Eliciting policy requirements for critical national infrastructure using the IRIS framework.

FAILY, S. and FLÉCHAIS, I.

2011

Eliciting Policy Requirements for Critical National Infrastructure Using the IRIS Framework

Shamal Faily, University of Oxford, UK

Ivan Fléchais, University of Oxford, UK

Abstract

Despite existing work on dealing with security and usability concerns during the early stages of design, there has been little work on synthesising the contributions of these fields into processes for specifying and designing systems. Without a better understanding of how to deal with both concerns at an early stage, the design process risks disenfranchising stakeholders, and resulting systems may not be situated in their contexts of use. This paper presents the IRIS process framework, which guides technique selection when specifying usable and secure systems. The authors illustrate the framework by describing a case study where the process framework was used to derive missing requirements for an information security policy for a UK water company following reports of the Stuxnet worm. The authors conclude with three lessons informing future efforts to integrate Security, Usability, and Requirements Engineering techniques for secure system design.

Keywords: *Computer Aided Integration of Requirements and Information Security (CAIRIS), Integrating Requirements and Information Security (IRIS), Knowledge Acquisition in autOmated Specification (KAOS), Misuse Cases, Personas*

1. INTRODUCTION

There is no longer any obvious reason why designing secure and usable systems should be so difficult, especially when guidance on applying Security and Usability Engineering best practice is no longer restricted to the scholarly literature. Several years ago, Nielsen claimed that cost was the principal reason why Usability Engineering techniques are not used in practice (Nielsen, 1994), but technology advances have reduced the financial costs of applying such techniques. Similarly, practical techniques for identifying and mitigating security problems during system design are now available to developers in an easy to digest format (e.g., Schneier, 2000; Swiderski & Snyder, 2004).

Problems arise when considering how to use these approaches as part of an integrated process. Accepted wisdom in software engineering states that requirements analysis and specification activities should precede other stages in a project's lifecycle (Ghezzi et al., 2003). However, Information Security and HCI

DOI: 10.4018/jsse.2011100101

proponents argue that their techniques should instead come first. For example, ISO 13407 (ISO, 1999) states that activities focusing on the collection of empirical data about users and their activities should guide early design, but security design methods such as Braber et al. (2007) suggest that such stages should be devoted to high-level analysis of the system to be secured. Invariably, the decision of what concern to put first is delegated to the methodology followed by a designer. The designer has many approaches to choose from, some of which include treatment for security or usability concerns. To date, however, no approach treats both security and usability collectively, beyond treating them both as generic qualities contending with functionality.

The IRIS (Integrating Requirements and Information Security) framework was first introduced by the authors in Faily and Fléchais (2009) to explore the challenges of designing systems with both information security and HCI in mind. This framework encompassed three elements: a meta-model for usable secure requirements engineering (Faily & Fléchais, 2010), a user-centered design method (illustrated in Faily & Fléchais, 2010), and complementary tool-support (Faily & Fléchais, 2010). However, although the second element was described as a *method*, this is more aptly defined as a *methodology*. While a method describes a concrete procedure for getting something done, a methodology is a higher level construct motivating the need for choosing between different methods (Iivari et al., 1998). Because the terms method and methodology are used interchangeably, the principles of information system methodologies have been encapsulated in several process *frameworks* that have, in recent years, emerged in Software, Security, and Usability Engineering. A framework can be defined as a set of milestones indicating when artifacts should be produced, as opposed to a *process* describing the steps to be carried out to produce the artifacts (Haley, 2007).

In this paper, we present the IRIS process framework, which is used for selecting techniques for specifying usable and secure systems. Building on the meta-model described in Faily and Fléchais (2010), we describe the different perspectives of IRIS, and how IRIS concepts and techniques are situated within these in Section 3. We propose a number of exemplar techniques for each perspective, and describe modifications, which are necessary to situate them within an IRIS process. In Section 4, we describe how the IRIS process framework was used to devise a user-centered approach for eliciting information security policy requirements for a UK water company. The management imperative for responding to the Stuxnet worm (Control Engineering UK, 2010) meant that policy decisions needed to be made where there was both a lack of time for data collection and restricted stakeholder availability. Finally, in Section 5, we describe some of the lessons learned carrying out this study, which, we believe, inform future approaches for secure system design.

2. RELATED WORK

Although frameworks exist for dealing with security and usability as quality requirements (e.g., Chung et al., 2004), we are unaware of existing frameworks dealing explicitly with both usability and security from a requirements perspective. There have, however, been processes and frameworks purporting to deal with each.

2.1. RESCUE

RESCUE (REquirements with SCenarios for a User-centered Environment) is a user-centered Requirements Engineering process (Maiden & Jones, 2004). Although not explicitly defined as a framework, the earlier phases of RESCUE afford leeway in technique application. RESCUE consists of the following four concurrent system engineering streams: Human Activity Modelling, i* system modelling (Yu, 1995), Use Case and Scenario Analysis (Cockburn, 2001), and Requirements Management. Human Activity Modelling involves analysing the way work is carried out, and partitioning the analysis of the problem domain into different aspects, such

as the work domain, control task, and social organisation. When the system boundary has been agreed, i* models are used to map both the dependency network between actors, and the intentional description of activities, and the rationale relationships between actors and related resources, goals, and tasks. Use cases are then identified based on actor or system objectives, which are graphically represented in a use case model, and authored using a template based on style guidelines prescribed by Cockburn (2001). Requirements elicited as part of RESCUE are specified using a template based on the Volere Requirements shell (Robertson & Robertson, 2009) and managed using commercial requirements management tools.

RESCUE implicitly assumes that a secure and usable system will result by following its prescribed guidelines. Security and usability are both considered as non-functional requirements, and while the activity-centric nature of RESCUE means it is likely that the system's core functionality will be situated for its actors, the results of specifying security requirements may circumvent these activities. As it stands, security requirements may lead to use case changes, but it remains the task of the analyst to spot use cases, which become unusable as a result of security constraints. Moreover, when problems are identified, the analyst needs to manually modify and maintain the traceability relations between i* models and downstream artifacts. RESCUE also fails to stipulate specific techniques for dealing with security concerns, making it an exercise for the analyst to select both the appropriate techniques and the points to apply them.

