is derived from understanding the needs and working practices of customers and other stakeholders in the system.

Contextual Design consists of seven parts:

1. Contextual Inquiry: uncovers who customers really are and how they work on a day-to-day basis. This helps identify their needs, desires and approaches to the work at hand.

2. Work Modelling: captures the work of individuals and organisations in diagrams to provide different perspectives on how work is done.
3. Consolidation: brings data from individual customer interviews together. This allows the identification of common patterns and structures without losing individual variation.
4. Work Redesign: uses the consolidated data to drive conversations about how to improve work by using technology to support the new work practice.
5. User Environment Design: captures the floor plan of the new system. It shows each part of the system, how it supports the user's work, exactly what function is available in that part, and how the user gets to and from other parts of the system.
6. Mock-up and test with customers: Paper prototyping develops rough mock-ups of the system to represent windows, dialog boxes, buttons, and menus.
7. Putting into practice: Prioritisation, planning and flexibility to organisational limitations are necessary to help the transition to implementation.

By following these steps, Contextual Design provides practitioners with the tools to identify customer needs and design, test and deploy solutions that are acceptable to the users.

In the next section, the field of software engineering is reviewed and in particular how software engineering addresses matters of security.

2.4 Software Engineering

Software engineering is the application of a systematic, disciplined, quantifiable approach to the development, operation, and maintenance of software [67]. Software engineering as a discipline is concerned with all the elements of the *development lifecycle* of a software system, from the initial feasibility study to the requirements elicitation and management, design, implementation, testing, deployment, and maintenance up to the final decommissioning of the system. In theory, software engineering provides all the tools and the techniques necessary to build systems that fulfil customer expectations, including security. In practice, the growing number of security incidents in industry [1, 45], coupled with the increasing amount of research that shows that security is not well-suited to human factors (see sections 2.3.3 & 2.7), indicates that current software engineering practices are not enough.

2.4.1 Requirements

With regards to security, software engineering classifies a security requirement as a *non-functional requirement.* A *functional requirement* is a requirement that specifies a function that a system or system component must be able to perform [67], whereas a non-functional requirement is usually defined as a constraint on the manner in which the system can behave. Therefore usability, performance and security are all considered to be non-functional requirements. In their software engineering roadmap for security, [46] identify that security requirements generally occur late in the requirements capture, usually after the functional requirements have been completed, leading to security being added on as an afterthought. One solution to this are *abuse cases* [80], an adapted form of *use cases,* which have been proposed as a means of capturing security requirements in a manner that is compatible with current requirements capture tools and notations. Abuse cases work by capturing a scenario in which actual harm comes to the system, and documenting the interactions and privileges abused in order to later inform the design of the system and prevent this. Although practical in their integration into current engineering tools and approaches, by focussing on privilege abuse, abuse cases place the emphasis on access control, which can detract from other security concerns such as audit, backup, accountability or dependability.

2.4.2 Design

A major aspect of software engineering is that of modelling the system in order to identify architectural issues and problems that can be solved without having to rework the actual programming. A general tenet of the argument for using a structured software engineering approach is that the earlier in the lifecycle the need for modifying the system is identified, the cheaper the cost of actually implementing that modification.

With regards to security, much effort has been placed on producing models that lend themselves to security analysis. An interesting example of this is UMLSec [72], an extension to UML that provides developers with a means of evaluating a UML specification for vulnerabilities, a view of security in the context of the system, and a means of specifying necessary security properties.

Formal security models have long been proposed for security-critical areas, for example in order to achieve the Orange Book B2 Mandatory Security Level [43] a formal security model of the system must be present. A major problem with formal modelling, however, is that the act of modelling the system introduces assumptions about the surrounding environment. These assumptions either simplify the environment to the

point where attacks are possible but not taken into account by the model, or complicate the model to the point where actually building and using it is too complicated [42]. This argument can be extended to cover other security modelling techniques, such as UMLSec, and makes the point that these techniques are useful but their limitations should be recognised. They do not provide a silver-bullet solution to security analysis – rather a useful tool in the overall analysis process.

Reusing successful designs is a key aim of software engineering, and in the same way that *software patterns* can provide access to successful software design approaches, *security patterns* [2] have been proposed as a means of reusing successful security architectures. Security patterns offer a clear, concise and practical means for developers to access good security design knowledge (see section 2.5.1). Although useful in documenting good security design, security patterns are not sufficient in achieving successful security design because they do not help with the reasoning and decision making parts of security design (see section 2.5).

2.4.3 Implementation

Implementation errors are some of the most frequent sources of security vulnerabilities. The most famous of these is the buffer overflow, where a write operation in a program is unbounded allowing a specially tailored input to overwrite other sensitive areas of memory, resulting in the system being compromised. Other common implementation problems for example are race conditions, predictable pseudo-random number generators or bad cryptographic protocol implementations [116]. The ideal solution, from a software engineering point of view, is automated code generation directly from the design model. In the absence of this, experience and knowledge [116] about secure programming practice and testing for security are the best solutions.

2.4.4 Testing

Testing is currently the only way in which security problems with the actual implementation can be detected. Automated security analysis tools (such as Nessus[1], Retina[2] or Satan) exist that allow an implementation to be tested against known vulnerabilities, but these only work for systems that are built using existing software.

---

[1] http://www.nessus.org

[2] http://eeye.com

Automated testing, such as unit testing, is currently useful during implementation to ensure that the functionality of previously written code is maintained throughout the development, and benefits from being customisable to the system. When it comes to security, however, unit testing is rather limited. The inherent difficulty is that testing for security means identifying an absence of bugs – something that for non-trivial sized projects is not currently possible. Not finding any security problems is not synonymous with their absence. As a result the most effective type of testing is that conducted by security experts that have both an understanding of attacks and experience in reviewing code.

Despite the limitations of testing, cryptographic algorithms for example currently only gain credibility after long periods of peer review. There is no guarantee that the algorithms are completely secure, but despite the uncertainty, the amount of testing they undergo is the only means of measuring their strength. This is currently the source of a strong debate between advocates of open versus closed source systems. On the one hand, proponents of open source systems argue that only systems that are completely open and subject to scrutiny can achieve security. On the other hand, the counter argument - referred to as 'security by obscurity' - is that opening the source to the community provides attackers with a greater knowledge about the system, and therefore makes the likelihood of successful attacks being found higher. Given the lack of alternative means of assessing security, this characteristic highlights that security design is still very much a craft discipline (see section 2.6.3).

2.5 Computer Security Design

It is commonly argued that the security must be designed into a system and not added as an afterthought [15]. However in practice, most systems have security added on, which can result in security that is ill-suited, expensive to maintain and inefficient. The reasons why some systems continue to be designed without security can fall into three categories:

1. Security is deliberately sacrificed in the design. For example because of time and cost concerns.
2. Security is not viewed as important. For example many security problems in the Internet or email have their roots in the fact that security was not considered to be necessary during their initial design – a design that was originally viewed as a proof of concept.

3. Security is desired, but the wrong decisions are made during design and implementation.

Computer security design aims to address the problem identified in point 3, which is to ensure the means of building good security. There are three separate aspects to designing security that are necessary for a successful system:

1. Knowledge of security.
2. Reasoning and decision making.
3. Knowledge of the system.

2.5.1 Knowledge of Security

Much of the literature about computer security relies on anecdotes to relay information about past exploits and crimes [15, 76, 89, 93, 103, 104]. These give insight into the inventiveness, mistakes and methods that attackers have used in the past. Anecdotes prove to be useful in two distinct ways:

- They supply information about past attacks and defences.
- They provide an insight into future attacks.

The disadvantage of using anecdotes is that they are not:

- Succinct.
- Versatile.

The information contained within an anecdote is hard to summarise and therefore hard to impart to third parties. In addition, the lessons that can be learned from one given attack can only be applied to a different area with great care, as the environment will probably be completely different. It would be very unlikely that the conditions that held for the anecdote would apply to other situations without considerable effort expended to identify the differences and analyse their significance.

Other popular sources of security knowledge are advisories issued by security related bodies, such as CERT[3], BugTraq[4] or RISKS[5]. These issue regular warnings about vulnerabilities in software, generally consisting of what software is affected and how. The difficulty here lies in their purely technical nature of the warning, which requires very specific knowledge to understand and act on. Another problem is that as a result of the specificity of these advisories, it is necessary to actively keep up-to-date.

---

[3] http://www.cert.org

[4] http://www.securityfocus.com/

[5] http://catless.ncl.ac.uk/Risks

This also applies to the software patches that software and hardware companies tend to publish together with announcements of security vulnerabilities. These announcements are usually accompanied by a severity rating. The problem with any rating of this sort is that it is only useful if the assumptions and knowledge that went into the rating are apparent. It is very possible that security patches deemed 'critical' by the issuing company are addressing an issue that has no relevance to a specific system, given that the implicit context in which the vulnerability may be exploited is not transferable to this system.

A separate type of information about security is also necessary to design a secure system: knowledge of security countermeasures. There exist many sources of information about specific security countermeasures, e.g. [15, 58, 81, 89, 103]. Although this information is not provided in a standard format, they all provide detailed information about given mechanisms and their application.

A slightly different source of security knowledge is that of security patterns [2] (see section 2.4.2). In the same way that software programming has benefited from reusing successful programming approaches through the adapted use of architectural patterns, capturing successful security designs in architectural patterns will provide better understanding and application of good security. Security patterns provide two advantages over other non-structured approaches:

- They provide knowledge about good system design – where the other approaches only limit themselves to specific technologies.
- They provide knowledge in a standard format – presenting developers with a universal interface to understanding the different techniques and technologies.

2.5.2 Reasoning and Decision Making

The most widespread methods for reasoning and making decisions about security can be grouped into four categories:

- descriptive and *ad hoc* methods,
- checklists,
- guidelines,
- risk management.

2.5.2.1 Descriptive Methods

Descriptive and *ad hoc* methods tend to follow from the experience of an individual or a small group of practitioners. Much of the advice generally given in these methods is in the form of stories or case studies. They describe in detail many aspects of computer security and illustrate them with appropriate examples. They do not, however, prescribe a method for securing a system or developing a system securely. *"Engineering Secure Software"* by Ross Anderson [15], for example, provides an encyclopaedic look at security (thereby providing knowledge about security – see previous section), but does not provide much insight into how to design a system securely. As such it serves as a useful reference tool, but not as an engineering methodology.

Other methods [89, 93, 103] tend to follow in the same vein as this, and rely on imparting as much information as possible about security, without providing a comprehensive framework in which to use this information.

2.5.2.2 Checklists

Checklists are fixed, sometimes numbered, lists of steps that you are told to take in order to secure a system. A number of different checklists abound, varying in their scope from basic security for home networks, to operating system or software specific instructions such as the Unix Security Checklist [1], the Windows 2000 Server Baseline Security Checklist [95] or SQL Server Security Checklist [57].

The advantage of checklists is that they provide easily used and applied information about security and also provide a means of measuring or auditing the security in a system.

A main disadvantage with checklists lies in their rigidity, making them ill-suited for more specialised security tasks. Although checklists can be useful in providing a baseline for computer security, they are not intended to be complete and the temptation exists to assume that by complying with a checklist no further security actions are necessary. This can be interpreted as a psychological problem caused in part by a failure to take responsibility for achieving good security (responsibility and other related factors are discussed in much greater detail in sections 4.5, 5.5 and chapter 7).

2.5.2.3 Guidelines

Guidelines are words of wisdom, intended to impart security developers with the principles they should follow in order to make computers secure. Sometimes

mislabelled as checklists, they impart advice as to desirable properties of a system, but do not describe means of achieving this in practical terms.

The biggest problem with guidelines is that they do not provide practical assistance in building security – rather they state what the intent of the security should be. A particular example relating to the need for usable security [73] states frequently that the system should be *"usable"* and *"understandable"*, yet fails to explain means of achieving these laudable goals.

2.5.2.4 Risk Management

Risk is defined as the probability that a threat will act on a vulnerability to cause an impact. In security terms, a threat is the potential source of an attack and a vulnerability is an area of the system that is susceptible to exploitation. Risk management [96, 104, 116] is the method many experts consider to be the most flexible and comprehensive for securing a system.

Risk management is the process of conducting a risk analysis and following it with a risk mitigation exercise (a more detailed description of a risk analysis can be found in section 3.6.3). Through risk analysis, risks are identified by determining the vulnerabilities of the assets, the threats that can exploit these vulnerabilities, the likelihood of these threats occurring, and the impact that a successful attack can have. Through risk mitigation, the onus is put on developing countermeasures to reduce risks which are deemed to be too high in relation to the impact of an attack.

It should be noted, however, that some experts [89] believe that a risk management approach to securing systems is not desirable. They argue that this is a costly and time-consuming exercise and that making the whole security of a system dependent on risk measurements does not necessarily lead to an acceptable standard of security, which in turn can lead to accusations of negligence. Risk analysis is highly dependent on the accuracy of the measurement of risk, which can be higher than reality, leading to unnecessary expenditure on security countermeasures, or lower, leading to an unacceptably high exposure to danger. But even a very accurate risk judgement can be problematic. This is because successful countermeasures prevent security incidents from happening, and it can be difficult to justify the expense when nothing bad ever happens. A good summary of the problems with risk analysis can be found in [37].

Despite this, BS7799 as well as other internal control standards regard risk management as essential for successful security management. In addition to this, the conduct of risk management is also a statutory requirement for US federal information systems.

2.5.3 Knowledge of the System

In HCI design techniques, such as Contextual Design [23] (see section 2.3.3), knowledge of a system is gained by gathering and understanding information about the users of the system, as well as the technical requirements of the system. In security design, knowledge of the system tends to be gained through the *evaluation* of the security of the system.

Original techniques, such as the Trusted Computer Systems Evaluation Criteria [43] (also referred to as orange book) later subsumed in the Common Criteria [86], laid the groundwork for this type of approach. Security evaluations tend to be used to analyse the security of a system and measure this against a definition of good security in order to identify areas in need of improvement. A more complete review of security evaluation can be found in Dhillon & Backhouse [48] (pp 136-137).

Many risk management tools, such as COBRA[6] or CRAMM[7] (both based on BS7799/ISO17799 [28]) or RISKPAC[8], perform an evaluation of the security of the system in order to inform the risk analysis. To achieve the evaluation, information gathering takes the form of questionnaires which users (frequently managers) have to fill out. Once the information has been gathered, the resulting security analysis is conducted by experts in isolation from users. This leads to a situation where security is developed without any significant involvement from the users of the system (who know the most about the system) as questionnaires are designed to only gather data in predetermined areas. Therefore questionnaires gather information from users in areas where the security analyst expects to gather information, which effectively removes all possibility of serendipity, and is typical of the *functionalist paradigm* which this type of approach adopts (see section 2.6.1).

---

[6] http://www.securitypolicy.co.uk/

[7] http://www.cramm.com/

[8] http://www.csciweb.com/riskpac.htm

2.6 Information Systems Security

2.6.1 Functionalist Security

According to the framework developed by Burrell & Morgan [32] and used by Dhillon & Backhouse [48] to classify information security research, the computer security perspectives presented so far belong to the *functionalist* paradigm. This paradigm *"...approaches the subject from an objectivist point of view, concerned with the 'regulation' and control of all organizational affairs."* This rationalist approach is particularly widespread in security research and practical applications (as can be seen in the endorsement of checklists and risk management approaches by regulatory bodies or standards, such as the US Government or BS7799/ISO17799 [28]).

2.6.2 Interpretive Security

There exists a smaller body of literature that fits into another research paradigm: *interpretivism*. This type of approach is characterised by the notion that the social world is open to interpretation and results from the shared understanding of the individuals in the society. As a consequence, these approaches to security are based on understanding the social aspects of the environment in which the technical system operates.

A small number of approaches, dubbed *"integrative approaches"* by Siponen [107] exist in the interpretive paradigm which attempt to consider organisational needs.

These include responsibility modelling (Structures of Responsibility [16], and abuse cases [80]), a managerial approach to systems risk [112] or security modifications to existing information systems development methods [65, 69].

A review of the existing integrative approaches by Siponen [106] suggests that existing methods for developing secure systems are fragmented and suffer from a number of problems:

1. Comprehensive modelling support is not available in any of these methods. The different approaches cover the different levels of information security, namely organisational, conceptual and technical, but no single method provides support for all.
2. Most approaches suffer from *development duality*, which refers to the conflict between the functionality of a system and its security. Development duality arises as a consequence of developing security and functionality separately.
3. Existing approaches tend to restrict the flexibility and autonomy of developers by prescribing a specific process and "toolkit" of methods. This means that if

developers want to address security in their system, they have to abandon their preferred design tools and techniques.

2.6.3 Epistemology

Summarising and following from the discussion in [99], there are three distinct epistemological viewpoints into the field of design: scientific, craft and engineering.

1. The scientific approach aspires to building theories and models through the creation and testing of hypotheses. From the point of view of design, scientific models are predictive and are used to inform the design process. The difficulty with using this type of approach is that this knowledge is not prescriptive and therefore difficult to use in practice.
2. The craft approach is characterised by informal heuristics, developed from experience in implementation and evaluation of systems. The emphasis in this case is based on practical knowledge. From a design point of view, whilst craft heuristics are easy to apply, a major problem with this approach is that the effectiveness of solutions is not guaranteed and this knowledge is not generalisable. Because this knowledge is highly specific, the cost of gaining more knowledge is particularly high.
3. An engineering approach aspires to categorising knowledge of the field into engineering principles that can be used to specify and implement a system. This approach is intended to reconcile scientific and craft knowledge into prescriptive and quantifiable design solutions to specified problems. The ultimate aim of the engineering approach is to introduce reliability and practicality into the design process.

As can be seen from the review so far, security design has elements of all three viewpoints. For example, the scientific viewpoint can be seen in formal models of security, and the engineering approach can also be seen in security patterns, or risk management – both of which aim to provide reusable means of designing good security. Despite this, computer security design is still very much a craft. This can be seen in the widespread use of anecdotes as a means of recording and propagating security knowledge, the broad variety of different approaches to analysing security, or the fact that expertise in security is essential in order to be able to successfully analyse a system. As a consequence considerable time and energy in spent in order to gain expertise in security, making this knowledge inaccessible to most stakeholders – particularly

developers who have to maintain a large body of additional knowledge. As a result, security design is either carried out by developers who do not have expert security knowledge, or by security experts who lack the developers' knowledge of the system.

As described in the previous section, a small number of approaches have been proposed to integrate the functional and security development aspects of a system. Helen James' proposal [69] is an example of one of these approaches, and actively seeks to involve users, developers and other stakeholders in the design phase of security in addition to security experts. The most important aspect of this is that, because of the involvement of all these different stakeholders, the relevant knowledge about the system (gained through users, and developers) and the expert knowledge of security (gathered from security experts) is present and accessible. It is further argued that the involvement of users in the design of security also has the added benefit of empowering them and improving their understanding of security matters.

Whilst this is a promising approach, a number of issues have to be addressed in further detail, namely the need to integrate the functional and security development issues more closely. Another matter of interest would be to gain a better understanding of the process of designing security, detailing the specific activities which have to take place for good security design. James approaches security design from an organisational perspective, which allows the inclusion of social factors, for example identifying the different tensions that may exist between managers and other employees, or the representation of different stakeholder viewpoints at crucial stages in design. This is a useful approach; however it is not the most suitable means of addressing technical security issues. As a result of this organisational focus, software and system developers who need to design and build new technical systems can face problems reconciling technical functionality and organisational security.

The following section looks at the field of HCISec, which addresses the problem of developing secure technical systems that are also suitable for their human users.

2.7 Human-Computer Interaction in Security

2.7.1 Usability and Security

Kahn [74], cited by Anderson [13], *"... attributes the Russian disasters of World War 1 to the fact that their soldiers found the more sophisticated army cipher systems too hard*

to use, and reverted to using simple systems which the Germans could solve without great difficulty".

This statement expounds the notion that mechanisms for strong security are hard to use. Bruce Schneier [103] makes the point that *"... security is only as good as its weakest link, and people are the weakest link in the chain"*. Other authors [10, 73, 92, 118] also argue that secure systems are broken through human issues, e.g. because an administrator makes a mistake in configuring a system. This indicates that ease of use is necessary in order to get people to behave securely, and therefore good security should be easy to use if it is to be applied.

The notion of *psychological acceptability* [98] was first proposed in 1975 as a desirable property for computer protection mechanisms. Although this can be seen as a call for increasing the usability of security mechanisms, and originally was intended as a call for good interface design, it is important to make distinctions between *who* benefits from psychological acceptability. Zurko & Simon [123] identify three major groups of people whose usability needs must be addressed:

1. Users
2. Administrators
3. Developers

Other research [101] also suggests that actually improving the usability of secure systems for users results in more effective security mechanisms, which benefits a fourth group:

4. System owners

2.7.2 Usability of Security and Dependability

"A computer is secure if you can depend on it and its software to behave as you expect." [58]. This definition is controversial in that it implies that security exists in the reader's expectations of computer and software behaviour (which is obviously incorrect in the many cases of people not knowing much about security or computers). It is useful, however, in underlining the importance of dependability in computer security. It has been argued that an emerging sentiment in security research is *"'correctness' is not the issue; 'dependability' is"* [17]. The point is that knowing that the system will behave and be used in the expected manner (dependability) is as much of a problem as knowing that a system will counter a threat if used correctly (correctness).

Although many authors argue that usability is beneficial for secure systems [10, 73, 117, 123], it is necessary to examine *who* benefits from usability in security and *how*.

In the following, a system involves both human and technical components and an interaction refers to behaviour from the social, technical or a combination of social and technical parts of that system

The ***owner*** of the system benefits from security by **dependably**:

1. allowing *desirable* interactions
2. avoiding, deterring, preventing, detecting and reacting to *undesirable* interactions.

The ***user*** of a security mechanism in that system benefits from good usability because it:

1. Facilitates the correct execution of the user's *desired* interaction
2. Impedes the incorrect execution of the user's *desired* interaction

In effect, increased usability results in fewer errors and a smaller mental or physical cost to the ***user***.

As a result, the ***owner*** of the system benefits from usability because it increases the dependability of the security – *but only if the user's desired interactions match the owner's desired interactions*. It should also be noted that a user refraining from engaging in a *desired interaction* can be considered an *undesirable interaction*.

This now highlights three different security issues (see Table 1):

1. A user intentionally desires an interaction the owner does not with malicious intent (e.g. criminal intent)
2. A user intentionally desires an interaction the owner does not without malice
    i. The user does not perceive the interaction as being detrimental to the owner. Either the user comes to an inaccurate conclusion through incomplete information (about security, the system, the risks, etc.), or comes to a different conclusion based on the same information (differences of judgement in security)
    ii. The user has greater incentive than disincentive to engage in the interaction (e.g. a user may decide to break the security policy because the user's bonus is only tied to achieving production targets – not achieving security that interferes with these targets. Another example may be that disciplinary measures for security

breaches are not enforced, therefore the disincentive for breaking the policy is very small)

iii. The user cannot behave in the manner desired by the owner

3. A user unintentionally desires an interaction the owner does not (e.g. misunderstanding, errors or confusion)

Points (2) and (3) are generally those that are exploited through *social engineering* [81], either by manipulating users through their desire to be helpful (2.i) or by deceiving them into unintentionally breaking the rules (3). Social engineering is the term used to refer to attacks that target authorised users of the system – as opposed to technical components – and attempt to manipulate them into revealing information (such as passwords) or otherwise compromising the system.

By improving the usability of a specific security technology, the use of that technology is more efficient (e.g. faster) or effective (e.g. fewer errors). This addresses (3) by reducing the number of errors (unintentional and undesirable interactions), and also to some extent (2.iii) by matching the demands of the specific technology to the user's capabilities, thereby reducing the specification of behaviour that the user cannot engage in.

Interaction	Intentional		Unintentional
Desirable	Good security		Chance
Undesirable	1. Malicious	2. Non-malicious	3. Error
	Crime	i. User perceives interaction as harmless ii. Greater user incentive for undesirable interaction iii. User incapable of desirable interaction	

Table 1: Analysis of user intention vs. desirability of an interaction

A review of research on the usability of specific security mechanisms is presented in section 2.7.3.

2.7.3 Usability in Security Technologies

2.7.3.1 Passwords

Studies into why people compromise passwords [10, 29, 117] have shown that the technologies are not well suited to the expectations and needs of users, and neither are the social and cultural contexts conducive to fostering secure behaviour. The authors

identify that in order to achieve more effective security interactions, systems should be designed to suit user needs (See section 2.7.4 about HCI and security design).

2.7.3.2 Encryption

A usability evaluation of PGP encryption software [118] also revealed that users experience considerable difficulties in achieving what are considered to be simple objectives. The conclusions of this research were that better user interfaces are needed, but there is no recognition that the problems encountered go beyond interface design and may also be a consequence of the complexity of asymmetric encryption itself.

2.7.3.3 Email

There have also been attempts to improve the usability of secure email. The proposals have focussed on automating email security, for example with the use of a security proxy for encrypting email [31]. These proposals have failed to gain public or widespread acceptance. This gives more support to the notion that the usability problem of secure email is not due to the software interface or the effort that users have to expend, but instead is due to the difficulty that users face in both understanding asymmetric encryption and understanding the need for that security technology.

2.7.3.4 Visibility of Security

Dourish and Redmiles [50] propose to improve the visibility of the state of security in the system, and their hypothesis is that this will allow users to make informed choices which, in turn, yield a more effective and more secure system use. They argue this is necessary to address the problems of *disembodiment* from the context of use and *dissociation* from one's actions identified by Bellotti and Sellen [22] as the primary source of a number of potential privacy and security problems.

Although useful in identifying problems with current design solutions, the research into improving usability of security technologies is largely focussed on the user. This ignores the developer group identified previously and the problems that they face when developing secure systems. Security development is already complex, time-consuming and error-prone, yet most of the conclusions from this research are that security development should also make systems more usable – adding another burden onto the load of developers.

More importantly, this research approach does not resolve the intentional but non-malicious undesirable interactions which users may engage in as seen in (2) of Table 1

(see section 2.7.2). These occur as a result of a conflict between the way in which users are expected to behave in the system design and the way in which they actually behave. In order to resolve this it is necessary to adopt a systemic approach that reconciles both social and technical aspects of the system, with the goal of assuring that the system owner's desired interactions match with the actual interactions undertaken by both users and technology.

In section 2.7.4, a review of the state of HCI research at the design level is presented.

2.7.4 HCI and security design

Zurko & Simon [123] identified that software developers require as much security usability as users and security administrators. According to the DTI Information Security Breaches Surveys [45], 32% of UK businesses surveyed in 1998 suffered a security incident, rising to 44% in 2000, 74% in 2002, reaching a massive 94% in 2004, which reinforces other figures about the growing number of security vulnerabilities and attacks [1]. This indicates that the activity of *designing security* is itself in need of an overhaul in order to *get the right design.*

Current HCI security design techniques fall into two categories, design guidelines and usability evaluations of secure systems.

2.7.4.1 Design Guidelines

A set of user interaction design guidelines for secure systems was proposed [73]. Some examples of these guidelines are:

"Path of Least Resistance. The most natural way to do any task should also be the most secure way.

Appropriate Boundaries. The interface should expose, and the system should enforce, distinctions between objects and between actions along boundaries that matter to the user.

Revocability. The interface should allow the user to easily revoke authorities that the user has granted, wherever revocation is possible."

Whilst these guidelines are interesting and useful in reminding developers of the need for taking user needs into account, they are not prescriptive and do not provide assistance into how to achieve the aim of the guideline. For example the *revocability* guideline specifies that users should be able to easily revoke authorities – which is a

very important part of the security of the system – but how to achieve this is left as a problem the developer should resolve.
In terms of helping in the design of usable and secure systems, these guidelines are only useful in setting out desirable goals.

2.7.4.2 Usability Evaluation of Secure Systems

Currently the only effective means of ensuring that a secure system is usable is to periodically conduct evaluations and test user responses. As can be seen from the PGP usability evaluation [118], this is useful as a means of uncovering problems. One major problem with this is that as a design practice it has no prescriptive value, and therefore does not inform developers about how to achieve a usable and secure design.
An additional problem is that designing, conducting and interpreting an evaluation currently requires specialist knowledge. Whilst in the field of HCI this is common practice, this knowledge is not widespread in the security community and this poses an additional difficulty.

2.7.4.3 Shortcomings in HCI security design

The need for change in the design of security has been called for by many authors. Blakley [24] advocates moving away from the military information fortress model, arguing that the foundations for such a model are not applicable in today's computing environment. Whilst this is undoubtedly true, the problem facing abandoning the military approach is twofold. Firstly, most existing security technologies have been developed by the military or are based on the military model [10, 24]. This means that most security tools are biased towards a military and *functionalist* mode of operation, regardless of the environment in which they actually operate (e.g. authentication mechanisms operate along the lines of "us against them", whereas in a commercial setting individual people might be much more self-centred and adopt a more self-serving attitude). Secondly, the military model of social interactions is much simpler in security terms because of the explicit presence of a chain of command, and the existing dedication to ensuring discipline. Despite evidence and arguments that the military model is unsuitable for general use, it is much simpler to continue to assume that orders (or prescriptive security policies) will be followed, as opposed to having to design the mechanisms whereby policies are communicated, understood, promoted or enforced. This added complexity is the biggest difficulty that has to be overcome before alternative paradigms can be successful.

Anderson [13] suggests that elements from safety-critical system design should be adapted to security and [30] have proposed a security design method based on the GEMS safety-critical design method [94]. Smetters & Grinter [109] propose a shift from the design of usable security technologies to the design of useful secure applications (from the user perspective).

The *interpretive* approaches [16, 65, 69, 112] described previously, whilst they provide a means of addressing user issues in security, also fall short. They either do not integrate seamlessly into the development cycle (forcing developers into tools, models and techniques they may not know or wish to adopt) or fail to address issues of development duality. In addition to this, *"... explanations come enshrouded in complexity, largely because of the sophisticated sociological and philosophical bases, and as a result the audience for such security approaches remains just a small group of academic researchers."* [48]

This problem becomes even more apparent when considering the problem of designing security using the framework described in section 2.6.3. As is evident from the importance of expert knowledge, anecdotes and testing, security design can generally be described as a *craft* discipline. One of the characteristics of this is that gaining security knowledge is hard and time consuming, and most system developers do not have the luxury of spending time developing this knowledge. The problem arises from the need for security to be designed from a socio-technical point of view. This approach requires extensive knowledge of the system, its context and stakeholders. Whilst security experts possess the relevant security knowledge, they lack this vital information. As a consequence, considerable time and energy has to be expended acquiring information that current developers already possess. It is therefore a much more practical and effective solution to provide developers with a simple process for effectively addressing security, without requiring them to learn specialist security knowledge.

2.8 Summary and Conclusions

Software engineering approaches to security are generally focussed on providing technical security, but there is growing evidence [98] that these approaches do not consider the needs of people sufficiently [118, 123]. Despite having identified this problem, the HCISec field is focussing nearly exclusively on improving the user interface to security tools. The field of HCI, however, has long advocated the need to take a systemic approach to design, as opposed to simply sticking a pretty interface on

top of a product. The user interface is a very important part of making security more usable to the user, but it does not address the issues related to among others:

- Complex design and concepts: for example asymmetric encryption, which employs terms such as *'private'*, *'public'* and *'keys'* that that are a mismatch to the everyday meaning most people assign to them.
- Unreasonable assumptions about users' and administrators' capabilities and motivation: i.e. changing an infrequently used password regularly and expecting it to be remembered yet not written down.
- Conflicting demands: such as having to complete production tasks as well as installing patches, updating virus signatures and rebooting the computer.

