

# Combining Functional and Security Requirements' Processes

Martyn Fletcher, Howard Chivers, Jim Austin

Department of Computer Science, University of York, Heslington, York, YO10 5DD, UK.

[martyn.fletcher@cs.york.ac.uk](mailto:martyn.fletcher@cs.york.ac.uk), [harchivers@iee.org](mailto:harchivers@iee.org), [jim.austin@cs.york.ac.uk](mailto:jim.austin@cs.york.ac.uk)

## Abstract

Requirements elicitation for a new system requires extensive involvement with stakeholders who usually have varying aims, backgrounds and disciplines, and this process is complicated further if the system has critical security, or other non-functional requirements. Established approaches to the elicitation and analysis of functional and non-functional requirements are very different - the former focussing on boundary behaviour and the later on threats to assets. This paper describes how requirements were established for the Distributed Aircraft Maintenance Environment (DAME) system, showing how established practice from both disciplines can be effectively combined, and the interaction and iteration that is needed between functional and non-functional requirements processes in order to meet both their objectives in practice.

## 1. Introduction

Requirements engineering is an interdisciplinary activity that involves the stakeholders of a system in the process of discovering, confirming, documenting and analysing requirements [1]. 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 [2]. A recent large-scale, industrially based Grid Project, the Distributed Aircraft Maintenance Environment (DAME) [3] required the elicitation and management of both functional and non-functional (security) requirements. Applying established requirements practices to both aspects was successful, but the interaction between accepted methods in the two disciplines highlighted fundamental differences in their nature, and provided important insights about their respective contributions and the iteration between them.

The development of functional requirements focuses on how a system interacts with its environment; progressively establishing goals for the business processes supported by the system and then the boundary of the system itself. Eventually a process of design

decomposes the resulting requirements into functions that can be assigned to components.

In contrast, the development of security requirements focuses on risk, in particular the risk of particular unwanted outcomes to the business, and its assets. The underlying practice in security is therefore markedly different to functionality: functional requirements are clearly behaviour-led, whereas security requirements are asset driven. As a consequence these processes are rarely fully combined, security is often dealt with as an afterthought, and with apparent justification: since security focuses on assets, security requirements elicitation relies on a degree of functional design before it can begin.

The risk in deferring security requirements elicitation is that it is disruptive of established functional requirements, and of the system design and architecture. Security goals motivate functionality (e.g. audit) as well as protection (e.g. access control), and simply viewing the system from an asset, rather than behavioural, direction provides a new perspective from which new functional requirements may emerge. Of course, security requirements may also invalidate a design, but that is beyond the scope of this paper. There is therefore a good case to elicit and manage requirements for functionality and security as part of a single integrated process, and recognise that there will be interaction and feedback between them

This paper reports the experience of developing security and functionality requirements as part of the DAME project; we outline the processes, provide examples of the artefacts used, and highlight our approach to important issues, including:

- The problems of developing ‘black-box’ requirements in Grid scenarios, where organisations or developers have a limited visibility of the whole system.
- The interaction between security and functionality requirements processes.
- The relationship between business goals and security, and how to construct meaningful security goals.
- Developing other non-functional requirements.

This paper is organised as follows. Following an introduction to the DAME system (section 2), section 3 provides an overview of the development of DAME functional requirements using a ‘black box’ view of the system. This is followed in section 4 by a description of the use of a ‘transparent box’ view to expose business assets for security analysis. Section 5 describes our practical experience in eliciting asset concerns and section 6 summarises the development of security goals, and shows how other non-functional goals are naturally captured in the process. Section 7 briefly describes elicitation of the security threat environment, and section 8 summarises the important lessons from this experience and concludes the paper.

## 2. Background: the DAME system

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. The system provides a collaborative environment where expert users in different organisations work together to interpret the data, supporting high-value contractual relationships. The project has two industrial customers: Rolls-Royce plc, and Data Systems & Solutions LLC. 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.

The requirements process focussed on the following main themes:

- Functional requirements, considering the system as a ‘black-box’.
- Asset Concerns (unwanted outcomes)
- Non-functional Goals
- The Threat Environment: potential attackers and their objectives.

