Secure and Dependable Patterns in Organizations: An Empirical Approach

Yudistira Asnar, Paolo Giorgini,
Fabio Massacci, Ayda Saidane
DIT- University of Trento, Italy
name.surname@unitn.it

Roberto Bonato, Valentino Meduri
Deep Blue srl, Italy
name.surname@dblue.it

Carlo Riccucci
Engineering SpA, Italy
name.surname@eng.it

Abstract

Designing a secure and dependable system is not just a technical issue, it involves also a deep analysis of the organizational and the social environment in which the system will operate. In this paper, we detail our experience in modeling and analyzing requirements for an industrial case (air traffic management system) using the Secure Tropos framework. Particularly, we focus on modeling and reasoning about trust and risk relations within the organizational structure; we discuss pros and cons of Secure Tropos stemming from our experience and lessons learned which might be general interests for RE methodologies.

1. Introduction

The literature in requirements engineering has recently highlighted the importance of analyzing security and dependability (S&D) issues since the early phases of the software development [13, 9], and in particular for safety-critical domains where human lives are at stake [3].

It is also accepted that S&D cannot be considered as purely technical issues but should be analyzed together with the organizational environment. In this direction, goal-based approaches have gained momentum in the community showing their relevance to model and analyze security issues within an organizational setting. Among the various proposals, the SI*/SecureTropos methodology, an extension of the *i**/Tropos methodology, has been gaining acceptance both in the RE [9] and Security [10, 2] communities. It introduces concepts such as ownership, trust, risk and delegation within the requirements model in order to capture security and trust requirements. Automated reasoning is supported to verify S&D properties.

Still, Secure Tropos is a research-oriented methodology and, although it has been used in many significant case studies, the modeling of system requirements has always been led by researchers. The objective of this paper is to look at what happens when practitioners use the Secure Tropos framework in modeling and analyzing requirements in a real safety-critical scenario set in the context of Air Traffic Management. We will discuss the specific challenging problems that an ATM scenario introduces (Section 2) and how to choose an appropriate a methodology for modeling S&D requirements (Section 3) and how they have been handled with Secure Tropos. Particularly, we will focus on how to model the ATM organizational structure and associated trust and risk relations and present some S&D patterns for the ATM scenario (Section 4). We discuss strengths and weaknesses of the methodology (Section 5) and conclude with final considerations (Section 6).

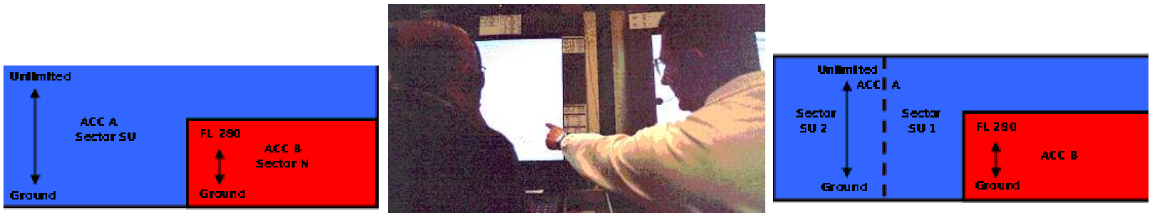
2. S&D for Air Traffic Management

This work is part of the SERENITY project[1], whose objective is to provide S&D patterns for the protection of critical infrastructures such as the air transportation system. In order to engineer S&D patterns, it is mandatory to be able to capture S&D requirements. This is a daunting task because we must (i) capture the context of the overall socio-technical system with a shared unambiguous language and (ii) validate and assess the S&D properties, and in the end (iii) make them suitable for automated processing by a run-time system supporting the S&D properties of the overall system. This introduces the need of coupling between computer-supported RE with ethnographic approach.

In a nutshell, Air Traffic Management (ATM) is an aggregation of services provided by ground-based Air Traffic Controllers (ATCOs). ATCOs must maintain horizontal and vertical separation among aircraft. They must ensure an orderly and expeditious air traffic flow, providing flight context information to pilots, such as routes to waypoints, weather conditions, and (last but not least) issuing orders and directions. These services are provided by Airport Control Towers for the arrival and departure flight phases and Area Control Centres (ACCs) for the en-route flight phase.

