

Security and Functionality Requirements in Practice: Towards a Combined View

Martyn Fletcher, Howard Chivers

Department of Computer Science, University of York, Heslington, York, YO10 5DD, UK.
martyn.fletcher@cs.york.ac.uk; hrchivers@iee.org

Abstract

Non-functional requirements, such as security, are critical in practical systems, but despite promising academic theory, established practices in functional and non-functional requirements remain disjointed: the development of functional requirements focuses on external behaviour, and non-functional requirements are developed using risk-based criteria. This paper draws on the experience of developing requirements for a grid-based project, the Distributed Aircraft Maintenance Environment (DAME), which is a collaborative industrial system with critical security requirements. We conclude that a fully integrated process for functional and non-functional requirements would greatly benefit the stability of the subsequent development, and show that such an approach is possible, by proposing a general model that shows how the artefacts generated for both purposes interact. The interaction model captures important experience, and can be used to understand and integrate functional and non-functional requirements processes in practice.

1. Introduction

An important aspect of requirements engineering is the treatment of non-functional as well as functional requirements, but satisfactory approaches for capturing and analysing non-functional requirements have yet to mature [1]. A recent large-scale, industrially-based grid project, the Distributed Aircraft Maintenance Environment (DAME) required the elicitation and management of both functional and non-functional (security) requirements. Applying established requirements practice to both aspects was successful, but the interaction between accepted methods in the two disciplines highlighted fundamental differences in their nature, and suggested an integrated requirements

management approach that can improve the stability of both types of requirement.

Functional requirements were developed via scenarios, which were initially described at the boundary of the system, then elaborated to show the interaction of the many services that make up DAME. Functional requirements for the whole system were first developed, then analysis and high-level design proceeded to the point at which the services offered by the different collaborators and systems could be identified.

The starting point for security requirements is a risk-based appraisal of the assets in the system, which includes an assessment of critical unwanted outcomes for those assets (known as stakeholder *concerns*). These concerns are concrete security goals rather than operationalized requirements, since security control requirements (or security policies) are not necessarily close to the assets they protect; further analysis is needed to establish true requirements, and this is reported separately [2].

In order to establish meaningful asset concerns, the main assets in the system have to be exposed and understood. This results in feedback between the functional and non-functional requirements processes. This feedback is the subject of this paper, which proposes a more unified process, based on this practical experience.

The contribution of this paper is to:

- describe the practical experience of managing both functional and non-functional requirements in a complex distributed system;
- show the interaction between the main components of the requirements system, and how they expose fundamental differences between the functional and non-functional perspectives; and
- propose a single-process for functional and non-functional requirements, based on this experience.

The next section (2) introduces DAME and the requirements process used in this project. This is followed by a summary of related work in section 3. Section 4 describes our practical experience, and

highlights tension, interaction and feedback between functional and non-functional aspects. Section 5 summarises the lessons learned, and proposes how these processes can be better integrated. Section 6 concludes the paper.

2. Background: DAME and Requirements

DAME is an e-Science pilot project to produce a diagnostic system for aero engines, implemented as a set of collaborating services using the Grid computing paradigm [3]. The input to DAME is monitoring and sensor data obtained during flight, and the system provides a collaborative environment where expert users in different organisations work together to interpret the data.

The DAME collaborative process spans several companies and supports a high-valued contractual relationship. The project has two industrial customers (Rolls-Royce plc, and Data Systems & Solutions, LLC) and the development work was distributed between four university-based development teams (Universities of York, Leeds, Sheffield and Oxford) and Cybula Ltd, each of which brought specific skills and industrial properties to the project.

This project is a demanding test case for requirements engineering, since it has essential non-functional requirements arising from the contrasting needs of different industrial partners, and a wide range of stakeholders with contrasting viewpoints.

DAME used established processes for eliciting and documenting functional requirements, based on the use of scenarios and use-cases [4]. The process started with the definition of the system scope, boundaries, actors, user goals, and outermost use cases. System use cases were developed hierarchically from the outermost use cases and the users' goals. The purpose of defining the system boundary was to precisely determine the design scope of the system. Outermost use cases showed how the system goals contributed to a wider context, and acted as a check on the purpose of user goals.