2.2. SQUARE

SQUARE (System QUAlity Requirements Engineering) (Mead et al., 2005) is a methodology which aims to build security into the early stages of a project lifecycle. It is applied within the context of a project by carrying out several steps. After agreeing a consistent set of security terms, project security goals are agreed in participatory workshops. With the aid of assistance of both domain and security experts, artifacts are then developed to support the definition of security requirements; these may include use cases and misuse cases (Sindre & Opdahl, 2005). After performing a risk assessment to identify and category the threats that need to be mitigated, security requirements elicitation techniques are selected and then applied. These elicited requirements are categorised as essential or non-essential, and of system, software, or architectural significance. A requirements prioritisation technique is then selected and applied with the aid of stakeholders. The final step involves selecting and applying a requirements inspection technique.

Although the selection of elicitation techniques is described as contextual, the selection of techniques for developing artifacts is not. Moreover, while several artifacts are recommended, no explicit guidance is afforded on how different techniques might contribute to each other besides reports of SQUARE applications.

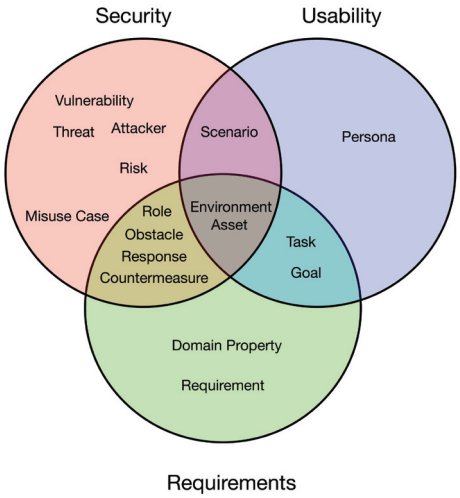
3. THE IRIS FRAMEWORK

3.1. IRIS Perspectives

To provide guidance for eliciting concepts from the IRIS meta-model, the meta-model was divided into three intersecting groups: Usability, Requirements, and Security. This is illustrated in Figure 1. These groups are called *perspectives* because each views the specification process subjectively through a lens coloured by its related concepts. Each perspective also views the design process in a different way, and techniques situated within them share certain characteristics.

The Usability perspective views the design process as a means of understanding how a system can be situated in its contexts of use. Consequently, the techniques situated by this perspective aim to model this understanding. The techniques associated with this perspective are often described as *user-centered*, but this is a misnomer. Instead, these techniques centre on one or more concepts within a context of use. Techniques within this perspective are also sensitive to human values. IRIS does not state

Figure 1. IRIS perspective concepts



what human *values* are conceptually, nor does it explicitly describe how these are portrayed from a stakeholder perspective. Instead, values are unpacked and elucidated by the usability techniques that elicit and analyse empirical data.

The Requirement perspective views the design process as a means of specifying the system being built. As such, the techniques situated by this perspective aim either to objectively specify the system being designed, or elicit models of how inter-related concepts situated in their contexts contribute to an objective specification. The techniques associated with this perspective can be described as *requirement-centered*. The ultimate ends of techniques situated in this perspective are prescriptive, objective, unambiguous, and bounded statements of system intent.

The Security perspective views the design process as a means of understanding how a system can be securely designed; specifically, how the design can make a system more secure. The techniques associated with this perspective can be described as *risk-centered*, because these aim either to understand what these are, or what design decisions are necessary to adequately respond to them. While what might be considered as *adequate* is subjective, it is, nonetheless, rational to the involved stakeholders. Consequently, techniques situated in this perspective not only discover what these risks are, but how they contribute to design in general.

The premise underpinning this framework is that the specification development process is hermeneutic rather than iterative. Nuseibeh alludes to this in his twin-peaks model (Nuseibeh, 2001), which talks about the dialogue between requirements and architectural activities. We assert that when specifying requirements, insights can occur at any time. While guidance is needed in a design process, the techniques adopted need to be agile enough to switch from one perspective to another should new insights dictate.

3.2. Converging Concepts

As Figure 1 illustrates, these perspectives are not mutually exclusive.

The concepts of Environment and Asset are shared by all perspectives, but for different reasons. In the Usability and Security perspective, these concepts are exploratory, while these reference or constraint concepts in the Requirements perspective.

Goals and tasks are conceptually shared between the Usability and Requirements perspectives; this is because user needs embodied by task descriptions in specific contexts eventu-

ally need to be reified by objective goals and requirements holding for all contexts of use. Similarly, roles, obstacles, responses, and countermeasures are conceptual interfaces between the Security and Requirements perspectives because they are used by techniques in both. Only scenarios are situated between the Security and Usability perspectives because, in both, these are used for exploring contexts of use rather than specifying elements within them.

3.3. Framework Techniques

We now consider techniques for eliciting the concepts from within these three perspectives. We have selected several candidate techniques, as summarised in Table 1. Because a *technique* is a particular way of carrying out a procedure or task, techniques can be construed as processes in their own right, albeit ones which require knowledge and experience in their use.

While this table does not purport to describe all techniques capable of eliciting IRIS concept, previous work has validated the effectiveness of these particular techniques when integrated as part of a larger process (Faily & Fléchais, 2010).

The framework is instantiated by devising an IRIS process: an ordered collection of techniques, informed by the context within which it is applied. This developmental context may be shaped by similar factors to those reported in Section 2.2 for SQUARE. Collectively, the techniques in a process should elicit all the concepts stipulated by the IRIS meta-model. There are no specific rules about what class of constraints should follow others, or what techniques can run concurrently with others. This is because the application of a technique from one perspective may lead to insights informing the analysis carried out using a technique in another. In general, however, the following principles apply:

- At the start of a process, one or more Requirements perspective techniques should be used to establish the scope of the system. Subsequent techniques assume that a context of design has been both agreed and specified.
- As the end product of a process is a specification, then a technique incorporating the validation of requirements is singularly or concurrently the last technique applied.

We describe these techniques in more detail in the following sections. For each technique, we present a brief overview of its key features, the rationale for using it, followed by interpretations and variations in its application as part of IRIS.

3.4. Grounded Theory

Grounded Theory is a qualitative data analysis technique for generating theory from observed real-world phenomena (Corbin & Strauss, 2008). These theories, and the sense-making activities associated with carrying them out, form the raw material that User-Centered Design artifacts can build upon. Although not traditionally construed as a design technique per se, it has been used for theory building in security and privacy research. For example, Fléchais (2005) used Grounded Theory to induce and refine a model of the factors affecting the design of secure systems, based on empirical data gathered from several different case studies.

Although we propose Grounded Theory be used for theory development in IRIS, induced theories are not developed for dissemination beyond the design team and system stakeholders. Instead, the sense-making associated with applying Grounded Theory is primarily used to support the elicitation and specification of other IRIS concepts.