The airspace managed by each ACC is organized into adjacent volumes (called Sectors) with a predefined capac-

[1] EU-IST-IP 6th Framework Programme - SERENITY 27587 - http://www.serenity-project.org



(a) Airspace before re-sectorization (b) EC (Left) and PC (Right) cooperate (c) Airspace after re-sectorization

Figure 1. Managing a sector with unexpected increase in traffic

ity (i.e. number of flights that can be safely managed at the same time). Each sector is operated by a team of two ATCOs, consisting of a Planning Controller (PC) and an Executive Controller (EC), who work together as a team and share the responsibility for safe operations of the sector.

ECs are in charge of all air-to-ground communication. They monitor aircraft in the sector and provide pilots with instructions on route, altitude, speed, information on weather, and air traffic condition. When an aircraft approaches the sector boundary, they pass it off the responsibilities to the EC on adjacent sector. PCs assist ECs, coordinating entry-exit flight-point and altitude with adjacent sectors, in order to ensure a smooth air traffic flow. They also monitor the traffic within their sector and in most cases they update the ACC system with the ECs clearances.

Groups of neighboring sectors are coordinated by a supervisor. Supervisors are also responsible to define sectors' configuration in order to manage the traffic forecast for the next period. The predefined procedures of sector configuration require the availability of a sector team (PC and EC) and resources (e.g., radar display, workstation, etc.) to safely manage the traffic (Figure 1(b)). Supervisors should monitor, assist, and, if needed, temporarily take over the roles of controllers. Supervisor have also the responsibility to promptly avoid delays in crucial information transmissions due to partial failures of automatic information systems. They typically accomplish this task by communicating directly with controllers.

The story below is an example in which airspace *re-sectorization* resolves an unexpected increase of air traffic.

1. It's 11 am. Paula (PC for Sector SU of ACC A, see Figure 1(a)) notices from the planning traffic display that there will be an increase of the number of flights for next hour which exceeds the capacity of Sector SU.
2. Paula asks Robert (Supervisor) to reach her position and to proceed with the horizontal splitting of Sector SU into two sectors. As shown in Figure 1(c), Sector SU is divided along the dotted line, so that only Sector SU 1 is the neighbor of Sector N from ACC B.
3. At 11.15 am sector re-configuration is carried out and is registered by the system in the daily log.
4. As soon as the re-sectorisation is operational, Paula informs neighboring sectors of ACC A, providing them with the radio frequency of Sector SU 2 and instructions for the traffic handover.
5. Luke (EC Sector SU 1) hands over the air traffics above Sector SU 2 to the new control team of Sector SU 2. The system shows in the displays of both sectors (SU 1 and SU 2) the traffics that are changing Sector.
6. It's 12 am. Paula and Luke can manage the expected peak of traffic safely.

The Challenges

Activities such as air traffic controlling and supervision, significantly depend on ATCOs' expertise on specific domains and often rely on casual, relaxed and proactive communications among them: all these elements contribute to create what is called *shared context awareness*. Context awareness refers to the ATCOs' understanding of processes and relevant aspects in dynamic environments where they interact [18]. In ATM, it is crucial especially when ATCOs have to deal with uncommon situations and they are required to find solutions that have not covered yet by current procedures. Context awareness in ATM is achieved through a strong design commitment in the following issues:

Collaboration among workers is critical where the tightly coupled or shared activities among ATCOs rely on overhearing-overseeing one another's moves and on shared knowledge.

Example 1 *Paula (PC) and Luke (EC) work cooperatively because they play two roles which are intertwined for the safe accomplishment of an activity. An activity is typically performed throughout a set of actions with given goals and a set of operations defined by activity conditions [17].*

Shared use of artifacts contributes to the consolidation of context awareness among workers. In a sector team, representations of information are multiple and the use of some artifacts can be shared.

Example 2 *Paula (PC) and Luke (EC) share Radar Display which is replicated in their working stations.*

Emergent behaviors frequently appear to encounter dynamism in complex systems. Actors' action effect others goals unintentionally. This relation can not be modeled by contribution relations among actors because typically they are used for intentional behavior.

Temporal nature of work is important to capture the nature of work in terms of sequence, routines and rhythms which usually are not formalized.

3. Choosing a Modeling Framework

Most of the challenges, mentioned in the previous section, are related to S&D properties. Many works have been proposed for modeling and evaluating dependable mission critical systems [1, 12, 19, 20]. The most appropriate type of model framework depends on the complexity of system behavior and the S&D requirements to be taken into account. Analytical modeling, such as Monte-Carlo simulations, can be used during the early stages of the design to evaluate several alternative architectures and select the most suitable for S&D requirements [1].