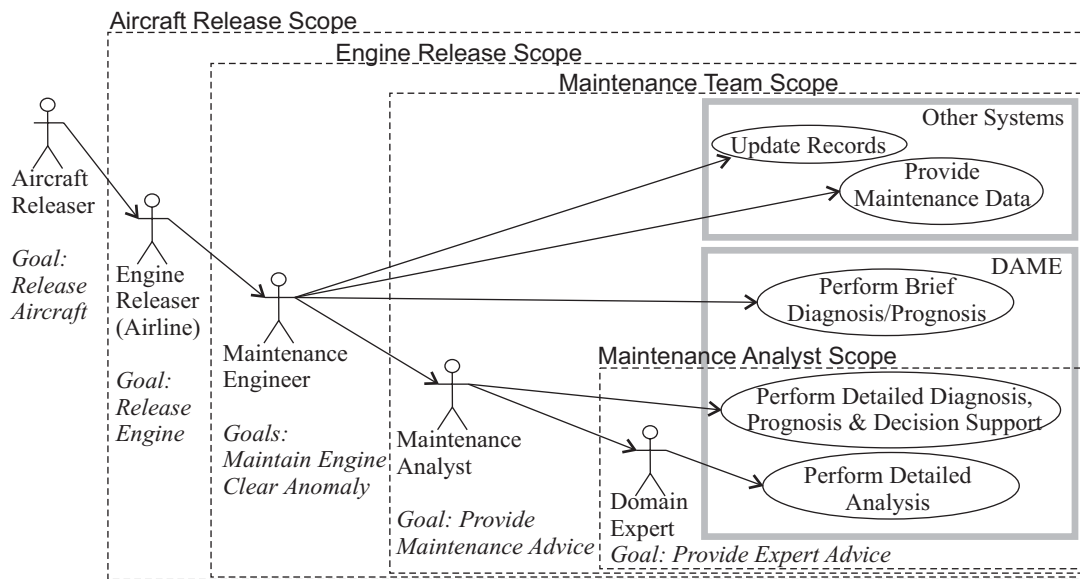
## 3. Viewing the system as a black box

The process of eliciting functional requirements from stakeholders using use-cases includes the definition of the system scope, boundaries, actors, user goals, and outermost use cases. System use cases are developed from a hierarchical breakdown of the outermost use cases and the users’ goals.

Functional requirements characterise the whole of the system under design, and to achieve this it is necessary capture significant behaviours of business actors, and of other related systems. For DAME, this requires an understanding of how it is integrated within the existing aircraft maintenance environment, and how its introduction changes existing working practices.

After defining the stakeholders, the first step is to agree the context of the system, and hence determine the design scope for the rest of project. For DAME this was complicated because the customers were still formulating ideas about how the system would interface to existing systems, and these systems were also evolving. Identification of the top-level use cases required extensive discussion with stakeholders (both customers and developers together) using the standard techniques and templates (see [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.



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

The outermost goal, Release Aircraft, 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, including 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 shows the breakdown of the Release Aircraft outermost goal, only for 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 context of DAME was elicited and the primary business process was identified. DAME supports and enhances problem 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 and primary actors were defined:

- Perform Brief Diagnosis / Prognosis (Maintenance Engineer)
- Perform Detailed Diagnosis, Prognosis and Decision Support (Maintenance Analyst)
- Perform Detailed Analysis (Domain Expert)

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

The process used to establish security requirements is asset driven, but the black-box perspective described in this section does not provide sufficient exposure of internal assets for the security requirements process. The development of the asset perspective is described in the next section.

#### 4. Exposing business assets

The process used to establish Grid security requirements [5] is based on standard security risk analysis [6]. The aim of the process is to evaluate risk via a risk matrix that contrasts 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. Risks that are judged to be unacceptable need to be mitigated by reducing either the likelihood or the impact or both. The unwanted outcome, or realised threat (also known as a *concern*) to an asset is quantified in business terms. Generally an asset is a resource of value to an organisation: hardware, software, data, people and soft assets (reputation or intellectual

property). Business assets are specifically data or software that are meaningful to stakeholders, including the system's customers, and about which it is possible to elicit concerns quantified in business terms. (In the remainder of this paper references are sometimes made to "asset", for brevity, however this is intended to mean "business asset".) The security requirements process is clearly asset driven: it seeks to identify stakeholder concerns, or potential threats, for assets in the system and the impact if those threats were realised. It also elicits a wider model of the system environment, to identify possible attackers, their motives and quantify the likelihood of attacks.

A precursor to security requirements elicitation is therefore the identification of relevant assets. Some business assets are evident when considering the system as a black box, for example, data that crosses the system boundary to other systems or actors. However, business assets may also be hidden within the System; so to identify these it is necessary to construct 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 we need to view the system as both a black box and a transparent-box. However, we need to determine how deep into the transparent-box view we need to go; clearly, if we fail to limit the depth of view, then internal design features will be generated that customer stakeholders find difficult to assess in business terms. The criterion of *business relevance* was therefore used to limit the amount of high-level design before the security requirements process. Interaction diagrams were used to show how use cases were implemented by internal services, and to identify the business assets within the design.

In addition to the need to expose internal assets for security analysis there were other reasons why it was desirable to establish a 'transparent-box' view of the system:

- To check the sufficiency of the black box perspective
- To accommodate the viewpoints of diverse development teams

In general, the black box perspective of the system is adequate for user interactions since it specifies what users need to achieve. Where this was found to be inadequate was in 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 present design uses a distributed storage capability for historical sensor data within DAME; in contrast the initial requirements that modelled access from DAME to an external centralised external data store.