3.5. Personas

Personas are archetypical specifications of indicative user behaviour. These were first introduced by Cooper (1999) to deal with programmer biases arising from the word *user*. These biases lead to programmers introducing assumptions causing users to bend and stretch to meet these needs; Cooper called this

Table 1. Techniques overview

Technique	Perspective	Input	Settings	Elicited Concepts	Output
Personas	• Usability	• Rich Picture	• Fieldwork • Analyst • Workshop	• Personas	• Persona specifications • empirical data
Activity Scenarios	• Usability	• Empirical data • Goals	• Workshop • Analyst	• Tasks • Scenarios • Usability Attributes	• Tasks
Grounded Theory	• Usability	• Empirical data	• Workshop • Analyst		• Qualitative models
Rich Pictures	• Requirements • Usability	• Empirical data	• Workshop • Analyst	• Roles • Environments	• Rich picture diagrams • Goals
KAOS	• Requirements • Security	• Empirical data • Goals	• Workshop • Analyst	• Goals • Obstacles • Domain Properties • Dependencies	• Goal Model
Volere	• Requirements	• Empirical data • Requirements	• Workshop • Analyst	• Requirements	• Requirements specification
AEGIS	• Security	• Empirical data	• Workshop • Analyst	• Assets • Security Attributes • Vulnerabilities • Attackers • Threats • Risks • Responses • Countermeasures	• Risk Analysis Model
Misuse Cases	• Security	• Risks	• Workshop	• Misuse Cases • Scenarios	• Risk Analysis Model • Task Model

phenomenon *designing for the elastic user*. We chose personas as a user proxy for IRIS because of their grounded representations of users. Although personas are viewed as a narrative structured by different behavioural variable types, their authorship is the least time-consuming element of their construction. In IRIS, data is explicitly collected and analysed to identify clusters of behaviour. This data is collected from representative users or related stakeholders, ideally within contexts the system needs to be situated in.

Personas are fully developed (or at least as fully developed as possible) at an early stage using qualitative data analysis techniques; in Faily and Fléchais (2010), we demonstrated an example of such an approach. However, friction may arise when personas are introduced into a project environment when developers are unhappy about system features directly appealing to some aspect of a persona. To deal with this concern, recent work by the authors on *Persona Cases* demonstrated how, during an IRIS process, Grounded Theory models can be bridged with persona narratives using argumentation models (Faily & Fléchais, 2010). These models provide a means for validating personas inspecting the assumptions made during qualitative data analysis.

3.6. Activity Scenarios

Activity scenarios centre around activities performed by users, rather than on the users themselves. In Scenario-Based Usability Engineering (SBUE) (Rosson & Carroll, 2002), these are preceded by problem scenarios, which

describe how hypothetical stakeholders tackle current practice; these scenarios may be based on empirical data or assumptions. Notable positive or negative consequences to stakeholders in problem scenarios are marked as *Claims*, which suggest possible design criteria rather than specific requirements. Activity scenarios are written to explore claims arising from problem scenarios. Claims may also be identified from activity scenarios, although these are used to describe the pros and cons of different approaches (Rosson & Carroll, 2008).

In IRIS, activity scenarios can be applied in individual and participatory settings, and used to explore the impact of possible design decisions. However, while the scenario component of activity scenarios and its contribution to the larger design process remains unchanged, IRIS is less generic about the role of stakeholders. Rather than treating users as hypothetical stakeholders, IRIS makes explicit assumptions about the use of personas, and the usability attributes of a persona (or personas) involvement with the scenario. Despite the implicit rationale links between activity scenario claims and requirements, personas and usability attributes act only in an exploratory role.

3.7. KAOS

The KAOS (Knowledge Acquisition in autOmated Specification) method (Dardenne et al., 1993) is a systematic approach for analysing, specifying, and structuring goals and requirements. KAOS defines a goal as a prescriptive statement of system intent; this is synonymous with the definition of goal in the IRIS meta-model, where a goal model represents an objective model of system requirements. This differs from agent-oriented goal modelling approaches such as i* (Yu, 1995) where goals describe an agent's intention; these may or may not be conducive to system objectives. The KAOS approach involves modelling goals from both a top-down and a bottom-up basis; goal models are annotated graphs of goals linked via AND/OR links. In addition to being refined to sub-goals, goals in KAOS may conflict with obstacles: conditions representing undesired behaviour and prevent an associated goal from being achieved (Lamsweerde & Letier, 2000). Like goals, obstacles can be modelled using an AND/OR refinement tree, where the root of the tree is an obstacle node associated with the goal it obstructs.

KAOS bridges between Usability and Security perspective techniques in three different ways. First, rather than operationalising goals by operations, we instead choose to operationalise goals using tasks. Although less precise, this modification affords the complementary use of goals and tasks for imparting the impact of goals on tasks, and vice versa in design sessions where non-technical stakeholders are present. Second, obstacles may obstruct goals, but we interpret these as both the accidental or intentional obstruction of goals. As Faily and Fléchais (2010) indicates, these can be associated with threats or vulnerabilities. Third, we apply recent work (Lamsweerde, 2009), which describes how *concern links* can be associated between goals and classes; these can be used to model traceability links between goals and any assets they reference or constrain.

3.8. Rich Pictures

Rich pictures are a diagrammatic way of representing the main concepts associated with a problem, and the relationships between them. Rich pictures are not based on any particular syntax, although the meaning of the imagery drawn be they boxes or sketches, are meaningful to both the stakeholders and the situation being described. Although representing problems with pictures is timeless, rich pictures were first introduced by Checkland and Scholes (1990) in their Soft Systems Methodology.

Checkland and Scholes propose the use of rich pictures to help obtain a *Root Definition*: a succinct definition of a system under investigation or being built. Despite the close relationship between a rich picture and a root definition, this latter concept has not been incorporated into IRIS for two reasons. First, contributing artifacts from a rich picture used in IRIS are

prescriptive high-level system goals, subject to refinement, rather than a joint descriptive and prescriptive statement about the system. Second, when a system is being scoped we are interested in *all* roles at play in a system, rather than differentiating between different classes of stakeholder. Understanding the politics associated with social classes may be useful from a Usability perspective, but not from a Requirements perspective unless these admit of requirements which need to be explicitly represented in a system specification. In such cases, the rationale behind these requirements also needs to be explicit, which, again, motivates the use of goals as an output from this stage of analysis rather than a root definition.