Threat and fault analysis identifies the causes of security and dependability failures, such as a set of uncertain events that can obstruct the correct functions of the system. Failures correspond to the system states where the S&D requirements are no more fulfilled. Each failure is assessed in terms of likelihood and the level of losses, and analysts treat them according those assessments.

The analytical models can be classified into non-state space models and state space models. Reliability block diagrams, fault trees, and reliability graphs are non-state space models that are commonly used to study dependability of systems [12]. They are concise, easy to understand, and moreover have efficient analysis with the support of suitable evaluation methods and tools. However, they cannot model concurrency, synchronization, or server failures, since these violate the model assumptions. Furthermore, those works traditionally put more focus on the technical-system and overlook the organization as an integral part of the system.

State-space based models include Markov models and high level approaches which have an underlying Markov model. These models provide great expressiveness for dependability specifications and have been extensively applied to model the dependability and to analyze the hardware-software reliability [21]. This class of models includes queuing networks [5] and Generalized Stochastic Petri Nets [7, 4], which have been the most commonly used. Recently, many frameworks based on UML [8] have been proposed. Generally, they propose to embed Petri Nets for describing the system properties and validating the solutions.

In the safety and reliability community, the classic models (e.g., FTA [19], FMECA [6]) have been commonly used by engineers. However, these works have been applied for modeling and valuating computer based systems without considering the dependability of the organizations where the system-to-be will operate. Answering to the last challenge, several attempts have been made by extending requirement modeling frameworks[2] (e.g., KAOS [20], *i** [13, 16]) such that it can model and analyze a critical system. However, those two approaches are not adequate enough. The former only analyzes the system-to-be without considering the organizational aspects, while the latter models S&D requirements as non-functional requirements which are too vague to be verified. Moreover, we argue that S&D requirements are properties of a system that must be verified up to a certain extent (i.e., not just a boolean value).

4. Modeling Framework and S&D Patterns

To address the mentioned challenges, we used a modeling framework, called Secure-*i**/(SI*), which unifies Tropos Goal-Risk Model [2] and Secure Tropos [10]. SI* can be used to analyze the organizational structure in terms of agents/actors that are stakeholders of the system (i.e., humans, software, or machines) and the relations among them. Each of them is in charge of a set of goals to be fulfilled under any circumstance. An agent might be responsible to fulfill a goal even if it does not have capabilities or capacities to do so. The methodology underlying the framework is summarized in Table 1.

The modeling starts by identifying stakeholders and categorize them into three notions: role, agent, and group. Afterwards, strategic-interests of each stakeholder/actor are identified and follow by interdependency among them. In SI*, there are two types of dependencies: an agent (delegater) may delegate to another agent (delegatee) to fulfill a goal and delegater gives permission to the delegatee to use a goal. Note that, the delegatee not necessarily has full capability to fulfill the goal (delegatum), because it could be the case the delegatee does a further delegation. By delegating the fulfillment of a goal, the delegater becomes vulnerable. The delegater can be failed because the delegatee fails to fulfill the delegatum. It is also the case for giving permission to another agent, the delegater takes a risk because the other agent could misuse the resource that has been granted to him. Indeed, the dependability of a system does not only depend on the reliability of a machine, but also depends on the interdependency among agents in the system.

Afterwards, analysts identify trust relations among stakeholders. SI* allows us to have a delegation/a permission to the un-trusted[5] agent. When an agent (source) grants permission to another agent (target), it does not imply that the source agent trusts the target one. Sometimes, the source should grant permission/delegates a goal fulfillment to the one which is un-trusted or distrusted. This sit-

[2] These frameworks consider the organizational structure where the system will be operated