Security requirements were established via risk analysis [2], which contrasted the likelihood of a security threat (potential harm) with the impact (damage or cost to the business) to the stakeholder if that threat were realised. Unacceptable risks can be mitigated by reducing their likelihood or impact; the former eventually results in the need for security controls or functions within the system. The requirements baselines for risk analysis are the unwanted security outcomes to system assets (known as *concerns*), which are quantified in business terms. An asset is a resource of value to an organisation, including hardware, software, data, people and soft assets (reputation or intellectual property). This process is *asset*

driven: it identifies stakeholder *concerns*, or unwanted outcomes, for assets, and these provide concrete security goals. It is also necessary to elicit a wider model of the system environment, including possible attackers and their motives, in order to estimate attack likelihood.

It is clear that the black-box perspective of the functional requirements process does not provide sufficient exposure of internal assets for an asset-driven security requirements process. This problem was resolved by allowing requirements development to progress to the point at which the internal business assets had been identified, prior to security requirements elicitation. Control of the granularity of the resulting assets is critical: if the design is developed in too much detail, assets are not meaningful in business terms to stakeholders and it is difficult therefore to elicit security concerns. The level of detail needs to be just sufficient to describe the internal business processes of the new system.

3. Related Work

Cockburn [4] deals extensively with functional requirements documentation through use cases including the use of 'black box' views and where necessary 'transparent box' views. His use of outermost use cases to explore how a new system fits into its environment proved particularly valuable in this project.

The use of different views to adapt to the perspectives of different stakeholders is also a feature of our work. Sommerville [5] provides a survey of viewpoint-oriented approaches, and suggests ways of overcoming the practical problems of introducing them to industry.

The established decision criterion for security requirements is risk to specific system assets, and this is established in both standards [6] and practice [7].

These approaches amount to current best practice, but are not integrated; our aim was to explore the extent that established practices for both functional and non-functional requirements can be used together in a project.

In contrast, there are other requirements approaches that are under development, but are not yet widely deployed. The main alternative to eliciting security requirements via consideration of assets is to use high-level goals. Traceability of security constraints to goals is a feature of goal-based requirements development [8, 9] which has also been tested in industrial settings. Goal attributes (rather than separate goals) have been suggested as a means to specify non-functional requirements including security [10, 11]. Other authors have suggested eliciting and refining security and functionality goals separately, and then merging the resulting operationalized requirements [12] or direct refinement of security goals [13].

An alternative approach is the use of abuse cases, which aim to elicit unwanted behaviour in a similar way that use-cases elicit wanted behaviour [14, 15].

The goal community has recently begun to consider objects that are nearer to the assets understood by the security community [16, 17]; however, neither goal-based requirements management, nor abuse cases have yet been reconciled with established risk-based security engineering.

4. The Requirements Processes in Practice

This section describes our experience in following the processes outlined in section 2. In particular we focus on problems caused by the separate development of functional and security requirements, and how they were overcome; from this we are able to suggest ways in which the processes can be better integrated.

4.1. Functional Requirements

Functional requirements need to characterise the whole of the system under design, and to achieve this it is necessary capture significant behaviours of external actors and systems. This requires an understanding of how DAME works within the existing aircraft maintenance environment, and how its introduction changes the way people work. After defining the stakeholders, the first step was to agree the context of the

system, and hence the design scope for the rest of project. This was complicated because stakeholders were still formulating ideas about how the system would work with existing systems, and these were also evolving. Identification of the top-level use cases required extensive discussion with stakeholders (both customers and developers) using the techniques and templates and described in [4]. Much time was spent eliciting scenarios and goals, and analysing outermost goals and use cases, to develop a consistent view that fitted the customers' business models.

Identifying the outermost goals (outside the system) that DAME must support helped the stakeholders develop an understanding of the functional boundary. Each outermost goal was refined into sub goals for other actors and eventually into use cases for existing systems and for DAME.

Figure 1 shows how the external behaviours were identified and successively broken down from the outermost goal 'Release Aircraft' through the 'Release Engine' goal to the DAME use cases. The outermost goal encompasses all checks and work done on an aircraft after it lands necessary to ensure it is ready for the next flight. To achieve this outermost goal, many sub goals have to be completed, ranging from cleaning the cabin to maintenance checks that include the Release Engine goal. The Release Engine goal encompasses all the checks and operations necessary on each engine to ensure that it is released for the next flight.