3.9. Volere Requirements Specification

Volere is a framework for Requirements Engineering encapsulating industry best practice for scoping, eliciting, specifying, and validating requirements (Robertson & Robertson, 2006). A characteristic element of Volere is its Requirements Specification template. This describes the format a specification document should take, and the elements of a single requirement. These elements are described as a *Requirements Shell* and include attributes such as a unique identifier, requirements type, description, rationale, priority, history, customer satisfaction, and fit criterion.

While the IRIS requirements specification template is largely based on that prescribed by Volere, attributes of an IRIS requirement do not include specific fields for history and customer satisfaction / dissatisfaction. In the case of the former, this is an attribute for tool-support, rather than something an analyst should have to manually maintain. In the case of satisfaction and dissatisfaction, it is arguably difficult to obtain reliable values for these attributes. The source of satisfaction or dissatisfaction may originate in early, contributory analysis, or may arise from a participant's subjective stake in the project. Although understanding the underpinnings of such value is useful, these are attributes best dealt with by Usability perspective techniques.

3.10. AEGIS

AEGIS (Appropriate and Effective Guidance for Information Security) (Fléchais et al., 2007) is a method for participative risk analysis. Under the guidance of a facilitator, and with the aid of one or more security experts, participants carry out an asset modelling exercise, identify risks from threats and vulnerabilities, select responses to these risks, and propose high-level security controls for risks requiring mitigation. Of the several approaches to dealing with the analysis and management of risks, which are typically frameworks like CORAS (Braber et al., 2007), AEGIS is a lightweight technique providing a simple, comprehensive, and straightforward structure for security analysis. For example, AEGIS prescribes the identification of vulnerabilities and threats, and the elicitation of requirements, but provides no explicit guidance on how these activities should be achieved. While this might be construed as a limitation, this is an advantage for IRIS as these activities can be carried out by complementary techniques.

When adopted in an IRIS process, focus groups are no longer a mandatory element of AEGIS. Although useful, participatory design activities can be time-consuming to participants, and the overall process of design may drag on if key participants are unavailable, or become compromised if participants are non-representative of users ultimately using the system. In lieu of multiple participants commenting on an asset model, insights are gleaned by interfacing AEGIS asset models with multiple techniques. For example, the asset-modelling phase in AEGIS may precede obstacle modelling in KAOS to identify vulnerabilities; these can be examined in more detail with other vulnerabilities and threats by AEGIS. Similarly, goal modelling to identify the requirements a countermeasure needs to satisfy may be injected as a stage between AEGIS risk response and countermeasure selection.

3.11. Misuse Cases

A misuse case is a sequence of actions, including variants that a system or other entity

can perform, interacting with misusers of the entity, and causing harm to stakeholders if the sequence is allowed to complete (Sindre & Opdahl, 2005). Misuse cases extend the UML use case diagram notation by extending the actor node as a stick figure which is a black, as opposed to white, head, and adding additional relations between use cases and misuse cases, e.g., threaten and mitigate. Like use cases, and scenarios in general, misuse cases add context to risk analysis by modelling the attacker, and describing how an attacker can exploit or misuse the system. The simplest way of eliciting misuse cases is informed brainstorming, guided by a security expert asking questions about the system likely to have weaknesses; this activity mirrors the way attackers might think (Hope et al., 2004).

Although IRIS obtains the same value from misuse cases as the literature suggests, it reaches the final end using a different means. Rather than using them to elicit and explore threats, IRIS uses misuse cases to validate risks. For a risk to be considered valid, a misuse case needs to be authored commensurate to the risk's impact, which, in turn, is commensurate to the attributes of the exploited vulnerability, and the threat taking advantage of this. If a valid scenario cannot be written based on this analysis, then the underlying analysis needs to be re-visited. The IRIS meta-model is such that, with suitable tool-support, it should only be necessary for analysts to write a misuse case narrative because contributing information, such as the misuse case objective and attacker can be determined from existing risk analysis data.

4. CASE STUDY: A PLANT OPERATIONS SECURITY POLICY

Information security policies in Critical National Infrastructure (CNI) organisations need to be balanced. The growth in technologies such as distributed control systems and smart grids have meant that media reports about cyber-warfare and terrorism have heightened the security awareness of senior managers. However, the impact of such policies extends beyond the board rooms and offices where they are drafted. Poorly written policies that constrain the ability of staff to carry out their day-to-day work might compromise operations, leading to the introduction of vulnerabilities to get around them. These problems can be compounded by unforeseen events causing organisations to re-think their current stance on information security. When under pressure, the perception that security design is time consuming may lead policy decisions to be driven by fear rather than rationality. Because few people are fired for making policies too secure, as long as usability and security continue to be treated as qualities to be traded off against each other, policies will err on the side of constraint over freedom of action.

To evaluate the IRIS framework, we wished to understand how successful Usability and Requirements Engineering techniques might be for eliciting organisational Information Security requirements.

Because this evaluation would take place in a real-world context rather than a controlled environment, this case study was carried out as an *Action Research* intervention (Lewin, 1946). Action Research is an iterative research approach involving the planning of an intervention, carrying it out, analysing the results of the intervention, and reflecting on the lessons learned; these lessons contribute to the re-design of the social action, and the planning of a new intervention. Although primarily used in social science and educational studies research, Action Research has also been used to validate security design methods (e.g., Fléchais et al., 2007). The objective of the intervention was to elicit and specify missing requirements for an information security policy, as indicated in section Introduction. For reasons of confidentiality, this company will hereafter be known as ACME.

An earlier version of the IRIS framework demonstrated how Usability and Security Requirements Engineering techniques could be aligned in system design without considering Security and Usability as trade-off concerns

(Faily & Fléchais, 2010). However, because this study took place over a period of several months, it is difficult to determine how useful these techniques might be when working to a tight deadline. In such situations, limited time is available for collecting empirical data and running focus groups or workshops.

The Action Research methodology used in this paper is that proposed by Baskerville (1999), who breaks an intervention into five distinct phases:

- Diagnosis: Identifying the influencing factors in the organisational context impacting the design of the intervention.
- Actions Planned: Devising the planned process to be adopted in order to meet the intervention's objectives.
- Actions Taken: Carrying out the planned steps taken as part of the intervention.
- Evaluating: Evaluating the outcome of the intervention.
- Specifying Learning: Stating the actions, which need to feed forward to future interventions or research.

For reasons of brevity, we will describe the actions planned and taken in Sections 4.3, 4.4, 4.5, and 4.6; the discussion in Section 5 constitute the results of the *Evaluating* and *Specifying Learning* phases for this intervention.