Although developers have been identified as being a group that requires usable security [123], no efforts seem to have been made to ensure security development methods are well suited to the needs of developers. In fact, by putting the onus on the developers to build better user interfaces, the current trend in HCISec research is arguably adding to the complexity of building secure systems. And whilst *interpretive* security approaches (see section 2.6.2) have started to address the needs for a socio-technical approach to secure systems, they are still falling short in providing a practical and relatively simple means for developers to achieve this. In addition, given the *craft* nature of security design (see section 2.6.3), developers also face problems in acquiring the necessary knowledge and expertise to build secure systems.

It is therefore necessary for a new approach to be instigated that provides a clear and simple method for developers to build a secure system that accommodates human factors and incorporates easily into the overall functional design process.

3 Methodology

3.1 Introduction

Real-world empirical research in security design is difficult, particularly from a logistical point of view. One of the key difficulties of conducting empirical security design research lies in gaining access to resources [33]. Few organisations, projects or information managers are willing to open their systems to scrutiny when it comes to security. Even fewer are willing to field test different approaches or what can be termed experimental security, simply because the field is so sensitive and unproven methods seldom inspire confidence. Whereas in many research experiments it is common to compensate the participants for their inconvenience, pecuniary recompense is not a suitable means of encouraging organisations to participate in the research. Interventionist research approaches, such as Action Research, have benefited the participating organisations by actively seeking to intervene and improve on specific problems within the organisation. This promise of immediate assistance seems to be much more persuasive in gaining organisational interest than *"... altruistic arguments about how (the) research might benefit software engineers generally"* [33].

An additional difficulty in engaging in empirical security research is that, because of the highly disparate nature of different organisations and their approaches to security design, identifying benchmark data for statistical comparison of results from different organisations is impractical. Combined with the difficulty of gathering sufficient participants, a positivist and quantitative research methodology is particularly difficult for empirical research into security design.

In order to address this fundamental difficulty in secure information systems research, it has been proposed to adopt social sciences research methodologies, such as Action Research [19, 20] or Grounded Theory [9, 10]. A few proposals have already successfully employed both Action Research [69, 105, 112] and Grounded Theory [9] as a means of empirically researching security. The Action Research studies consisted of applying different design approaches to real world organisations, and the Grounded Theory research was concerned with identifying and modelling users' perceptions of privacy in multimedia communications.

From this, it can be seen that both research methodologies are appropriate for the research in this thesis:

1. Action Research as a means of researching the practical application of the AEGIS socio-technical design approach.
2. Grounded Theory as a means of analysing and exploring the broader issues surrounding the application of a socio-technical design process.

In this chapter, section 3.2 consists of a review of Action Research and section 3.3 presents a review of Grounded Theory. The details of the research methodology used in this thesis are described in section 3.4, and the validity of this research approach is argued in section 3.5. Finally, the secure socio-technical design method AEGIS is introduced in section 3.6 which describes the underpinning concepts behind the interventions actually applied during the empirical studies. Since AEGIS was extensively revised throughout the empirical studies, section 3.6 presents the basic principles of AEGIS whilst in Chapter 7, sections 7.2-7.4 describe the specifics of the revised AEGIS process.

3.2 Action Research

Action Research [19, 71, 77, 84] describes a research strategy in which the researcher actively intervenes with the research material, as opposed to being a simple observer. Originally coined by Kurt Lewin in the 1940's, Action Research is a structured research approach that *"... identifies a question to investigate, develops an action plan, implements the plan, collects data, and reflects the findings of the investigation."* [71] Whilst originating in the social and medical sciences, towards the end of the 1990s, Action Research became increasingly popular for scholarly investigations of information systems [19]. One of the reasons for this is because the Action Research results provide very relevant information grounded in practical action, whilst simultaneously informing theory. This makes Action Research a very useful methodology for researching the practical application of a secure design approach, and examples of this can be found in [65, 69, 105, 112].

One of the key characteristics of Action Research is its association with the *interpretive* viewpoint of research. Grounded in the philosophy of phenomenology [27], *interpretivism* is concerned with the subjective understanding that individuals attribute to their social settings [48]. This is in opposition to the *positivist* approach which is dedicated to an objective, measurable and rational viewpoint.

The importance of this is that interpretive researchers argue that meaningful research cannot be conducted by reducing or avoiding the importance of complex social systems. Action Research therefore provides a means of studying these complex systems by introducing changes in the social system and studying the effects of these changes.
According to Baskerville [19], this has two consequences: first is that Action Research adopts an idiographic viewpoint, second is the need for qualitative data and analysis. Since each social setting is unique and involves different human subjects, it is necessary for the research to assume an idiographic viewpoint and study the role of individuals in the overall setting. Action Research accomplishes this by incorporating human subjects as collaborators, directly involving them in the change experience.
Quantitative data and analysis techniques are not well-suited to the problem of gathering and interpreting subjective responses which form the basis of interpretive and idiographic research. Instead qualitative analysis techniques such as hermeneutics, theoretical sampling or deconstruction are frequently associated with Action Research.

As taken from Baskerville's Action Research tutorial [19], the Action Research approach can be seen as a five phase process:

1. Diagnosing
2. Action planning
3. Action taking
4. Evaluating
5. Specifying learning

Throughout the Action Research process, researchers and practitioners collaborate to achieve the desired outcomes. In the diagnosing phase, the main problem and underlying causes are identified. Using the theoretical framework to determine both the desirable outcomes and means of achieving them, the next phase is to plan the specific actions that will be undertaken. During the action-taking phase, the plan is then put into motion. During the evaluation phase, once the actions are complete, outcomes are evaluated to determine whether the expected effects of the actions were realised and whether these had an effect on the problems identified in the diagnosis phase. The final phase reflects on the knowledge gained from the research. This includes:

1. Knowledge gained by the participating organisation itself and any changes that may come out of that knowledge.

2. Greater understanding about the general research problem from the failures and successes of the actions undertaken.
3. Increased knowledge about the theoretical framework as a result of its application.

3.2.1 Validity and Quality of Action Research

"An account is valid or true if it represents accurately those features of the phenomena, that it is intended to describe, explain or theorise." [62]

The validity and means of evaluating action and interpretive research have frequently come into question [19, 20, 82]. For example one physics teacher who used Action Research stated: *"... coming from a physics point of view, I keep asking 'What is the data? How do we really know if we're doing anything better or not?' In physics we see research as more controlled experiments, variables and data, and so forth, which is not what we're doing here."* [52]

It can be said, however, that applying validity criteria from other research paradigms is flawed in that these are not suitable for the aims of the research [49]. In this case, interpretive Action Research is concerned with the notion of the social construction of reality, as opposed to the positivist notion that reality is an objective and measurable state.

Nevertheless, because the researcher is actively involved in the researched subject matter, the question of validity in Action Research is an important one and has been addressed by many different researchers [19, 21, 69, 77, 105, 112]. This has resulted in a number of validity criteria being proposed [19, 21, 75]. The most important criteria are presented here:

1. A theoretical framework must be present as a premise of Action Research [19].
2. Data collection methods should be carefully selected [19, 21], and capable of capturing both intended and unintended effects [52].
3. The researcher should actively intervene in the research setting [21].
4. The immediate problem in the social setting must have been resolved during the research [21, 102].
5. The Action Research approach should be cyclical. The use of multiple cycles allows the early conclusions of the researcher to be scrutinised and refined in the later stages [19].[49]
6. Generalization about the results should be tempered with an interpretation of the extent of similar settings to which the theory can be expected to apply [19].

3.3 Grounded Theory

According to [78], Grounded Theory is *"... an inductive, theory discovery methodology that allows the researcher to develop a theoretical account of the general features of a topic while simultaneously grounding the account in empirical observations or data."* An excellent review of Grounded Theory can be found in [9]. The author uses Grounded Theory as a tool for exploring the field of users' perceptions of privacy in multimedia communications. Since Grounded Theory is a theory-building qualitative analysis tool, it is argued to be particularly suitable for areas in which little is already known. This makes the use of this research methodology ideal for investigating the factors and issues that affect the practical application of a socio-technical secure system design approach.

The origins of Grounded Theory come out of the social sciences as a means of focusing on *theory generation* as opposed to *theory testing* [59]. A definition of Grounded Theory is a *"... theory that was derived from data, systematically gathered and analysed through the research process"* [113]. In the process of building the theory, the researcher does not begin a project with a preconceived theory in mind, but begins with an area of study and allows the theory to emerge from the data.

The main analytical process of Grounded Theory consists of taking data, breaking it down, conceptualising it and reassembling it into new forms. The process for achieving this can be broken down into three stages: open, axial and selective coding. Tool support for these analytical steps exists in the form of ATLAS.ti, a hermeneutics analysis package.

3.3.1 Open coding

Open coding is the analysis process through which *concepts* are identified in the data. Concepts are the building blocks of the Grounded Theory analysis and are generated through the detailed labelling of the data. The granularity of these concepts can vary from individual word labelling, to whole paragraphs or documents being labelled.

The concepts are then compared to see if they pertain to a similar phenomenon. Those that do are then organised into *categories*, a more abstract unit of analysis. The final stage of open coding consists of identifying different *properties* and *dimensions* of the different categories. *"Whereas properties are the general or specific characteristics or*

attributes of a category, dimensions represent the location of a property along a continuum or range." [113]. That is to say that whereas properties consist of attributes of the category, dimensions represent the measured extent or importance of a property.

3.3.2 Axial Coding

The axial coding stage refers to the process of relating categories to their subcategories, and identifying the high-level phenomena present in the data.

This stage is important in identifying and relating issues of *structure* and *process*. Structure is defined as being the conditional context in which a category (phenomenon) is situated.

Process is defined as the sequences of action/interaction pertaining to a phenomenon as they evolve over time.

Structure therefore refers to the circumstances that explain *why* a particular event is occurring, and process relates to *how* these circumstances are responded to.

In order to achieve an analysis that relates structure and process, axial coding first identifies *conditions* that pertain to the phenomena. These can be:

1. Causal conditions: events that directly influence a phenomenon.
2. Intervening conditions: events that mitigate or alter the impact of causal conditions. This frequently arises out of contingencies (unexpected events).
3. Contextual conditions: These result from the crosscut of causal and intervening conditions. Therefore contextual conditions are the result of causal conditions mitigated by intervening conditions.

The purpose of this is to assist in identifying the complex interweaving of events that lead up to a phenomenon.

The next step is to identify *action/interaction strategies* that individuals engage in as a result of these conditions. These describe the way individuals handle, manage and respond to conditions leading up to a phenomenon, and therefore are useful in linking process with structure.

Finally *consequences* are identified that represent the outcome of *action/interaction strategies* in response to *conditions*.

During the analysis, a category is *saturated* when no new information (properties, dimensions, conditions, actions/interactions or consequences) seems to emerge during coding.

3.3.3 Selective Coding

In this final stage, the theory is integrated and refined. One of the key parts of this process is the identification of the *core category*. This refers to the central phenomenon around which all the other categories are integrated. *"A central category has analytic power. What gives it that power is its ability to pull the other categories together to form an explanatory whole. Also, a central category should be able to account for considerable variation within categories."* [113]. The most important requirements for the core category are that it must be central (all other major categories can be related to it) and it must appear frequently in the data.

Once the central category has been identified the final step is a description of the *storyline* of the theory, comprised of the key elements and their interrelationships.

3.3.4 Validity and Quality of Grounded Theory

As with Action Research, it is necessary to address the issues of quality control and validity of Grounded Theory. The main success criterion that can be applied to a Grounded Theory analysis is the degree to which it *fits* with the data. However, issues of researcher subjectivity and bias are important in determining this and it has been argued that instead of ignoring bias, efforts should be made to acknowledge and address these issues [91, 119].

Some proposals for validating Grounded Theory results have included *testimonial validation* [111] as a means of ensuring that the analysis seems cogent to the respondents, and *negative case analysis* [47], where cases are actively sought out that challenge the emerging theory. Unfortunately these either require access to the respondents after the analysis has taken place or assume that analysis and data collection is happening concurrently.

Reflexivity, the notion that researchers should acknowledge and document how their views change during the course of the analysis is argued to be useful in determining evaluation criteria [111, 119]. This demonstrates a level of *permeability* [111] to the data and a willingness to adapt and change the theory in response to new data.

Strauss and Corbin [113] also underline the importance of maintaining objectivity. They argue that *thinking comparatively*, comparing incident to incident in the data, enables the researcher to better stay grounded in the data and maintain a level of distance from the data. They also mention the importance of gaining multiple viewpoints in the data, possibly through *triangulation* of data gathering techniques or approaches. *Theoretical*

sampling refers to one means of gaining alternative viewpoints by sampling according to emerging concepts during the analysis.

Finally, presenting information about the *context* in which the studies are completed also provides additional information about the transferability of the study to other contexts. Contextual information should for example consist of details about the participants, circumstances of the projects involved in the research, the duration of the studies, etc.

3.4 Research Approach

3.4.1 Research Methodology

As discussed in Chapter 1, the aims of the research presented in this thesis are to explore the areas of secure system design, propose a socio-technical secure development process, and evaluate this process through practical empirical application. This approach requires a research methodology that is well-suited to the problem field and allows the flexibility necessary for exploration in this area.

While a positivist, quantitative, hypothesis-testing approach to research would stay more in keeping with traditional research methods, the nature of the research and the realities of the field make this an infeasible proposition for the following reasons:

1. One of the aims of this research is to identify factors that affect the design of security. This can only be achieved through an exploratory and explanatory framework, not a theory testing approach.
2. The data collection required for this type of research is costly and particularly difficult when considering the rarity of willing subjects and the timescales involved.
3. This type of research requires that observations be made by an independent observer. In the case of empirical research into security, because security is such a sensitive area for most research subjects this is particularly difficult. As a consequence an observer is also a participant in the events that are observed – compromising the validity of this type of research approach.

The needs of this research therefore dictate that an alternative research methodology be selected.

As seen above, the strengths of Action Research and its previous use as a research methodology in secure information systems design research favour it as a means of practically applying a socio-technical secure system design approach.
Grounded Theory is particularly suited to the theory building aspect of the research in this thesis, namely exploring the real-world factors that affect and surround the development process of a socio-technical secure system design approach.

As a result, the strategy in this research has been to adopt Action Research as the methodology in which AEGIS is applied and used as the theoretical framework informing the interventions. The success, strengths and weaknesses of AEGIS are identified through this practical application. In addition to this, the data gathered during the Action Research case studies has been analysed using Grounded Theory in order to identify, interrelate and model the different issues and factors that affect secure socio-technical design.
Since the two research areas (evaluating AEGIS and identifying real-world factors) overlap, it is argued that a further benefit is gained in combining these two strategies as they complement, further validate and reinforce the results.

Research Theme	Research Strategy
Real-world empirical evaluation of AEGIS	Action Research
Identification of real-world factors in socio-technical secure design process	Grounded Theory

Table 2: Research approaches for addressing the research goals

3.4.2 Case Studies

The empirical research in this thesis consisted of four case studies in the relatively new area of Grid computing. Grid computing research has recently been the recipient of large amounts of funding, with figures for UK funding in excess of £100 million, and European Commission (EC) funding topping €50 million in 2003 (with a further €52 million being announced in December 2004) [90]. This funding has gone into initiatives such as the EC funded Gridstart and the UK e-Science programme, dedicated to investigating means of applying grid technologies to traditional sciences.
"e-Science will refer to the large scale science that will increasingly be carried out through distributed global collaborations enabled by the Internet. Typically, a feature of such collaborative scientific enterprises is that they will require access to very large

data collections, very large scale computing resources and high performance visualisation back to the individual user scientists." [7]
It is thanks to the existence and support of the e-Science Security Task Force that access to Grid projects was obtained and the research in this thesis could be carried out.

The next sections define what Grid applications are, examine issues and motivations for security research in Grid projects, and finally identify the limitations of this research.

3.4.2.1 Defining Grids

Grid applications are somewhat problematic in that there have been many attempts to define what they are and what they are intended to do. Despite this, there does not seem to be a consensus in the industry as to precisely what a Grid application is.
"Grid computing is a form of distributed computing that involves coordinating and sharing computing, application, data, storage, or network resources across dynamic and geographically dispersed organizations." [121]
This definition describes Grid computing as a technical architecture, and also mentions a social aspect to Grid applications, namely that it is meant to accommodate organisations that can be fast changing and widely distributed in the physical world.
A somewhat different definition defines Grid applications as satisfying three requirements [55]. A Grid application:

1. coordinates resources that are not subject to centralised control,
2. uses standard, open, general purpose protocols and interfaces,
3. delivers non-trivial qualities of service.

From both of these definitions, Grid applications can be seen as the next step in networked computing. That is to say that the purpose of Grid applications is to provide a platform on which a business (or group of different businesses) can operate with a high quality of service unfettered by geographical constraints. What makes Grid applications distinct from other networked products is their dedication to the notion of a *virtual organisation* – an organisation that does not have to exist in the bricks and mortar sense. A virtual organisation is a new concept in that it not only operates from different physical locations; it is made up from a variety of different self governing entities such as businesses, individuals or academic departments for example. This is a radical departure from previous networked business applications which have tended to operate on the implicit assumption that while the users of the system may be varied, the owners of that system at least belong to the same organisation.

3.4.2.2 Grid Projects and Security Research

Conducting security research with Grid projects has a number of advantages. Because of the recent drive for funding Grid research, a large number of projects have been undertaken simultaneously. Whilst these projects all have different fields of application, their commonality in using Grid and Grid-like technologies provide them with a large degree of similarity which makes these very useful research candidates. Comparisons drawn between different projects are more meaningful as a result. An additional operational benefit of the considerable funding into Grid research is a relative abundance of Grid projects which provides an unusually rich pool of research subjects.

One of the characteristics of these Grid projects is that they are generally intended to lay the foundations for the commercial application of Grid technology. As a consequence, the need for accommodating commercial needs and concerns, namely in the area of security, is also important for the success of the project. However, despite the involvement of commercial companies in some Grid projects, these projects are generally academically driven. This has resulted, in some cases, in situations where academic interests in the research aspects of the projects have overshadowed the commercial requirement of addressing the issues of security by focussing on achieving the functionality of the Grid project with little thought to securing it. This is problematic as can be seen in the widespread security problems of technologies that have evolved from academic proofs of concept that did not address matters of security – such as the Internet, newsgroups or email.

Other than having to address issues of scale, and heterogeneous operational environments, one of the key security issues of Grid projects resides in the concept of a *virtual organisation*. As seen in section 3.4.2.1, virtual organisations are collections of different self-governing entities which cooperate through the Grid environment. As discussed in section 2.7.4.3, most current security technology and techniques are derived from the military world, although they are arguably unsuitable for modern computing environments. Virtual organisations, by virtue of their lack of centralised management or hierarchy, are even further removed from the military information fortress model than most ordinary organisations.

It is very important to note that decentralisation is not only a characteristic of the eventual Grid application, but can also be a characteristic of the development environment of the Grid project. Because of the involvement of a variety of different organisations in these Grid projects, the development teams of most Grid projects tend to be geographically distributed over a variety of areas. Whilst technology does allow

communication to be undertaken between these different teams remotely, this requires a specific act as opposed to the natural by-product of face-to-face interaction. The increasing trends of outsourcing software development mean that decentralised development environments are increasingly common, making the study of security development in Grid applications all the more relevant.

3.4.2.3 Limitations of Research Based on Grid Projects

Research into security based on Grid projects is not without limitations however. As mentioned previously, the field of Grid computing is relatively recent and still largely the province of academia. It is therefore very important to mark the distinction between academic and commercial environments, especially when it comes to security. In academia, the need for open collaboration is a strong factor against the adoption of security. Whilst some commercial organisations are still disinterested in security, the majority is concerned and considers matters of security to be of great importance. Therefore whilst research conducted on current Grid projects is applicable to academic software development, care must be taken when generalising any research conclusions to commercial software development.

Also, as a consequence of the decentralised software development approach common in Grid projects, generalising any research conclusions to more centralised development environments can be made more difficult. This should only be undertaken with great care and consideration for the possible differences between these different environments.

3.4.3 Indicators for Success and Failure of the Research Approach

As seen in sections 3.2 and 3.2.1, indicators of success with Action Research lie mainly in feedback from participants. This feedback can take the form of participants reporting positive outcomes, expressing satisfaction with the intervention, or changing their operations as a result of the intervention. In accordance to this, factors that indicate failure of the intervention also lie in feedback from participants, either through direct criticism or through indirect events such as a lack of engagement or an absence of change during and following the intervention. In order to capture this type of feedback, particular care should be given to record the intervention and analyse this impartially to identify both positive and negative feedback. In this respect, the combination of Action Research and Grounded Theory as a means of analysing the intervention is very useful as a means of systematically identifying both good and bad feedback.

With the Grounded Theory aspect of this research approach, the main indicator of success is the identification of a theory that fits and explains the data gathered. Without a theory that coherently explains the data gathered, and can be logically argued to generalise to fields other than the immediate research experiment, the Grounded Theory analysis can be deemed a failure.

An additional factor for success can be seen in the identification of substantive findings from the studies, such as the existence of security problems and the identification of potential solutions. These findings reinforce the claims that the intervention methodology is effective at identifying and addressing security problems. Failure to identify any security issues during the intervention could indicate a significant problem with the methodology, and strongly undermine claims of success.

3.5 Validity of Research

The case studies presented in this thesis follow the six Action Research criteria mentioned previously:

1. **A theoretical framework must be present as a premise of Action Research [19].**

   The theoretical framework that informs the studies is visible in the secure socio-technical software development process AEGIS as presented in section 3.6.

2. **Data collection methods should be carefully selected [19, 21], and capable of capturing both intended and unintended effects [52].**

   All the case studies have been recorded and transcribed as a means of capturing the qualitative data on which subsequent analyses could be carried out.

3. **The researcher should actively intervene in the research setting [21].**

   In all the case studies, there was active involvement from the part of the researcher, which resulted in actions being undertaken that would not have done otherwise.

4. **The immediate problem in the social setting must have been resolved during the research [21, 102].**

   As discussed in more detail in chapters 4, 5 & 6, all the case studies have shown evidence of improved understanding of security, and a greater interest in addressing human factors in security.

5. **The Action Research approach should be cyclical. The use of multiple cycles allows the early conclusions of the researcher to be scrutinised and refined in the later stages [19].**

The two case studies presented in chapters 4 & 5, took place over several sessions, which allowed refinement and analysis of the data to be carried out throughout the actual study. The case studies presented in chapter 6, although they only occurred during a single session, built on top of the lessons learned during the two previous studies. Therefore it can be seen that the research presented in this thesis as a whole consisted of several iterations of application and reflection, undertaken with a total of four different projects.[49]

6. **Generalization about the results should be tempered with an interpretation of the extent of similar settings to which the theory can be expected to apply [19].**

   All the generalisations made on the basis of the research are accompanied with a clear argumentation as to why these are valid.

In addition to this, published papers and presentations by the participants in the workshops detailing benefits and outcomes are referenced as a further means of validating the approach.

The Grounded Theory component of the research is validated through:

1. Theoretical sampling – selecting case studies that highlight different attitudes towards security. This has lead to four case studies being selected, two of which (CLEF & DCOCE) are specifically concerned with developing a security technology. The other two studies are more concerned with achieving a functional product that requires security.
2. Reflexivity – is shown through the gradual refining and expanding of the analysis to encompass each new case study. In addition, during the process of coding the workshop transcripts, the thoughts of the researcher were recorded in memos as a means of documenting the evolving interpretation of the data. A sample workshop transcript together with the coding and memos is presented in Appendix A.
3. Constant comparison to the data. During the analysis of the transcripts, the emerging categories and subsequent relationships were continually compared to the data to ensure that the theory fit with the data.
4. Detailed information about the context in which these case studies operate. As part of the Action Research component of this thesis, a close look and description of the context of the studies is given which is used to ground the intervention.

3.6 AEGIS

AEGIS is a socio-technical software engineering methodology for creating secure systems based on asset modelling, security requirements identification, risk analysis and context of use.

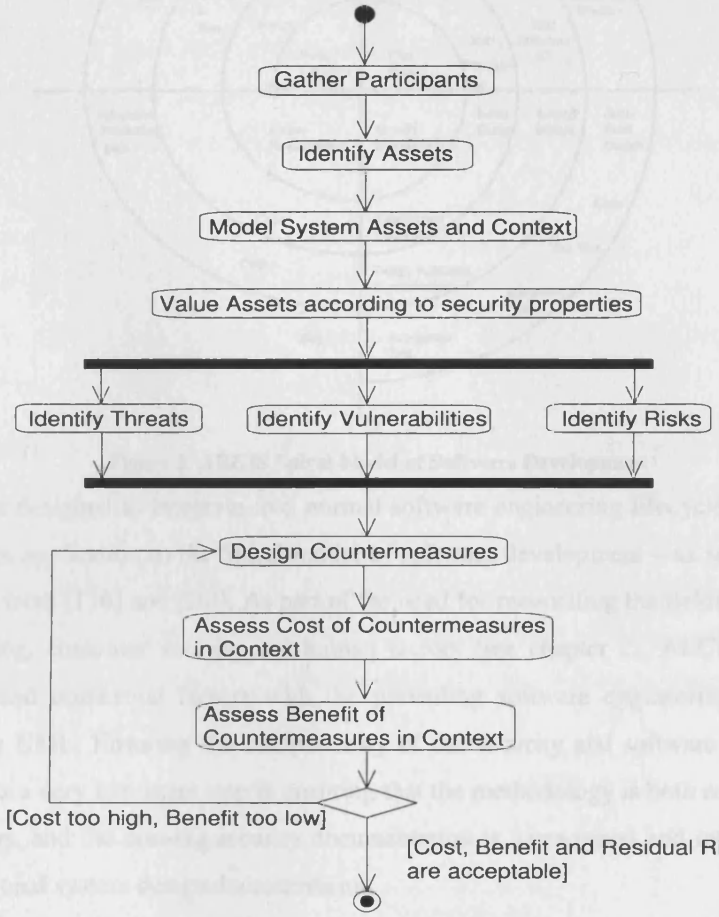


Figure 1: AEGIS activity diagram

The purpose is to provide developers with simple and intuitive tools for developing a secure system that takes user needs into account and promotes security buy-in. The core process of AEGIS can be seen in Figure 1. This consists of gathering participants in the design process (see section 3.6.1), identifying the system's assets, modelling them in the context of operation and identifying security requirements based on these (see section 3.6.2). The next step is a risk analysis in which vulnerabilities, threats and risks are identified which then informs the security design process (see section 3.6.3).

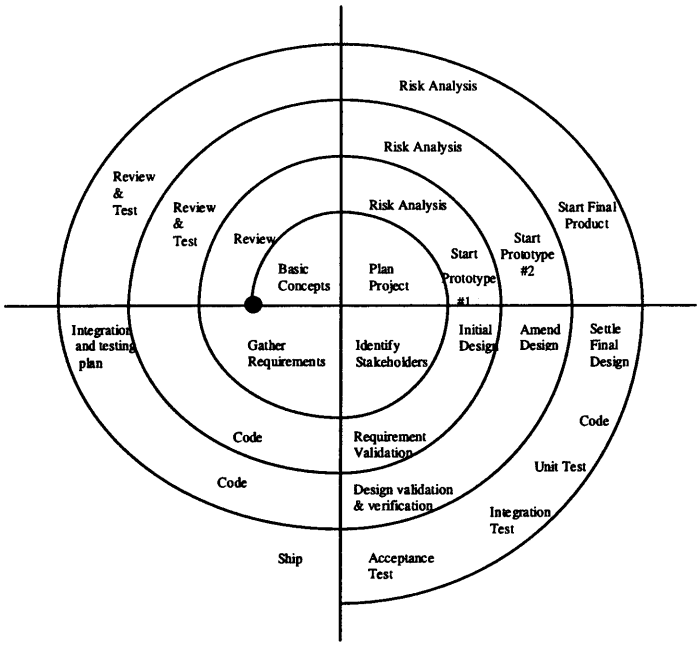


Figure 2: AEGIS Spiral Model of Software Development

AEGIS is designed to integrate into normal software engineering lifecycles, as can be seen in its application to the Spiral model of software development – as seen Figure 2 (inspired from [116] and [26]). As part of the need for reconciling the fields of software engineering, computer security and human factors (see chapter 2), AEGIS integrates security and contextual factors with the prevailing software engineering modelling technique UML. Ensuring the compatibility of the security and software engineering notation is a very important step in ensuring that the methodology is both easy to use by developers, and the ensuing security documentation is harmonised and integrated into the functional system design documentation.

The inclusion of contextual elements draws upon research into *contextual design* (see section 2.3.3.1), which is an HCI technique for designing usable systems based on the identification of contextual information at the design stage. The principle behind contextual design is that greater understanding of the context in which users operate is necessary for designing a system that is well-suited to its users. By providing a model that encapsulates context and security, this ensures that human factors and security are visible throughout the development process (more detailed information about the modelling technique is presented in sections 7.3 & 7.4).

It is particularly important to note that AEGIS is not intended to provide solutions or answers to security questions (like checklists for example), but is instead aimed at

giving developers a process through which relevant information pertaining to security is identified and decisions are made based on this.

3.6.1 Gather Participants

The first step in the process requires the stakeholders in the system to be identified and assembled: developers, users, owners, security experts, etc. It is important to have a variety of stakeholders participating in the analysis (i.e. owners/management and different users should be represented). The reason for involving these stakeholders is to ensure that:

1. all contexts in which the system is used are represented, and
2. stakeholders become aware of each others' needs.

3.6.2 Identify and model assets and security requirements in context

The foundation of AEGIS is to base every security decision on knowledge of the assets in the system. Inspired by the work of [63], a UML compatible notation is used to model the system, its assets and the context of operation. A simple example of this notation is presented in Figure 3. This notation models context through the use of packages – in this example the accountant and secretary both work in a main office and both use the same workstation. The workstation contains a salary database and a word processor, and is connected to the Internet.

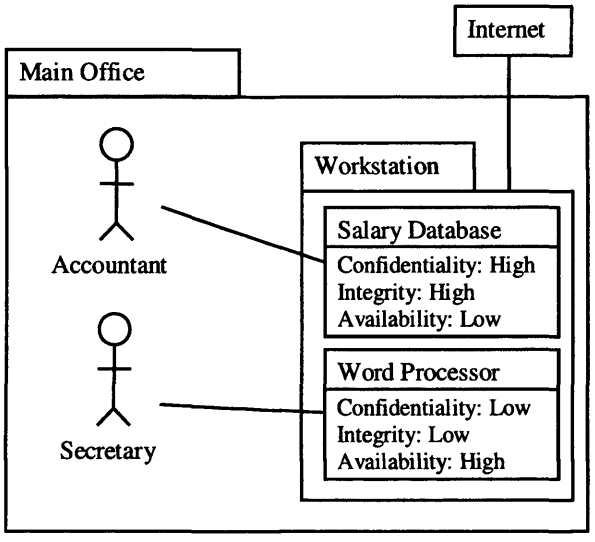


Figure 3: Simple Model of AEGIS Modelling Notation

The security requirements of these assets are described for the salary database as high in confidentiality and integrity and low for availability. This reflects the sensitive nature of

the salary data, the need to ensure it is not modified and the relative infrequency of the need to access it. The word processor however has a low confidentiality and integrity requirement, reflecting the fact that the work carried out is not sensitive, but the availability requirement is high as the secretary needs it to be able to work.