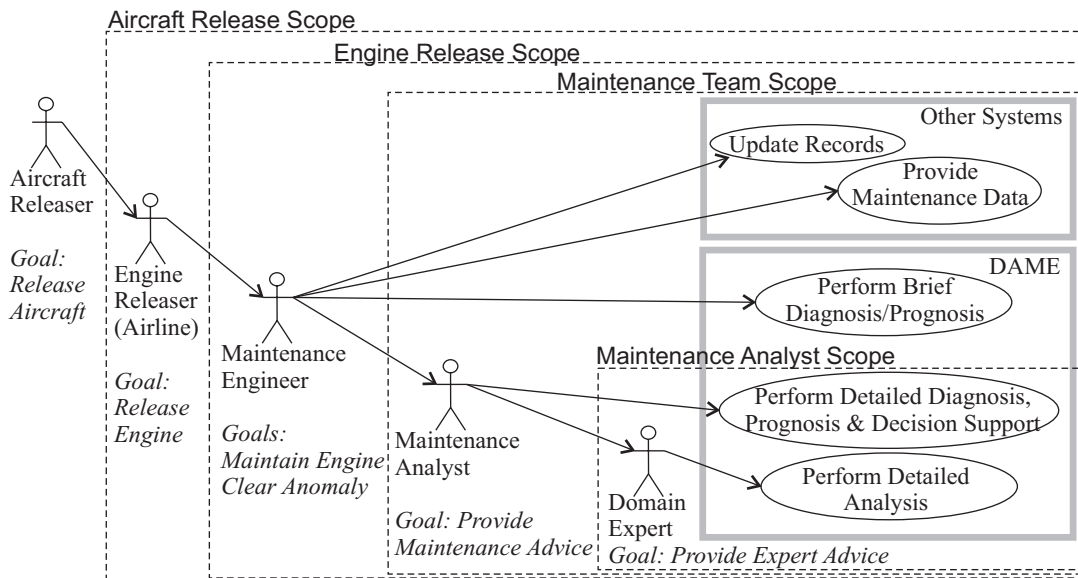


Figure 1. Hierarchical breakdown of 'Release Aircraft' outermost goal (Release Engine goal only)

Also, figure 1 shows the breakdown of the Release Aircraft outermost goal, for only the Release Engine sub goal; many other sub goals exist but they do not involve DAME. This sub goal is broken down through the various actors and scopes until the use cases are identified.

The primary business process in DAME is problem identification, escalation and analysis. Problems detected may be escalated from airport Maintenance Engineers to Maintenance Analysts at a maintenance data centres, and eventually to Domain Experts at the engine manufacturer. Three main DAME use cases were defined:

- Perform Brief Diagnosis/Prognosis;
- Perform Detailed Diagnosis, Prognosis and Decision Support; and
- Perform Detailed Analysis.

The behaviour of DAME was therefore defined in conjunction with behaviour outside the system, resulting in an agreed boundary and a set of business processes to be supported by the system. Our experience confirms the value of establishing the wider context during this process, and the high investment in programme and stakeholder time required to achieve that understanding.

4.2. Inside the System Boundary

In the DAME project it also proved necessary to establish a 'transparent-box' view of the system in order to:

- check the sufficiency of the boundary perspective;
- accommodate the viewpoints of diverse development teams; and
- expose internal business assets for security analysis.

The transparent-box view of the system used interaction diagrams to show how its use cases were implemented by internal services, and identify the data items that flow between them.

The Boundary Perspective: the boundary view was adequate for user interaction with the system, but inadequate for the definition of the interfaces to other systems. The interface between DAME and other customer systems evolved significantly as the top-level model of DAME developed, and some of these changes were major. For example, the final design uses a distributed storage capability for historical sensor data within DAME; in contrast, the initial requirement for DAME was to access the data from a centralised external data store.

Development Teams: the requirements problem for developers was the technical capabilities and limitations of key industrial properties. The system ultimately depends on exploiting these technologies, but the

designers that understand their capabilities and limits are concerned with the performance of sub-systems and services, rather than the total system behaviour. Given multiple development teams, it is necessary to expose some the internal detail of the system in order to expose functional issues of feasibility and technical capability.