4.1. Influencing Factors

In July 2010, early reports of how the Stuxnet worm had infected several industrial plants around Europe began to appear. These reports shook up senior management at ACME for several reasons. First, a long held assumption that the obscurity of their SCADA (Supervisory Control and Data Acquisition) systems made them immune to security was dispelled; the virus explicitly targeted the same type of SCADA software used by ACME. Second, by combining knowledge of zero day threats with a realistic means of spreading the virus, i.e., via USB sticks, plant control software no longer seemed as isolated as it once was. Finally, although the motivation of the attacker was, at the time, unknown, the technical sophistication of the virus suggested that the virus was professionally developed to cause harm. While ACME didn't believe they were the virus' target, they were acutely aware that the impact of being infected was largely unknown. They did, however, agree that an effective means of mitigating the likelihood of being threatened was to devise a specific information security policy for those staff working in plant operations.

Although many of ACME's sites were unstaffed, the planned policy would cover staffed clean water plants and sewage works serving large urban areas. These plants were staffed by operators responsible for the running of the plant, and its treatment operations. Plant operators were acutely aware of the safety implications of clean and waste water treatment. If not properly treated, waste water effluent could have a significant impact on the ecosystem and the food chain. The clean water treatment processes are also critical enough that quality warnings are automatically forwarded to ACME chemists and quality assurance teams. Plant operators were also made aware of the security implications of deliberate attacks on the clean water infrastructure. Like other employees at ACME, information security communiques were regularly sent to all ACME staff, and police periodically visited clean water treatment plants due to the perceived risk of possible terrorist action. There was, however, a feeling held by the information security team that plant operators perceived the threats described in these communiques as irrelevant to their work.

The new security policy would need to cover both the existing infrastructure, and a new Enterprise SCADA system currently being rolled out to other parts of ACME. There were, however, two issues, which would need to be considered when designing policy requirements for this system. First, access to stakeholders working in this project was limited. The project relied on external contractors, several of whom were paid a substantial amount of money for their expertise. Their insight

would be required for this intervention, but their time needed to be carefully managed. Second, several technical requirements had been stipulated by the Enterprise SCADA system manufacturer. At the start of the intervention, it was unclear what impact these might have on the security policy, and ACME's ability to enforce it without compromising this new operating environment.

4.2. Approach Taken

Based on the influencing factors, we determined that the intervention needed to be completed in a timely manner; this would ensure that the initial analysis would be available to senior managers quickly. We also determined that stakeholders working at water treatment plants, from plant operators to managers, would need to be engaged in the process without underselling or overselling the importance of security and usability in policy decisions. Finally, design activities would need to be informed by the on-going design of the new Enterprise SCADA system, and access to resources working on the Enterprise SCADA project would need to be carefully managed.

To meet these criteria, we devised a user-centered process for eliciting policy requirements. This process was *user-centered* because of its early focus on the needs of the policy's users and tasks, and the grounding of these needs in empirical usage data. After agreeing the scope of the policy, a Fieldwork phase was undertaken; this involved holding in-situ interviews with users who would be affected by the policy at sites where the policy would be applied. This data was used to build a qualitative model of security perceptions held by plant operators. The results from this phase informed two further phases: Usability & Security Analysis, and Requirements & Risk Analysis.

Usability & Security Analysis entailed developing personas and, using scenarios, describing the typical activities they carried out. In parallel with this activity, KAOS goal trees were developed to model the policy requirements, and possible ways these requirements could be obstructed. These obstructions were modelled using KAOS obstacle trees. Where possible, obstacles were resolved at this stage using policy requirements. Risks were elicited on the basis of obstacles that could not be resolved without being first discussed by stakeholders.

The Requirements & Risk Review phase involved creating misuse cases (Sindre & Opdahl, 2005) to describe the impact of the identified risks, and holding a focus group with key stakeholders to agree possible policy requirements for mitigating them.

As the UML diagram in Figure 2 suggests, several different models were generated as part of this process. The artifacts elicited during the analysis and review phases were managed using the CAIRIS (Computer Aided Integration of Requirements and Information Security) Requirements Management tool (Faily & Fléchais, 2010). CAIRIS builds upon the IRIS meta-model (Faily & Fléchais, 2010), which describes how the concepts underpinning these artifacts are linked. As a corollary, entering data about these artifacts into the tool automatically generates the different models according to the meta-model relationships.

Once the scope had been agreed, a little less than two weeks were set aside for carrying out the Fieldwork phase and supporting qualitative data analysis activities. Usability & Security Analysis took place over the following 3-week period before initial policy requirements were available for the Requirements & Risk Analysis review.

Further information about the process and how the different artifacts were generated are described in subsequent sections.

4.3. Agreeing Scope

Existing documentation about ACME information security policies was provided as an input to this process. On the basis of this input data, an initial rich picture diagram of the policy scope was developed. Due to time constraints, this was developed off-site and distributed to stakeholders via ACME's Information Security Manager. Although preparation for Fieldwork

commenced during this stage, the scope of investigation was bounded only when the rich picture diagram was agreed with ACME. The feedback received from the ACME stakeholders involved word changes which seemed minor, but were semantically significant to ACME. For example, an association was drawn between one system in scope to another box named after the physical location of ACME's head office; ACME's telemetry group and their servers were located at this site. Although the association was valid, the box was renamed to Bunker to emphasise the data flow to the telemetry group rather than other groups located at the physical location; the name was commonly used to refer to the group because they were located in a bombproof building.

4.4. Fieldwork

The objective of the Fieldwork stage was to develop a qualitative model of plant operations security; this would be used to derive one or more personas representing plant operators for later design activities. We visited 4 different water-treatment works (2 clean water and 2 waste water) to hold in-situ qualitative interviews with plant operators and related stakeholders. Although these interviews were largely open-ended, high-level questions dealt with the nature of work undertaken, including what plant operators were responsible for, whom they worked with, and how they obtained help if necessary. Plant operators were also asked about important work items and activities, and the problems they often faced. Following the interviews, qualitative data analysis was carried out on the interview transcripts and, from this, a qualitative model of plant operator security perceptions was derived.

In addition to these fieldwork activities, goal modelling also commenced at this stage. The documents used to drive this activity included a draft security policy that ACME had prepared, an ACME information handling guidelines document, and ACME's organisational security policy. As the aim of the intervention was to elicit missing requirements from the first of these documents, this was the primary document used to elicit goals. For each policy recommendation in this document, a goal was defined. As they were elicited, a goal tree was induced based on statements, which relied on the satisfaction of other goals. Where supplemental documents were referenced, the referenced statements also formed the basis of goals.