In this phase of the design process, using the modelling notation described above, stakeholders must build a model of the system representing various assets and their relationships. Particular attention must be placed on modelling the *context* in which people are interacting with the system. This includes the physical and cultural environment, the particular roles that people must assume and the tasks they must perform [23]. Security requirements can then be gathered from the stakeholders based on the specific assets in question (Figure 3 shows a simple example of the kind of model that should be generated).

Putting a value on security properties of an asset can be done either quantitatively (for example through a monetary value) or qualitatively through some kind of judgement of the stakeholder. Given that the value of a security property of an asset is rather difficult to accurately judge quantitatively, using a qualitative rating in this situation makes more sense in that it captures the stakeholders' judgement directly. A comparison of the qualitative values against each other can then be carried out to identify which aspects of the system are of most importance to the different stakeholders.

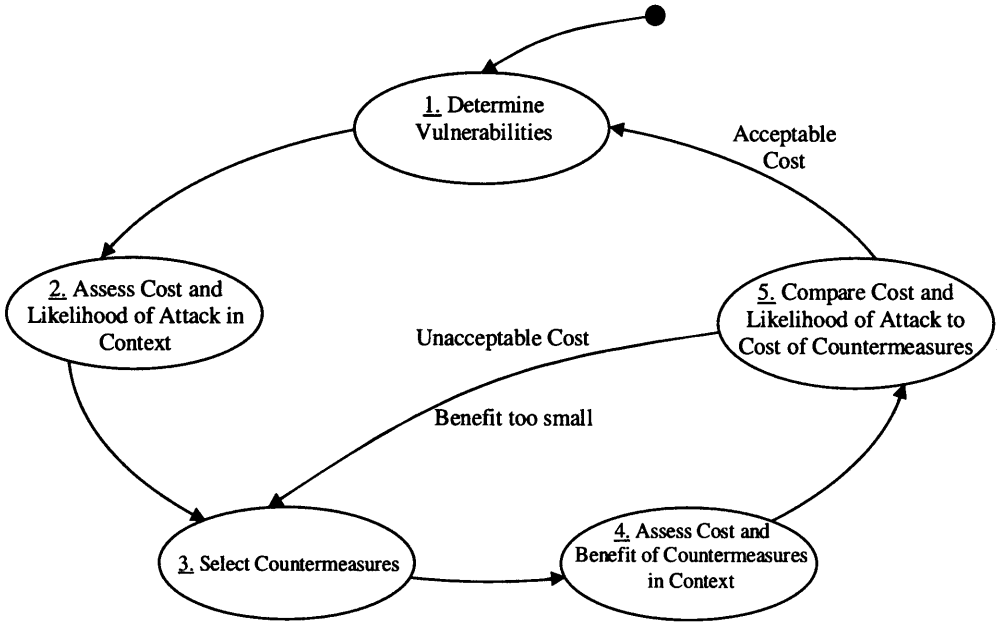


Figure 4: Risk Analysis and Security Design Process

3.6.3 Risk Analysis and Security Design

This phase focuses on identifying risks, vulnerabilities and threats to the system, and designing the appropriate countermeasures. Figure 4 shows the process of risk analysis (see section 2.5.2.4 for an overview of risk analysis) and security design.

1. Determine *vulnerabilities*

A *vulnerability* is an area which is susceptible to an undesirable action. There are many kinds of vulnerabilities, which can be broadly divided into two categories: technological vulnerabilities and social vulnerabilities. Both should be considered equally.

2. Assess *cost and likelihood of attack in context*

This step is necessary to establish how damaging an attack on the asset (utilising the vulnerability) will be, and how likely it is to happen in the context of use.

As John Adams asserts, *"... risk is subjective. It is a word that refers to a future that exists only in the imagination."* [12]. He also shows that any risk compensation affects the risk being compensated for and that subsequent behaviours can create different risks [11]. Adams illustrates this with evidence that seat-belt legislation has reduced the number of injuries in car passengers, but has increased the number of injuries to pedestrians. This is because seat belts provide the driver with an added sense of safety and their behaviour becomes less risk averse as a result. Assessing risk is therefore a complex endeavour which, as [25] state would benefit from adopting a structure which allowed the sharing of information.

Quantitatively evaluating risks and damages, such as the ALE (a product of the probability of the risk occurring and the financial damage it would incur [25]), allows an easily used and shared measure for risk and damages. Another example of a widely used quantitative risk measurement is the security metric accompanying CERT vulnerability disclosures [1], which is based on a number of factors including the impact of the vulnerability being exploited, the ease with which it can be exploited, the number of systems at risk, etc.

One problem with this is that only easily financially estimated assets can make use of this. Non-tangible assets such as reputation, goodwill, staff morale, etc. cannot be assigned a meaningful quantitative financial cost, and this does not take account of non-financially motivated attackers.

Furthermore, the usefulness of sharing quantitative ratings (such as the CERT security metric) – thereby reusing some of the acquired knowledge in the field – is currently badly affected by their lack of contextual information. Without this information, it is impossible to know whether the value has any use in a given environment.

In this step, it is important to seek accurate knowledge in order to achieve an informed decision and both quantitative and qualitative measurements should be used where most appropriate. Since risk is ultimately subjective, a consensus should be reached with security experts and stakeholders, based on available information – which can include existing risk assessments, field experience, numbers of past incidents, environment of the asset, dependencies between assets, etc.

When determining the cost of a potential attack, one method of assessing this is to have users of the system evaluate the consequences of an attack. This information should be gathered from as many users as possible. Once obtained, this information can be correlated with other sources, such as legal requirements, industry standards and dependent assets so as to gather a good picture of the cost of an attack.

3. Select *countermeasures*

Countermeasures are chosen to address a vulnerability. The decision can be:

1. to deploy no countermeasure,
2. to put countermeasures into the system, i.e. means of deterrence, prevention, detection and reaction to attacks,
3. to transfer of liability and responsibility (through insurance or third party intervention).

4. *Cost-benefit assessment* in context

Cost of countermeasures

Cost in this section not only addresses financial issues, but also refers to the effort a user will expend employing the countermeasures. The context refers to the environment in which the attack can occur and in which the countermeasures are deployed. For example, if a system forces a user to change his password whilst he is simultaneously being urged to achieve a production task for which he needs the system, the cost will be very high both in terms of loss of productivity and in frustration of the user.

Benefit of countermeasures

Benefit in this section refers to whether the controls actually reduce the risk, as well as establishing whether they provide any advantages to the user. It is important to put the

control in context with other security controls as well as the rest of the system. Taking the previous example, the benefit of forcing a password change may not be particularly evident in the face of the potential problems. It may be that a different or additional countermeasure would be more beneficial. A different countermeasure - such as a physical authentication token - or an additional countermeasure - such as user training in selecting passwords - would provide additional benefits to the user, at the cost of greater financial expenditure and the potential creation of different risks (such as having the token stolen).

5. Compare *cost* and *likelihood of attack* against *cost of countermeasures* in context
This is to establish whether the vulnerability poses sufficient risk and potential damage to justify the cost of the countermeasures. If the cost proves to be unacceptable, or the risk still too great, you must return to step 3. Otherwise you must go on to step 1 and conduct a new determination of the vulnerabilities taking the new countermeasures into account. If no further controls have been added, the assessment is over.

3.7 Summary

In this chapter, the research methodologies of Action Research and Grounded Theory have been presented (see sections 3.2 & 3.3), together with an argumentation of their relevance to the research problem. The approach adopted in this research of applying Action Research to Grid project case studies and further analysing the results using Grounded Theory has also been presented (see section 3.4), together with an argument about the validity of such an approach (see section 3.5).
Finally the AEGIS process has been introduced (see section 3.6) as the guiding methodology informing all the interventions in the studies.

4 Empirical Research: EGSO Case Study

4.1 Introduction

This chapter describes the first case study in which AEGIS was applied. The study involved the European Grid of Solar Observations (EGSO) project, and details of this Action Research study are presented in section 4.2. As part of the methodological validation of the design method, the application of AEGIS is described in section 4.3, with the conclusions of the Action Research in section 4.4. This is then followed in section 4.5 by the Grounded Theory analysis of the transcripts which is aimed at identifying substantive factors affecting the application of the AEGIS. These range from ascertaining the importance of motivation and responsibility in the design of security to recognising issues of communication and the role that stakeholders play during the design of security.

4.2 Description of study

4.2.1 What is EGSO?

The EGSO project is run by a consortium of different global partners (including among others British, French, Italian or American institutions), with a heavy emphasis on academic participation. EGSO is funded under the Information Society Technologies (IST) thematic programme of the European Commission's Fifth Framework Programme. The project is one of many partners from across Europe that co-operate through the EU GRIDSTART initiative. [8]

The purpose of EGSO is to provide a Grid making the solar observations of a number of different observatories and institutions available to customers, scientists in particular. EGSO is intended to operate as a virtual observatory, providing a platform through which scientists can access solar observation data from around the world. In addition to providing access to solar data, EGSO also intends to provide a distributed computation service for analysing the data.

4.2.2 Details of the study

The case study consisted of an initial three interviews with up to four different project members of EGSO in order to determine what the aims and requirements of the project were, and also to establish the current state of security in the project. This was followed up with a series of four workshops held at University College London on 7/02/2003,

28/02/2003, 11/4/2003 and 6/6/2003 with up to three project members (two developers and one manager/user) in which the AEGIS methodology was applied. The interviews were recorded and used to provide background information about the project. Each of the workshops lasted between 2 and 3 hours and was recorded, transcribed and a Grounded Theory analysis was conducted using the qualitative analysis package ATLAS.ti.

4.2.3 Grounded Theory model semantics

All the models presented in the Grounded Theory analysis were generated using ATLAS.ti and consist of network diagrams that relate the different categories identified. The semantics of the notation are as follows:

- == means *is associated with*
- => means *is a cause of*
- [] means *is part of*
- <> means *contradicts*

A simple example of this is in the following diagram:

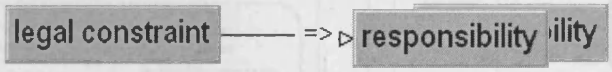


Figure 5: Example Grouned Theory network diagram

This diagram means that a "legal constraint" *is a cause of* "responsibility".

In the rest of this chapter, the Action Research application of AEGIS is presented, followed by the details of the Grounded Theory analysis.

4.3 Application of AEGIS

4.3.1 Asset Identification

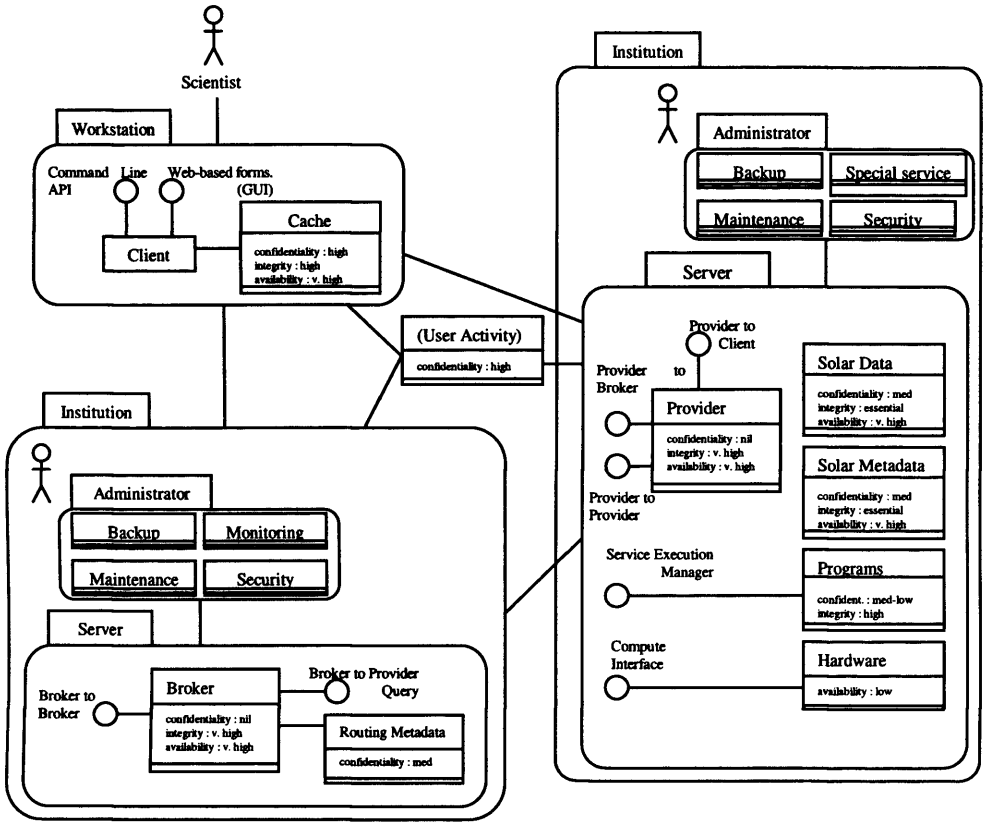


Figure 6: EGSO Asset Model

The process started by focussing on identifying the major assets of EGSO. The participants in the process were asked to draw a model of EGSO identifying assets, their relationships and the context in which they existed. Because of the distributed nature of GRID applications, there can be many iterations of identical components throughout. For the purposes of simplicity, it was decided to build a model that represented all the different types of assets without necessarily depicting how many of these assets were going to be deployed.

The natural inclination was to draw the system isolated from its environment, and the participants were encouraged to describe where people were involved in the system and the kinds of environments various different parts of the system existed in. The wide range of possible environments for EGSO users prevented the modelling of too much detail, although in such cases it was possible to model an abstract type of environment (such as "institution" where this could equally represent a university, private company or a small observatory).

4.3.2 Security Requirements Capture

Once the main assets of the system had been modelled, the process of identifying security requirements began. The concepts of *confidentiality*, *integrity* and *availability* were defined for the participants. Then specific assets were examined and the participants were asked to rate them qualitatively according to these three terms. Since an abstract approach was proving confusing, these ratings were derived by evaluating what the impact would be on the system should a specific scenario of attack occur.

For example, this is how the solar data asset was rated:

- Availability: *"What would happen if users were unable to access this information?"*
  Answer: The system needs to be *"robust within reason"*. Identifying levels of availability was *"not something that's been clearly defined."* Availability was therefore rated by the participants as being a 'very high' requirement.
- Integrity: *"How important is it for the information held at the providers to be what users and providers expect it to be?"*
  Answer: *"If there was no data, there would be no system"*. Similarly, if the data was modified in any way so as to mislead would be unacceptable. The Integrity requirement was therefore rated as being 'essential'.
- Confidentiality: *"Does the solar data have to be kept secret from anyone?"*
  Answer: *"Some providers may want to restrict the access to the data for a period of time"*, but *"they may not want to use EGSO for that type of data"*. The Confidentiality requirement was rated as 'medium'

Capturing security requirements based on the asset model proved to be useful for three reasons:

1. Participants had to look systematically at their system and identify a wide range of security requirements for every part of the system (many people tend to forget that requirements other than confidentiality are also important).
2. It allowed the explicit description of implicit assumptions, which in turn uncovered important information (c.f. responsibility in section 4.5.2).
3. The final outcome, although it consisted of qualitative ratings, allowed the easy identification of the most important assets in the system, and a ranking of these according to their relative importance.

One point of note is that with regards to modelling assets, a number of instances appeared where aspects of value in the system were not assets as defined, the most obvious example being the reputation of the project. Whilst this was not modelled

explicitly, it was reflected in the model through the importance rating of the assets which were deemed to be able to damage the reputation.
The full asset model, complete with the identified security requirements can be seen in Figure 6.

4.3.3 Risk Analysis

Prior to any active involvement with EGSO, during the interviews, there had been a debate about whether or not to use digital certificates for the purposes of authentication and authorisation. The perceived cost and complexity of employing certification was driving the discussion, but the full consequences of either path of action had not been fully analysed.
Before even starting the risk analysis, a strong desire to avoid having to use digital certificates was voiced, justified by the statement that only a few users would need access to sensitive computing services. In the risk analysis it was identified that although few users may have actually needed access to these computational facilities, other aspects of the system (such as personal user spaces) were also sensitive and at risk (see section 4.3.4 for details about how this risk was addressed).

The risk analysis started by identifying the various dependencies between the assets of EGSO. This highlighted, for example, that the availability of the solar data (rated as very high) was completely dependent on a wide range of factors such as data provider administrators, broker administrators, routing, hardware operation, network links and their traffic.
Whilst a comprehensive and extensive risk analysis was not conducted during the workshops with the participants, a number of new vulnerabilities, mainly in areas of availability of services and integrity of data were identified and extrapolated from the scenarios used to identify security requirements. The focus was put on the scenarios which were deemed to be more important to the system based on the importance of the asset ratings, or the potential for damage. This included the scenarios in which the data that was assumed to be public could be modified to suit a particular attacker, where user software running through the user executable code service could be used to attack the system, or where a third party gained access to a user's personal space.

4.3.4 Security Design

Security design was also focussed on those few areas which were deemed to be most important to the system, and the identification of the dependencies in the beginning of the risk analysis highlighted the total dependency on system administrators and prompted the need for specifying their technical duties in a security policy. The specified need for the cost incurred by data providers to be low also prompted a design proposal whereby larger organisations could act as a proxy, taking responsibility for security procedures and alleviating the administrative burdens of running the system for smaller providers.
Other areas were also identified where policies would have to be detailed, such as the expansion to different providers, data update and integrity control, and acceptable use.

The user executable code service proved to be an interesting source of discussion, where the risk analysis highlighted serious difficulties with allowing arbitrary code to be run on the system when combined with the desire for access control mechanisms to be as lightweight as possible. In addition, other services available to users included a personalised user space which would store information about the queries and details of the previous work run on the system. The confidentiality of this information, having been rated as high, was also at risk from weak authentication methods. In the face of these potential failures of the system, this discussion led to the participants reviewing a long held assumption that 80% of users of the system would not need to be subjected to a strong access control mechanism. Different mechanisms were discussed to address this such as digital certificates, username and password combinations or IP address filtering. The particular costs of deploying each mechanism were also discussed, such as user costs, administrative costs, as well as the different strengths of each of these methods, or their likelihood to be applied appropriately (it was readily recognised, for example, that digital certificates are expensive to administer and could cause problems with users).

In a similar vein, although not as potentially dangerous as allowing the execution of arbitrary code, the fact that EGSO would be executing third party software on its systems as a service to its users also highlighted the exposure to software exploits from a source that was not under the direct control of the system. Means of addressing this were discussed and ranged from code review, strong limitations on the variety of different packages available and sandboxing of the applications.

4.4 Action Research Summary

The initial interviews uncovered the presence of very competent software engineers in the project, who advocated that a rigorous software engineering approach should be applied to EGSO. This could be seen in documented use cases, requirements validation, user interface design and UML system design. The need for security had been acknowledged and some use cases, albeit in vague terms, described the need for some security mechanisms (e.g. the need for *"direct access to satellite data in near real-time, perhaps only with necessary authorisation"*).

During the interviews, a number of undocumented security needs were voiced, such as *"users want their results to be protected"* and data providers need to protect their resources from being swamped and attacked.

The interviews also uncovered, however, that to the best knowledge of the interviewees, *"no one is in charge of security"*. Furthermore it was also stated that security had not been considered in depth because the project was *"still in (the) early stages (of) going from requirements to design"*. A final comment justified a lack of concern for security by insisting that functionality was much more important at this time, and that security would be addressed later. In this case, security was considered as a *non-functional requirement* and the decision to address security at a later stage is an example of *development duality* [106] (see section 2.6.2).

Inaccuracies in the understanding of security technology were also uncovered such as, for example, the notion that middleware would *"take care of the PKI"* (Public Key Infrastructure) in a digital certificate scheme. The underlying assumption in this statement being that a PKI only requires the design of software, but as the acronym describes, an *infrastructure* is necessary, and therefore a human infrastructure needs to be designed and implemented.

Despite the presence of very competent software engineers and actively recognising the need for security, the project had not taken any systematic approach towards making their system secure. Thanks to the enthusiasm for security advice, however, four workshops were organised. In all four workshops two developers were present and in two of them a project manager/user was also present.

AEGIS was particularly effective in its use of a graphical asset based notation, which provided a means of ensuring that all the participants were talking about the same parts of the system, and focussed the discussions onto the various assets of the system. This

proved to be very useful in refining general statements about security, by grounding discussions about security into the specific assets in question. Given that the development team was already using UML, the specific notation for the asset models was easy for the participants to understand.

Another key strength of AEGIS was that the participants became more aware of security issues and constraints. In the words of one of the developers, *"... what [AEGIS] is doing is reducing the unknown unknowns and converting the unknown unknowns into known unknowns."* This illustrates and enforces the point that the process of AEGIS is not intended to provide or prescribe security solutions. Instead it is aimed at identifying the issues that need to be addressed during the design of security (e.g. what type of security is necessary, how much is necessary, what are the human constraints associated with a technology), and providing a process for assisting developers in designing the most appropriate security for the system.

A presentation [61] given by one of the developers at a workshop on practical security for e-Science projects highlighted that AEGIS had *"... improved understanding of [the] problem space"*, and the modelling approach had fostered a *"commitment to [a] shared conceptual model"*.

Some of the difficulties encountered in this case study revolved mainly around complexity. Whilst a sample risk analysis was conducted, complexity and time constraints restricted the extent to which risks could be assessed. This was exacerbated by the fact that many issues were focussed on to the detriment of others. That is to say that the natural inclination of all the participants was to identify an area of concern and then focus the discussions on that area. Whilst these discussions were useful in identifying security issues, the coverage of the whole of the system was not achieved. This reflected the need for AEGIS to provide more structure and more rigour as a means of ensuring the equal coverage of the whole system. As seen in section 7.3.4, the risk analysis was revised to include more structure, and the need for a facilitator was identified. The role of facilitator is intended to ensure the smooth running of the process, ensuring that all parts of the system are actually analysed and addressed.

4.5 Grounded Theory Analysis

4.5.1 Introduction

The Grounded Theory analysis of the transcripts of the workshops allowed the identification of the properties and dimensions of four main factors in the design process of secure systems: *responsibility*, *motivation*, *communication*, and *stakeholders*.

Throughout the following quotes A, B and C stand for the EGSO participants, and R stands for the researcher.

4.5.2 Responsibility

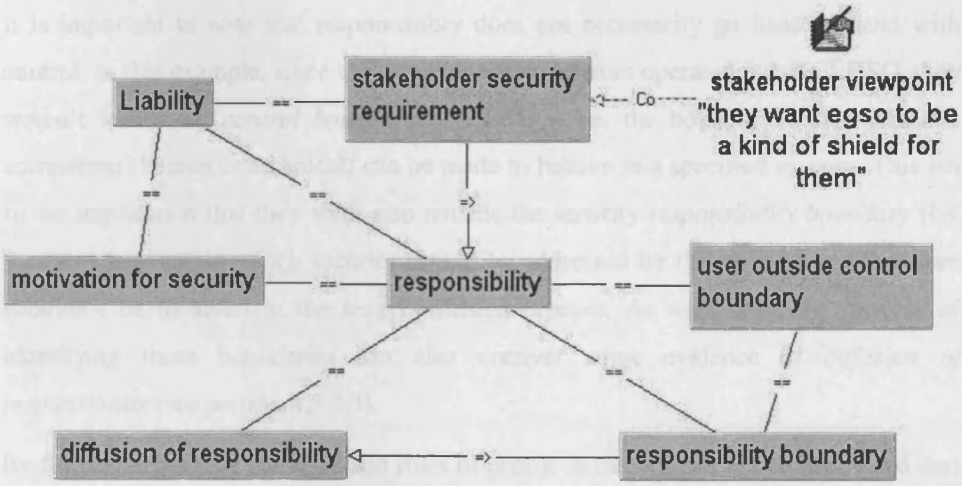


Figure 7: Responsibility Model: EGSO case study

> *"The difficulty is that politically, and organisationally, the project is kind of broken down into people who are pursuing their own research interests."*

In the diverse development environment of EGSO (with multiple institutions from countries around the world participating in EGSO), each party tended to be focussed on their *own research* interests.

As shown in Figure 7, responsibility for security is tightly linked with motivation, liability (see section 4.5.3), and stakeholder security requirements. Stakeholder security requirements meant that the EGSO project had the responsibility for providing security to these stakeholders, such as data providers who wanted *"EGSO to be a kind of shield for them"*. However the responsibility for security within EGSO was not assigned to anyone in the project, and as illustrated by the previous quote, this resulted in a distinct lack of coordination and even interest in security from other parts of the project.

4.5.2.1 Boundaries of responsibility and control

During the AEGIS process, identifying the roles of people in the system and the environment in which they operated generally led to the identification of boundaries:

1. of *control*
2. of *responsibility*

> *"R:(...) do you assume that there is an admin at each provider?*
> *A: yes it is assumed that there will be somebody who has the role of an admin, whether that is there...*
> *B: I believe they're outside EGSO*
> *A: yes they're outside EGSO*
> *B: they don't have to know the nitty gritty*
> *R: but you're assuming that someone at the provider end is in charge of the resource and is capable of modifying access and things like that."*

It is important to note that responsibility does not necessarily go hand in hand with control. In this example, since the provider administrators operated outside EGSO, they weren't inside the *control boundary* of EGSO – i.e. the boundary within which a component (human or technical) can be made to behave in a specified manner. This led to the implication that they were also outside the security *responsibility boundary* (i.e. the boundary within which security should be addressed by the project) and therefore shouldn't be included in the security design process. As seen here, the process of identifying these boundaries can also uncover some evidence of *diffusion of responsibility* (see section 4.5.2.3).

By further identifying the tasks and roles of people in the system, it was uncovered that the system was designed with the implicit assumption that the provider administrators would achieve specific tasks pertaining to EGSO, such as maintenance, providing special services to specific customers, or ensuring security. Since these tasks directly impact the confidentiality, availability or integrity of assets, it became clear that these administrators were within the security *responsibility boundary* of EGSO whilst being outside of the direct *control boundary* of the project, and therefore should be included in the design process of security.

Another example of this is that responsibility for ensuring the confidentiality of a user's data resided with EGSO. However some of the measures put in place for this could fall outside of the control of EGSO, such as ensuring that users adhered to usage policies for access control mechanisms for example.

4.5.2.2 Control and Usability of Security

> *"B: I actually think for the provider administrator here, we've got very little control over them as well have we?*
>
> *C: We've said that because we want to encourage as many providers as participants as possible, including the very small resource-poor providers, that we need to do the thing in a way that is as little burden as possible on the providers. I mean ok, there are some providers who are quite happy to do things with you, but there will be a few, and all they want to do is to say this is where you get at the data, and don't knock on our door too much (laugh). So, for the resource rich providers, the administrator might be able to do quite a lot for us, but again we don't want to put a huge burden on them."*

In situations where there is a lack of control over parts of the system, yet security has to be provided – such as at the provider administrator, or user authentication level – the stakeholders expressed the need for making security as lightweight as possible. This can be seen as an indication that low security overheads and easy-to-use security mechanisms are important for situations where control (i.e. enforcement, monitoring, or auditing for example) is not possible. This is a key argument in favour of usable security since it provides a means of addressing the lack of control of security.

4.5.2.3 Diffusion of Responsibility

Another property of responsibility that has been identified is the propensity to assume that another party will or should take care of security. This is what social psychologists call *diffusion of responsibility*: the notion that everyone assumes that someone else will take care of a particular problem [41] (see Figure 7). This has been identified at the system level through undocumented assumptions that administrators will take care of maintaining access control lists, backup systems and perform special services.

It has also, to some extent, been identified at the project level:

- Where lack of control over a component (as identified in section 4.5.2) leads to a reaction of avoiding the consideration of security for that component.
- Where some security issues are assumed to be the province of another party (namely governing bodies such as Gridstart, or the eScience Security Task Force). For example, in order to use certificates in the system, a network of trust between certification authorities has to be in place.

> *"It really shouldn't be up to EGSO to establish this network of trust itself. It should be relying on people to certify people within countries and organisations."*

As seen in this quote, the responsibility for setting this up and administering it was argued to be in the hands of a third party, possibly the governing bodies of the Grid

projects – but no discussion had arisen between the project and the governing bodies with regards to resolving this issue.

4.5.3 Motivation

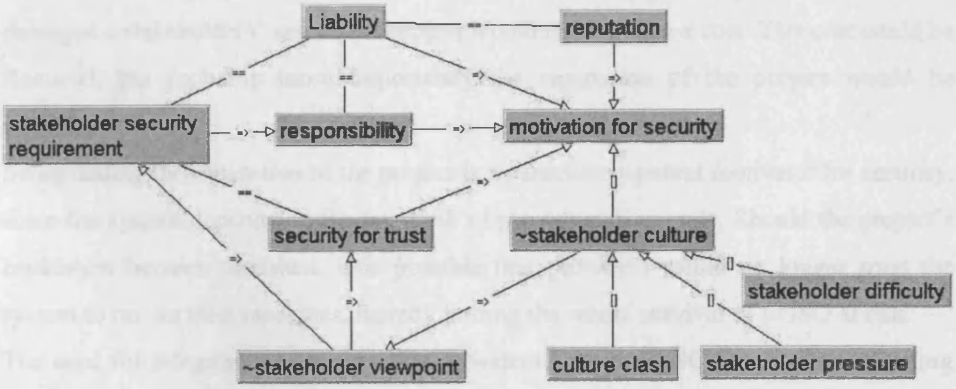


Figure 8: Motivation Model: EGSO Case Study

During the interviews, the motivation to apply security in EGSO varied from enthusiastic to uninterested. It was stated, for example, that security would be addressed once functionality was finished. This led to parts of the development team of the project that was not involved in the workshops ignoring any matters pertaining to security (even when part of the project *was* addressing security issues, the rest of the project tended to be completely uninterested, e.g. "*He's* [the architect] *not 'all right, what's been happening with the security, what's coming out of it?' do you know what I mean?*").

A detailed analysis shown in Figure 8 describes all the factors that affect the motivation for security during the development process. The main factors affecting motivation for security in EGSO are:

- Responsibility
- Liability
- Reputation
- Trust
- Customer culture.

Responsibility is a key motivator for addressing security. In this case study, the manager was keen to take responsibility for security (even though he was not explicitly given this responsibility) and facilitated and participated in the security design process.

Liability is a refinement on the notion of responsibility in that it represents what a third party expects to be the responsibility of the project – not necessarily what the project recognises or accepts as their responsibility. Liability is a motivator for security in the sense that if the project were to facilitate or inadequately guard against an attack that damaged a stakeholders' assets, the project would have to face a cost. This cost could be financial, but probably more importantly the *reputation* of the project would be tarnished.

Safeguarding the *reputation* of the project is a particularly potent motivator for security, since the system depends on the goodwill of providers to operate. Should the project's *reputation* become tarnished, it is possible that providers would no longer *trust* the system to run on their machines, thereby putting the whole survival of EGSO at risk.

The need for safeguarding the trust that providers have in EGSO is a strong motivating factor for addressing security.

The *customer culture* in this case seriously affects the motivation for security.