Assets and Security: the security requirements process is asset driven, so the first step is to ensure that business assets within the system have been identified and understood. By 'business asset' we mean an asset that is meaningful to stakeholders, including the system's customers, and about which it is possible to elicit concerns quantified in business terms.

Some business assets are evident when considering the system as a black-box; for example, data assets exchanged between the system and other systems or actors can be identified because they cross the system boundary. However, business assets also occur within the system, requiring the development of a top-level system design.

For example, the functional requirements of DAME require an engine simulation, the ability to search for previous occurrences of similar symptoms in sensor data, and the need to consult the maintenance history of related engines. The last of these – the maintenance history – is held in a different system, so it is fully exposed at the system boundary. However, the other two – simulation algorithms and historical sensor information – determine the need for assets within DAME that are both meaningful, and critical, in business terms.

Therefore, to identify the significant business assets it is necessary to view the system as both a black-box and a transparent-box. However, it is important to limit the depth of the transparent-box view, otherwise internal design features are generated that stakeholders are unable to relate to their business. The criterion of business relevance was therefore used to limit the amount of high-level design carried out prior to the security requirements process.

4.3. Eliciting Asset Concerns

Once the assets had been identified, each was individually assessed. This entailed discussing with the stakeholder what *concerns* they had about each asset (i.e. determining unwanted outcomes for each asset) and the impact should the concern be realised. The concerns were documented in the stakeholder's language, using appropriate domain vocabulary; the usual security keywords were used (confidentiality, integrity) but the discussion naturally introduced other related concerns including availability, reliability and provenance.

Table 1. Extract from Asset Analysis Table: Specific Concerns, Impacts and Goals

(Extracted from) Asset Table 3 Specific DAME Data Asset threats (concerns)				
Data Assets	Confidentiality	Integrity	Provenance	
3.2 Engine Data Record Performance. (These concerns may change if the data are deployed outside DAME)	RR / DS&S Could divulge proprietary information to 3 rd party	RR / DS&S Need to protect accuracy of reference data	RR / DS&S. Protect the reference source of decision data	Notes
	I Engine Performance	III Reliability (L)	IV Record decision basis	Goal
	C.A Unauthorised Access	I.A Loss or Corruption	P.B Source of Reference Data	Concern
	Medium	Low	Medium	Impact

The impact of a concern being realised was classified in business terms on a scale from zero (not significant) through low and medium to high (prejudicial to part of the business for a long period). It was then possible to produce a set of generic concerns (such as reliability), which applied to all services, as well as those specific to each asset. Table 1 is an extract from the asset analysis table, describing the concerns, impacts and goals produced during the security requirements process for one asset. (Goals are described in section 4.4.) In table 1, for the data asset “*Engine Data Record Performance*” the stakeholder has expressed several concerns. One is related to confidentiality: the stakeholder is concerned about unauthorised access to the asset. The concern has a business impact of Medium and a goal from which it is derived; the goal is abbreviated in the table but the full title is “*To maintain the Confidentiality of Detailed Engine Design and Performance Data*”. The DAME security requirements capture process identified and assessed 20 service assets and 33 data assets.

When using the black-box view it proved difficult to identify concerns with some assets because of their composite nature. Their definition was acceptable for functional purposes, but there were significantly different concerns, or levels of impact, associated with different aspects of the asset. To avoid non-functional over-specification these instances were identified and assets partitioned accordingly.

At the transparent-box level, inconsistencies within the system can be identified. As far as possible the asset requirements elicitation avoided indirect concerns – those that apply to an asset because of a concern on another; this is dealt with in the subsequent security analysis. However, stakeholders naturally reason about the implications of their decisions and this can lead to the confusion of primary and secondary concerns in the elicitation process. Traceability is a great help in resolving this problem – each concern is traceable to something, which may be a primary goal or another concern. Concerns that are traceable to another concern can then be examined for relevance.

One situation where indirect concerns are relevant is when a security concern for one asset requires extra functional behaviour, which may in turn have a non-functional concern. For example, a concern to maintain the provenance of an asset implies the need to record the lifecycle of that asset. Recording is a functional requirement, but the new asset (the record) inherits a concern about its integrity, traceable to the original provenance concern.