In parallel with other activities, an asset model was progressively developed and, by the end of this stage, was mature enough to form the basis of analysing possible security issues. This asset model was based on the AEGIS Asset Model notation (Fléchais et al., 2007). The qualitative data analysis carried out indicated that the two prevalent contexts of interest to plant operations staff would be activities taking place during daylight hours (Day) and the hours of darkness (Night). With this in mind, assets and security values that stakeholders appeared to hold about them were modelled for each of these two contexts. Data about what constituted *Low*, *Medium*, and *High* value assets were based on ACME's own risk management documentation.

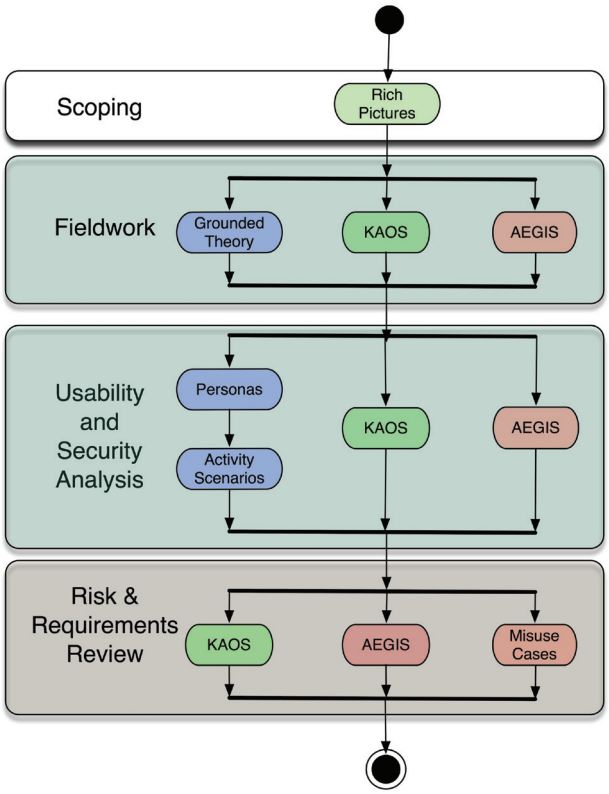
4.5. Usability and Security Analysis

4.5.1. Security Analysis

At this stage, the goal tree was analysed to find obvious vulnerabilities requiring further analysis. Although no obstacles were forthcoming, a number of goals suggested policy requirements needing to be present in order for them to be satisfied. One such requirement was *Authorised STCS network point data shall be available to authorised plant operators on the ACME portal*. This requirement arose from a goal stating that information about authorised network points should be available to authorised plant staff; this was necessary to allow plant staff to identify network points, which might be unauthorised.

Although no obstacles were obvious from the goal tree, examining the empirical data collected during the Fieldwork stage identified several vulnerabilities. Figure 3 provides

Figure 2. Policy requirements elicitation process



an example of how these were integrated into the goal tree. During the site-visits, cabinets containing network infrastructure were found in publicly accessible areas of certain plants. Based on this, the *Exposed ICT Cabinets* obstacle was introduced; this obstructed pre-existing goals for securing the physical infrastructure. This particular obstacle was mitigated with the requirement *Key ICT equipment shall be stored in a restricted access computer room*. In the figure, this requirement is abbreviated with the label ICAB-1 because it references the ICT Cabinets asset (abbreviated as ICAB).

When all possible vulnerabilities were mitigated, a threat analysis step was carried out to identify possible attackers and the threats they might carry out. From the empirical data, two classes of attacker were identified. The first related to thieves attempting to break into plants to steal scrap metal or other equipment. Several plant operators expressed concern about these attackers because the damage to monitoring equipment they cause is inevitably greater than the value of the items stolen. Plant operators were also worried about their own personal safety should they be required to confront them out-of-hours. A *Kit Theft* threat was defined to model the impact of this attack.

The second class of attacker arose from a general indifference that plant operators and engineers held about information security threats. Even after describing the recent reports of Stuxnet, participants interviewed were still unconvinced that "hackers" were as convincing a threat as the press and information security communiques would have them believe. Consequently, to portray an attacker that would be believable, a profile was developed based on a penetration tester that could, potentially, be commissioned by ACME; this attacker

was grounded in a number of open-source intelligence resources and texts on penetration testing. Based on this attacker, several threats were identified, such as war-dialling modems, *foot-printing* to determine information about possible ACME network services, and enumeration of possible passwords using known defaults for software applications. Although several obstacles were elicited based on these threats, no mitigating requirements could be identified without further discussing the threats and their consequences with ACME stakeholders.

4.5.2. Usability Analysis

A plant operator persona (Rick) was derived from the qualitative model developed in section Fieldwork using the Persona Case technique introduced in Section 3.5.

Once the personas were ready, 3 scenarios were developed to describe how Rick would carry out his activities during the Day and Night contexts; these scenarios were modelled as tasks in CAIRIS, and textual narratives described how the task was carried out in each context. For example, the narrative associated with the *Resolve reservoir alarm* task during the Day context was as follows:

Rick looks at the SCADA monitor nearest to him and notices that the levels of the reservoir nearby are unusually high. When the level gets too high, the entire works need to be shutdown. In this situation, Rick knows exactly what to do. After stopping the alarm, Rick logs into the ICT PC next to the SCADA workstation, and clicks on the Xtraview icon. After logging into Xtraview, he finds the location of a pumping station 10 miles upstream on the map and connects to it. After a few moments, he masters the main pump before switching it off. Rick then returns the pump to its normal slave setting before shutting down Xtraview. The alarm periodically starts and stops again but, after about an hour, the reservoir level normalises again.

Although the above task was identical for both Day and Night contexts, there were a number of variations in other tasks. This was due to the necessity for on-call technicians to resolve problem that on-site staff could have fixed during working hours.

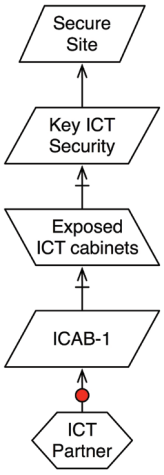
4.6. Requirements and Risk Analysis

The final stage involved running a focus group with ACME stakeholders and presenting the misuse cases encapsulating the unmitigated risks. These stakeholders were operational managers responsible for plant security and a representative ICT manager. Because only a limited amount of time was available, the presentation of the analysis was centred around a discussion of risks of most interest to ACME: a virus-infected SCADA workstation, and a site-break in. The misuse case associated with each risk was presented, discussed and, based on the outcome, mitigating strategies were proposed. For each discussed misuse case, a misuse case model was developed; as indicated at the beginning of the section, each model was generated automatically by CAIRIS.