For the development teams, the main requirements issues were 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 elicit functional issues of feasibility and technical capability.

In summary, to facilitate asset analysis for security, it is necessary to establish some design features, but it is also important to not commit to a detailed design before security requirements are established. The criterion of *business relevance* (i.e. that stakeholders understand the resulting components in business terms) is an important guiding principle for the detail that should be exposed. Top-level design to the sub-system level was also necessary to check that overall system functional requirements were consistent with the capability of specialised technical sub-systems.

The assessment of the individual assets is described in the next section.

## 5. Eliciting business asset concerns

Once the assets had been identified, each was individually assessed. This entailed discussing with the customer the *concerns* they had about each asset (i.e. identifying particular unwanted outcomes) together with the impact should the concern be realised.

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

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

Concerns were documented in the customer’s language (using appropriate domain vocabulary); the usual security keywords were used (confidentiality, integrity) as guidewords but the elicitation naturally introduced other non-functional issues including availability, reliability and provenance.

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). The resulting concerns were divided into generic concerns (such as reliability, that applies to all services) and concerns that are specific to individual assets. Table 1 provides an extract from the asset analysis table, describing the concerns, impacts and goals produced during the security requirements process for a typical asset. (Goals are described in section 6.) In table 1, for the data asset EngineDataRecordPerformance the customer has expressed several concerns. One is related to confidentiality: the customer is concerned about unauthorised access to the asset. The concern has a business impact of “Medium” and a parent goal I (which 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 elicitation process identified and assessed 20 service assets and 33 data assets in this way.

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 it was necessary to recognise these instances and partition the assets accordingly.

As far as possible asset concern elicitation avoids indirect concerns (those that apply to an asset only because there is a concern on another), this is dealt with in 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. Avoiding indirect concerns where possible allows the designer maximum freedom to change the design to meet the primary goals: the minimum requirement, provided by concern traceability is to allow a designer to distinguish base requirements from implications resulting from the current design.

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, the need to maintain the provenance of an asset implies the need to record the life cycle of that asset. Recording is a functional requirement, but the new asset (the record) inherits a concern about its integrity, traceable to the provenance requirement. In the case of DAME, some provenance requirements were initially identified as functional requirement, but a systematic review of the assets revealed that only part of the necessary functionality had been specified, as well as identifying related security requirements. Asset analysis therefore proved to be useful in the identification of inconsistencies and omissions in the functional requirements.

One interesting issue that did arise was that real requirements are not necessarily static; the impact to a customer 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 customers reviewed them carefully.

In summary, the use of an open asset-based concern elicitation, rather than one that was

constrained to a prior security checklist, naturally resulted in the statement of other non-functional concerns, such as availability and provenance. Asset-based elicitation produced new functional requirements, and required minor design changes to better match the granularity of assets to the developing security concerns. Finally, traceability between concerns allowed stakeholders to distinguish between core concerns and those that were implied by the current design.

Asset concerns are a form of concrete goal, but business-level non-functional goals were also developed for the system, as described in the next section.

## 6. Developing non-functional goals

The aim of business goals is to provide objectives that motivate more detailed requirements. Success in a goal is beneficial to a business; failure to meet a goal will have an adverse impact. Goal development was carried out as an integral part of the elicitation of asset concerns, because 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. An example of a DAME goal is presented in table 2. Goals consisted of four parts: Title, Owner, overall business impact, and descriptive clarification.

**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.

There is a considerable difference between this type of goal and non-specific goals such as 'secure' or 'confidential'. 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, i.e. without specific 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 customers are comfortable thinking in terms of assets and concerns. The strategy used was therefore to establish initial asset concerns, and then cluster these concrete examples into putative goals, which were shaped by customers to reflect their underlying business concerns.

In DAME the development of non-functional goals using this technique proved very effective; clustering the asset concerns suggested seven top-level goals, which were then revised by customers and widely agreed. The assets and concerns were then rechecked against the goals for consistency. This was an iterative process that proved an important 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 iterative development of two complimentary views, where each can be used to crosscheck the validity of the other.

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

Some non-functional requirements may relate to either behaviour or assets. For example, timeliness may be thought of as relating to a system function or to asset availability or delivery. In DAME there is a timeliness concern on completion of functional requirements such as Provide Brief Diagnosis; this requirement may therefore emerge during the elicitation of either functional behaviour or asset-based concerns.

In summary, non-functional business goals provided a means of testing the completeness and pertinence of concerns, which are effectively a more concrete form of goal. For business-level goals to be useful they must be specific about threats, objects of protection and motivation. To achieve this clarity in abstract is difficult, but goal consolidation from concrete examples provides a good starting point for goal elicitation.