> *"They (customers) want something that they can sit down and physically play with, rather than something which is presented on paper. (...) They'd be happy with code, whether it works or not, they'd be happier with seeing some code rather than seeing some abstract representation of some high-level app..."*

As illustrated by the quote, the customer culture in this case is perceived by the workshop participants to be particularly keen on achieving functioning prototypes as quickly as possible without necessarily going through structured engineering approaches. The pressure is therefore put on the developers to provide functionality as quickly as possible, to the detriment of security.

4.5.4 Communication

4.5.4.1 Confusion

Initially, participants required detailed explanation of the security concepts because of their extremely precise and abstract nature. During the discussion the differences between integrity, confidentiality and availability could become confusing, particularly after discussions had been ongoing for long periods of time.

> *"R: what about availability? Availability of this program. Judging from what you've said I don't think that availability is that...*
>
> *B: should it not match the most important resources?*
>
> *R: that's an interesting question... I think it's a bit beguiling, because we're not seeing the picture as it is.... No no no I'm sorry my mistake, I think availability is high, is a very high level, you're right... I'm just a bit confused."*

This is made somewhat worse by the fact that dependencies between assets can link two different security concepts in two different assets. For example the integrity of the broker node can directly affect the availability of the solar data, or the confidentiality of a user's activities.

Addressing issues of confusion, particularly for the purposes of eliciting the security requirements of each asset of the system, became very important. As mentioned in section 4.3.2, the participants were finding it difficult to understand what was meant by rating the security properties of each asset, therefore the question was clarified by using a *scenario* in which this particular property was compromised and asking the participants to rate how damaging this would be to the system.

The decision to use scenarios in the security requirements elicitation came from the observation that the communication of more complex security concepts throughout the case study generally took place in the form of *anecdotes* and *scenarios* which will be described in the following sections.

4.5.4.2 Scenario

Generally used as a means of overcoming the complexities associated with abstract security talk, scenarios occurred throughout the case study. As well as being used to elicit security requirements, scenarios were used to propose potential threats, suggest design solutions and describe how these would behave.

In more complex discussions, scenarios were also supported through the use of simple graphical representations (on a white board for example). These allowed the clarification of complex communications between different participants, and ensured that people did not talk at cross purposes.

It should be noted that abuse cases [80] are a form of documented scenario for the purposes of identifying security requirements. The difference here is that abuse cases are directly used to identify vulnerabilities by modelling attacks, whereas in this case scenarios are used to elicit information from the participants. As such these scenarios are not a security analysis tool like abuse cases, but a communication tool which then supports security analysis through the participation of others. Despite this difference, the notation employed by abuse cases (namely UML use cases) can easily be used here to document the scenarios.

4.5.4.3 Anecdote

As stated in section 2.5.1, knowledge about security in the literature is generally expressed in the form of anecdotes – either personal or vicarious stories. In this case study, security anecdotes were used by participants to explain and justify the possibility or likelihood of a particular security problem. For example:

> *"C: well yes again, X was telling me the other day that he's got some files that have vanished, and it's because he was running a mirror.*
>
> *B: mirror?*
>
> *C: he had a mirror running that should have been stopped, somebody deleted some data at the other end, so his data disappeared as well because he had got the delete switch on. Now this is the problem with the, not necessarily with the administrator, directly doing something, but inadvertently they allowed a change in the system at the other end to affect the copy at your end."*

Anecdotes about past attacks gathered from stakeholders who have experience in the field can also be a useful source of information. In the absence of other risk measurements, anecdotes can serve during the risk analysis as a means of informing the estimation of the likelihood of an attack occurring. Whilst this is not an ideal solution, it is frequently the case that there is no other information available on which to base this estimate.

4.5.5 Stakeholders

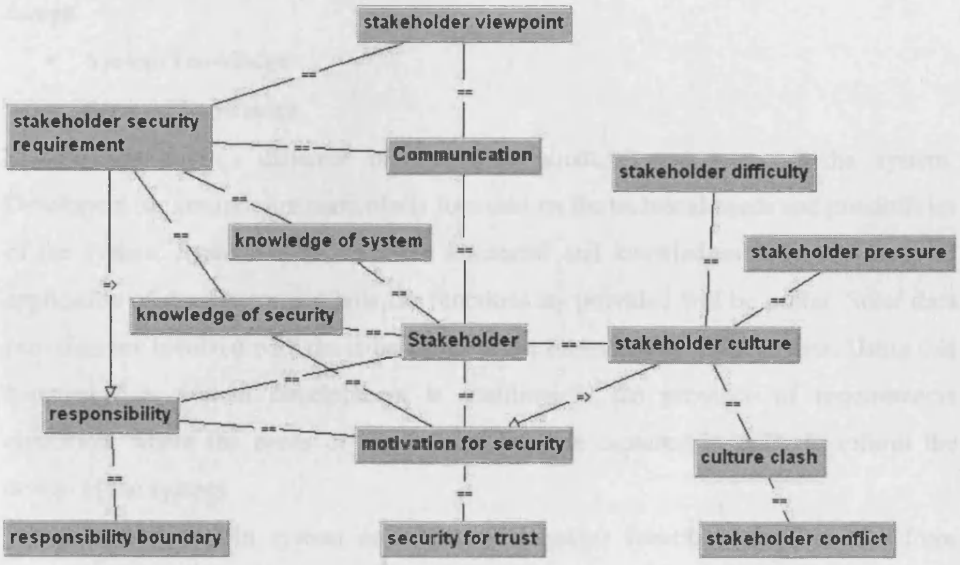


Figure 9: Stakeholder Model: EGSO Case Study

4.5.5.1 Stakeholder Viewpoint

The biggest benefit of involving stakeholders in the security process was that it was possible to directly elicit their point of view. This provided very rich information about security needs, constraints and limitations that would be acceptable to the different stakeholders. As an example, other than lack of control, another factor driving the need for providers to have easy to use security (see section 4.5.2.2), was the stated need for low buy-in. This was necessary in order for the project to secure as many providers as possible, thereby creating value in their system.

It is interesting to note that even with a relatively small set of stakeholders participating in the process (three in this case) it is possible to identify points of view from a variety of other stakeholders as related through the participants. These points of view can serve as a basis for identifying the value of the security properties of the system's assets. For instance it was initially stated that there was no need for confidentiality of the solar data. When asked about the point of view of the organisations supplying the data, it was identified that some solar data providers did have a requirement for temporarily ensuring the exclusive access to their data.

4.5.5.2 Stakeholder Knowledge

Different stakeholders have different types of knowledge that are relevant to the system design:

- System knowledge
- Security knowledge

Stakeholders have a different understanding about diverse areas of the system. Developers for instance are particularly focussed on the technical needs and possibilities of the system. System users are more interested and knowledgeable in the areas of application of the system and how the functionality provided will be useful. Solar data providers are involved with the collection and dissemination of research data. Using this knowledge in system development is traditionally the province of requirements elicitation, where the needs of the stakeholders are captured in order to inform the design of the system.

Whilst it is typical in system development to gather functional requirements from stakeholders, security development tends to adopt a different approach. As seen in sections 2.5.3 & 2.6.1, functionalist approaches to security derive security requirements from checklists, risk assessments based on questionnaires or the analysis of the system in order to determine what security is necessary. This type of approach only makes use

of security experts' security knowledge, and whilst they have the most security knowledge of all the stakeholder groups of the system, they are not the only source of security knowledge or needs. The needs of users, data providers, administrators, and developers are also important for security. As such the knowledge of these other stakeholders is also particularly relevant in the identification of security requirements that reflect accurately what the stakeholders want.

Without an approach adopting other stakeholder points of view, specific needs for ease of use (see section 4.5.2.2), identifying varying organisational attitudes towards security (see section 4.5.5.5) or identifying the need for documenting policy about administrative tasks (see section 4.3.4) may not have been identified.

It is also important to realise that stakeholder knowledge of security can be limited or even flawed. Mistakes, preconceptions and misunderstandings can affect the direction of the discussion. As an example, a load balancing mechanism for EGSO was proposed as a means of addressing denial of service attacks. Although this mechanism would be useful under heavy but normal operation, it would not be particularly effective against a targeted attack. Stakeholder conflict (see section 4.5.5.5) can arise out of mistakes, however in the case of mistakes or misconceptions, this can usually be resolved through communicating the reasoning behind different positions.

4.5.5.3 Stakeholder Security Awareness

To the best knowledge of the author, directly involving stakeholders in the security process has only been done by [69], and then only in the security planning of information security in an existing organisation – not in the development of a technical system.

One of the findings in that study was that awareness of security needs was raised after the involvement of the participants. In this case study there is also evidence that involving stakeholders in this process has raised awareness. In the words of one participant who was paraphrasing a politician at the time:

> *"... what [AEGIS] is doing is reducing the unknown unknowns and converting the unknown unknowns into known unknowns."*

Essentially, the AEGIS process allowed the participants to become more aware of security issues which had previously not been known about.

4.5.5.4 Stakeholder Discussion

A further benefit of involving stakeholders in the design process of security was identified when these stakeholders started discussions amongst themselves. These led to

the identification of various issues ranging from security to organisational or even functional that they had not necessarily been aware of. The following quote gives a good illustration of this, where B identifies a functional problem with the current architecture's searching mechanism:

> *"B: I'm wondering about integrity of searches in progress, especially they're a pipeline because you've potentially got the query being distributed out to several providers simultaneously and if then that synchronisation between them gets messed up then they can't get back together and they can't identify the consumer to return the information to.*
>
> *A: Yes*
>
> *B: Integrity of distributed state is ...*
>
> *C: once your request is ... this ... then it can be fielded out to many providers, they're asynchronous all of these requests, and they have to resynchronise.*
>
> *B: I'm thinking about requests where there's an analysis component*
>
> *C: why should you not be able to? You've got to be able to have some token, even if the token does not identify the original person with the request it's got to identify the broker that the request came from, so it should at least be able to get back to the broker saying here I am I've got what you asked for.*
>
> *B: but I thought the technique was direct communication back to the client?*
>
> *C: That's what we said isn't it..."*

4.5.5.5 Stakeholder Conflict

This category serves to illustrate the cases when different stakeholders do not agree. For instance, this can be seen when the need to provide confidentiality was stated as necessary for some Japanese data providers, but American data providers wanted a policy of open access.

In order to move on from this, it becomes necessary to reach an understanding with the two stakeholders regarding the disagreement. Either the disagreement arises out of incomplete knowledge (such as ignorance of a particular threat, for example) or it arises out of genuinely differing but equally valid points of view.

If these conflicts occur within the workshop, they can usually be resolved after both sides have argued their position, and possible solutions can then be explored. In this particular example, a possible solution is to avoid hosting the Japanese data at other institutions, and another is to refrain from making the data available to EGSO until it stops being confidential.

4.5.6 Grounded Theory Summary

Category	Details	Description
Responsibility	Liability	Legal or implied responsibility for security
	Boundary of Responsibility and Control	Problems of responsibility for security with limited control of the environment
	Stakeholder Security Requirement	Stated need for security
	Diffusion of Responsibility	Degree to which security is assumed to be taken care of by someone else
Motivation	Responsibility	The importance of responsibility in the motivation for security
	Liability	Consequences of liability for security and their motivating ability
	Reputation	The importance of reputation in the motivation for security
	Trust	Security can be used for other purposes such as building trust
	Stakeholder Culture	Depending on the stakeholder culture, security may or may not be actively pursued
Communication	Confusion	Talking about security can be confusing
	Scenario	The most widespread communication tool for security is the scenario
	Anecdote	A common means of communicating security knowledge is through anecdotes
Stakeholders	Stakeholder Viewpoint	The impact that involving stakeholders can have on the design process
	Stakeholder Knowledge	The type of knowledge that stakeholders bring to the process and its value
	Security Awareness	Stakeholder security awareness
	Stakeholder Discussion	Discussions between stakeholders as a consequence of being involved. These can have providential consequences and lead to the discovery of previously unknown issues
	Stakeholder Conflict	When stakeholders disagree, it is interesting to identify if it is as a result of misunderstandings or whether it is as a consequence of genuine and valid differences of opinion.

Table 3: Summary of Grounded Theory Analysis

4.6 Chapter Summary

By involving stakeholders in the security analysis, AEGIS provided increased awareness of security in the participants, allowed them to identify a number of problems and issues with security themselves, and provided a wealth of information about the needs of stakeholders. This information was elicited and recorded in the asset model.

Identifying security requirements based on this technical and organisational model highlighted a number of issues, mainly in the area of policy documentation. The success at this level is an encouraging step towards bridging organisational and technical needs in the design of security.

As a consequence of identifying the difficulties of communication in the design process, the use of scenarios was shown to be effective as a means of eliciting information and providing a means of reasoning about security. Given that the notation of AEGIS is completely compatible with abuse cases, the further support for scenario documentation is an easy step.

Some discussions, however, had a tendency to stagnate, or focus on a single area to the detriment of others. This has led to the identification that whilst AEGIS has a good means of identifying the breadth of security requirements, there is a need for a means of analysing the breadth of security needs. One way of doing this is to have a facilitator or moderator included in the process whose role is to ensure that coverage is achieved, as opposed to being involved in the security analysis directly.

5 Empirical Research: CLEF Case Study

5.1 Introduction

The Clinical e-Science Framework (CLEF) was the subject of the second study reported in this thesis. It should be noted this study took place concurrently with the EGSO case study (see section 4), and therefore some of the issues identified in the previous section (such as the need for a moderator, or providing more support for risk analysis) were not implemented in this study.

A major distinguishing feature of this study, when compared to the EGSO case study, is that CLEF is a medical project and therefore has significantly different objectives and constraints. The most significant difference is the presence of legal and ethical requirements to ensure the confidentiality of patient information. Another difference is that the participants in this study consisted of the people who were responsible for ensuring the security of CLEF, whereas in the previous study the participants were not assigned this responsibility.

In section 5.2, details of the study are presented, followed in section 5.3 by the description of the Action Research undertaken. The methodological contributions of this research are presented in section 5.4. Finally the Grounded Theory analysis of the transcripts of this study presented in section 5.5, extend and refine the substantive insights gained from the previous study.

5.2 Description of study

5.2.1 What is CLEF?

"CLEF aims to develop rigorous generic methods for capturing and managing clinical information in patient care and for integrating that information into clinical and basic bioscience research. CLEF will focus on cancer, but the goal is to produce a robust framework which can be used in many areas of clinical medicine and research based on emerging knowledge management techniques within the E-Science/Grids programme." [38]

CLEF is a 3 year project funded by the MRC (Medical Research Council), commencing October 2002. The purpose of CLEF is to provide a framework through which clinical patient information can be accessed by medical researchers in order to conduct research. The *"capture, integration, and presentation of descriptive information is a major barrier to achieving such a framework. Clinical histories, radiology and pathology*

reports, annotations on genomic and image databases, technical literature and Web based resources all typically originate as text. Often they are dictated and then typed; alternatively they are laboriously coded or annotated manually, usually in incompatible formats that lack rigour and hence cannot be scaled up or aggregated effectively." [38]
Because of legal and ethical constraints placed on clinical research, one of the main areas of research for CLEF is in the areas of security, and how to preserve the confidentiality of patient information whilst achieving a useful research framework.

5.2.2 Details of the study

The case study consisted of an initial meeting with one project member (henceforth referred to as "A"), followed up with two workshops held at University College London on 30/04/2003 and 14/07/2003. The first workshop involved "A" and two researchers and lasted approximately two hours; the second workshop consisted of "A", a senior member of the project ("B"), two researchers and one independent security expert and lasted nearly three hours. The initial meeting was used to gain more information about the aims of the project, and the two workshops were recorded, transcribed and analysed using ATLAS.ti. The diagrams produced in this analysis consist of network diagrams refining on the categories identified in the previous study (for details of the diagram semantics see section 4.2.3).
The participants from the project both had extensive experience in the field of medical research and were also security experts responsible for the security in CLEF, although along slightly different lines. "A" was responsible for ensuring that the system as a whole was not open to abuse, whereas "B" was responsible for the automated clinical coding aspect of the project. The clinical coding refers to the process whereby the original data is coded in order to:

1. Standardise the format of the clinical data
2. Remove identifiable information from the data in order to safeguard patient confidentiality

Efforts were made to involve additional stakeholders, such as developers or users, however this proved to be impossible. This case study therefore also describes how AEGIS operates when used solely with security experts (without any additional stakeholders).

5.3 Application of AEGIS

5.3.1 Asset Identification

Once the AEGIS approach had been explained to the participants, the first stage in this study was to identify the major assets of CLEF. Using an existing diagram of the project, a high level description of the operation of the system was given. Based on this description, the participants were then encouraged to identify the main technical assets in the system, and the environment in which these operated.

In order to identify the location, role and tasks of people in the system, questions were raised based on the actions and events described in the system. For example, when data was transferred from one institution to another, the question was raised as to whether this was an automated process or a manual one. This type of questioning uncovered a number of people that played a part in the system and had not been mentioned, such as the administrators of the systems, the clinicians that provided the data, or the people in charge of supervising the process of anonymising the data.

5.3.2 Security Requirements Capture

Throughout the identification of the assets, the discussion extensively revolved around the legal and ethical need for protecting the confidentiality of patient records. This raised a question from the participants about how to model the confidentiality requirements of the project. The need for protecting the patient clinical *information* was described as somewhat different from protecting patient clinical *data*. Protecting the clinical *information* is necessary for preserving the *privacy* of patients, but does not have to rely on keeping the clinical *data* secret. This is because identifiers in the data could be removed, making the identification of a specific individual impossible.

This was resolved by deciding to model the clinical data asset and using the confidentiality requirement to represent the need for both *data* and *information* confidentiality. The reason for this was to avoid modelling an entity that is neither physical nor transactional, but a property of the asset. This is similar to the approach taken with EGSO where reputation was not modelled as an asset, but represented in the importance of the security ratings of assets that could affect it.

The need for ensuring the confidentiality of patient records was a direct consequence of both legal and ethical constraints placed on the project. As such, most of the discussions centred on the procedural and technical means that were available to CLEF to achieve

this. When asked to rate the importance of the confidentiality of the clinical data, the answer was immediately "essential".

Other security requirements were also identified, such as availability and integrity needs. With the exception of the customer serving sections of CLEF, availability as a whole was not judged to be particularly important, simply because most of the processing would be occurring in batch jobs, and this was therefore rated as being low.

> *"A: In terms of availability it would definitely be batch based. Certainly not – a long way from real time. It may be a once a week, once a month extract. Probably once a week to keep numbers down."*

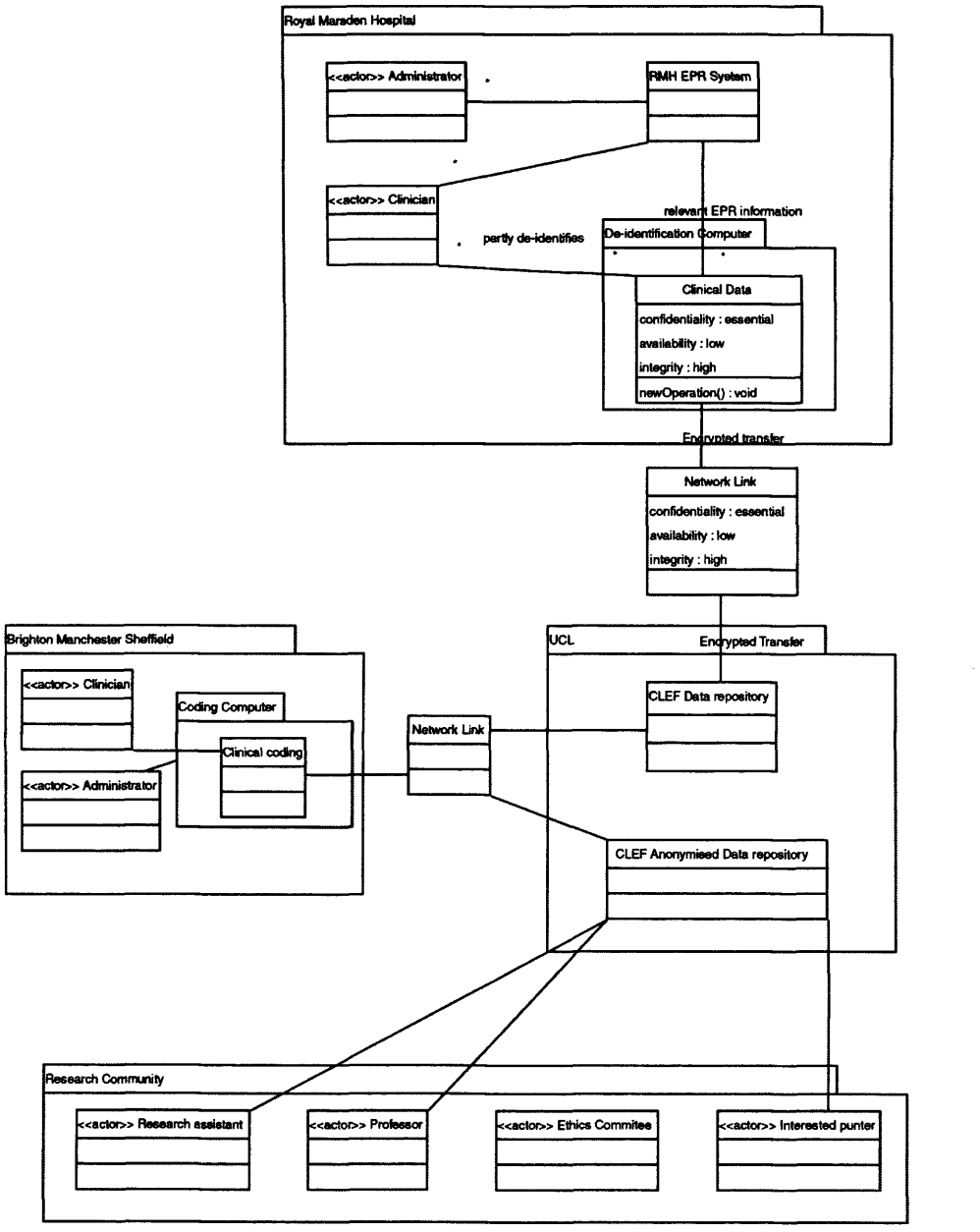


Figure 10: Asset and Security Requirement Model of CLEF

Integrity, however, had to be maintained in order to ensure that the data could be used to support meaningful research. When asked to rate the integrity need for data, the answer was:

> *"A: umm. Well my instinct is to say high, because if someone could interfere with it then it would invalidate the whole basis of the data set. But if someone fiddled with an individual record it wouldn't necessarily be material. So it's not as high as say a banking transaction where each transaction is of itself important. Some will be quite trivial in terms of the impact and others could be extremely significant."*

The reason why integrity was not rated as essential as confidentiality was because any research conducted on this data would use aggregated datasets, and research methods in which conclusions would have to be based on statistically significant results. Small changes in a small number of records might not affect that research. However, deliberate modifications that were intended to mislead and affect research could seriously affect the research. As a result the integrity of the data was rated as being high.

The full asset model, complete with the identified security requirements can be seen in Figure 10.

5.3.3 Risk Analysis

As a result of the very strong needs for ensuring confidentiality in the system, prior to the study, the threats to the system had mainly been considered in areas where they would compromise the confidentiality of the clinical data. In terms of security, it was quickly identified that the clinical data was the most crucial asset of the system and as a consequence most of the risk analysis focussed on the threats and vulnerabilities that could compromise this data.

The main threat to confidentiality was seen as a third party attempting to gather personal clinical information about an individual. This type of threat could result in a successful attack if the system was either vulnerable to *unauthorised access* to directly identifiable data, or an inferential attack on the anonymous data provided by the system.

One area of concern that was raised was for the need for integrity in the data. The anonymisation process described by CLEF consisted of removing what they called directly identifiable information from the data, names, addresses and dates of birth of patients. The anonymisation process also consisted of coding the medical information held in the patient records into a standard format, further reducing the possible existence of directly identifiable data. The concern that was raised was related to how much this

process of changing the records affected the integrity of that data, bearing in mind that the ultimate purpose of the data was to support clinical research.

Since the project was designed to research issues such as this, the participants were not able to quantify the impact of this process; however they were able to present three reasoned arguments:

1. Clinical research cannot be conducted on data without first coding it in some fashion. Whilst the clinical coding does change the data, it is necessary to support any research.
2. This would only be critical if the data changed to the point where the conclusions drawn from its analysis would be different from those conducted on the identifiable data. It was argued that research methodologies allowed for some degree of variation in the data, and that therefore the changes to the data would be taken into account.
3. The purpose of CLEF is to provide a means of informing new clinical studies, as opposed to replacing them. Therefore the data provided by CLEF is intended as a means of facilitating the identification and specification of new research.

5.3.4 Security Design

Prior to this study, CLEF had already developed a policy and system design to address the legal needs of data protection legislation and the ethical concerns of the ethics committees. The policy and architecture were formulated to provide clinical research with a means of operating legally, ethically and securely. The policy described a number of steps designed to:

- Anonymise the patient records.
- Isolate the various parts of the system from each other, in order to prevent the identification of the records in case of unauthorised access.
- Restrict access to the data to individuals on projects that have Ethics Committee approval.
- Restrict the type of data available for research:
    - Generally, only return aggregated results, except in special circumstances.
    - Only grant access to the type of information relevant to the study.
- Prevent individuals who work on more than one authorised project from knowingly accessing the same patient record.

Having reviewed this policy and the associated architecture, together with the security requirements and the asset model, the question of whether this was a cost-beneficial security solution to the problem was raised. The cost of implementing this policy was judged to be rather high. This was based on a judgement of the costs of training, enforcing and monitoring all the different aspects of this complex policy. In addition the potential for compromising the integrity of the data systemically (as opposed to providing data without bias) was raised as a potential cost to the system. Finally the need for both anonymising the clinical data *and* restricting access to authorised parties (either of which is sufficient to satisfy the needs of patient confidentiality) was questioned as being particularly costly for little apparent gains.

In the discussion about the costs and benefits of this architecture, the participants made the point that in order to comply with ethical regulations, either patient data must be demonstrably anonymous, or patient consent must be given in order for identifiable personal information to be used in a clinical study.

> *"The thing is that when an ethics committee gives approval to research, it normally does require consent of the data subjects and if it's very anonymous, and demonstrably anonymous, then fine it does not always require any consent, but it usually still has a fairly specific domain over which it's happening. I mean most research takes place in a given institution, etc."*

Securing patient consent for an online research framework such as this was described as much too complex to be feasible.

> *"The problem is getting a consent matrix that contains a precise enough specification of a person's wishes full disclosure and non-disclosure, to enable a particular research project to decide if it's inside or outside the scope of that individual's consent, to manage the future evolution of research queries and the domain of medicine as a whole. In order to have an anticipatory framework that withstood even a hundred years or even that person's lifetime is difficult enough. And then you've got the fact that you need an information system to record that against each person - much more complex than say an organ Donor Register – that has to be consented and consulted before any research analysis took place. And then you say: I want to do this on a quarter of a million patients please, how long might it take. And the answer is the query will take two minutes to run, getting the consent queries run across the main server (...) will take four days. (...)And then you know it's a skewed sample because you don't know about who's consented and which sectors of society give or don't give consent more readily."*

The need for providing anonymous patient medical information was therefore a consequence of ethical constraints placed upon clinical research in general. As well as being another regulatory requirement, the need for restricting access to approved projects was also a consequence of the lack of existing experience in this area and the result of needing to adopt a cautious strategy that would inform later security developments – which might include relaxing certain aspects of the security process.

"... we did feel that they needed to be from an authorised, recognised and registered party. Maybe we need them to actually sign up and be a registered person, as CLEF customer. Now that is on one level overkill to say the least, on the other hand very few detailed clinical repositories have been made available for queries outside of the institution that owns and protects them. Most of the time it's been made available in house. (...) we need to be very careful and to set ourselves up with a very high goal standard that would be quite tough. Because it's on that basis of confidence and that we can then knowingly relax certain parts of the process."

As another example of a cost benefit discussion in this study, the question was raised as to whether it would be necessary to be able to backtrack from the anonymised data back to the original data source, possibly as the consequence of relevant information discovered during research. By providing patients with the potential of directly benefiting from research carried out on their clinical information, this was seen as one way of improving the take-up and public acceptance of the system. The costs of developing this would initially be in the added complexity of ensuring the security of the backtracking mechanism, and also in the new risks inherent in adding the new mechanism – for example the potential for compromising the anonymity of patient records.

5.4 Action Research Summary

CLEF is a very different project from EGSO. Although they both share the description of being Grid projects, CLEF is both legally and ethically required to safeguard the high level of confidentiality and privacy of the data it provides, whereas EGSO has practically no confidentiality requirements and no legal or regulatory oversight.

A significant difference between the two case studies can be seen in the participants' difference of awareness and knowledge of security. Whereas the participants in the EGSO case study were developers and managers, the CLEF participants were security experts.

Although the ideal AEGIS process should involve a variety of stakeholders in the design process of security, applying it only with security experts has proven beneficial to the project. By explicitly modelling the people and environment in which CLEF operates, the organisational policies necessary for securing the system became clear. The operational complexities of the system also became more apparent. In addition to this, systematically looking at the assets of the system and their specific security requirements allowed the identification of a high need for integrity in the data, whereas most of the design and discussions about security in CLEF had revolved around confidentiality.

One of the main outcomes of the application of AEGIS in this case study was the cost benefit discussion of the proposed security against its stated requirements. In this case, the discussion mainly revolved around the complexity and cost of the proposed system design, when rated against the needs of researchers and the privacy requirements of the patients providing the data. Anderson [14], in the security review of the DeCODE Icelandic Health Database system, argued that there are two types of approaches that a system such as this can take. The first is for the system to hold potentially identifiable data, but restrict the people who can access it. The second is to anonymise the data and release it to the public.

The fact that CLEF proposed to combine both approaches led to raising the issues of cost and benefits. This cost benefit discussion allowed the participants to seriously question their design and justify it, and as such this process also served as a sanity check. This following quote illustrates this:

> *"I thought about that very hard, and I felt that actually although it was very helpful that you prodded me in the direction, I actually thought no I actually think he's wrong and I'm right, and that we are right."*

It is also interesting to note that the security requirements for privacy were not directly derived from patients, but instead imposed by legal precedent and ethics committees. This had the result of making the issue of being able to re-identify a patient (backtracking) complex. Given appropriate safeguards, patients might be interested in such a service as it would allow them to benefit directly from new research results. However, because of the need for complying with a perceived level of confidentiality, this facility was disregarded.

> *"A: I mean effectively, what we're going to do, we recognise that backtracking could be valuable in certain cases, in many ways in order to get it adopted we're saying we're not going to have backtracking, we won't have processes that allow backtracking, though technically, it might be possible. But as far as we're concerned we're not going to set up processes that are going to do that."*

Evidence of the success of the AEGIS process can be seen in the identification of the issues described above, as well as from feedback from the participants. This can best be seen in the fact that the researchers were subsequently invited to participate and apply AEGIS in the follow-up *CLEF Services* project. This project, also funded by the MRC, runs from January 2005 to December 2007. As seen from the project website [5], one of the key goals of the project is that *"CLEF Services addresses issues encountered in CLEF in managing privacy and security (...)"*.

5.5 Grounded Theory Analysis

5.5.1 Introduction

Building from the concepts identified in the analysis of the EGSO Case Study (see section 4.4), the Grounded Theory analysis of the transcripts of this case study provided a different look into the categories of *responsibility*, *motivation*, *communication* and *stakeholders*.