[5] "Un-trusted" means neither trust nor distrust relation

Step	Description	Example
Identify stakeholders	identify who are involved in the system	Executive Controller (EC), Planning Controller (PC), and Supervisor. A group (called Team SU) is responsible to manage Sector SU. Team SU consists of an EC (played by Luke) and a PC (played by Paula)and Roberts plays as a Supervisor.
Identify strategic rationale	analyze the intentions, capabilities, ownerships[3] of stakeholders about particular goals[4]	Luke must manage the traffic in Sector SU. Consequently, he must coordinate the exit point of an aircraft with related adjacent sectors. Robert entitles to allocate ACC's resources (working station, telephone, radio, radar display).
Identify strategic dependencies	analyze dependencies among stakeholders about a particular goal	Luke must coordinate the exit point of aircraft with adjacent sectors. Unfortunately, EC usually is busy enough managing current traffics in his sector therefore he does not have time to coordinate with adjacent sectors. Luke delegates this goal to Paula, such that she will coordinate with Luke's adjacent sectors on behalf of Luke.
Identify trust relations	identify trust relations among stakeholders about a particular goal	Luke must hand-over the controlling of aircraft to the adjacent sector when it is approaching the boarder. Even, Luke does not know who the next controller is. Luke must delegates to Paula the task of managing incoming traffic, because she acts as Planning Controller of his sector.
Identify risk factors	identify uncertain event that may obstruct the system and define the acceptable risk level	Controllers might ask for short relief of duty because of simple matters (e.g., receiving a call, going to the toilet).
Analyze the model	analyze model whether it has fulfilled the S&D properties and the risk is below the acceptable level	The S&D properties are checked as in "context" scenario described in Section 4. Analyze whether the risk of having a short relief of duty is acceptable.
Refine the model	improve the model such that it fulfills the S&D properties and the risk is acceptable	Introduce electronic flight-progress-strips to ease coordinating modification between controllers in a team. Establish trust relation, between team member.

Table 1. Modeling Steps of Secure $i*$

uation emerges when an agent must delegate the fulfillment of a goal or give permission in which it does not have any power to choose another agent (e.g., has been defined by the regulator) or it does not have any other option (e.g., no trusted target agents). SI* specifies a trust relation in three levels: trust, un-trust, and distrust; and each level implies different level of risk to the system.

The next step is identifying events or threats that can compromise our S&D requirements. The framework allows us to model an uncertain event that acts as a risk (negative effect) or an opportunity (positive effect). This event should be assessed against its likelihood and its effect to the related goals. Finally, the model is analyzed with the help of a suite of tools to ensure whether S&D requirements are fulfilled and calculate the risk level of the model. On the base of the results, analysts refine the model (the strategic rationale, dependency, or trust relations) such that all the S&D requirements hold and introduce a set of treatments (i.e., tasks) so that the risk levels are acceptable.

In the following, we present sample S&D patterns that we have identified in the SERENITY project using the SI*/SecureTropos methodology. These S&D patterns (adapted from [15]) are at organizational level, and not software/system patterns. These patterns are structured into three sections: *context* when the pattern can be applied, *requirements* are S&D properties that must be fulfilled by the pattern, and *solution* which describes how the new organizational setting must be implemented such that the requirements are fulfilled. In our work, the context and solution must be represented formally, besides natural language description. The formalization allows us to verify the fulfillment of S&D requirements formally and automatically.

4.1. Pattern 1: Collaboration in Small Groups

Context: Workers/agents have to cope with complex activities where tight coordination among them are crucial. Actor 1 (Paula) and actor 2 (Luke) are part of a team (Team Sector SU) which is responsible to achieve Main-Goal (managing aircraft in Sector SU), and each actor is in charge to achieve a subgoal (G1 or G2) of Main-Goal.

Example 3 *Luke must manage current aircraft and Paula must plan incoming traffic and assess the acceptability of the sector for incoming traffic.*

The failure of one subgoal has a bad impact to the satisfaction of another subgoal.

Example 4 *Once Paula fails to perform her responsibility to plan incoming traffic, then Luke will have difficulties to manage current traffic.*

The success of this pattern depends on the possibility to organize the work in small dedicated groups of workers aided by working arrangements and various artefacts.

Requirements: The Main-Goal is guaranteed to be available and reliable.

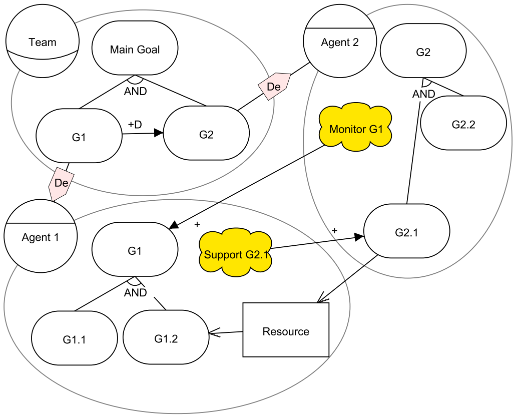


Figure 2. Solution of Coll. in Small Groups

Solution: To minimize the risk of G2 getting bad impact from G1, A new mechanism "Monitor-G1" is introduced to ensure the fulfillment of G1 which is performed by actor 1 and also vice versa. Actor 1 also adds the goal of "Support-G2.1" to ensure the availability of the resource that it needs from actor 2. The model can be seen in Figure 2[6]

Example 5 *Luke monitors the work of Paula in managing incoming traffic. Moreover, Luke supports Paula by organizing incoming flight progress strips, such that it helps Paula in annotating FPSs updating flight data.*

Moreover, applying this pattern also means defining the group size and its composition in terms of members' capabilities, expertise, and knowledge.