One interesting issue that did arise was that real requirements are not necessarily static; the impact to a stakeholder of some threats depended on the business cycle – for example the progress of a new contract. These issues were simply recorded, but they complicate the later lifecycle of the system, so stakeholders reviewed them carefully.

4.4. Non-functional Goals

Goal development was carried out as an integral part of the elicitation of asset concerns, but it is described separately here because of its importance. Its value is that it allows the completeness and pertinence of asset-based requirements to be established. Completeness and pertinence was judged by reviewing assets against goals and by maintaining traceability from requirements to goals, respectively.

The aim in drafting goals was to produce business-oriented statements with clear motivation and objectives. Goals (for example, table 2) consisted of four parts: Title, Owner, overall business impact, and descriptive clarification.

There is a considerable difference between this type of goal and top levels goals such as ‘secure’ or ‘confidential’, which are often quoted in the literature. The DAME goals are well suited to their function because they define threats, objects of protection and motivation, but as a consequence they are harder to elicit in abstract: without assets to consider.

The problem with developing goals in abstract is that it is difficult to establish an appropriate level of detail; on the other hand it is relatively easy to elicit requirements in

terms of specific business assets because stakeholders are comfortable thinking in terms of assets and concerns. One strategy is therefore to establish initial asset concerns, and then cluster these concrete examples into draft goals, which can be shaped by stakeholders to reflect their underlying concerns.

Table 2. A Typical DAME Goal

Title:	IV: To record the provenance of diagnostic decisions and identify individuals' actions in the diagnostic process
Owner:	Rolls-Royce and Data Systems & Solutions
Impact:	Medium
Description:	The process by which diagnostic decisions are made must be recorded with sufficient quality to allow the investigation of problems or marginal decisions after the fact. Individuals that contribute to the diagnostic workflow must be accountable for their contribution to the process.

In DAME the development of non-functional goals using this technique proved very effective; clustering the assets concerns suggested seven top-level goals, which were then revised by stakeholders and widely agreed. The assets and concerns were then rechecked against the goals for consistency. This was an iterative process that proved a useful check of the correctness and consistency of the asset concerns, and allowed traceability between asset-based concerns and business goals. This is an example of the use of two complimentary views, where each can be used to crosscheck the validity of the other.

4.5. Other Non-functional Requirements

Goals also emerged for other non-functional requirements - reliability and availability. When asked to express concerns about business assets, stakeholders do not immediately distinguish between different requirement types (e.g. security, reliability, etc.), they simply state their concerns, which may be related to one or more types of non-functional requirement. This demonstrates again the importance of an open elicitation process that is not constrained by preconceived security checklists, and also that the asset driven approach proves to be as suited to capturing other non-functional requirements as it is to security.

It may seem that some non-functional requirements relate to behaviour rather than to assets, for example, timeliness of a particular function. However, after further consideration it is apparent that this is a concern regarding the delivery of an asset. For example, in

DAME there is a timeliness concern on completion of functional requirements such as Provide Brief Diagnosis, but this can equally be expressed in terms of the availability of specific DAME outputs.

4.6. Attackers

Security risks are characterised by two factors: business impact and likelihood of a successful attack. The process described in the preceding sections determines the concerns or threats, and impacts; likelihood is determined by the frequency of attack, the type of attacker, the degree of access and vulnerabilities or paths in the system that can be exploited. The last of these is a design and implementation issue, but the others are best elicited with other contextual requirements. This is not described in detail here, because the process does not interact strongly with functional requirements management (see 3.2.2, Motivation and Threat Actions in [7], for an example).

In DAME, attacker elicitation was carried out after the non-functional goals had been agreed, and the resulting profile became another component of the baseline context.

5. Process Interactions and Integration

The project set out to develop functional and non-functional requirements. These processes cannot be carried out strictly in parallel, because the security requirements process is asset-driven, whereas the functional requirements can start with a black-box view.

The use of boundaries, and the ability to view them from either black-box or transparent-box perspectives, is a central theme in requirements capture. Functional requirements establish a boundary around the system, and to achieve this, the primary focus of requirements elicitation is outward. The same boundary is used for security requirements, but the primary focus of attention is inwards to the assets of the system. Security requirements elicitation can be carried out only after assets have been identified, since it needs to consider both the business assets that cross the system boundary, and those embedded within it.