5.5.2 Responsibility

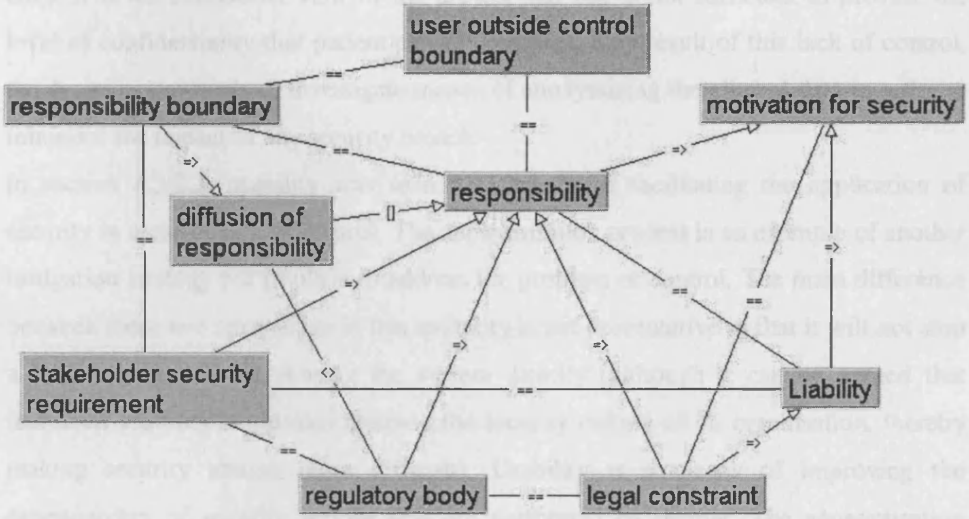


Figure 11: Responsibility Model - CLEF Case Study

In stark contrast to EGSO (see chapter 4), the responsibility for security in CLEF is particularly well addressed. The main factors behind this are that:

1. The project has a legal responsibility of ensuring that personal information is not misused, and the privacy of patients is protected.
2. In order to conduct any clinical research, such as that proposed by CLEF, it is necessary for proposals to be approved by ethics committees. Set up after the Second World War to ensure that clinical research be ethical and moral, these also address issues of patient privacy and the confidentiality of medical data. This overseeing body ensures that matters of security are being addressed prior to any projects can get approval.

The updated responsibility network model can be seen in Figure 11.

5.5.2.1 Boundaries of responsibility and control

An extension to the boundary of responsibility and control concept (see section 4.5.2.1), is that the purpose of CLEF is to address the need for conducting clinical research with modern tools and techniques, such as Grid technology. This means that the security control mechanisms that have existed so far have to be used or adapted to this new environment.

One of the basic premises of CLEF is that as a consequence of the distributed environment in which the project operates, controlling who has access to the data is not easy. It is the considered view of the project that this is not sufficient to provide the level of confidentiality that patient privacy requires. As a result of this lack of control, the decision was made to investigate means of anonymising the clinical data in order to minimise the impact of any security breach.

In section 4.5.2.2, usability was seen as a means of facilitating the application of security in areas of lack of control. The anonymisation process is an example of another mitigation strategy put in place to address the problem of control. The main difference between these two approaches is that usability is not preventative in that it will not stop a malicious user from abusing the system directly (although it can be argued that increased usability could also increase the security culture of an organisation, thereby making security abuses more difficult). Usability is a means of improving the dependability of security actions that are performed by people. The anonymisation process is a more preventative solution, which aims to make it impossible for individuals to be directly identified from the data, and also aims to make it as hard as possible for individuals to be inferentially identified from the data.

5.5.2.2 Diffusion of Responsibility

The issues of *diffusion of responsibility* identified at the project level in the EGSO case study are conspicuous in their absence here. It is important to note that *diffusion of responsibility* occurs only in areas where responsibility has not been assigned. The legal and ethical frameworks have ensured that CLEF has taken responsibility for security; therefore it is not surprising that there is no evidence of *diffusion of responsibility* at the project level.

Some of the evidence supporting *diffusion of responsibility* in the design of security in EGSO revolved around the need for specifying and documenting policy for the human

aspects of the system – in essence not taking responsibility for ensuring the human side of security.

In this case too there was some evidence that some of these organisational requirements had not been specified, such as identifying the importance of the system administrators or highlighting the role of clinicians in the anonymising process of the data. The anonymisation process was designed to enforce separation in between different data processing parts of the system to prevent information from being re-identified. Whilst on a technical level this had been well covered, identifying the people in the system showed the need for documenting policies needed to ensure the separation at an organisational level.

5.5.3 Motivation

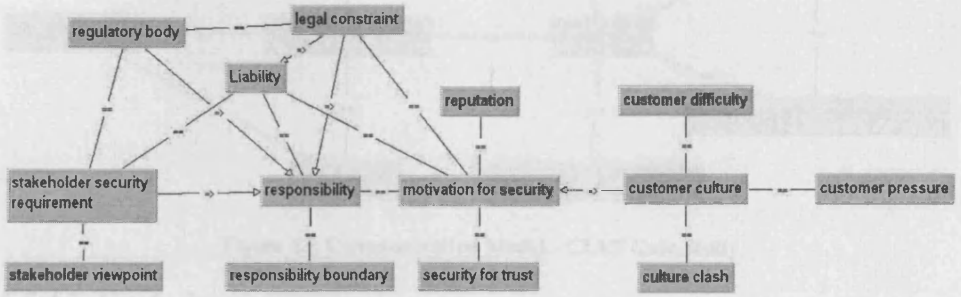


Figure 12: Motivation Model - CLEF Case Study

Motivation to achieve a secure design is again very different from EGSO. In this case, the main distinguishing features of motivation could be tied down to CLEF taking responsibility for achieving security as a consequence of legal requirements and regulatory oversight.

The existence of a regulatory body overseeing the approval for medical projects, thus ensuring that security needs are met is particularly important. This was readily apparent from the participants:

"A: you raise the question about are steps four, six and eight [of the anonymisation process] necessary. The answer is probably not, but the fact is if we don't have them there will never get the approval so nothing will be done."

Another significant factor is that both the participants were experts in security, and hence their main role and responsibility in the project was to achieve security. Finally, a major aspect of the project was to develop a means of securing patient records in such a manner that research could be carried out without compromising the confidentiality of

the data. As a result it is unsurprising that the motivation to achieve security was very high. The updated motivation model is shown in Figure 12.

5.5.4 Communication

The factors identified in the communication category are shown in Figure 13, and the most significant are described in the following section.

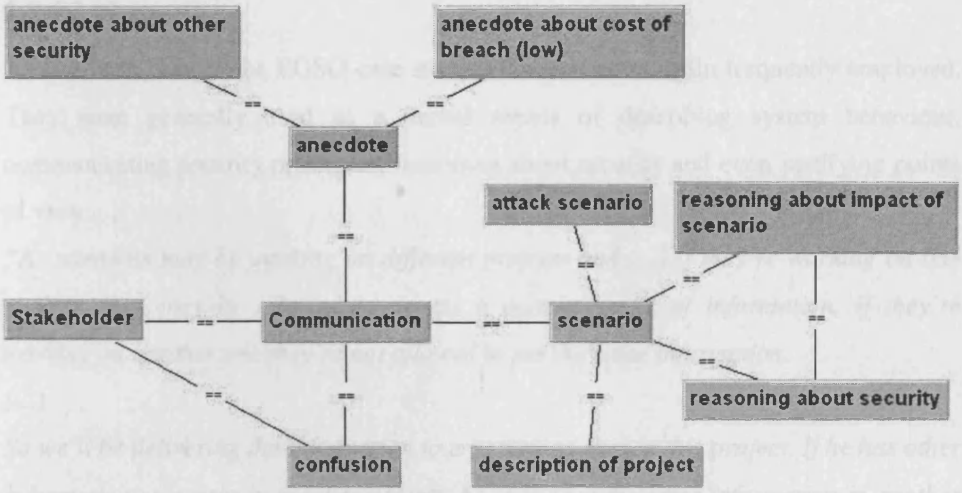


Figure 13: Communication Model - CLEF Case Study

5.5.4.1 Confusion

Confusion was not apparent in this case study. Both participants already had an extensive knowledge of the terms of confidentiality, integrity and availability. Confidentiality was the most important concept to address for CLEF, insofar as the purpose of the project was to provide a means of conducting research whilst protecting the confidentiality of patient records. When discussing this, it became apparent that the exact meaning of confidentiality as related to patient records was not only linked to the actual data, but more accurately to the information that could be determined from that data.

"A: What I was wondering whether to add in here, which is more the confidentiality side, it's almost a question of you've got the data and you've got the meaning, the semantic content. And even within that we'd probably for confidentiality have to say we've got the semantic content in the sense that it's not specific to a person and the semantic content that is specific to a person. And even within that we'd probably have to distinguish between those things that are reasonably unique to a person, I'm including by that things like NHS number or a person's name which we consider in a

social context is unique for someone, but there are quite a lot of John Smiths, so it's not as unique if you like, but in legal terms that is the person's true identifier."

As a result of this clarification, the discussions about the confidentiality of the patient medical records distinguished between the need for protecting the actual data, and the need for anonymising the data as a means of reducing the identifiable information.

5.5.4.2 Scenario

As had been seen in the EGSO case study, scenarios were again frequently employed. They were generally used as a verbal means of describing system behaviour, communicating security principles, reasoning about security and even justifying points of view.

"A: scientists may be working on different projects and (...) if they're working on one project, they may be allowed to access a certain range of information, if they're working on another one they're not allowed to see the same information.

(...)

So we'll be delivering the information to a person as user in this project. If he has other information as a user in another project, he might get the same information in another project but it will be a different pseudonymous identifier. So they can't link the two sets of data. Because they might be doing one which says 'yes they can get information about sexual conditions' or something and there might be another one where there might be more information about geography. The idea is they can't link the two sets of data together to start saying 'now I've got a person of this age in this area with this sort of condition', people might start saying 'well there aren't many of those, are there!' So that's one thing we, for our purposes, would have to drop in there."

Given that it is essential for different stakeholders to communicate during the design process, it is useful to note the predominant use of scenarios as a tool for communicating about security.

5.5.4.3 Anecdote

Anecdotes were again used by participants in this study to describe and justify security properties. Although similar to scenarios, anecdotes differ in that they claim a basis in fact, as opposed to scenarios that merely describe theoretical possibilities. This is to say that anecdotes can be used as a means of justifying a particular point by recounting a relevant event that is claimed to have happened.

"A: I think it was in the DeCODE database of the Icelandic people. I gather the prime minister was included and if you knew that if in he 1977 fell down some step and broke his leg (because it was on the news) and he had a bad bout of flu at some other time...
R: you could find the record...
A: you could narrow it down to enough people and could guess which one it was."

From the review of the literature (see section 2.5.1) about computer security knowledge, other types of security knowledge exist (advisories, risk measurements or security patterns). It is interesting to note, however, that none were presented in this study as a means of quantifying or qualifying threats to the system. This illustrates one of the fundamental difficulties of security, which is the difficulty of gaining accurate, relevant and succinct knowledge which can then inform the security design process. In the absence of this, expert knowledge and experience, presented in the form of anecdotes and scenarios, remains the most accurate and practical source of security knowledge.

5.5.5 Stakeholders

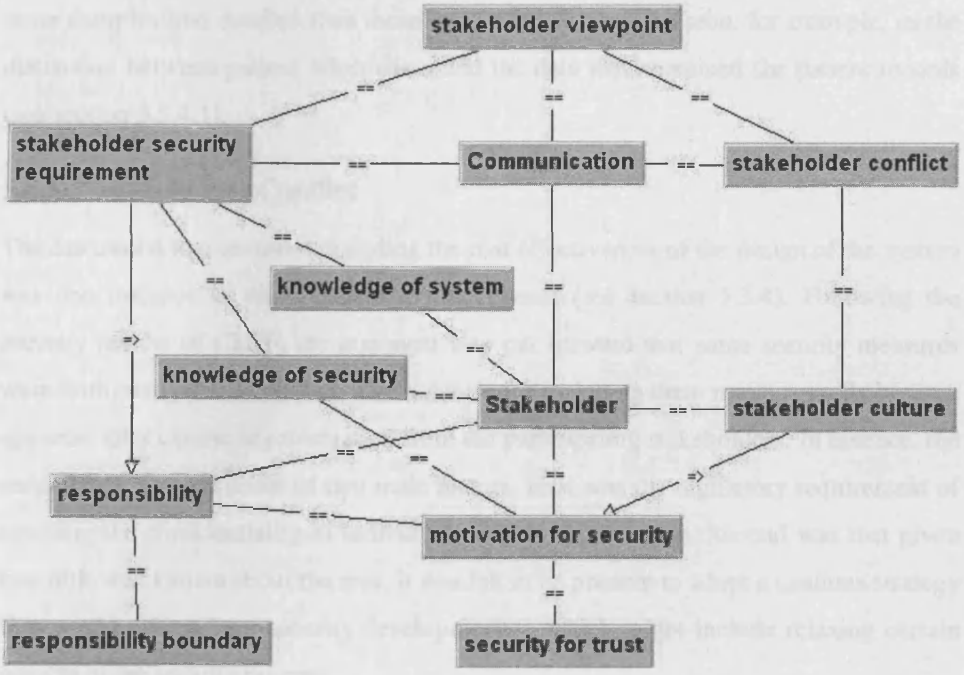


Figure 14: Stakeholder Model - CLEF Case Study

5.5.5.1 Stakeholder Viewpoint

This study only involved two stakeholders in the actual workshops. Both of these stakeholders had extensive knowledge of the legal and ethical constraints surrounding

the field of medical research. As a result of this, the participants were able to represent data protection officers, and ethics committee viewpoints. These were used to justify the vast majority of the security requirements in CLEF.

Efforts were also made during the workshops to ensure that the patient point of view was represented based on limited role-playing from the researchers. This approach resulted in the discussion to allow the re-identification of patient information as a means of improving take-up and acceptance (see section 5.3.4).

Whilst useful results were gained from the application of AEGIS with two security experts, a greater number and diversity of stakeholders and stakeholder viewpoints would be preferable and might have yielded additional findings.

5.5.5.2 Stakeholder Knowledge

Given that both stakeholders were experts in security and centrally involved in the development of the system, their security knowledge and system knowledge was very extensive. This is a significant difference when comparing the CLEF study to the EGSO study. As a result of this, many of the security discussions in the CLEF case study were more complex and detailed than those in EGSO. This can be seen, for example, in the distinction between patient information and the data that contained the patient records (see section 5.5.4.1).

5.5.5.3 Stakeholder Conflict

The discussion that occurred regarding the cost effectiveness of the design of the system was one instance of disagreement in the process (see section 5.3.4). Following the security review of CLEF, the argument was put forward that some security measures were both costly and redundant. The rationale for including these measures only became apparent after careful argumentation from the participating stakeholders. In essence, the redundancy was the result of two main factors. First was the regulatory requirement of ensuring the confidentiality of individual patients' information. Second was that given that little was known about the area, it was felt to be prudent to adopt a cautious strategy that would inform later security developments – which might include relaxing certain aspects of the security process.

As a result of this initial disagreement, the following discussion allowed the participating stakeholders to seriously question their design and justify it:

> *"I thought about that very hard, and I felt that actually although it was very helpful that you prodded me in the direction, I actually thought no I actually think he's wrong and I'm right, and that we are right."*

5.5.6 Grounded Theory Summary

A number of differences between EGSO and CLEF have led to the refinement of concepts identified during the EGSO analysis.

In the Responsibility category, a number of additional factors became apparent, such as the very clear assignment of responsibility as a result of the involvement of a regulatory framework in ensuring that security was taken care of. This was due to the medical domain of the project, which had a legal requirement to ensure that the security of patient records was ensured

The category of Motivation was informed by three factors. First was the legal requirement to ensure that security was addressed. Second was the fact that both participants were security experts and therefore already motivated to address security. Third was the fact that a significant part of the CLEF project was specifically dedicated to addressing matters of security.

In the Communication category, many more examples of anecdotes and scenarios were uncovered. Given that both participants were already knowledgeable in security matters, there was little evidence of confusion.

Finally, in the Stakeholder category, some additional insight was gained into stakeholder conflicts, and their role as a sanity check in security design.

5.6 Chapter Summary

In this second case study, the Action Research application of AEGIS to the CLEF project has been presented. Although only two project members participated in this study, they were able to present several different stakeholder viewpoints. The outcome of this was the identification of several issues, such as the usefulness of being able to re-identify individual patients from their anonymised records, the high cost of maintaining the security of CLEF in its current design, or the potential integrity problems of modifying the research data.

One of the most significant endorsements of AEGIS was the subsequent invitation join the follow-up CLEF Services project and apply the analysis from the start.

Based on the transcripts of the workshops in which AEGIS was applied, a Grounded Theory analysis was conducted. The core categories of Motivation, Responsibility, Communication and Stakeholders, identified during the EGSO Grounded Theory analysis, were further refined and validated as a result of this study.

6 Empirical Research: BioSimGrid & DCOCE Case Studies

6.1 Introduction

The experimental approach used in the following two case studies differs from the previous research. As part of Oxford University's Software Engineering Programme course on people and security, AEGIS was first taught to part-time graduate students who in turn conducted a short security analysis using the methodology on real-world projects. Since the analyses were conducted by people who were new to the methodology, these two case studies provide a more objective record of the application of AEGIS, and provide an alternative insight and means of evaluating the process.

6.2 Case Studies

6.2.1 Case Study 1: BioSimGrid

In this case study, the principles and processes of AEGIS were taught on 18/10/2004 to a group of six software engineering graduate students from Oxford University in a two-hour session. The basic principles of AEGIS were explained through a series of slides, as well as a sample asset model. Once this introduction was completed, the student-analysts were given a manual and access to two members of the biological Grid project BioSimGrid. The documented study therefore involved six graduate student-analysts, two stakeholders of BioSimGrid and one researcher, who assisted but did not conduct the exercise.

6.2.1.1 Project Description

BioSimGrid is a biological simulation project funded by the Biotechnology and Biological Sciences Research Council (BBSRC) and the Department of Trade and Industry (DTI). From the project website: [3] *"The aim of the BioSimGrid project is to make the results of large-scale computer simulations of biomolecules more accessible to the biological community. Such simulations of the motions of proteins are a key component in understanding how the structure of a protein is related to its dynamic function."* Since running these simulations is computationally expensive, they are currently performed by individual laboratories that have the resources to conduct this research. The purpose of this project is therefore to provide a data Grid of different simulations so that users will have a single point of access to this information.

6.2.1.2 Starting the Process

The two members of the project were able to represent both a developer and a system user point of view and their participation was secured for two hours and thirty minutes (although the user unfortunately had to leave after one hour). The student-analysts were given the tasks of identifying the security needs of the project and conducting as much of a security analysis as possible within the timeframe.

Initial questions from the student-analysts focussed on understanding what the project was about and how the basic architecture functioned. The Grid project was described as providing a group of universities a means of centralising access to different *"simulations of molecules of biological importance"*.

It was quickly identified that the project was expected to provide a secure environment for these different universities to operate in. In addition to this, there were long-term plans to expand the system to private sector pharmaceutical companies. The need to provide a secure environment was further reinforced by the fact that the private sector had very high confidentiality requirements, to the extent that *"it's really hard to convince them [pharmaceutical companies] to share their data with anybody – to even go outside of their own building"*.

Although academic use and provision of simulation data was free of confidentiality constraints, the private sector had very high requirements of confidentiality for their own simulations. A long-term aim of the project was therefore to provide their software to these private companies so they could federate their own databases in a compatible format and query the union of the private and public databases, but not allow queries from outside access to their own simulations.

The importance of the biological simulation data, also called trajectories, was further identified through the following questions *"would you place a high cost on producing the data? The manpower and equipment involved..."*, *"would the R&D of other pharmaceutical companies be interested [in this data]?"* Both answers were positive and showed that producing the data was expensive and the simulations could be very valuable to third parties.

6.2.1.3 Modelling the system

At this point, the student-analysts tended to want to focus on the specific security needs of the confidentiality of the database of simulations. After a quick reminder that the analysis should start by identifying all the assets and various stakeholder operatives, the

student-analysts started building an asset model later formalised as can be seen in Figure 15.

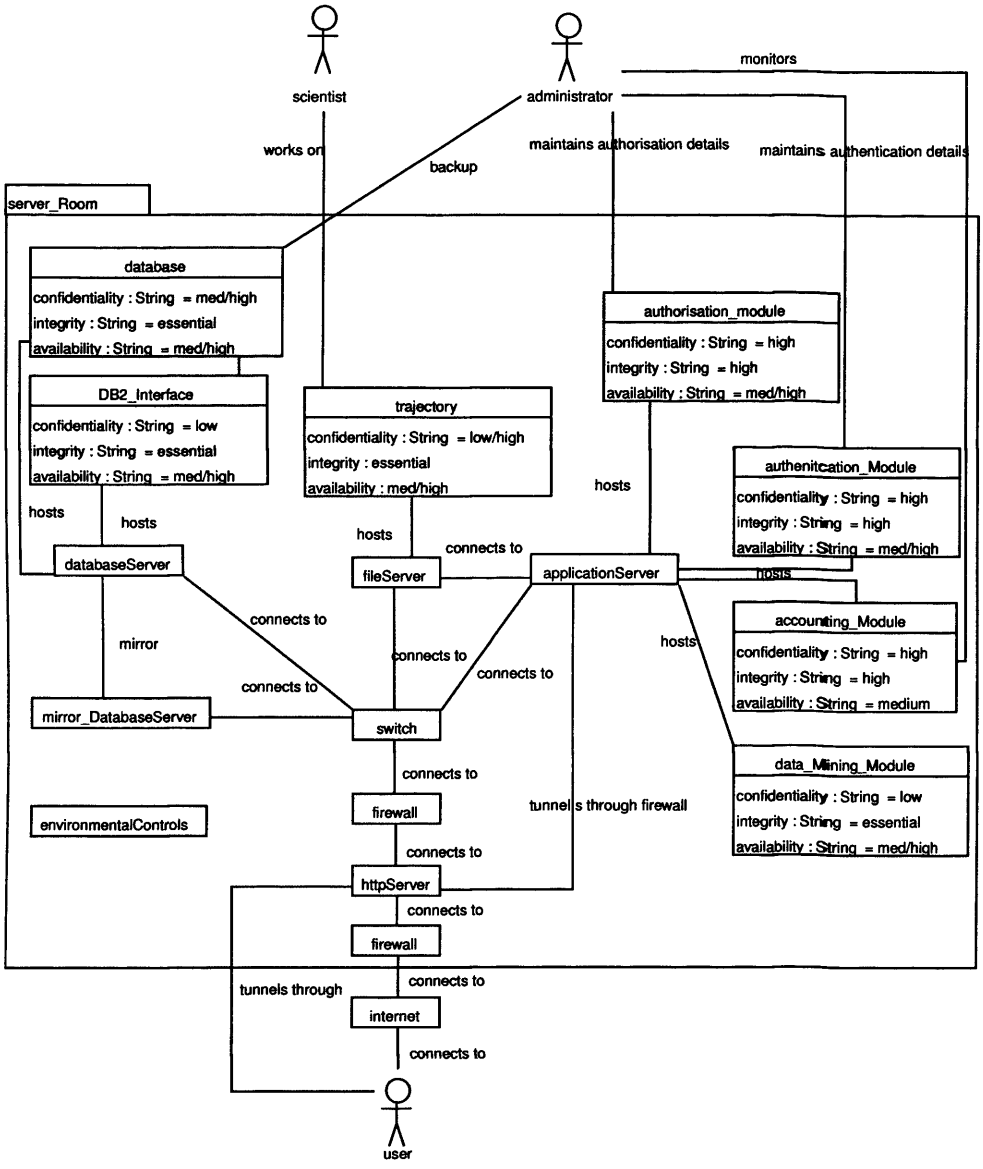


Figure 15: BioSimGrid Asset and Security Requirement Model

The Grid project representatives initially had trouble understanding what was required of them - *"what do you mean by* asset*?"* - although the student-analysts were quickly able to explain and lead them through an analysis. The modelling process consisted of one student-analyst drawing the asset model onto a whiteboard while the group of student-analysts as a whole asked detailed questions about the architecture that would inform the diagram.

For example, having identified that the project was geared to providing users with simulation data, the student-analysts then asked questions about how this data was

served to the user, what kind of server it resided on, where the server was housed, and so on. This in turn led to the identification of a number of other assets, such as the application server which provided authentication, authorisation and accountability services. Apart from the basic user, operatives were identified by asking leading questions, such as *"who is in charge of maintaining the system", "who has access to the server room", or "who supplies the information in the database"*. These operatives were then modelled as shown in Figure 15. The interactions between the operatives and the assets were identified throughout the process of building the model, such as the administrative task of maintaining the authentication mechanism which became apparent when the student-analysts identified the existence of that asset.

One finding was that many of the administrative duties in the system, such as backup, patching, maintenance of the authentication mechanism (in this case based on SSL digital certificates, and a username and password combination for users who do not have certificates), and maintenance of the authorisation mechanism (role-based access control) were not initially apparent. Identifying these required detailed and probing questions, for example when the representatives mentioned that the system was backed up (*"who backs the system up? Is there a policy for when and what to backup?"*), or that digital certificates were used to authenticate users (*"How do users get a certificate?", "Who do they apply to for access to the system?"*). What is interesting is that simply establishing that an administrator has to monitor, backup, and maintain the system – with little to no supervision or help – throws up a number of questions with regards to both the scalability of the system (can the tasks expected of the administrator be extended to cover one or two orders of magnitude more users?) and the effectiveness of the current system security (which in the absence of training, audit and documented policies is wholly dependent on the competence of the administrator – not on the technical countermeasures).

6.2.1.4 Identifying Security Requirements

Building on the discussion at the start of the analysis, student-analysts tried to get the representative to rate the confidentiality requirement of the trajectory files (simulation data). *"How important is it for you to be able to keep this secret?"* to which the answer was *"we have no need for confidentiality... At the moment."* When questioned further, *"from an academic user point of view, using your own words, how would you rate, how important would confidentiality be? Would it be low, unimportant, high, essential..."*

The answer was that the requirement for confidentiality was low, however from the *pharmaceutical company's* point of view, the requirement for confidentiality was deemed to be medium to high in some cases. However since the system did not currently involve pharmaceutical companies, the current requirement was originally judged to be low.

From a requirements point of view, capturing this information is important. On the one hand, the system *as it is* does not require that particular type of security, on the other, the system *as envisioned in a future development* may have a high requirement for this kind of security. Furthermore this also illustrates the need to identify and represent as many stakeholders in the system as possible to identify potentially conflicting viewpoints. As can be seen in Figure 15, the confidentiality requirement for trajectory was therefore rated as "low/high", which highlights this basic conflict.

Identifying the security requirements of other assets did not highlight any further conflicts, and it was quickly established that the integrity of the trajectory data was the most important security requirement of the system. This was because the whole purpose of the system was to provide accurate data. As a result of the dependencies between the trajectory data and database, the database was also judged to have an equal need for integrity. The availability of the data was not judged to be very important in the short-term, but in a future commercial application this would be more important. This was further justified by the fact that the project had already designed server mirrors in the architecture of the system.

Another series of assets that proved to be of interest were the authentication, authorisation and accounting modules. Although they were originally expected to resolve security issues in the system, the identification of high integrity and confidentiality requirements showed that they also raise security issues. This is one of the key strengths of AEGIS because it takes the point of view that every asset, including the security measures, has security needs. These effectively highlight the need for good usability, training, incentives and enforcement for security measures that require the involvement of an operative. Without those, the requirements of the security measures may not be met.

6.2.1.5 BioSimGrid Case Study Summary

As shown in this case study, a number of issues were identified through AEGIS. First of these was that the security roles of operatives in a system are frequently overlooked, and

technical security mechanisms are generally assumed to solve a security problem. By identifying security requirements on security mechanisms, these new security problems were highlighted. Modelling the tasks that operatives must perform in the system also helped to highlight some of these problems.

Although the case study shows that some confusion existed at the beginning of the process, the participants quickly adopted the method, and in a relatively short period of time new issues and requirements were identified. This importance of a moderator was also highlighted in this study as it was easy to get sidetracked on a particular area, whilst ignoring a multitude of other problems.

A final point concerns the resolution of conflicting requirements as seen in section 6.2.1.4. Different stakeholders in the system will have different points of view about what is important to them. This is typical of any reasonably large engineering project and establishes the need for making decisions based on conflicting data. With regards to security, it is very important to understand the need for a cost-benefit analysis of any security decision. The differences between the short and long-term security needs in the system do not necessarily have to cause serious difficulties. It is cheaper to compromise on a short-term implementation than it is to compromise on the long-term design. Any security mechanisms that have been designed but not implemented will be cheaper to implement at a later date than in a system where it is necessary to overhaul the original design.

6.2.2 Case Study 2: DCOCE

This second case study also operated on 18/10/2004 and involved six graduate students, one researcher and four stakeholders of the Digital Certificate Operation in a Complex Environment (DCOCE) project. Whilst the DCOCE project is not a Grid project per se, its main goal is to provide a middleware authentication service for online applications. As such it has many of the same characteristics of other Grid applications, such as a varied and distributed user base, the potential need for a decentralised management mechanism, or operating over unsecured networks such as the Internet.

6.2.2.1 Project Description

The DCOCE project is funded by the Joint Information Systems Committee (JISC) and is established to look at the use of digital certificates and public key infrastructures (PKI) within the complex environment of Oxford University, with its (semi-) autonomous colleges and departments. The emphasis for the project is to look into the

use of X.509 digital certificates for authentication to services. It should be noted here that the purpose of the system is to provide a security infrastructure on which other services to users can be supplied.

6.2.2.2 Starting the Process

In this case study, the four stakeholders of the project each represented a different point of view. These consisted of:

- The university point of view. The view of the organisation that owns and manages the project.
- A developer of DCOCE. The view from the principal developer of DCOCE
- A user of the system. The view from an academic who will use DCOCE as a means of pursuing research
- A data provider. The view of an organisation that provides access to its data based on the authentication of staff and students through DCOCE.

As with the BioSimGrid case study, the student-analysts were asked to identify the security needs of the project and conduct as much of a security analysis as possible within the two hours and thirty minute timeframe.

In this study, information was elicited by the student-analystss asking detailed questions of each stakeholder in turn. This started with an overall description of how the project had approached the development of the project, what the system was about and how it operated. As new issues were uncovered, the questions gradually became more specific. The questions were intended to identify the various stakeholders of the system, the different assets as seen by these different stakeholders, and what other operational parameters were – such as the need for usability for the academic users. The approach taken by the DCOCE project highlighted that issues of usability were of great concern, and it was keen to adopt a development approach that took stakeholder needs into account:

> *"The decision to look into PKI [was taken] and we're keen to go forward on that rightly or wrongly, and then we try and take it to our stakeholders and see how it fits and see what the needs are. So in a way we're kind of forced to go down the route of PKI, but we're trying to build something that will fit the stakeholders."*

Some of the challenges facing this project were identified from the university representative, such as:

> *"(The need to) have a system that is usable by my set of users at the university, that vary hugely in terms of their knowledge of IT and knowledge of security and*

knowledge of computing generally. (...) It needs to be scalable so you're issuing it to lots of new students that are arriving each year, so it's got to be scalable and dynamic and something that you can manage and keep on top of. Obviously it mustn't be too expensive, but that's part of the scalability issue."