4.2. Pattern 2: Reinforcing Overlapping Responsibilities for Robustness

Context: Several agents must accomplish a task that requires high level of safety. Therefore, those agents share responsibility for fulfilling it. A critical activity (Main-Goal) is responsible of a team that consists of agent 1 and agent 2. Agent 1 and agent 2 work together and share the responsibility for the safety-critical task. To achieve "Main-Goal", agents must have "Capability 1" and "Capability 2", as we have seen in the previous examples.

Requirements: To ensure the safe operations of a team. The Main-Goal must always be fulfilled (available and reliable), even if one of the team member is temporarily absent.

Solution: Foresee the explicit (in terms of procedures, manuals) interdependence of roles amongst tightly coupled group of agents in safety critical systems. Initially, a team mate is defined as "can play" the other task such that the team mate has similar capability from another agent. In normal conditions this pattern provides cooperation, supervision, advice and sharing of knowledge and in times of difficulty it allows for agents to take over the tasks and responsibilities of others with a certain degree of fluidity.

Moreover, this pattern requires the team members to have the capability playing the others' roles. Sometimes this requirement will be contradicted by the fact that novice controllers should work with the experienced ones because of safety reason or to train them.

5. SI* Strengths and Weaknesses

At first, by using SI* models we experienced a remarkable easiness in communicating the model between security analysts and domain experts. This point is really significant for the success of a requirement analysis, because it enables the domain experts to understand and criticize directly the RE model which might differ from their knowledges.

All modeling languages come with a rich choice of features and constructs. Modeling organizations was a key claim of the original *i** modeling language. We find out that the initial set of entities used by *i** could be reduced to four: "role" (loosely speaking for capturing classes), "agent" (loosely speaking for capturing instances), "play" and "part-of" relations, and the rest could be discarded. For example, positions can be substituted by a role inheriting from different roles. Sometimes we stretched the English meaning of the relations but adding more constructs would have decreased the readability of the models and hindered their communication to industry collaborators.

At the same time, the ATM models, and we believe many if not all organizational models, badly required some constructs. The first and foremost missing notion is *supervision*: namely the ability to "give orders" to somebody. The social relation among Boss and Employee cannot be captured by the part-of relation nor by the classical notions of dependencies in *i** nor the new relations of delegation of either execution of permission [10] in SI*.

The second notion is about *collective or group responsibility*, the most debated point that could not find a convincing representation with *i** and its descendants. Each member must fulfill its part, and all together they must ensure the fulfillment of the other members' parts, as in Example 2-3.

We noticed that designers are often prescriptive: they tell what the system should do (after all they are designing it!). In contrast, many S&D features require a notion that we term *mandatory non-determinism*: for example to ensure the fulfillment of their responsibility, each agent (Luke and Paula) is *possible to play* both roles. These circumstances cannot be modeled by a "play" relation, because the same agent will end both as EC and PC which is against the regulations. So, we introduced "can-play" relations and similar variations for or-decomposition.

Finally, one of the problems of using a research-driven modeling language is the lack of a document irrelevant for academics but essential for practitioners (e.g., manual).

[6] The colored constructs are the addition for the existing context

This might be a feature (the language is open to changes and adaptations) or a bug (people from industry having to read research articles with many small inconsistencies). This issue traverses to many RE methodologies and case studies. For example, in the paper combining *i** and e3value [11] no reference manual on either methodology is given thus revealing its academic conception.

6. Conclusions

In this paper we have reported our experience in using the SI*/Secure Tropos methodology for modeling and analyzing requirements of an Air Traffic Management system. This work has been very useful to understand the strengths and weaknesses of the methodology in practice.

The major asset of the methodology is its simplicity: after essentially a meeting to introduce the language, it was possible to communicate directly between experts from industry (who never saw a Tropos Model before) and members from academia (who never saw an EC or PC before) using directly SI*/Secure Tropos diagrams. Quite rapidly domain experts from industry were able to correct and propose alternative models. This was a significant point for the project since it allowed establishing a common language.