This suggested a serial process from the external context to the boundary definition, then to assets and concerns. Such a process would include feedback and iteration between each stage and the next; however, our work exposed more complex dependencies, which are summarised in figure 2.

Specifying internal business processes elicited functional constraints from technical experts, and also clarified the understanding of the system boundary. The process of eliciting asset concerns identified requirements

with both functional and non-functional components. Finally the process of goal development added to the system context, allowing the asset concerns to be revised and the attack environment to be identified.

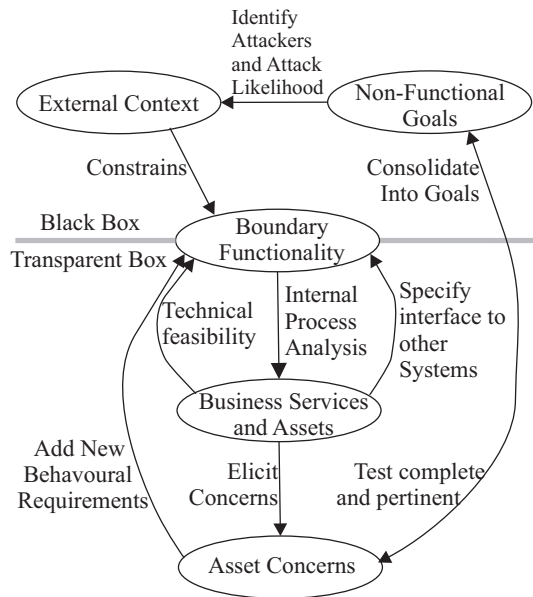


Figure 2. Dependencies between Functional and Non-functional Requirements Artefacts.

It is evident from these interactions that functional and security requirements processes should not be separated; to do so introduces the risk that work based on the functional requirements may be negated by changes flowing from the non-functional side. It is equally clear that they are not separate processes that can be safely merged later in the system lifecycle.

This evidence suggests the need for an integrated approach to functional and non-functional requirements, cognisant of the feedback paths identified here. The enabling factor in such a process is the early identification of internal business assets, which facilitates communication with technical stakeholders as well as providing the context for the elicitation of non-functional concerns.

An alternative way of regarding a combined functional and non-functional process is that it increases the number of views that are employed and consolidated; this increases the degree of stakeholder involvement in requirements elicitation, by providing a wider range of vocabularies with which to describe the system.

Finally, business pertinence proved to be a useful criterion for the amount of asset detail that is needed in practice, and building goals by consolidating asset concerns also proved very effective, despite its pragmatic rather than top-down, approach. It is a valuable

technique because it allows the stakeholder to relate asset concerns to business goals and allows the concerns to be checked for consistency from two perspectives.

6. Conclusions

This paper describes a successful attempt to combine functional and non-functional requirements processes in a significant grid-based project, DAME. The opportunity to document such a project is rare; the experience is therefore documented in sufficient detail to expose the practical issues that arise in such an activity.

The planned approach was to use established methods to develop both functional requirements (focussing on the context and black-box behaviour) and security requirements (risk-based), and apply these in series as the project progressed.

However, the experience shows that rather little design work is needed to expose the internal business assets of the system, but there is a considerable amount of feedback between the non-functional and functional parts of the process. This means that a more integrated approach is both viable and necessary, and the critical step in enabling such an approach is the early identification of internal business assets. Applying a business relevance criterion can control the depth to which internal detail is exposed, and the resulting process greatly benefits the stability of the resulting requirements.

In summary, it is feasible to integrate current best practice in functional and non-functional requirements management. Figure 2 illustrates the dependencies between the artefacts in a combined process, but the key features are that:

- the top level design is developed to a limited extent, since pure black-box requirements do not include sufficient concrete assets to allow security requirements elicitation; and
- asset-based elicitation of security requirements is closely coupled to the functional requirements management process, since it gives rise to functional iteration.

This work also demonstrates the utility of a pragmatic approach to developing non-functional goals: consolidating high-level goals from more concrete asset concerns. This combines the benefits of goal traceability with more apposite goals than are often achieved.