In addition to the user group, another important stakeholder group and their need for security was identified by asking what the performance measures of the system were:

"Where the performance really comes in, in terms of you need something that is secure enough to be trusted by the data service providers, that they're going to have faith in it, that it's working, that we're doing it properly. So we have to allow our users to access those resources, and for the data service providers to have confidence that our whole system is not so insecure that anyone in the UK or the world cannot get hold of credentials to log onto their system."

Therefore the need to ensure security was essential for ensuring the trust of the data providers, and also the reputation of the university. A further complication with regards to managing the system was discovered when a student-analyst asked about the logistics of the subscriptions to data service providers:

"Student-analyst: with the actual subscriptions with the data service providers, is it done on a university basis, or is it done on a more 'by department basis', so some departments would have access to these resources whereas others wouldn't, or if one department has it, the others wouldn't?

University: That's a very good question actually. Within Oxford, if you want to have access, there's an agreement that you really should go through the systems and electronic resources department, so that these deals can be brokered like that. However one of the things that we were to find out was that there are some departments, there are some colleges that have got their own deals, so that was an interesting question for us. So yes they do do that."

6.2.2.3 Usability Requirements for Security

As had been stated, there was a need for ensuring that DCOCE provided usable security. The presence of a user in the case study provided a means of directly eliciting that stakeholder's point of view. When asked about current organisational policies for security, the user responded that the policies varied depending on location and who was managing the computers.

"Student-analyst: do you think that's a good thing, different policies, or would you like to see one policy that covers everything?
User: I'd like to see one policy that covers everything. Because I have to do different things according to where I am
Student-analyst: So is there a central place within the university which dictates policy for security, or is it up to the individual departments to set their own
User: it's up to the different departments and colleges"

Some usability information was uncovered by asking how much effort the user was willing to go to for the added security of the public key infrastructure of DCOCE:

"User:I don't want it to be any more difficult than it is already.

Student-analyst: and how difficult is it
User: Just username and password"

It became clear to the student-analysts that the priority of users is not security, but the ability to achieve their production tasks. This was further illustrated by the question:

"what sort of frequency of password change would you find acceptable?
User: oh, I never change my email password..."

Despite this lack of interest in actively pursuing security, the user was made to assess the impact that a malicious attack on their personal data would have:

"Student-analyst: so if somebody else got access to that data [user's research website] and changed it, would it have any impact on you or the work that you do?
User: it would probably affect my credibility
Student-analyst: Would that impact maybe on the university or the department or an outside agency
User: well if my credibility slips that means that other people don't want to work or collaborate with me on research."

The questioning approach prescribed by AEGIS and used by the student-analysts allowed the identification of a number of issues in the system, for example the lack of a centralised policy setting in security, a user group that was unmotivated to behave in a secure manner, the need to ensure that data providers could trust the security of the system.

6.2.2.4 Modelling the System and the Security Requirements

The approach taken by the student-analysts in this case study was to identify as much information from the stakeholders as possible with regards to assets, stakeholder needs, and security requirements. Armed with this information, the student-analysts build a partial asset model of the system, together with their security requirements. In addition to this, a sample threat and risk was evaluated that could compromise the confidentiality of the passphrase used to secure the digital certificate. This process was carried out in the absence of the stakeholders. The student-analysts then presented the asset model to the stakeholders and asked for comments and suggestions in order to ensure it was an accurate representation of the system. Given the short amount of time available in this case study the modelling, requirements capture, risk analysis and security design were intentionally incomplete in order to achieve as much coverage as possible.

It is very important to note that this is a departure from the approach described in the previous case studies where the elaboration of the model is achieved with the direct participation of the stakeholders. One of the arguments for adopting this approach with

regards to the asset model is that the stakeholders may not know or be interested in the modelling notation.

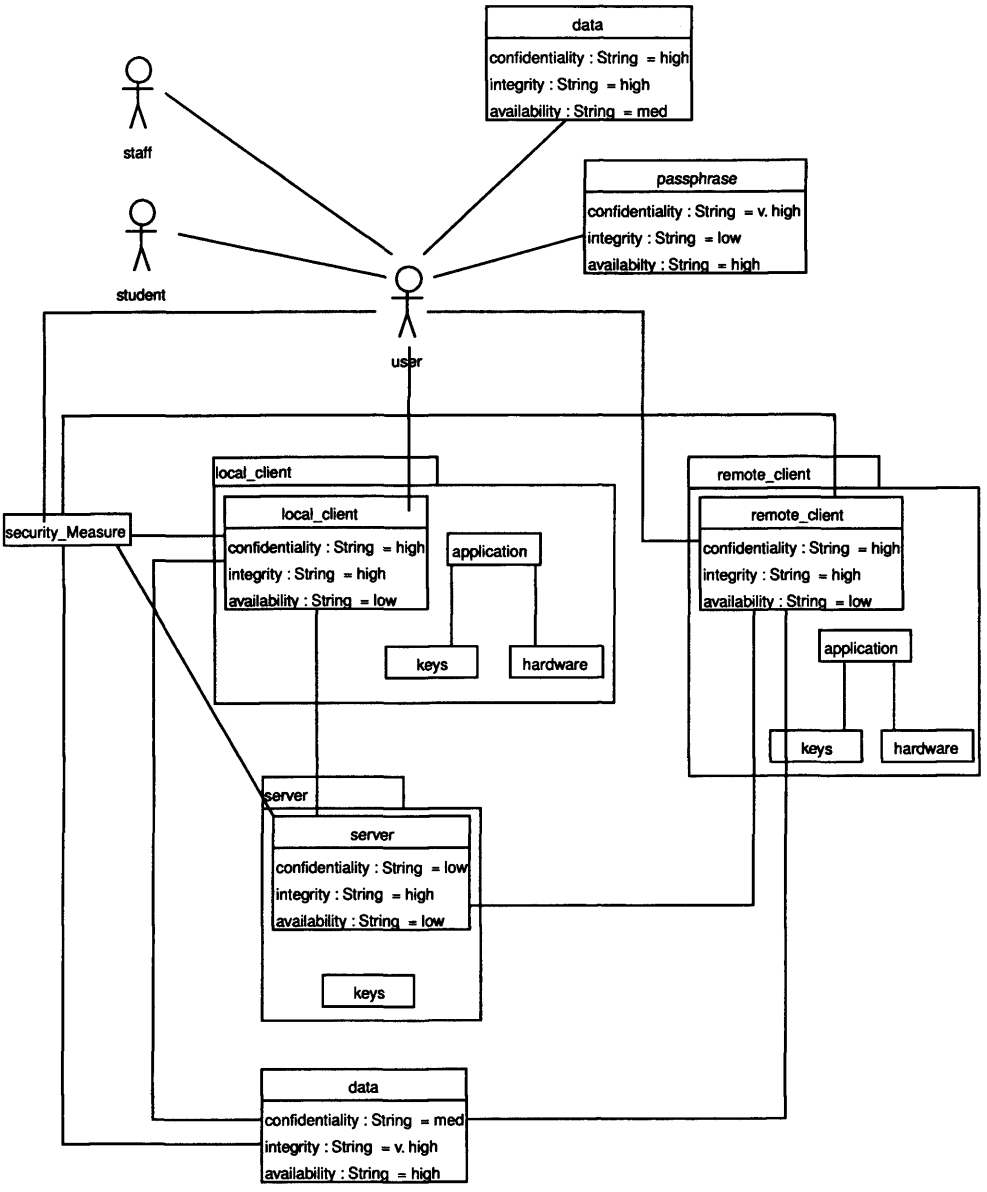


Figure 16: Partial DCOCE Asset and Security requirement model

Another argument in favour is that modelling the details of the system can be done outside of the workshop, saving time and energy during the face-to-face meetings. Should this be done, it is very important to ensure that the resulting asset model be made available, and explained, to the stakeholders. This is necessary for the stakeholders to understand the asset model and correct or inform any issues or mistakes that may appear.

Evaluating the security requirements of the system however should *not* be done without the direct involvement of the stakeholders. In this case, whilst an attempt in the absence of the stakeholders was made by the student-analysts to value these requirements based on previous comments, the security requirements were again elicited based directly from the asset model. This is particularly important in that one of the guiding principles of AEGIS is to involve stakeholders in the analysis of security, therefore isolating them from this phase would be antithetical.

Another problem that became apparent from this modelling attempt was that the student-analysts do not instantiate the assets to the specifics of the DCOCE study. For example, the service provider data was labelled data, and the user's data was also labelled data. Another example can be seen in the labelling of all the security measures 'security measure' as opposed to specifying each and every single one (such as server backups, or firewalls for example). Whilst this can be attributed to the inexperience of the student-analysts, it also highlighted the need for specifying the semantics of the notation as a means of standardising the models.

Once the asset model was presented to the participants, the process of identifying the security requirements took place. Parts of this became confusing for the student-analysts and stakeholders, and can be partly attributed to the inexperience of the student-analysts with AEGIS. For example the need for availability of the user's private key had been identified:

> *"Student-analyst: If the data wasn't available lets say tomorrow, how drastic would that be to your research and work?*
>
> *User: It wouldn't have a tremendous impact for that day, but I would want to be able to access it within a few days..."*

Due to the fact that the model of the asset contained two copies of the key, there already existed a certain amount of redundancy in the availability of these keys. In order to capture the original requirement of *"not much more than 24 hours"* without the key, and after a prolonged discussion, it was eventually agreed that for each of the copies of the key, the availability requirement was low, as a consequence of the design of the system which already provided a key retrieval mechanism. This, however, failed to capture the stakeholder requirement of availability of the key. Instead it captured the system design's point of view, which can be paraphrased as 'since there are two copies of the key, each copy has a reduced requirement for security'.

In cases where mechanisms in the system provide security, the requirements should still be modelled as though the security mechanism was not present or effective – in essence the rating of both the local and the server copy of key should reflect the 24 hour

requirement (probably a medium value since this is the value of the availability requirement for the user's data). The fact that the system provides multiple copies of the key should be seen as fulfilling that requirement.

Other examples of this confusion have occurred in other case studies, most notably in cases where data is encrypted and therefore rated as having a confidentiality rating of low. Another example was seen in this case when the confidentiality of a user passphrase was argued by the student-analysts to be higher in the case of a remote client. The point was made, however, that the need for confidentiality did not depend on the environment, only the risk of an attack. It was therefore agreed that the need for confidentiality of the passphrase was very high, and later in the risk analysis the differentiation was made between a user accessing the system from a remote client, and a user employing a local client.

Although the student-analysts had not had much experience with AEGIS, they clearly managed to understand the other concepts of the methodology. Their confusion highlighted the need to clarify the relationship between the value of an asset according to different security properties, the exposure of these assets to risks according to the different environments of the system, and the actual design of countermeasures as a means of reducing that exposure.

6.2.2.5 Sample Threats, Risks and Security Design

As mentioned previously, the aim of this study was to achieve as much of a security review as possible in two hours and thirty minutes. This restricted student-analysts to identifying one sample threat and risk to the system and briefly looking at the design of a security measure to address this.

Threat	Shoulder Surfer
Goal	Unauthorised access to data from the data service provider or the user
Target	User data, service provider data
Resources	Low – has access to an internet café where users access DCOCE through a remote client
Risk-Aversion	Med – does not want to be caught but because he's a shoulder surfer in an internet café, the chance of him being caught is probably slim, and the punishments are probably slim.

Table 4: Sample DCOCE Threat

Table 4 depicts the threat of a shoulder surfer wanting to gain unauthorised access to data from the service providers or the user.

Give the time constraints on the workshop, it was decided to focus on identifying a single risk. The risk described in Table 5 shows the assessment that was made of a shoulder surfer compromising the user's passphrase by observing the user in an internet café. The judgement of the stakeholders was that this was a serious risk to the system, based on the impact to both the user and the university, together with the estimation that it would be a reasonably likely scenario in practice.

Risk	Shoulder surfer accessing DCOCE through a compromised user passphrase
Threat	Shoulder surfer
Vulnerability	The user's passphrase being observed whilst it is being typed in an internet café
Likelihood	Medium to High – In a successful deployment of DCOCE, the value of accessing services and data through DCOCE would be known to attackers. Given the relative cheapness of the attack, this was judged to be a fairly likely scenario. *"Imagine this system is successful, it will be used a lot. So hopefully there will be enough people out there who do realise that it is valuable. And so in a successful situation the risk might be quite high, or medium high I'd say"*
Impact	The shoulder surfer can access the data provided by the data provider and also access, modify or delete the user's data. The reputation of the university, the credibility of the user's research could also be affected.

Table 5: Sample DCOCE Risk

As a result the analysis of this risk, a short discussion took place to identify potential countermeasures. The only solution that was proposed before the end of the workshop was to increase awareness of the problem amongst users through training and education.

6.2.3 DCOCE Case Study Summary

This study benefited from the inclusion of a wide variety of different stakeholders, who were each able to represent their individual viewpoints. For example, this resulted in the users of the system voicing their need for a simple and usable system (see section 6.2.2.3):

"User:I don't want it to be any more difficult than it is already.

Student-analyst: and how difficult is it
User: Just username and password"

Evidence was also uncovered that the users were not necessarily committed to following existing security policies:

"User: oh, I never change my email password..."

Other needs were voiced, such as the need to allow specific departments within the university to agree individual deals with data service providers (see section 6.2.2.2). The administrative repercussions of allowing this level of management in DCOCE were clearly an issue that needed addressing.

Whilst the model of the system should have instantiated the different assets to represent specific parts of the DCOCE, the model was still useful in grounding discussions about security into the assets of the system. This had the benefit of providing a large group of eleven people with a clear means of communicating about specific security issues in the system. As a result, and whilst there was evidence of confusion about the meaning of security properties among both the project stakeholders and the student-analysts (see section 6.2.2.4), the depiction and understanding of the system itself was not problematic.

The sample risk analysis conducted in this study also identified the problem of having a shoulder surfer compromise a user's passphrase if used in an internet café. This was used to drive a brief discussion with the user in order to identify means of addressing the problem, with the outcome that user awareness of security was an issue that had to be raised, possibly through training or education.

6.3 Action Research Summary

The two studies are useful in validating the approach as a socio-technical secure system design process. A significant difference between these two studies and the first two was that they were conducted by a third party. Whilst researchers were still present and did get involved, the main body of the analysis was conducted by student-analysts who had only been taught the AEGIS process in a two-hour session prior to the workshop. And whilst mistakes and misunderstandings did occur, the overall process allowed the identification of issues that had not been considered by the projects beforehand. Given the brevity of the involvement, this further validates AEGIS as a reasonably simple and effective means of addressing socio-technical security design.

In both these studies, a moderator role was assumed by the researchers as a means of ensuring the smooth running of AEGIS, as had been recommended from the EGSO and

CLEF studies. This proved to be successful in ensuring that discussions did not become too polarised on single issues.

With regards to improving the process of AEGIS, the main issue that came out of these studies was the need to specify the semantics of the modelling technique. Whilst the overall brevity of the workshop may have been to blame, the DCOCE model did not instantiate any of the assets and simply modelled the system as data, security measure, and application. Furthermore, in order to be able to reuse the model and communicate accurate information about security, the existence of clear semantics is necessary.

6.4 Grounded Theory Analysis

During the coding of these studies using ATLAS.ti, no further insight was gained into the existing categories of Responsibility, Motivation, Communication and Stakeholder. As described in section 3.3.2, this is evidence that the existing categories have reached saturation point. The next phase in the analysis is selective coding in which the different concepts identified are interrelated in order to create an overall model. This model is presented in section 7.1.

6.5 Summary

In this chapter, two case studies in which AEGIS was applied by student-analysts to two different projects – BioSimGrid and DCOCE - were presented. One of the key benefits of using student-analysts to apply AEGIS was to gain a degree of objectivity about the design process in action. Whilst the time spent both teaching and applying AEGIS was limited, further evidence of some of the strengths of the process was gained from both studies. This included providing a structured means of communicating about security, the value of involving stakeholders in the security discussions, and using an asset model to model the environment and the security requirements of the system.

However, as can be seen in the approach from the DCOCE case study in which the decision was made to model generic assets, the semantics of the notation were also identified as being in need of specification.

7 Discussion

This chapter describes the Grounded Theory model of the factors that affect the socio-technical design of secure systems based on the results from the four cases studies.

A discussion of the strengths and weaknesses of AEGIS is then presented tying into the insights gained from the Grounded Theory model. This includes the success of providing a simple approach to socio-technical security design, the usefulness of the modelling process as a means of supporting communications and involving different stakeholders in the design process of security. Published documents describing the application of AEGIS in other projects (not included in this thesis) are also referenced in the discussion as further validation of the process.

Finally, the final version of the AEGIS process is presented which incorporates the lessons learned from the practical case studies.

7.1 Model of factors and issues in socio-technical security design

The model described in this section is the result of the Grounded Theory analysis of the transcripts of the case studies presented in this thesis. Following the principles of Grounded Theory, the transcripts were initially coded (open coding phase) using ATLAS.ti. With the help of this tool, these codes were then organised into categories (axial coding). The categories consisted of Motivation, Responsibility, Communication and Stakeholder, and were presented in detail in the EGSO and CLEF case studies (see sections 4.5 & 5.5). It was identified during the axial coding of the BioSimGrid and DCOCE studies that no further information was being added to the categories, they had reached saturation (see section 6.4).

The final stage of Grounded Theory analysis consists of organising these concepts around a central category, and creating a 'storyline' explaining the resulting theory. This storyline is presented here, and describes the fundamental factors that influence an interpretive socio-technical approach to security design. A model of these factors is presented in Figure 17. It should be noted that whilst these factors were identified from information gathered from the application of AEGIS, efforts were made to ignore the issues that were the direct result of AEGIS (for example modelling assets, or eliciting security requirements based on these assets). Instead the focus was put on identifying the general factors that *affected* the socio-technical security design process, thereby providing a better understanding of the more general act of designing security.

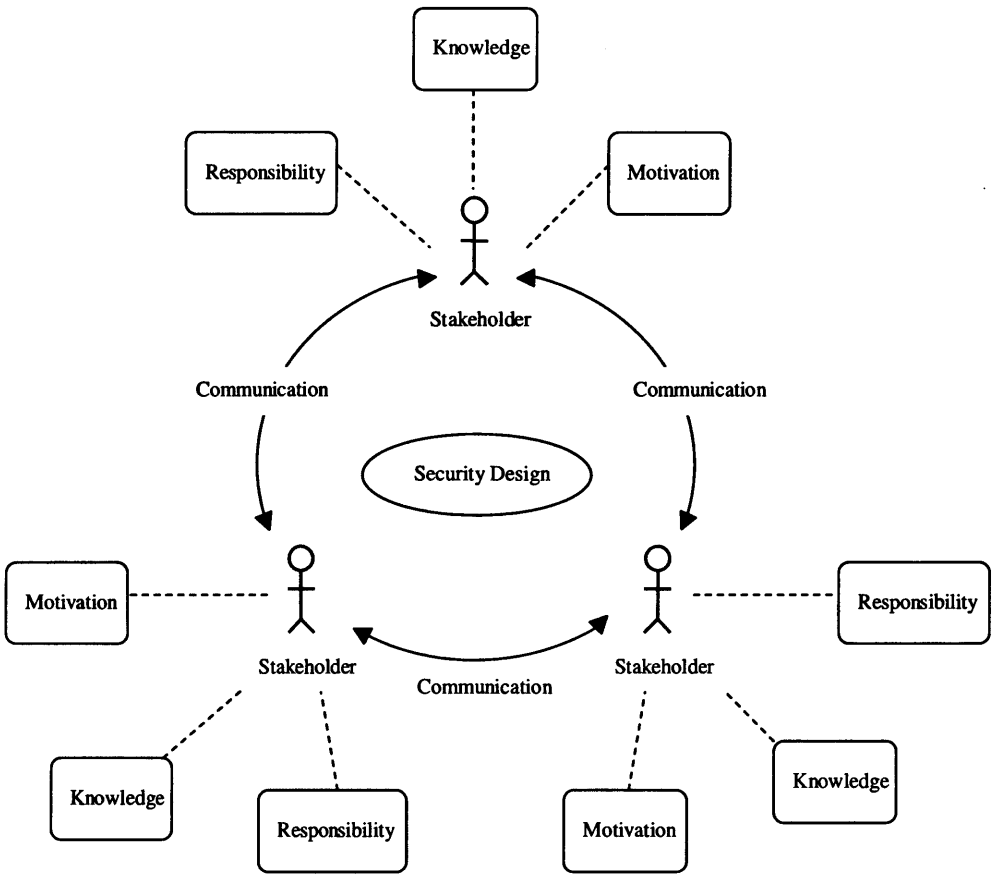


Figure 17: Grounded Theory Model of Socio-Technical Secure System Design

The aim of security design is to create a system that is adequately protected from undesirable events in that system. It is important for the success of the security that relevant knowledge be present during the design process. This includes the need for accurate knowledge about:

1. the system
2. security practice

Knowledge about the system refers to information that describes what the system is intended to do, how it should operate and any other information about the system relevant to the design of security. The best source for this information exists in the knowledge of stakeholders such as users, developers, system owners, system administrators, etc.

Knowledge about security practice refers to security concepts and principles, the understanding of the need for security to be usable, insight into threats, vulnerabilities

and risk, etc. This type of information exists in the knowledge of stakeholders such as security experts, but also as part of the security design process itself, which informs and directs the act of designing security. In the case studies, the AEGIS process provided some knowledge about security practice, whilst also recommending the involvement of security experts as a means of providing the essential knowledge gained through experience. Other design processes exist, as was reviewed in section 2.5.
Given that any stakeholder (for example a user or a security expert) has a degree of knowledge about both the system and security practice, the model does not distinguish between the extent and type of knowledge that different stakeholders have.

In order to ensure that this knowledge is available, it is necessary for all the relevant stakeholders in the system to be identified and represented in the design process. These stakeholders include people who are more knowledgeable about the system – for example users, developers or administrators – and people who are more knowledgeable about security practice, namely security experts.
One of the best ways of achieving this representation is to ensure the participation of stakeholders in the process. However, each stakeholder has different security responsibilities (which range from significant in the case of security experts to very low in the case of some users) and various levels of motivation to address security needs. As was seen in the EGSO case study, this can result in certain stakeholders declining to participate in the security design, either through a perceived lack of responsibility (for example through *diffusion of responsibility*) or insufficient *motivation* (for example with the need to achieve functionality taking precedence). Despite this, ensuring that the relevant viewpoints of stakeholders in the system are represented in the security design is critical in making sure that the final design addresses the needs and requirements of these stakeholders.
The issues of motivation and responsibility can also affect the security design process beyond determining whether stakeholders decide to participate or not. That is to say that issues of responsibility boundaries, in other words the perception of the limits of responsibility of a particular stakeholder, also limit the extent to which stakeholders decide to consider security needs. In practice, this can be seen when stakeholders decide not to address specific issues because they perceive the problem to be the responsibility of another party. The problem arises here when there is no communication between the two parties and both of them assume that the issue is the responsibility of the other (another instance of *diffusion of responsibility*).

As shown in the model, the interpretive process of socio-technical security design is fundamentally a communication exercise between the different stakeholders in the system. Without effective communication the relevant design information – such as requirements, constraints or necessary discussions – would be impossible. As a result, the primary purpose of a socio-technical security design process is as a means of *facilitating* this communication.

As was seen in the case studies, the concept of communication was very strongly tied in practice to anecdotes and scenarios as means of expressing security knowledge and reasoning about security. Anecdotes were frequently used to communicate knowledge about real security issues. Scenarios, however, were even more widely used as a means of communicating security concepts, reasoning about security principles and justifying points of view.

The most basic need for successful communication is to allow stakeholders to share their knowledge and points of view with as little bias and as few misunderstandings or confusion as possible. This is necessary to ensure that the design process remains focussed on matters of security design, as opposed to having to address issues of semantics or other unrelated concerns. The aim is to ensure that stakeholder conflicts, in other words disagreements, are related to genuine and valid opposing points of view, as opposed to differences that arise out of miscommunications or confusion. Resolving these disagreements is a key aspect of the design of security.

7.2 Discussion of AEGIS

One of the key elements of a socio-technical secure design process, identified in the Grounded Theory analysis, is the need for facilitating communications between stakeholders. AEGIS is particularly effective in this respect by providing a simple tool for supporting the communication of security concepts in the form of the asset and security requirements model. The fact that the chosen modelling notation is capable of supporting scenarios, through use cases, is an added benefit. As seen from the case studies, the use of this model in practice was rather straightforward, and allowed the identification of a number of security and organisational concerns. And whilst the lack of specified semantics did lead to some confused notations (in the DCOCE case study), the model support did prove to be an effective means of identifying security requirements.

An additional strength of the AEGIS process has been the importance attached to involving different stakeholders in the design of security. Whilst security design can be a jealously guarded secret among experts – possibly due to a belief in the principle of security through obscurity (see section 2.4.4) – including stakeholders proved to be a useful step. The involvement of these stakeholders provided a means of gaining more relevant information about both the system design and specific needs that had not necessarily been identified, for example the need for specifying administrative duties in EGSO (see section 4.3).

Given the difficulties that were encountered in convincing some stakeholders to participate in the design process, AEGIS also proved to be useful in situations were the ideal representation of stakeholders was not available. In those cases, the participating stakeholders had sufficient knowledge of the system to allow the identification of assets, and security requirements. A good example of this can be seen in the CLEF and BioSimGrid cases studies.

The final key strength of AEGIS has been the ability to introduce human factor issues into the design of a technical secure system. Putting people in context with the technical assets of a system has allowed the explicit statement of certain implicit assumptions. For example in the EGSO study, one implicit assumption was that the data provider administrators would conduct specific tasks pertaining to EGSO, such as maintenance, providing special services to specific customers, or ensuring security. It was only when these assumptions were made explicit that the need to address this was identified.

One of the difficulties of adopting a physical asset-based modelling notation has been the difficulty in representing non-tangible assets, such as reputation or information. The notation is capable of modelling the importance of these non-tangible assets, through the security ratings of assets that could impact them. This is, however, not an ideal solution, in that it is not readily apparent that these assets have been taken into account.

A further problem arose in the need to identify the dependencies between assets. This is necessary in order to be able to easily identify the impact of an attack on the system. The solution to this problem was to firstly assign the same value to the relevant security properties of dependent assets. Secondly, during the risk analysis phase, all the vulnerabilities identified in the system should include a list of all the assets that could be compromised, thus reflecting the chain of dependency. Whilst both these solutions

allow the modelling of dependencies, they are also not an ideal solution in that dependencies are not readily apparent from the resulting model.

Despite this, the Action Research results of the application of AEGIS described in this thesis provide good evidence of the success and usefulness of the process. Further evidence of successful applications of AEGIS, not reported in this thesis, can be found in [108] & [110]. In the next section, as a result of the empirical application of AEGIS, a revised and improved version of the process is presented.

7.3 AEGIS

As a consequence of the empirical application of AEGIS a number of issues were identified that needed addressing. First was the need to ensure that the semantics of the AEGIS modelling notation were specified in order to ensure consistency. As described in section 7.4, the semantics of AEGIS were specified using the MOF extension mechanism.

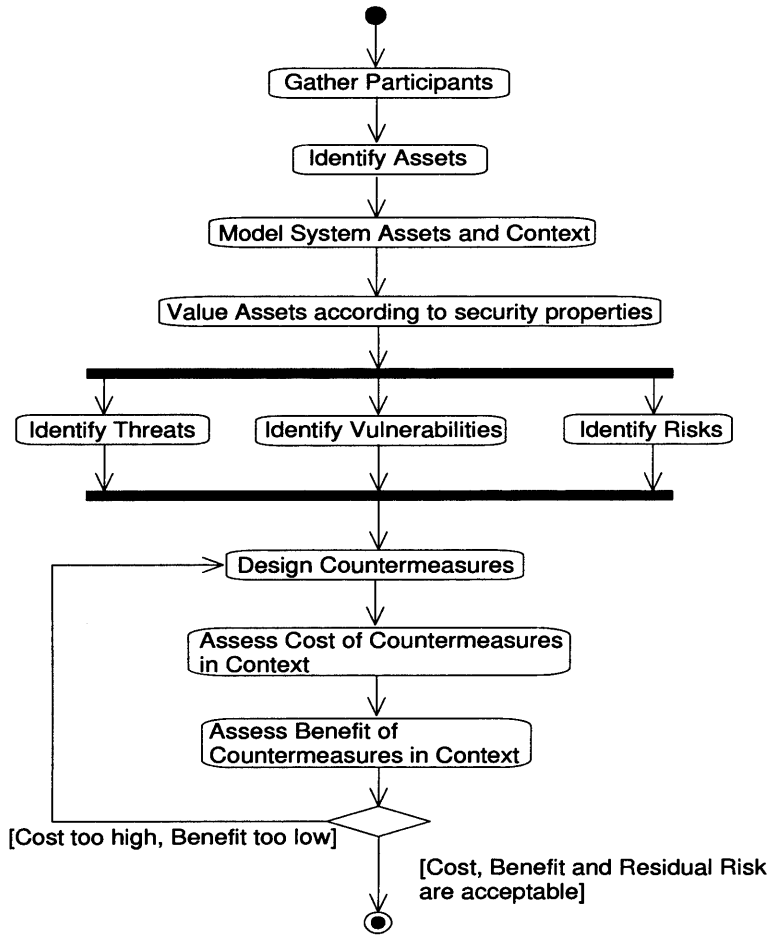


Figure 18: AEGIS activity diagram

The need for identifying dependencies between assets has been an issue that occurred during the application of AEGIS. Recording these dependencies is now done in two ways:

- The security attributes of dependent assets have to be the same
- The vulnerabilities identified in the risk analysis list all the assets that can be compromised.

Appropriate and Effective Guidance for Information Security (AEGIS) is a software development process for secure and usable systems.
AEGIS follows a standard incremental and iterative software development process as described in [116]. An iterative process allows for rapid implementation of software and requirement gathering from all the stakeholders in the system. In more detail, the activity diagram in Figure 18 describes the core activities of AEGIS, which consist of identifying and securing the correct participants, getting them to model the system's assets in context using the semantics defined through the UML [88] meta object facility [87], assign a value on these assets, conduct a risk analysis and, finally, design the countermeasures that address the risks in a cost effective way.

7.3.1 Gather Participants

The first step in any system design process is to identify and secure the commitment of the people who will participate in that design. There are four main types of roles that can be differentiated (although an individual can play more than one role):

- Decision makers. They consist of project management, owners (customers commissioning the system), and anybody else given a decision-making role in the development of the system.
- Developers. They are the technical aspect of the design team, responsible for the capture and analysis of the system requirements down to the design and implementation. These include programmers, designers, security experts, interface designers, etc.
- Users. They are the people that the system should be designed to work with, and as such are a major source of system requirements.
- Facilitators. They are the people who run the AEGIS process, document the meetings and serve as mediators in general.

Despite being traditionally regarded simply as a technical problem, the design of security is instead a socio-technical issue [10, 54] – i.e. designing and building security must involve both a technical and a social undertaking. Developers are the best equipped to handle the technical aspects of security; however the social aspects of security are generally the responsibility of the owners and higher management, who have the authority to institute, encourage and enforce policies. This is why it is essential to ensure the participation of these groups of people: the decision makers – who are better suited to deal with the enforcement of the social requirements of security, the developers – who are necessary for the technical implementation of security, the users – who are the ultimate source of usability requirements of the system, and the facilitators – who ensure the smooth running of the design process.

The most important aspect of this phase is to determine a single individual who will have leadership for the security of the project. The responsibility associated with this role is to document decision-making, citing the arguments and reasons for the decision, and to provide a final say in any disagreement that may occur during the process.