For the first risk, a policy requirement was added to remove USB access to SCADA workstations. Responsibility for the second risk was provisionally passed to ACME's facilities management department.

After updating the CAIRIS model based on these discussions, a revised specification document was re-issued to ACME. Because of the limited time available during the focus group, a more detailed review of the analysis took place at ACME's head-office several weeks later. In this one-to-one session with ACME's Information Security Manager, the goal model and elicited policy requirements were validated, and the risk analysis results were reviewed. The purpose of this session was to ensure that all goals and requirements were assigned a responsible role and responses were elicited for each risk.

Figure 3. Goal tree fragment associated with exposed ICT Cabinets vulnerability



On completion of the study, 106 separate policy goal statements had been elicited. The vast majority of these were associated with the Day context; this reflects the many day-to-day concerns that participants had with regards to security policy coverage. Similarly, the threats most evident from the empirical data were based on attacks expected to take place during daylight hours.

5. DISCUSSION

We believe the outcome of the intervention was a success for two reasons. First, despite the challenging time constraints, the study was completed comparatively quickly without compromising the quality of the artifacts created. As Section 4 reported, the study was largely completed in just over one month with ACME involvement limited to occasional email discussions, in-situ interview participation and a single focus group session to discuss key misuse cases. Second, all elicited policy requirements were accepted by ACME. Moreover, the design models created during the study were used to help with other security issues in ACME. For example, the Rick persona was subsequently used to inform design decisions about user account profiles. In the following sections, we describe three findings from this case study that, we believe, inform future efforts to harmonise Security, Usability, and Requirements Engineering techniques as part of secure system design.

5.1. Fieldwork is Security Sense-Making

Focusing on security design activities at the same time as Fieldwork activities heightened awareness of possible threats and vulnerabilities at an early stage. For example, on one site-visit, questioning the purpose of one particular PC led to the discovery that not only was it superfluous to plant operations, but the modem attached to it was vulnerable to war-dialling attacks. On another visit, a chance conversation about a car driving up to the plant's main gate on a CCTV screen led to the discovery that the plant had a second gate, and the access control system for this plant entrance was particularly weak. Based on these observations, we believe that fieldwork makes two important contributions to security design. First, de-familiarisation activities associated with in-situ interviews leads to identification of hitherto unseen affordances; these affordances are potentially exploitable by attackers. Second, opportunities for identifying

and analysing vulnerabilities happen at any time and, quite often, such insights might have otherwise remained hidden.

5.2. Threats Without Up-Front Threat Analysis

Useful information about attackers and threats was collected without an upfront threat elicitation exercise. This is because threat analysis could be informed by the sense-making activities associated with other analysis. There are two reasons why this is an improvement over security design methods relying solely on anecdotal information from stakeholders or security experts to derive threats (e.g., Braber et al., 2007; Fléchais et al., 2007). First, threat elicitation is not exclusively contingent on participatory approaches, which rely on getting stakeholders together in a single location. Second, the task of eliciting attackers followed by threats is easier than trying to elicit attacks in their own right. While the empirical data can point to possible attackers, further research is often necessary to determine what threats these attackers can give rise to and, as a result, which assets might be threatened.

5.3. Misuse Cases as Cases

Misuse cases were useful for spotting more general fallacies made when arguing against the feasibility of a risk. In particular, we noticed a tendency by stakeholders to undermine the impact of the threat or the severity of the vulnerability by focusing solely on the threat's likelihood and the asset directly under threat. During discussion of the *Site break-in* misuse case, some participants highlighted the limited number of staffed sites, coupled with the relatively high frequency of PC theft, as a reason why incorporating policy requirements to mitigate this risk might be infeasible. However, when it was highlighted that the PCs themselves were less important than the monitoring they facilitated, and that the quantity of staffed and unstaffed sites had little bearing on the impact of the risk, it was agreed to transfer responsibility of the risk rather than ignore it. When discussing the risk during the follow-up meeting with ACME's Information Security Manager, it was highlighted that transferring the risk in its entirety was inappropriate. Consequently, the policy goals related to securing physical sites were reviewed to determine which were the responsibilities of ACME's facilities management department, and which needed to be pro-actively managed by ACME's own security team.

6. CONCLUSION

In this paper, we have presented the IRIS process framework; this builds upon the IRIS meta-model by grouping concepts by Usability, Security, and Requirements perspectives. Based on these perspectives, the framework guides the construction of individual IRIS processes for eliciting secure system requirements using a number of exemplar techniques. To illustrate the framework, we have described a case study where the framework was used to devise a process for eliciting requirements for an information security policy for a CNI organisation. This paper has made three particular contributions towards improved harmonisation between usability, security, and software engineering.

First, we have successfully evaluated the efficacy of integrating selected usability, security, and Requirements Engineering techniques. Specifically, we have demonstrated that rather than adopting a single process model, judiciously selecting and applying appropriate design techniques for the organisational context can be economical in terms of manpower and time.

Second, we have motivated and presented the results of an Action Research intervention in a real-life context of contemporary interest; specifically, CNI following the initial outbreak of the Stuxnet worm.

Finally, we have illustrated how, by focusing on the up-front development of Usability, rather than Security, Engineering artifacts, we can re-use the sense-making activities and empirical data to elicit hitherto unseen vulner-

abilities. From this, we can also glean insights about possible system attackers and threats.

ACKNOWLEDGMENT

The research described in this paper was funded by EPSRC CASE Studentship R07437/CN001. We are very grateful to Qinetiq Ltd for their sponsorship of this work.

REFERENCES

Baskerville, R. L. (1999). Investigating information systems with action research. *Communications of the Association for Information Systems*, *2*(19), 1–32.

Checkland, P., & Scholes, J. (1990). *Soft systems methodology in action*. New York, NY: John Wiley & Sons.

Chung, L., Nixon, B. A., Yu, E., & Mylopoulos, J. (2000). *Non-functional requirements in software engineering*. Boston, MA: Kluwer Academic.

Cockburn, A. (2001). *Writing effective use cases*. Reading, MA: Addison-Wesley.

Control Engineering, U. K. (2010, 20 July). *'Stuxnet' Trojan Targets Siemens WinCC*. Retrieved from http://www.controlenguk.com/article.aspx?ArticleID=35267

Cooper, A. (1999). *The inmates are running the asylum: Why high tech products drive us crazy and how to restore the sanity* (2nd ed.). Upper Saddle River, NJ: Pearson Higher Education.