Finally, although the risk-based asset-driven process was adopted because of its relevance to security, it naturally extended to a range of other non-functional requirements. This suggests that the combined approach recommended here is a good step towards the integration of all types of requirements capture.

This paper documents a practical attempt to combine existing security and functionality requirements practice.

Our conclusions suggest that this is a productive approach that justifies further exploration; the next step is to conduct further practical work to test the feasibility of the tightly-coupled combined process suggested here.

7. Acknowledgements

This work was undertaken as part of the DAME project, with grateful assistance from Rolls-Royce plc, Data Systems & Solutions, LLC and Cybula Ltd and the teams at the Universities of York, Leeds, Sheffield and Oxford. This research was supported by the UK Engineering and Physical Sciences Research Council (Grant GR/R67668/01), the Royal Academy of Engineering, and through contributions from Rolls-Royce plc, and Data Systems and Solutions, LLC.

8. References

- [1] B. Nuseibeh and S. Easterbrook, "Requirements Engineering: a Roadmap," in *The Future of Software Engineering (Special Volume published in conjunction with ICSE 2000)*, A. Finkelstein, Ed., 2000, pp. 35-46.
- [2] H. Chivers and M. Fletcher, "Applying Security Risk Analysis to a Service Based System," *Software Practice and Experience: Special Issue on Grid Security*, vol. 35(9), pp. 873-897, 2005.
- [3] I. Foster and C. Kesselman, "The Grid 2: Blueprint for a New Computing Infrastructure," Morgan Kaufmann, 2003.
- [4] A. Cockburn, *Writing Effective Use Cases*: Addison-Wesley, 2000.
- [5] I. Sommerville and P. Sawyer, "Viewpoints: Principles, Problems and a Practical Approach to Requirements Engineering," Lancaster University, Computing Department, Cooperative Systems Engineering Group, Technical Report CSEG/15/1997, 1997.
- [6] "Information Security Management: Part 1 Code of practice for information security management," British Standards Institution BS 7799-1:1999, 1999.
- [7] "Risk Management Guide for Information Technology Systems," National Institute of Standards and Technology (NIST) SP 800-30, January 2002 2002.
- [8] A. Dardenne, A. v. Lamsweerde, and S. Fickas, "Goal-directed Requirements Acquisition," *Science of Computer Programming*, vol. 20(1-2), pp. 3-50, 1993.
- [9] A. v. Lamsweerde, "Goal-Oriented Requirements Engineering: A Guided Tour," presented at Fifth IEEE International Symposium on Requirements Engineering (RE '01), Toronto, Canada, 2001.
- [10] L. Chung, "Dealing with Security Requirements During the Development of Information Systems," presented at 5th International Conference on Advanced Information Systems Engineering (CAiSE '93), 1993.
- [11] A. I. Antón and J. B. Earp, "Strategies for Developing Policies and Requirements for Secure Electronic Commerce Systems," *Recent Advances in Secure and Private E-Commerce*, 2000.
- [12] J. D. Moffett and B. A. Nuseibeh, "A Framework for Security Requirements Engineering," University of York, Department of Computer Science YCS-2003-368, 20 August 2003 2003.
- [13] H. Mouratidis, P. Giorgini, and G. Manson, "Integrating Security and Systems Engineering: Towards the Modelling of Secure Information Systems," presented at Conference on Advanced Information System Engineering (CAiSE 03), 2003.
- [14] J. McDermott and C. Fox, "Using Abuse Case Models for Security Requirements Analysis," presented at 15th Annual Computer Security Applications Conference, Phoenix, Arizona, 1999.
- [15] I. Alexander, "Misuse Cases: Use Cases with Hostile Intent," *IEEE Software*, vol. 20(1), pp. 58-66, 2003.
- [16] A. v. Lamsweerde, "Elaborating Security Requirements by Construction of Intentional Anti-Models," presented at 16th International Conference of Software Engineering (ICSE'04), Edinburgh, Scotland, 2004.
- [17] L. Liu, E. Yu, and J. Mylopoulos, "Security and Privacy Requirements Analysis within a Social Setting," presented at 11th IEEE International Requirements Engineering Conference (RE'03), Monterey Bay, California, USA, 2003.