Besides the definition of the roles of the participants, AEGIS provides an asset and risk based approach to designing security, which then supports a meaningful cost-benefit analysis. The first step in this analysis is to identify the assets of the system, and is described next.

7.3.2 Identify and Model Assets

AEGIS defines three major categories of assets – operatives, hardware and data.

- Operative
    - User
    - Administrator
    - Developer
- Hardware
    - Network link
    - Computer
        - Processing node
        - Storage
        - ...
- Data
    - Application
    - Information

Operatives identify the people interacting with the system, whether users, developers or administrators. These assets tend to be the most overlooked of all, because they are not generally perceived as being a part of the system, but a customer of the system. In practice, operatives will always play a central role in the success or failure of any security countermeasures, and some of the most devastating and successful attacks [103, 104] involve legitimate operatives either maliciously or inadvertently breaking the security [81].

Hardware assets are the physical entities in the system which need to be protected. For simplicity we only consider a short list of the possible ones. From a security standpoint, an attacker who has physical access to the hardware is much more likely to succeed than one who does not. Identifying the presence and role of the physical assets in the system is therefore vital in the overall design of the security countermeasures.

Data assets are subdivided into applications and information. Applications refer to the software that runs on various hardware assets. These will generally correspond to the more traditional architecture for the system (which concerns itself with the software architecture). The information asset is not to be confused with the notion of data. Information is a concept that is unfortunately rather fluid in its definition. From a security point of view, it is possible to have data but not information. For example intercepting an encrypted message gives an attacker the data, but without the decryption key(s), the data holds little information. On the other hand, a pseudonymised medical record where all identifiable information has been removed or altered may still hold information to an attacker in what is referred to as an "inferential attack". That is to say that the correlation of the pseudonymised record with another dataset already in the possession of the attacker can give a positive match and yield information to the attacker.

- Security Measure
    - Operative
    - Hardware
    - Data

On top of these categories there are security measures, which consist of any of the previous assets. Security measures can take the form of operatives (e.g. guards, administrators checking system logs, or users having secure passwords), hardware (e.g. dedicated optical networks that are much more resistant to interception, dedicated encryption hardware or random number generators) or data (e.g. a security policy

governing the backup of information, an encryption algorithm, a firewall application or an encryption key).

It is important to understand that while the categories are defined in AEGIS, the assets themselves are different from system to system. Regardless of whether the system already exists or not, the identification of the assets of the system should be done by the participants. In the case where the system has not yet been designed the process can still take place, but instead of identifying assets that already exist, assets should be identified from the most likely architecture. This will then be refined at a later date, when more is known about the system, possibly in a typical incremental fashion.

Based on this, building a model representation of the system's assets also serves as a very useful common ground for discussing security requirements. Frequently, simply building the model throws up a number of issues, which should be recorded by the facilitators for future discussion.
A more detailed description of the AEGIS asset model can be found in Section 7.4.1. The next step consists of assigning a value to these assets according to various security properties as described in the following section.

7.3.3 Value Assets According to Security Properties

In order to elicit security requirements from the participants, it is necessary to first explain and agree on the meaning of security properties. The three most common security properties are defined as follows [60]:

- Confidentiality: property of security that concerns unauthorised disclosure of information.
- Integrity: property of security that concerns unauthorised modification of information.
- Availability: property of security that concerns unauthorised withholding of information.

Security requirements elicitation is achieved, for each of the assets, by having the participants judge the importance of the asset in terms of the security properties defined above
A qualitative rating system is recommended, based on natural language which gives flexibility in the rating of the assets; however it is equally possible to adopt quantitative

rating systems (such as the Annualised Loss Expectancy [25]) that may integrate better into more formal risk analysis methods. The qualitative approach allows participants to use their own words to define how important assets are, and by ranking the results hierarchically, a breakdown of the most important security properties for the assets can be identified according to each participant.

Experience has shown that scenarios are very useful in making participants both understand what the security property means, but also, how important it is in relation to the asset. These various scenarios should be documented, possibly in the form of abuse cases [80] – UML use cases of an attacker and the actions he/she may take to conduct an attack.

For more information on the semantics of modelling the security requirements for the assets, see Section 7.4.2. The following step should consist of a risk analysis to identify threats, vulnerabilities and risks to the system.

7.3.4 Risk Analysis

Risk analysis attempts to determine which threats and risks the system faces in order to feed into the design of security countermeasures that are appropriate to the threats and are cost-effective to the risks. Knowledge of existing and past threats and vulnerabilities is essential, as is the presence of expert security knowledge in order to interpret and adapt this information to the situation at hand. In the absence of hard evidence (which is a rare commodity), much of risk analysis is based on opinion and is widely open to argumentation – however by using a structured approach, it is possible to address this in a systematic manner.

This step is not about dictating the security needs of the system, it is about painting the picture of the threats, vulnerabilities and risks to the system in its current form. The designers, the developers and decision makers should then use this information to decide if, what, and how much security should be built into the design.

A risk analysis generally goes through a three-step process of:

- Identifying Threats – Threats are the potential sources of attacks to the system. Things that characterise threats include the attacker, their motive, their target, their resources and their risk-aversion.
- Identifying Vulnerabilities – Vulnerabilities are areas of the system that are amenable to exploitation. This is where security advisories, security scanners, good knowledge of the technologies being used and information about past attacks become important. It is also vitally important to recognise that

vulnerabilities in a system apply to human as well as technical characteristics. It is the case that the most devastating and successful attacks to date have generally resulted from social engineering and not technical wizardry.

- Identifying Risks – Risk is the likelihood of an attack successfully exploiting one or a sequence of vulnerabilities in order to compromise an asset. Since a risk is a forecast of the future, it is only a "best guess" based on information such as past information and knowledge of the system. This information is generally best acquired from security experts who have the knowledge and experience necessary to assess these risks.

7.3.5 Security Design

This next phase is an iterative process of designing potential security countermeasures and assessing their respective costs and benefits in the context in which they will be used. The aim of this is to reduce all the risks identified previously to an acceptable level, whilst ensuring the reliability of the system by providing suitable mechanisms, education, incentives and disciplinary measures to motivate people in the system to behave in the expected manner.

Security measures come in five different flavours that work best when combined together [60]:

- Avoidance: Measures that discontinue the possibility of a given threat, or transfer liability to a third party
- Deterrence: Measures that discourage the abuse and damage of an asset
- Prevention: Measures that prevent assets from being damaged
- Detection: Measures that afford the knowledge of when, how and who has damaged an asset
- Reaction: Measures that stop ongoing damage and recover from damage to assets

Some security measures will belong to more than a single class of security, an example of which can be prosecuting offenders – which is both a reactive and a deterring measure.

The design of the security should be driven by the risks identified previously, with attention being paid to those which are deemed to be most important. During this design, the cost of the implementation, deployment, operation and maintenance of the

resulting secure system should be assessed. Cost refers not only to the financial price of security measures, but also to the user cost of having to apply them.
In addition to identifying cost, the benefit of each particular security measure should be looked at in order to determine how effective it is at reducing risks and whether different or additional measures are necessary.
Since a secure system is a socio-technical system, it is also necessary to design and implement the social aspects of the system as well as the technical. This is because each individual user playing a part in applying security measures will undergo their own personal cost-benefit analysis about the need to follow security policy, weighing a number of factors such as the effort involved in following the policy, what the punishment may be for violating the policy or how exposed the individual feels to the threat.

The next section describes the UML meta-model that is used to give semantics to the asset model specified by the participants.

7.4 UML Meta-Model for Asset Diagram

7.4.1 Asset Model

The semantics for an asset diagram are described using the UML Meta Object Facility [87] as can be seen in Figure 19. The meta-model defines the semantics for models of assets which can then be built by the participants. The reasons for choosing UML for this kind of modelling are obvious: UML is a well-understood notation among developers, it is widely supported and easy to extend (through the meta object facility). The simplicity of the extension means that non-experts can also relatively easily understand and use this as a starting point if provided with a basic introduction.
Four new objects are defined in the meta-model in Figure 19:

- `asset`
- `operative`
- `physicalEnviroment`
- `culturalEnvironment`

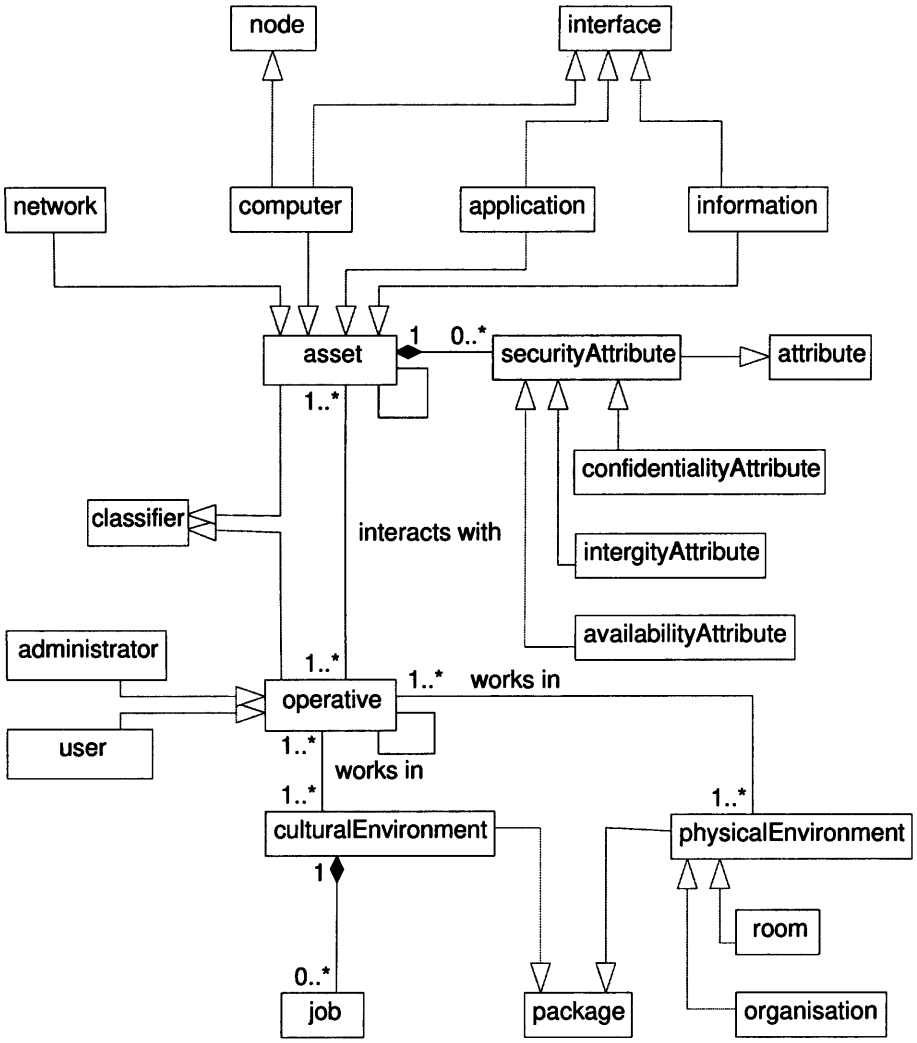


Figure 19: Asset diagram meta- model

Although we have previously identified operatives as being assets, the AEGIS meta-model refines the semantics with a distinction between `operative` and `asset`. This is to accommodate the differences of interaction that operatives have with other assets and other operatives. Bearing in mind the similarity of an `operative` to a UML user, the same look was chosen to depict an `operative`, as seen in Figure 20.

Figure 20: Diagram for `operative`

In addition to the assets and operatives, the physical and cultural context surrounding both the assets and operatives can also be depicted through the `physicalEnvironment` and `culturalEnvironment` components. Asset and operative both extend the UML MOF `classifier` and should therefore be modelled as such (for more information and examples on how to model various elements see Sections 7.3.2 & 7.3.3). `physicalEnvironment` and `culturalEnvironment` both extend the core UML `package` and should therefore be modelled as packages. These two components can thus contain `assets` and `operatives` and serve to represent the boundaries of both physical environments (such as rooms) and cultural environments (such as security culture). For example, a system administrator operative and a secretary operative sharing the same room should be apparent.

7.4.2 Security Requirement Modelling

In order to document the security requirements, `Asset` is a classifier (Figure 21) that contains `securityAttributes`, which have been defined as `confidentialityAttribute`, `integrityAttribute` and `availabilityAttribute`. These attributes should be used to record the value of each asset.

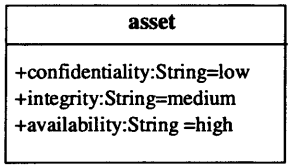


Figure 21: Diagram for `asset`

Additional attributes can also be defined, such as for example a non-repudiation attribute or dependability attribute, depending on the needs of the system. These `securityAttributes` extend the core `attribute` element of the UML MOF and can thus be depicted in a similar manner. Thus, an asset can be drawn as shown in Figure 21.

7.5 Chapter Summary

In this chapter, the final Grounded Theory model describing the storyline of the factors that affect the socio-technical design of secure systems was presented. This included the importance of involving different stakeholders with relevant system or security knowledge. In addition, the importance of communication in the design process was also described, including the fundamental need to achieve clarity. In practice this is achieved through the use of scenarios and anecdotes as means of illustrating security concepts or reasoning about security.

A discussion of the strengths and weaknesses of AEGIS then followed, tying into the insights gained from the Grounded Theory model. This includes the successes of using the modelling notation as a simple means of communicating about security, involving different stakeholders in the design process, and the importance of introducing human factor issues in the design process of a technical system.

Finally, the final revision of the AEGIS process was presented. This included the specification of the semantics of the modelling notation, opening the way for better tool support, and providing more detail about the risk analysis phase of the process.

8 Conclusions

8.1 Summary

Traditional computer security focuses on technical tools and techniques to secure a system, such as secure communication protocols, encryption algorithms or network scanners. Whilst this is useful, problems have been identified with the usability of technical security mechanisms [118]. In addition to this, many authors are claiming that the most effective means of attacking a secure system is through social engineering [81, 103]. In contrast to this, other security design approaches in the field of information systems address the issues of security from a non-technical perspective – i.e. looking at organisational processes and describing appropriate policies, measures and mechanisms for preventing potential security violations. The problem here is that these do not integrate well into the design of a technical system.

As a means of addressing this problem, there has been some research into considering socio-technical approaches to information system security [80]. Whilst this research provides useful information to the problem of designing secure systems that accommodate human factors, there remains a lot of ground to cover. In spite of the fact that some of these methods have been empirically tested [69], there has been no investigation into practical real-world factors and their effect on a socio-technical approach to secure system design. The usefulness of this theoretical perspective would be a greater understanding of the environment, pressures and limitations surrounding socio-technical secure system design. This in turn can be used to inform future research and improve on existing security design methods.

The research question addressed in this thesis was: **'how can the design of usable and secure socio-technical systems be better *understood* and *supported*?'**

Drawing from research in the fields of Software Engineering, Computer and Information Security, and HCISec, the socio-technical secure system design process AEGIS was presented. The purpose of the research in this thesis was directed according to two themes. The first was to empirically apply AEGIS in order to both refine and validate the process through Action Research, thereby providing better support for the design of usable and secure socio-technical systems. The second theme sought to achieve a theoretical perspective into socio-technical secure system design based on the exploratory Grounded Theory analysis of the application of AEGIS.

	EGSO	CLEF	BioSimGrid	DCOCE
Number of workshops	4	2	1	1
Dates	7/2/2003 28/2/2003 11/4/2003 6/06/2003	30/4/2003 14/7/2003	18/10/2004	18/10/2004
Location	University College London	University College London	Oxford University	Oxford University
Number of participants	3 project members	2 project members	2 project members 6 student-analysts	4 project members 6 student-analysts

Table 6: Summary of Research Interventions

In order to address the first research theme, AEGIS was applied to a total of four case studies: EGSO, CLEF, BioSimGrid, and DCOCE – see Table 6 for a summary of the research interventions. Although DCOCE was not a technically a Grid project, it had much in common with the other three Grid projects. The similarity of the technologies and academic project management between these four different projects provided a means of easily comparing one study to another. As a result, these case studies were ideal data subjects for the application and validation of AEGIS.

The first two case studies were conducted on EGSO and CLEF. These involved the researcher directly in the application of AEGIS, and the success of the intervention was seen in the successful identification of security issues and human factors which had not previously been discovered. Further evidence of the success of AEGIS was seen in the feedback from the participants who in the EGSO study stated that AEGIS had *"improved understanding of (the) problem space"* [61]. The endorsement from participants in the CLEF study came in the form of an invitation for the researchers to participate and apply AEGIS in the follow-up *CLEF Services* project.

The final two case studies also involved the researcher in the application of AEGIS, but in this case the task of conducting the analysis was given to student-analysts to whom the process had been taught in a 2-hour session. Whilst the analysis process was rather short due to practical constraints, the application of AEGIS again yielded positive results both in areas of security and human needs. Whilst there was evidence of greater confusion and the need to clarify the issue of notation semantics as a result, this is also understandable due to the fact that the student-analysts were not familiar with the process. Despite this, positive outcomes did come out of these studies, lending further support to the value and validity of the AEGIS approach.

The second research theme of this thesis was addressed by conducting a Grounded Theory analysis of the transcripts of the design sessions. The transcripts were coded using the hermeneutics analysis tool ATLAS.ti to identify significant issues and factors that occurred during the design process of security that could provide insight into the socio-technical secure system design process. Four main categories were uncovered, namely those of responsibility, motivation, communication and stakeholders. The final output of the Grounded Theory analysis was a model of these factors and their specific relationship to a socio-technical security design process. This highlighted that an interpretive socio-technical design approach relied on the need to represent different stakeholder viewpoints, possibly through the direct inclusion of the stakeholder in the design of security. As a result, the act of designing security relies heavily on the communication between the different stakeholders, together with the need for clarity in that communication. This is necessary to avoid problems of confusion and misunderstanding that can arise out of the complexity of security design.

Based on the results from both research themes, a discussion of the strengths and weaknesses of AEGIS was presented, followed by a description of the final version of the process. This final version includes a formal definition of the semantics of the modelling notation using the UML Meta Object Facility, together with clarifying the purpose of each of the different steps.

8.2 Contributions

Coming back to the original problem statement, the research presented in this thesis is directed according to two themes:

1. The identification of issues in the development of secure technical systems. This consists of:
   - **1.1.** A theoretical perspective of security design issues based on the literature.
   - **1.2.** An empirical perspective of security design issues based on the analysis of the case studies of AEGIS (developed in **(2.1)**), culminating in a model of the relevant factors.
2. The presentation and evaluation of a socio-technical design method for secure systems. This consists of:
   - **2.1.** The formulation, based on **(1.1)**, of Appropriate and Effective Guidance for Information Security (AEGIS).

2.2. The practical application of AEGIS in order to validate and evaluate the benefits and disadvantages of that process.

2.3. The refinement of AEGIS, based on the results of **(1.2)** and **(2.2)**.

In order to address these two themes, the following substantive contributions were made in this research:

1. A critical review of the state of the art of security design, specifically as it applies to addressing human factors in technical security design (in chapter 2)
2. The socio-technical secure system design process AEGIS (in chapters 3 and 7)
3. An evaluation of the AEGIS design process through empirical research (in chapters 4, 5 and 6)
4. An analysis and model of the real-world factors that affect the socio-technical process of developing secure systems (in chapters 4, 5 and 7)

An additional methodological contribution made in this thesis has been the application of Grounded Theory to the transcripts of Action Research as a means of providing greater theoretical understanding (described in chapter 3). Combining these two approaches allows the researcher to gain additional validity from the results as both methodologies compliment each other. Action Research is useful in gaining a broad understanding of the issues and the validity of the research, whilst Grounded Theory provides the tools to analyse more detailed factors, thereby gaining a better understanding as to why the Action Research was successful.

8.3 Discussion

Based on the results of the Action Research studies, AEGIS has been shown to be valuable and effective at bridging the gap between technical and human factors in security. One of the added benefits of AEGIS has been the importance of identifying security requirements based on assets before engaging in countermeasure selection. This has allowed participants to specify their need for security and later evaluate their security design decisions based on these requirements, providing a helpful sanity check of the security design. With regards to providing support in the design process of security, this led to a simple means of separating goals and technologies – allowing stakeholders that did not have technical know-how to specify their security needs in a simple fashion.

The most important conscious decisions in the elaboration of AEGIS was the need to ensure the simplicity and clarity of the process. This was necessary in order to provide practical security knowledge to non-experts, without requiring them to learn significant amounts of specialist information. As a result of this, the stakeholders that possessed the most knowledge about the system (yet were not very knowledgeable about security) could become involved in the design of security, thereby improving the quality of the overall process. The need for involving a security expert, especially during the risk analysis and security design phases, was identified as the most effective means of gaining access to the specialist knowledge required for security design. However by involving developers and users in the security design, the role of a security expert becomes somewhat akin to that of a consultant advising about the relevant security problems and possibilities. This approach is useful in avoiding the problems of development duality in which the design of the functionality of a system and its security are separate.

Another fundamental decision was to adopt a process that fits into a software engineering design approach. The intention behind this decision was to provide better integration and support for developers who were in charge of building the functionality and the security. The initial adoption and later specification of a UML compatible modelling notation, as well as the seamless integration of AEGIS into software engineering approaches are a testament to this.

In answer to one part of the research question – how to better support usable and secure socio-technical system design – both these aspects of AEGIS (identifying security goals before looking at technologies, together with using a UML compatible notation) are persuasive and effective contributions.

In answer to the second part of the research question – how to better understand usable and secure socio-technical system design – the detailed analysis of the application of AEGIS yields useful results.

From a methodological point of view, combining Action Research with a detailed Grounded Theory analysis provides a useful means of exploring the research question, and surmounting the problem of conducting real-world empirical research in security. This approached provides a flexible, yet systematic means of gathering detailed information whilst maintaining rigour in the conclusions, It should be emphasised that great care was taken during the Grounded Theory analysis to separate and discount

factors that were directly the result of the Action Research component. In this case, all the factors that were attributed to the AEGIS methodology were ignored (such as the modelling approach, or the risk analysis and cost-benefit decision making aspects) in favour of factors which were not attributable to the actual intervention. This is particularly important in order to be able to generalise the findings of the Grounded Theory to areas outside specific experiments, and also to reduce the amount of bias that could be introduced by the researcher actively intervening in the research subject.

From a substantive point of view, the Grounded Theory model of the factors and issues surrounding socio-technical secure system development provides a useful theoretical framework that can be used to analyse the reasons for the success or failure of a given socio-technical design method. This can be useful for future research as a means of evaluating other secure system design methodologies and identifying some of their strengths and weaknesses, together with proposing areas in which security design methodologies can be improved.

When applying the model to AEGIS, some of its key strengths relate to the inclusion of a variety of different stakeholders, as well as the dedication to facilitating the communication process of these different stakeholders (see section 7.2). However according to the model, some other problems could be resolved by employing security design methods that address responsibility issues – such as, for example, the Structures of Responsibility [16] security design approach.

In addition to helping to evaluate other methodologies, the factors identified in the model have implications for secure system design in general. Motivation and responsibility are two key aspects that extend beyond the scope of a design methodology. Specifically assigning responsibility to individual stakeholders, or ensuring their motivation to address the security issues, is not in the power of a design methodology to enforce. However it can be a strong recommendation that one of the first steps any security design exercise should take is to ensure the clear assignment of responsibility. This is necessary to ensure that everyone in the project knows who is in charge of security, and addresses the problem of diffusion of responsibility. A final point is that responsible stakeholders that do not have the authority to implement security decisions will encounter significant difficulties. As a consequence, responsibility should only be assigned to people already in authority (senior management for example), or authority should be given to those who are charged with ensuring security.

The assignment of responsibility also has an impact on the motivation of stakeholders – namely increasing it in those who are responsible for ensuring security. Maintaining the momentum and motivation for security can be a difficult task when faced with competing demands (e.g. functionality, cost, time-to-market), and the complexity of security design. An initial motivation for achieving security drives the desire to adopt a secure system design methodology. It is therefore important for security practice to understand the motives behind the need for security and address these. The motives can relate to legal requirements (such as the CLEF project – see chapter 5), fears of exposure to attack, or having valuable assets. Clearly addressing these underlying issues should be a key element of any security design methodologies if they are to keep in touch with the original motivation for security. Furthermore, the process of security design should be engaging, inclusive and understandable to all the participants in order to avoid discouragement. Motivational and organisation psychology are ideal fields of research from which further insight into these issues could be gained.

The identification of anecdotes and scenarios in secure system design has more practical and immediate implications for security design in general. One of the key elements of any design exercise is ensuring the good communication of the participants. Security methods that adopt and support scenarios are much more in tune with the way people tend to communicate about security. As such they have a greater chance of being understandable, facilitating the clear communication between stakeholders. This is not to say that scenarios should be the only means of modelling or reasoning about security. Some of the weaknesses of scenarios are related to the difficulty in generalising their information content (see section 2.5.1). That is to say that scenarios are highly specific descriptions of particular events in a system, whereas security needs have to encompass the system as a whole. Scenarios should therefore be used in conjunction with other security analysis techniques, as a particularly useful method for explaining security concepts and reasoning.

Similarly, anecdotes used in security design discussions are not necessarily accurate or representative of the problem space. However, anecdotes have the benefit of being a persuasive source of security knowledge during these discussions. In the eyes of many, information related from anecdotes holds the advantage of having actually occurred, as opposed to being a theoretical possibility. Simply relying on anecdotes as the sole source of knowledge informing security design is risky and prone to error. However used in conjunction with other sources of knowledge, anecdotes have the benefit of

being a useful source of information that can be easily communicated, understood and accepted by the different stakeholders.

8.4 Critical Review

Whilst there have been encouraging successes in this research, it is also important to identify areas of weakness. One of the main concerns about the evaluation of AEGIS can be seen in its exclusive application to academic research projects. These differ from commercial environments significantly. Some of these differences can include different priorities from stakeholders, as well as completely different application fields. Therefore the effectiveness of AEGIS for commercial use is hard to ascertain, and the methodology may require adjusting in order to achieve satisfactory results.

Another criticism is that AEGIS does not provide a formal decision making process, providing a framework enabling stakeholders to make decisions based on rational information. The benefit of such an approach is that the decisions are made as a result of the security process as opposed to the (possibly questionable) judgement of the security designers. An example of such an approach is Butler's Security Attribute Evaluation Method [34], which provides an objective framework in which the cost effectiveness of security countermeasures can be identified. The key counterargument to this is that the purpose of AEGIS is not to provide an objective means of making security decisions – a hallmark of functionalist approaches to security design. The process instead aims at providing the important stakeholders in the system with a means of identifying and gathering the relevant information about security in order to allow them to make their own decisions. Given this, a formal decision making process would be too inflexible to take into account the large numbers of relevant and important information about the system. That is not to say that better support for the decisions making process would not be useful, and this has been identified as one direction for future work on AEGIS.

A final criticism about the research can be directed at the Grounded Theory analysis of the factors and issues that affect socio-technical security design. Leaving aside arguments about the validity of Grounded Theory as a research methodology, the factors that were identified may only be relevant in a small number of cases, or even only in the cases that were studied. This is, of course, a possibility, however the inclusion of four different case studies in this research provides a greater degree of assurance that the factors identified are not specific to these particular projects. Given

the first criticism about the limitation of the research to academic projects, it can be argued that these results are only relevant to those types of environments. Whilst this is a possibility, the lack of similar undertakings in other fields does not provide a means of comparing the results. Therefore, whilst the findings may only be relevant to academic endeavours, they still provide valuable information about socio-technical design approaches. In addition, there is evidence from expert feedback about this research that the findings are relevant to other domains of application.

8.5 Directions for Future Work

Given that this is the first study into analysing the factors affecting socio-technical security design, further work into this area would be invaluable in providing added knowledge in the area. Since the studies described in this thesis are all related to academic projects, a major direction for future work in this area would be to analyse in more detail factors and issues that pertain to other sectors, such as commercial or governmental areas. This would be useful in establishing this type of interpretive approach to security as a "serious" design method, as well as necessary in order to gain a theoretical understanding of the other approaches to designing security. One possible area that could benefit from such a stakeholder driven approach is that of electronic voting (as envisioned by the UK e-government[9] programme for example). Electronic voting is an area that involves large numbers of very disparate stakeholders, and adopting a strategy to accommodate these different needs seems paramount to the success of the undertaking.

Further research would be useful in investigating how issues of organisational cultures can be influenced from the standpoint of improving security. One approach to this could be to analyse in greater detail some of the mental models that stakeholders associate with security in a bid to identify those which improve security and those which do not, thereby increasing understanding of how security is perceived. As mentioned in section 8.3, the fields of organisational and motivational psychology would also be promising areas for interdisciplinary research into understanding and supporting security cultures. This research would fit very well into recommendations of the *Foresight Cyber Trust and Crime Prevention*[10] project, which looked at means of achieving secure standards

[9] http://www.odpm.gov.uk/

[10] http://www.foresight.gov.uk

for online activity. One of the key proposals was that fostering a secure culture was central to the achievement of "appropriate online activity" [51]. The model presented in this thesis could be used as a means of further exploring some of the issues surrounding the promotion of security cultures, possibly by exploring in further detail the issues of responsibility and motivation.

Also of interest would be to gain a better understanding of the interaction between people and security technology. This would, in the first instance, provide useful information about the way in which people relate to – and view – security technology. Such knowledge is necessary to gain a better insight into the underlying principles of security cultures. Secondly, additional research into understanding the interactions between social elements and technical mechanisms would also provide means of evaluating the effectiveness of security technology. This could help to improve the overall design of security through a better understanding of the strengths and weaknesses of different technologies.

With regards to future directions for the AEGIS process, providing tool support for modelling and security analysis would be very useful. Although the first step was taken through the modelling of the semantics of the notation, additional tool support could be given during the risk analysis by providing access to lists of common threats and vulnerabilities. Whilst this would not address issues of novel risks and vulnerabilities, it would provide a useful support for assessing the majority of commonly know threats.

An additional area of interest would be to provide a decision-supporting framework. As mentioned in section 8.4, this type of framework should not be intended to become a decision making tool. It would be useful, however, in providing a clearer means of assessing the benefits and disadvantages of specific countermeasures. Whilst the work by Butler [34] provides a useful starting point, the difficulty that would need addressing here is the need to factor in user costs and user benefits of security technologies.

Somewhat related to the previous point would be to identify means of clearly, but simply, modelling dependencies between different security properties of assets (see section 7.2). By providing a means of simply creating this interrelationship, the impact of particular threats exploiting vulnerabilities would be easier to assess. However, in order to retain the ease of use of the model (thereby maintaining support for the communication of security properties by non-experts), it would be necessary for this

notation to remain simple. Instead of complicating the model, one solution to this could be achieved through tool support which would allow the specification of dependencies during the risk analysis phase.

9 References

[1] *CERT.* http://www.cert.org

[2] *Security Patterns.* http://www.securitypatterns.org/

[3] *BioSimGrid.* 2004. **www.biosimgrid.org**

[4] *Cambridge Dictionary.* 2005. http://dictionary.cambridge.org/

[5] *CLEF Services.* 2005. http://www.clef-user.com/page2.html

[6] *Digital Guards Glossary.* 2005. http://www.digitalguards.com/Glossary.htm

[7] *eScience.* 2005. http://www.nesc.ac.uk

[8] *European Grid of Solar Observations.* 2005. **http://www.egso.org**

[9] Adams, A. *Users' perception of privacy in multimedia communication.* Unpublished Ph.D.Thesis, School of Psychology, University College London, UK 2001.