**Table 3. A Typical Attacker Record**

Access Type	Attacker	Goal	Frequency	Notes
Legitimate users	Domain expert Maintenance analyst Maintenance engineer	IV	Low	Users may seek to change or remove records of inappropriate decisions or actions.

## 7. 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.

The type of attacker, the frequency of attack, the degree of access to a system, and the availability of exploitable vulnerabilities determines the likelihood of attack. The last of these is a design and implementation issue, but the others are best elicited with other non-functional requirements. Attacker elicitation was carried out with stakeholders after the non-functional goals had been agreed, since these goals are identified as attack objectives.

A typical attacker record is shown in table 3, which describes a potential attack on the goal shown at table 2.

In DAME, potential attackers are grouped by access type into five main categories: administrator, legitimate user, other user, external with significant resources (e.g. journalist or competitor), and external with limited resource (e.g. hacker).

Attack frequency is quantified on a four point scale, which is defined in business terms: High and Medium likelihood attacks are respectively equivalent to many, or at least one, attack in any financial year. Low likelihood and Unlikely attacks are respectively likely, or not, during the system's operational life.

In DAME an attack checklist was initially generated from asset concerns, and from the roles and organizations identified in the system context. Elicitation with stakeholders used this base material as a focus. This will not be described in more detail here, because the process is straightforward and does not interact strongly with the rest of the requirements process.

In summary, attacker modelling is a necessary part of security requirements elicitation; it is based on the system context and non-functional goals, but interacts with the design process, via security risk management, rather than the functional requirements process.

## 8. Conclusion

This paper describes the experience of combining functional and security requirements elicitation and management processes in a significant grid-based project (DAME) and the lessons learned from applying this approach. Examples of some of the requirements artefacts have been included to clarify the approach used, and the project experience has been documented in sufficient detail to expose the practical issues that arose during the activity.

Brief summaries are given at the end of each section, but important observations include:

- Because of the innovative nature of grid systems in business terms, there is a high investment in programme and stakeholder time needed to achieve an understanding of how these system fit into their business context. Establishing the context for functional requirements is a lengthy and iterative process.
- In a distributed development it is not possible to complete requirements definition by regarding the system simply as a 'black box'. It is necessary to design to the sub-system level to check that overall system requirements are consistent with specialised technical sub-systems.
- Internal assets must be exposed to allow the elicitation of security concerns. However, it is important not to proceed too far with the design before security requirements are established: the criterion of *business relevance* (do the stakeholders understand the components) is the guiding principle for how much detail should be exposed.
- The use of an open asset-based concern elicitation, rather than one that is constrained by an a priori security checklist, naturally results in the discovery of other non-functional goals, such as availability and provenance.
- Asset-based elicitation provides a complementary viewpoint to functional elicitation, and therefore results in new

functional requirements as well as potential design changes resulting from security concerns.

- Non-functional business goals provide a means of testing the completeness and pertinence of asset concerns. However, to be useful, they must be specific about threats, objects of protection and motivation. Goal consolidation from concrete examples provides a good starting point for goal elicitation.

The experience reported in this paper highlights some important practical aspects of requirements management: the degree to which standard approaches for functional and security requirements processes can be combined, how their respective processes interact, and where iteration is needed. Although it is well known that security risk analysis may require design changes, the significant feature of this experience is the extent that asset-based elicitation, as used in security risk analysis, provides an orthogonal requirements view of the system, and therefore generates extra functional and non-functional requirements, as well as security goals.

## 9. Acknowledgements

The work reported in this paper was developed and undertaken as part of the DAME project with grateful assistance from Rolls-Royce plc, Data Systems & Solutions LLC and Cybula Ltd and all 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.

## 10. References

1. Sommerville, I., *Software Engineering*. Sixth edition 2001. ISBN 0 201 39815 X. Addison-Wesley.
2. Nuseibeh B. and Easterbrook, S. *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.
3. Jackson, T., Austin J., Fletcher M., and Jessop M. *Delivering a Grid enabled Distributed Aircraft Maintenance Environment (DAME)*, in *Proceedings of the UK e-Science All Hands Meeting 2003*, Nottingham, UK. At <http://www.nesc.ac.uk/events/ahm2003/AHMCD/>. 2003.

4. Cockburn, A., *Writing Effective Use Cases*. First edition 2000. ISBN: 0201702258. Addison-Wesley.

5. Chivers, H. and Fletcher, M. *Adapting Security Risk Analysis to Service-Based Systems*, in *Proceedings of the Grid Security Practice and Experience Workshop*, Oxford, UK. Technical Report YCS 380. University of York, Department of Computer Science. 2004

6. "Risk Management Guide for Information Technology Systems," National Institute of Standards and Technology (NIST) SP 800-30, January 2002.