Corbin, J. M., & Strauss, A. L. (2008). *Basics of qualitative research: techniques and procedures for developing grounded theory*. Thousand Oaks, CA: Sage.

Dardenne, A., van Lamsweerde, A., & Fickas, S. (1993). Goal-directed requirements acquisition. *Science of Computer Programming*, *20*(1-2), 3–50. doi:10.1016/0167-6423(93)90021-G

den Braber, F., Hogganvik, I., Lund, M. S., Stølen, K., & Vraalsen, F. (2007). Model-based security analysis in seven steps - A guided tour to the CORAS method. *BT Technology Journal*, *25*(1), 101–117. doi:10.1007/s10550-007-0013-9

Faily, S., & Fléchais, I. (2009). Context-sensitive requirements and risk management with IRIS. In *Proceedings of the 17th IEEE International Requirements Engineering Conference* (pp. 379-380).

Faily, S., & Fléchais, I. (2010). A meta-model for usable secure requirements engineering. In *Proceedings of the 6th International Workshop on Software Engineering for Secure Systems* (pp. 126-135).

Faily, S., & Fléchais, I. (2010). Barry is not the weakest link: Eliciting secure system requirements with personas. In *Proceedings of the 24th British HCI Group Annual Conference on People and Computers: Play is a Serious Business* (pp. 113-120).

Faily, S., & Fléchais, I. (2010). The secret lives of assumptions: Developing and refining assumption personas for secure system design. In R. Bernhaupt, P. Forbrig, J. Gulliksen, & M. Lárusdóttir (Eds.), *Proceedings of the 3rd Conference on Human-Centered Software Engineering* (LNCS 6409, pp. 111-118).

Faily, S., & Fléchais, I. (2010). Towards tool-support for usable secure requirements engineering with CAIRIS. *International Journal of Secure Software Engineering*, *1*(3), 56–70. doi:10.4018/jsse.2010070104

Fléchais, I. (2005). *Designing secure and usable systems*. Unpublished doctoral dissertation, University College London, London, UK.

Fléchais, I., Mascolo, C., & Sasse, M. A. (2007). Integrating security and usability into the requirements and design process. *International Journal of Electronic Security and Digital Forensics*, *1*(1), 12–26. doi:10.1504/IJESDF.2007.013589

Ghezzi, C., Jazayeri, M., & Mandrioli, D. (2003). *Fundamentals of software engineering*. Upper Saddle River, NJ: Prentice Hall.

Haley, C. B. (2007). *Arguing security: A framework for analyzing security requirements*. Saarbrücken, Germany: VDM Verlag Dr Müller.

Hope, P., McGraw, G., & Anton, A. I. (2004). Misuse and abuse cases: getting past the positive. *IEEE Security & Privacy*, *2*(3), 90–92. doi:10.1109/MSP.2004.17

Iivari, J., Hirschheim, R., & Klein, H. K. (1998). A paradigmatic analysis contrasting information systems development approaches and methodologies. *Information Systems Research*, *9*(2), 164–193. doi:10.1287/isre.9.2.164

International Organization for Standardization (ISO). (1999). *ISO/IEC 13407: Human-centered design processes for interactive systems*. Geneva, Switzerland: ISO.

Lewin, K. (1946). Action research and minority problems. *The Journal of Social Issues*, *2*(4), 34–46. doi:10.1111/j.1540-4560.1946.tb02295.x

Maiden, N., & Jones, S. (2004). *The RESCUE requirements engineering process: An integrated user-centered requirements engineering process (Version 4.1)*. London, UK: City University.

Mead, N. R., Hough, E. D., & Steheny, T., II. (2005). *Security quality requirements engineering (SQUARE) methodology* (Tech. Rep. No. CMU/SEI-2005-TR-009). Pittsburgh, PA: Carnegie Mellon Software Engineering Institute.

Nuseibeh, B. (2001). Weaving together requirements and architectures. *Computer*, *34*(3), 115–117. doi:10.1109/2.910904

Robertson, J., & Robertson, S. (2009, January 14). *Volere requirements specification template.* Retrieved from http://www.volere.co.uk/template.htm

Robertson, S., & Robertson, J. (2006). *Mastering the requirements process*. Reading, MA: Addison-Wesley.

Rosson, M. B., & Carroll, J. M. (2002). *Usability engineering: scenario-based development of human-computer interaction*. New York, NY: Academic Press.

Rosson, M. B., & Carroll, J. M. (2008). Scenario-based design . In *The human-computer interaction handbook* (pp. 1041–1060). Mahwah, NJ: Lawrence Erlbaum.

Schneier, B. (2000). *Secrets & lies: Digital security in a networked world*. New York, NY: John Wiley & Sons.

Sindre, G., & Opdahl, A. L. (2005). Eliciting security requirements with misuse cases. *Requirements Engineering*, *10*(1), 34–44. doi:10.1007/s00766-004-0194-4

Swiderski, F., & Snyder, W. (2004). *Threat modeling*. Sebastopol, CA: Microsoft Press.

van Lamsweerde, A. (2009). *Requirements engineering: from system goals to UML models to software specifications*. New York, NY: John Wiley & Sons.

van Lamsweerde, A., & Letier, E. (2000). Handling obstacles in goal-oriented requirements engineering. *IEEE Transactions on Software Engineering*, *26*(10), 978–1005. doi:10.1109/32.879820

Yu, E. (1995). *Modeling strategic relationships for process reengineering*. Unpublished doctoral dissertation, University of Toronto, Toronto, ON, Canada.

Shamal Faily is a post-doctoral researcher at the University of Oxford having recently completed his DPhil at the same university. His main research interests involve examining how the design of secure systems can be better supported with design techniques and tools from HCI and Secure Software Engineering. Shamal is also interested in understanding how entrepreneurship and innovation theories can be used to inform the design of security. Shamal is currently working on the EU EP7 webinos project where he is exploring how the security and privacy expectations of archetypical users can be built into the design of the webinos platform.

Ivan Fléchais is a departmental lecturer in the Software Engineering Programme at Oxford University and his main lecturing and research interests are in the area of computer security, in particular, given the important role that people play in secure systems; this involves researching how secure systems can be designed, implemented, and tested to take human factors into account. Prior to this, he graduated with BSc in computer science from University College London (UCL) and then stayed on at UCL to achieve a PhD researching how to design secure and usable systems, which resulted in the creation of the Appropriate and Effective Guidance to Information Security secure system design method.