[10] Adams, A. & Sasse, M. A. *Users Are Not The Enemy.* Communications of the ACM 1999. Vol. 42, No. 12 December

[11] Adams, J. *Risk.* 1995. UCL Press.

[12] Adams, J. & Thompson, M. *Taking account of societal concerns about risk: framing the problem.* Health and Safety Executive. Research Report 035 2002. http://www.geog.ucl.ac.uk/~jadams/publish.htm

[13] Anderson, R. *Why Cryptosystems Fail.* ACM Conf.Computer and Communication Security CCS'93 1993. pp 215-227.

[14] Anderson, R. *The DeCODE Proposal for an Icelandic Health Database.* 1998. http://www.cl.cam.ac.uk/~rja14/iceland/iceland.html

[15] Anderson, R. *Security Engineering: A Guide to Building Dependable Distributed Systems.* 2001. Wiley.

[16] Backhouse, J. & Dhillon, G. *Structures of responsibilities and security of information systems.* European Journal of Information Systems 1996.

[17] Baker, D. *Fortresses built upon sand.* Proceedings of the New Security Paradigms Workshop 1996.

[18] Barki, H. & Hartwick, J. *Rethinking the Concept of User Involvement.* MIS Quarterly, March 1989. pp 53-63.

[19] Baskerville, R. *Investigating Informations Systems with Action Research.* Communications of AIS 1999. Volume 2, Article 19

[20] Baskerville, R. & Wood-Harper, A. T. *A critical perspective on action research as a method for information systems research.* Journal of Information Technology 1996. 11 (3) , pp 235-246.

[21] Baskerville, R. & Wood-Harper, A. T. *Diversity in information systems action research methods.* European Journal of Information Systems 1998. 7 (2) , pp 90-107.

[22] Bellotti, V. & Sellen, A. *Design for Privacy in Ubiquitous Computing Environments.* Proceedings of the Third European Conference on Computer Supported Cooperative Work (ECSCW'93) 1993. pp 77-92. Kluwer.

[23] Beyer, H. & Holtzblatt, K. *Contextual Design : Defining Customer-Centered Systems.* 1998. Morgan Kaufmann Publishers, Inc.

[24] Blakley, B. *The Emperor's Old Armor.* New Security Paradigms Workshop Lake Arrowhead CA 1996.

[25] Blakley, B., McDermott, E., & Geer, D. *Information Security is Information Risk Management.* New Security Paradigms Workshop 2001. pp 97-104.

[26] Boehm, B. W. *A spiral model of software development and Enhancement.* IEEE Computer 1988. 21(5) , pp 61-72.

[27] Boland, R. *Phenomenology: A Preferred Approach to Research in Information Systems.* 1985. Research Methods in Information Systems , pp 193-201.

[28] British Standards Institution. *Information Security Management-Part 1: Code of Practice for Information Security Management.* 1999.

[29] Brostoff, S. & Sasse, M. A. *Are Passfaces more usable than passwords? A field trial investigation.* People and Computers XIV - Usability or Else! Proceedings of HCI 2000 (September 5th - 8th, Sunderland, UK) 2000. pp 405-424. Springer.

[30] Brostoff, S. & Sasse, M. A. *Safe and Sound: a safety-critical approach to security design.* New Security Paradigms Workshop 2001.

[31] Brown, I. & Snow, C. R. *A proxy approach to e-mail security.* Software - Practice and Experience 1999. 29(12) , pp 1049-1060.

[32] Burrell, G. & Morgan, G. *Sociological Paradigms and Organizational Analysis.* 1979. Heinemann, London.

[33] Butler, S. *Security Design: Why It's Hard To Do Empirical Research.* 2000. Workshop on Using Multidisciplinary Approaches in Empirical Software Engineering Research, affilated with the 22nd International Conference on Software Engineering (ICSE 2000). http://www-2.cs.cmu.edu/~Vit/paper_abstracts/secure.butler.html

[34] Butler, S. *Security Attribute Evaluation Method: A Cost Benefit Approach.* Proceeding of ICSE 2002 2002. http://www-2.cs.cmu.edu/~shawnb/

[35] Butler, T. & Fitzgerald, B. *A case study of user participation in the information systems development process.* Proceedings of the eighteenth International Conference on Information Systems 1997.

[36] Carlson, B., Burgess, A., & Miller, C. *Timeline of Computer History.* 1996. http://www.computer.org/computer/timeline/

[37] Charette, R. N. *The Risks with Risk Analysis.* Communications of the ACM 1991. 34 (6), pp 106.

[38] Clinical e-Science Framework. 2004. http://www.clinical-escience.org/

[39] CSI/FBI. *CSI/FBI Computer Crime and Security Survey.* 2004. **http://www.gocsi.com**

[40] CSO Magazine/U.S.Secret Service/CERT Coordination Center. *2004 E-Crime Watch Survey.* 2004. **http://www.cert.org**

[41] Darley, J. M. & Latané, B. *Norms and normative behaviour: field studies of social interdependence.* Altruism and Helping Behaviour. 1970. New York: Academic Press. J.Macauley & L.Berkowitz (eds).

[42] Denning, D. E. *The Limits of Formal Security Models.* National Computer Systems Security Award Acceptance Speech 21999. http://www.cs.georgetown.edu/~denning/infosec/award.html

[43] Department of Defense. *Trusted Computer System Evaluation Criteria.* DoD 5200.28-STD 1985.

[44] Department of Trade and Industry . *Information Security Breaches Survey.* 2002.

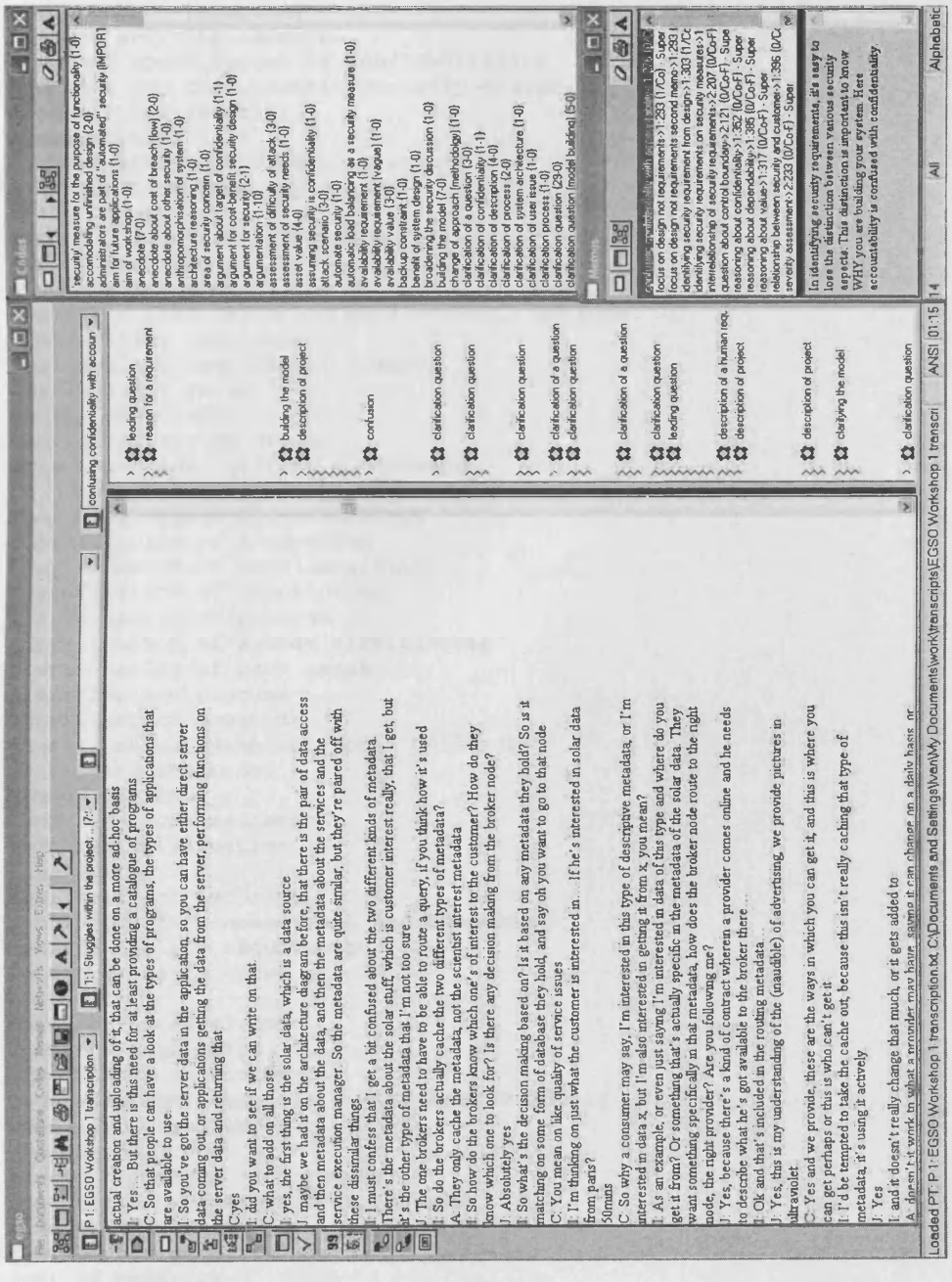
[45] Department of Trade and Industry . *Information Security Breaches Survey.* 2004. http://www.security-survey.gov.uk/

[46] Devanbu, P. T. & Stubblebine, S. *Software Engineering for Security: a Roadmap.* Proceeding of the Conference on the Future of Software Engineering 2000. pp 227-239. ACM Press.
[47] Dey, I. *Qualitative Data Analysis: A User-Friendly Guide for Social Scientists.* London: Routledge and Kegan Paul 1993.
[48] Dhillon, G. & Backhouse, J. *Current directions in IS security research: towards socio-organizational perspectives.* Information Systems Journal 11 2001. pp 127-153.
[49] Dick, B. & Swepson, P. *Appropriate validity and its attainment within action research: an illustration using soft systems methodology.* 1994. http://www.scu.edu.au/schools/gcm/ar/arp/sofsys2.html
[50] Dourish, P. & Redmiles, D. *An Approach to Usability Security Based on Event Monitoring and Visualization.* Proc.New Security Paradigms Workshop (Virginia Beach, VA) 2002.
[51] Edwards, J. *Cyber trust and crime prevention: towards generally accepted digital principles.* 2004. http://www.foresight.gov.uk
[52] Feldman, A. *Erzberger's dilemma: Validity in action research and science teachers' need to know.* Science education 1994. 78(1) , pp 83-101.
[53] Flechais, I. & Sasse, M. A. *Developing Secure and Usable Software.* OT2003 2003.
[54] Flechais, I., Sasse, M. A., & Hailes, S. M. *Bringing Security Home: A process for developing secure and usable systems.* New Security Paradigms Workshop 2003.
[55] Foster, I. *What is the Grid? A Three Point Checklist.* 2002. http://www.globus.org/research/papers.html
[56] Friedman, B. & Felten, E. *Informed Consent in the Mozilla Browser: Implementing Value-Sensitive Design.* Proceedings of the Thirty-Fifth Annual Hawaii International Conference on System Sciences 2002. http://www.ischool.washington.edu/networksecurity/outcomes.html
[57] Fukuyama, F. *Social Capital and the Civil Society.* 2nd Conference on Second Generation Reforms 1999. Washington, DC: IMF.
[58] Garfinkel, S. & Spafford, G. *Practical UNIX and Internet Security.* 1996. O'Reilly.
[59] Glaser, B. G. & Strauss, A. *The Discovery of Grounded Theory: Strategies for Qualitative Research.* 1967. Chicago, Aldine.
[60] Gollman, D. *Computer Security.* 1999. Wiley.
[61] Gryce, C. *Security Lessons from the EGSO Project - an Experience Report.* 2003. http://www.cs.ucl.ac.uk/staff/C.Gryce/papers/PracticalSecurityFinal.ppt
[62] Hammersley, M. *Some notes on the terms 'validity' and 'reliability'.* British Educational Research Journal 1987. 13(1) , pp 73-81.
[63] Herrmann, P. & Krumm, H. *Object-Oriented Security Analysis and Modeling.* Proceedings of the 9th International Conference on Telecommunication Systems - Modeling and Analysis 2001. pp 21-32.
[64] Hitchings, J. *Achieving an Integrated Design: The Way forward for Information Security.* Proceedings of the IFIP TC11 11th international conference on information security 1995.
[65] Hitchings, J. *A Practical solution to the complex human issues of information security design.* Proceedings of the 12th IFIP TC11 international conference on information security 1996.
[66] Institute of Electrical and Electronics Engineers. *IEEE Standard Computer Dictionary: A Compilation of IEEE Standard Computer Glossaries.* 1990.
[67] Institute of Electrical and Electronics Engineers. *Standard Glossary of Software Engineering Terminology.* Std 610.12-1990 1990.
[68] Irestig, M., Eriksson, H., & Timpka, T. *The Impact of Participation in Information System Design: A Comparison of Contextual Placements.* Proceedings Participatory Design Conference, Toronto Canada 2004.
[69] James, H. L. *Managing Information Systems Security: a Soft Aproach.* Proceedings of the Information Systems Conference of New Zealand 1996. IEEE Society Press, Los Alamitos, CA.
[70] Jarvinen, P. H. *Research questions guiding selection of an appropriate research method.* Proceedings of the 8th European Conference on Information Systems (ECIS 2000), Vienna, Austria. 2000.
[71] Johnson, B. M. *Why Conduct Action Research?* Teaching and Change 1995. 3(1) , pp 90-104.
[72] Jürjens, J. *UMLsec: Extending UML for Secure Systems Development.* LNCS 2003.
[73] Ka-Ping, Y. *User Interaction Design for Secure Systems.* 2002. http://zesty.ca/sid
[74] Kahn, D. *The Codebreakers.* 1967. Macmillan.
[75] Klein, H. K. & Myers, M. D. *A Set of Principles for Conducting and Evaluating Interpretive Field Studies in Information Systems.* MIS Quarterly 1999. 23 , pp 67-94.
[76] Landauer, T. K. *The Trouble with Computers: Usefulness, Usability, and Productivity.* 1996. Cambridge, MA: MIT Press.
[77] Lewin, K. *Action research and minority problems.* Journal of Social Issues 1949. 2 , pp 34-46.
[78] Martin, P. Y. & Turner, B. A. *Grounded Theory and Organizational Research.* The Journal of Applied Behavioural Science 1986. (22:2) , pp 141-157.
[79] McDaniel, G. *IBM Dictionary of Computing.* 1994. New York, NY, McGraw-Hill.
[80] McDermott, J. & Fox, C. *Using Abuse Cases for Security Requirements Analysis.* Proceedings of the 15th Annual Computer Security Applications Conference 1999.
[81] Mitnick, K. D. & Simon, W. L. *The Art of Deception: Controlling the Human Element of Security.* 2003. John Wiley & Sons Inc.
[82] Morgan, M. *Qualitative Research: A Package Deal?* The Psychologist: Bulletin of the British Psychological Society 1996. pp 31-32.
[83] Mumford, E. *Designing Human Systems - The Ethics Method.* 1983. **http://www.enid.u-net.com/C1book1.htm**
[84] Mumford, E. *Advice for an action researcher.* Information Technology and People 2001. 14 , pp 12-27.
[85] Mumford, E. & Weir, M. *Computer Systems in Work Design - The ETHICS Method.* 1979. Associated Business Press.
[86] National Institute of Standards and Technology. *Common Criteria for Information Technology Security.* Version 2.1 CCIMB-99-031 1999. **http://www.commoncriteria.org**
[87] Object Management Group. *Meta Object Facility (MOF) Specification.* Technical Report 2003.

[88] Object Management Group. *Unified Modeling Language version 1.5.* 2003. http://www.omg.org/technology/documents/formal/uml.htm
[89] Parker, D. B. *Fighting Computer Crime.* 1998. Wiley.
[90] Parsons, M. *Grid Computing.* Presentation at BCS Edinburgh 2003. http://www.edinburgh.bcs.org/02-03/grid_computing.pdf
[91] Pidgeon, N. & Henwood, K. *Grounded theory: Practical implementation.* Handbook of Qualitative Research Methods for Psychology and theSocial Sciences 1996. pp 86-101. Leicester, The British Psychological Society.
[92] Poulsen, K. *Mitnick to lawmakers: People, phones and weakest links.* 2000. http://www.politechbot.com/p-00969.html
[93] Putnam, R. D. *Bowling Alone: The Collapse and Revival of American Community.* 2000. New York: Simon & Schuster.
[94] Reason, J. *Human Error.* 1990. Cambridge University Press.
[95] Resnick, P. *Beyond Bowling Together: SocioTechnical Capital.* HCI in the New Millenium 2002. pp 242-272. Boston, MA: Addison-Wesley.
[96] Riegelsberger, J., Sasse, M. A., & McCarthy, J. *The Mechanics of Trust: A Framework for Research and Design.* International Journal of Human Computer Studies 2004. 62(3) , pp 381-422.
[97] Rousseau, D. M. *Managing the Change to an Automated Office: Lessons from Five Case Studies.* Office: Technology & People 1989. 4 , pp 31-52.
[98] Saltzer, J. H. & Schroeder, M. D. *The protection of information in computer systems.* IEEE 1975.
[99] Sasse, M. A. *Eliciting and Describing Users' Models. of Computer Systems.* PhD Thesis 1997. http://www.cs.ucl.ac.uk/staff/a.sasse/thesis/Frontpage.html
[100] Sasse, M. A. *Computer Security: Anatomy of a Usability Disaster, and a Plan for Recovery.* CHI 2003 2003.
[101] Sasse, M. A., Brostoff, S., & Weirich, D. *Transforming the 'weakest link': a human-computer interaction approach to usable and effective security.* BT Technical Journal 2001. 19 , pp 122-131.
[102] Schein, E. H. *The Clinical Perspective in Fieldwork.* 1987. Newbury Park, CA, Sage Publications.
[103] Schneier, B. *Secrets and Lies.* 2000. John Wiley & Sons.
[104] Schneier, B. *Beyond Fear Thinking Sensibly about Security in an Uncertain World.* 2003. Copernicus Books.
[105] Siponen, M. *Designing Secure Information Systems and Software: Critical Evaluation of the Existing Approaches and a New Paradigm.* PhD Thesis 2002. http://herkules.oulu.fi/isbn9514267907/
[106] Siponen, M. & Baskerville, R. *A New Paradigm For Adding Security into IS Development Methods.* Eighth Annual Working Conference on Information Security Management & Small Systems Security, Las Vegas, Nevada, USA 2001.
[107] Siponen, M. T. *An Analysis of the Recent IS Security Development Approaches: Descriptive and Prescriptive Implications.* Information Security Management: Global Challenges in the New Millennium 2001. pp 101-123. Idea Group Publishing.
[108] Slaymaker, M., Politou, E., Power, D., Lloyd, S., & Simpson, A. *e-DiaMoND: Risk Analysis.* Proceedings of UK e-Science All Hands Meeting 2004. http://www.allhands.org.uk/2004/proceedings/papers/54.pdf
[109] Smetters, D. K. & Grinter, R. E. *Moving from the design of usable security technologies to the design of useful secure applications.* New Security Paradigms Workshop.September 23-26, 2002, Virginia Beach, VA 2002.
[110] Stainforth, D., Martin, A., Simpson, A., Christensen, C., Kettleborough, J., Aina, T., & Allen, M. *Security principles for public-resource modeling research.* 13th IEEE International Workshops on Enabling Technologies: Infrastructure for Collaborative Enterprises (WETICE'04) 2004.
[111] Stiles, W. B. *Quality Control in Qualitative Research.* Clinical Psychology Review 1993. 13 , pp 593-618.
[112] Straub, D. W. & Welke, R. J. *Coping with Systems Risk:Security Planning Models for Managerial Decision-Making.* MIS Quarterly 1998. 22:4 , pp 441-469.
[113] Strauss, A. & Corbin, J. *Basix of Qualitative Research: Techniques and Procedures for Developing Grounded Theory.* 1998. SAGE Publications Inc.
[114] Susman, G. I. & Evered, R. D. *An assessment of the scientific merits of action research.* Administrative Science Quarterly 1978. 23 , pp 582-603.
[115] Tait, P. & Vessey, I. *The Effect of User Involvement on System Success: A Contingency Approach.* MIS Quarterly, March 1988. pp 91-107.
[116] Viega, J. & McGraw, G. *Building Secure Software.* 2002. Addison-Wesley.
[117] Weirich, D. & Sasse, M. A. *Pretty Good Persuasion: A first step towards effective password security in the real world.* New Security Paradigms Workshop 2001.
[118] Whitten, A. & Tygar, J. D. *Why Johnny Can't Encrypt: A Usability Evaluation of PGP 5.0.* Proceedings of the 8th USENIX Security Symposium, August 1999, Washington 1999.
[119] Willig, C. *Introducing Qualtiative Research in Psychology: Adventures in Theory and Method.* 2001. Buckingham, Open University Press.
[120] Wong, E. Y. W. *A Study of User Participation in Information Systems Development.* 1994. **www.is.cityu.edu.hk/Research/WorkingPapers/paper/9405.pdf**
[121] www.grid.org. 2004.
[122] Zakon, R. H. *Hobbes' Internet Timeline v8.0.* 2005. http://www.zakon.org/robert/internet/timeline/
[123] Zurko, M. E. & Simon, R. T. *User-Centered Security.* New Security Paradigms Workshop 1997.

Appendix A
Grounded Theory Analysis

This appendix contains documents relevant to the Grounded Theory analysis reported in this thesis. Given the large volume of data generated in these analyses, elements of the EGSO analysis are shown here as a representative sample of the overall analysis. The following screen shot shows a sample EGSO transcript together with associated codes and memos.



The following list details all the codes that were generated in the EGSO analysis.

```
Codes
=====
'security' measure for the purpose of functionality
accomodating unfinished design
administrators are part of "automated" security (IMPORTANT)
aim for future applications
aim of workshop
anecdote
anecdote about cost of breach (low)
anecdote about other security
anthropomorphisation of system
architecture reasoning
area of security concern
argument about target of confidentiality
argument for cost-benefit security design
argument for security
argumentation
assessment of difficulty of attack
assessment of security needs
asset value
assuming security is confidentiality
attack scenario
automate security
automatic load balancing as a security measure
availability requirement
availability requirement (vague)
availability value
backup constraint
benefit of system design
broadening the security discussion
building the model
change of approach (methodolgy)
clarification of a question
clarification of confidentiality
clarification of description
clarification of process
clarification of system architecture
clarification of user issue
clarification process
clarification question
clarification question (model building)
clarifying the model
Communication
communication resolution
comparative valuation
complexity
complexity of security
confidentiality assessment (unstructured)
confidentiality requirement
confidentiality value
conflict
conflict in development
conflict resolution
conflicting requirements
confusing confidentiality with accountability
confusing requirement
confusion
control boundary (for people)
cost benefit assessment of security
cost effectiveness
cost of bad security
cost of security
```

cost of security measure
countermeasure
critical counterargument about reasoning about backup
culture clash
current state of security
current vs future environment
customer funding
customer misunderstanding of design
customer requirement (opinion)
customer viewpoint
definition of accountability
definition of confidentiality
definition of dependability
definition of integrity
description of a human requirement
description of active monitoring
description of area which is vulnerable to attack scenario
description of culture
description of design mechanism
description of environment
description of people in the project
description of project
description of security measure
description of specific environment
description of third party
design
design hope
design of system is not finalised
design requirement
different interpretation of same architecture
different policies
difficulty with different modeling techniques
diffusion of responsibility
disagreement
disagreement about system architecture
distinction between design and requirements
distinction between stakeholders
enforcement vs trust
equating confidentiality with security
establishing security responsibilities
establishing security responsibility (in the product)
evidence of increase security knowledge
experience about project
explaining the modeling technique
explanation of asset
explanation of integrity
explanation of need for user inclusion
explanation of reasoning about vulnerabilities
explanation of security concepts
explanation of social countermeasures
explanation of term
explanation of the model
explanation of valuation process
explanation of vulnerability
financial incentive for security
financial responsibility
focus on a single issue
focus on design not requirements
focus on security requirement not current design
follow methodical process
functionality and security
grid specific aspect
hope about effectiveness of security
identify impact of scenario

identifying assets
identifying dependencies
identifying relative importance of security requirements
identifying security requirement from design
identifying security requirements on security measures
identifying severity of vulnerability
identifying user tasks (process)
identifying users (process)
identifying value
impact scenario
implication of human monitoring
implicit requirement of the system
incorrect assumption about security
inferential question
information about vulnerability
integrity valuation
interrelationship of security requirements
issue of granularity of model
issues about functional system desgin
justification for disagreement
knowledge of security
knowledge of system
lack of control
lack of incentive to apply security
lack of knowlege
leading question
leading question about users
legal and financial responsibility
legal constraint
Liability
means of reasoning value
miscommunication
misinterpreting the target of confidentiality
mistakes about security
misunderstanding of security vulnerability
misunderstanding the model
modelling users
motivation for low customer buy-in
motivation for security
motivation for security action
need for policy
new security concern
new security issue from grid specific environment
non-technical security measures
obstacle for security
on-the-spot design
open question
opinion about design
opinion about security design
opinion about user requirement
opinion of stakeholder culture
outsourcing vs internal security
paraphrase of technical requirement
participant debate about value
participant description of AEGIS
past field knowledge
people in security
perception of stakeholder position
perception of user group
personal interest in security
personal opinion
personal thoughts about policy
personal view of system design
position of stakeholder is unknown

possible design solutions
possible future technical solution to current problem
potential design
potential human security design
pragmatic approach to security
previous work
prior experience of security
prior involvement with security
problem with security
process helping with functional aspects
project requirement
project requirement in conflict with security
proposed human system
question about availability
question about backup reasoning
question about confidentiality
question about control boundary
question about functionality
question about functionality vs security conflict
question about future development
question about integrity
question about model
question about physical security network topography
question about policy
question about process
question about requirement
question about security mechanism
question about stakeholder needs
question about system environment
question about undocumented requirement
question about user cost and benefit
question about user motivation
question about user participation
question about user tasks
question about users
question about value
quote
reason for a requirement
reason for security
reasoning about availability
reasoning about backup
reasoning about confidentiality
reasoning about cost-benefit
reasoning about dependability
reasoning about impact of scenario
reasoning about integrity
reasoning about project future
reasoning about proposed solution
reasoning about security
reasoning about security aims
reasoning about security measure
reasoning about security priorities
reasoning about severity of vulnerability
reasoning about stakeholder viewpoint
reasoning about system architecture
reasoning about system design
reasoning about value
reasoning about vulnerabilities
refining security requirement
regulatory body
relationship between functionality and security
relationship between security and customer
relevance of vulnerability
reputation

reputation cost
requirement
requirement for a particular service
requirements gathering
responsibility
responsibility boundary
responsibility of users
scale impact on system design
scenario
security decision
security discussion leading to functionality questions
security expert viewpoint
security for trust
security is equated to confidentiality
security measure
security meta-requirement (low buy in for a certain user group)
security priority (low)
security requirement
security requirement (unstructured)
security requirement apparent from model
security requirement conflict
security solution for lower user cost
security vulnerability
severity assessment
simplifying end-user security
simplifying the model
social countermeasure
software engineering issue
Stakeholder
stakeholder conflict
stakeholder culture
stakeholder difficulty
stakeholder pressure
stakeholder requirement (functional)
stakeholder security requirement
stakeholder viewpoint
summary of vulnerabilities
system requirement
target of vulnerability
technical difficulty of synchronising replicas
technnical requirement
trust and confidence
trust and policy
trust and security
trust relationship
unclear current system design
understanding a vulnerability
understanding of human role in security measure
understanding of integrity
understanding of security
understanding the model
undocumented requirement
unranked security requirement
unrealistic expectations (naive)
user action
user base
user cost
user issue
user outside control boundary
user task
users undefined
visibility of security
vulnerability
worry about personal conviction

Presented here are the details of the memos generated during the analysis. Memos serve as a means of documenting the thoughts of the researcher throughout the analysis, and thereby serve as a means of documenting reflexivity.

```
Memos
=====

confusing confidentiality with accountability
   In identifying security requirements, it's easy to
   lose the distinction between various security
   aspects. This distinction is important to know
   WHY you are building your system. Here
   accountability is confused with confidentiality.

focus on design not requirements
   It is easy to look at the design of the system to
   modify or justify the value of a security
   requirement. We did this so it must be that. This is
   not necessarily the best way. Requirements should
   be about what the system should aim to achieve,
   the design is about what you are building to
   accomplish this

focus on design not requirements second memo->
   May also be a means of avoiding responsibility. i.e.
   we're building this so I'm not going to a) take the
   responsibility for it b) make an objective valuation
   of the need for it if it conflicts

identifying security requirement from design->1:303 {1/Co} - Super
   As opposed to not taking responsibility by pointing
   at current design, here is the notion of
   identifying an implicit requirement by looking at a
   design.

identifying security requirements on security measures
   It's interesting to try to identify integrity and
   availability requirements on a measure whose
   purpose is to provide integrity and availability.
   Security requirements that make more sense are
   those of dependability and accountability
   (monitoring, reaction...)
   Maybe there's different classes of security
   requirements for different classes of assets.
   dependability also seems tied to risk...

interrelationship of security requirements
   whilst identifying dependencies between different
   assets, some security properties are dependent on
   different other security properties. For example the
   availability of data is dependent on the integrity of
   the components delivering that data.

question about control boundary->
   control is an interesting addition to the decision
   making, knowledge, authority mix.
   It's to do with authority to institute change.
   Without control, security is seen as a barrier to
   customer adoption, and hence the need to design
   the system so that there is little need for 'costly'
```

security, This seems to be the argument for inobtrusive security - not the fact that it's hard to use, the fact that it can't be enforced. That's a difference in philosophy, either you force people to follow security policy (with money) or you make it so as little security as possible is necessary.

reasoning about confidentiality
confidentiality seems to dominate security discussion, both in complexity and in interest - it could be because confidentiality is the hardest to enforce and understand? integrity and availability are at first glance easier to handle, you're ensuring a positive result, something will happen (results will be accessible, results will be the same as before). Confidentiality is a negative result, people will not see or gain access to this...

reasoning about dependability
dependability seems to be the integrity of the action that a system performs - as opposed to the integrity of the system. So while the system might have integrity, factors from outside the system may still affect it in such a way that it does not behave in the correct manner, whilst being technically correct.

reasoning about value
there is a subtle distinction between a security rating and the value of an asset. The rating reflects the value of the asset from a particular point of view. For example, an asset may have a financial value, but a confidentiality breach may not lose any money, conversely an integrity breach may lose more than the monetary value of the asset.

relationship between security and customer
on the one hand the project wants security because it wants to be trusted by its customer, on the other it does not want security to interfere with the customer take up. Interesting paradox. The key to resolving this is to liaise with the customer to identify what they want.

severity assessment
the severity of a vulnerability being exploited can be illustrated through scenarios. The likelihood of these scenarios actually occuring is not captured in this excercise

stakeholder viewpoint
"they want egso to be a kind of shield for them"

user outside boundary (IMPORTANT)
Boundary setting is important in the diffusion of responsibility. They are not my problem because they are not part of what I have to consider.