Looking at expressivity, we could model most features [14] describing the foreground of the socially organized nature of work (awareness of work, distributed coordination, plans, and procedures) even though some addition was needed to deal with *collective* goals. Spatially oriented features, describing the physical nature of the work and observable arrangements within the workplace, have are still out of reach. Still, we do not see a viable alternative among other RE methodologies. Another limitation (shared with *i**/Tropos) is the difficulty of moving smoothly from a conceptual view to a process view. For instance, and-decomposition might be either sequence or parallel flow. This will be the subject of our future work.

Acknowledgments

This work has been partially funded by EU Commission, through the SENSORIA and SERENITY projects, by the FIRB program of MIUR under the ASTRO and TOCAI projects, and also by the Provincial Authority of Trentino, through the MOSTRO project.

References

[1] SQUALE: Security, Safety and Quality Evaluation for Dependable Systems. Technical report, SQUALE Consortium, 1999. Final Deliverable.

[2] Y. Asnar, P. Giorgini, F. Massacci, and N. Zannone. From Trust to Dependability through Risk Analysis. In *Proc. of the 2nd Int. Conf. on AReS*. IEEE Press, 2007.

[3] A. Avizienis, J.-C. Laprie, B. Randell, and C. E. Landwehr. Basic Concepts and Taxonomy of Dependable and Secure Computing. *IEEE TDSC*, 1(1):11–33, 2004.

[4] C. Betous-Almeida and K. Kanoun. Stepwise construction and refinement of dependability models. In *Proc. of DSN '02*, 2002.

[5] O. Das and C. M. Woodside. Layered Dependability Modeling of an Air Traffic Control System. In *Proc. of Workshop on Architecting Dependable Systems-ICSE'03*, 2003.

[6] US Department of Defense. Military Standard, Procedures for Performing a Failure Mode, Effects, and Critical Analysis. MIL-STD-1629A, 1980.

[7] N. Fota, M. Kaaniche, and K. Kanoun. Dependability Evaluation of an Air Traffic Control Computing System. In *3rd IEEE IPDS*, 1998.

[8] H. Gabor and M. Istvin. Quantitative Analysis of Dependability Critical Systems Based on UML Statechart Models. In *5th IEEE International Symposim on HASE*, 2000.

[9] P. Giorgini, F. Massacci, J. Mylopoulos, and N. Zannone. Modeling Security Requirements Through Ownership, Permission and Delegation. In *Proc. of RE'05*, pages 167–176. IEEE CS Press, 2005.

[10] P. Giorgini, F. Massacci, J. Mylopoulos, and N. Zannone. Requirements Engineering for Trust Management: Model, Methodology, and Reasoning. *Int. J. of Inform. Sec.*, 5(4):257–274, 2006.

[11] J. Gordijn, E. Yu, and B. Raadt, van der. E-Service Design Using *i** and e3value Modeling. *IEEE Software*, 23(3):26–33, 2006.

[12] A. M. Johnson and M. Malek. Survey of software tools for evaluating reliability, availability, and serviceability. *ACM Comput. Surv*, 20(4), 1988.

[13] L. Liu, E. S. K. Yu, and J. Mylopoulos. Security and Privacy Requirements Analysis within a Social Setting. In *Proc. of RE'03*, pages 151–161, 2003.

[14] D. Martin, T. Rodden, M. Rouncefield, I. Sommerville, and S. Viller. Finding Patterns In The Fieldwork. In *Proc. of ECSCW 2001*, 2001.

[15] D. Martin and I. Sommerville. Patterns of Cooperative Interaction: Linking Ethnomethodology and Design. *ACM Trans. on Comp.-Human Interaction*, 11(1):59–89, 2004.

[16] N. Mayer, A. Rifaut, and E. Dubois. Towards a Risk-Based Security Requirements Engineering Framework. In *Proc. of REFSQ'05*, 2005.

[17] B. Nardi. *Context And Consciousness Activity Theory And Human Computer-Interaction*. MIT Press, 1996.

[18] T. Sheridan. *Humans and Automation: System Design and Research Issues*. Willey, 2002.

[19] M. Stamatelatos, W. Vesely, J. Dugan, J. Fragola, J. Minarick, and J. Railsback. *Fault Tree Handbook with Aerospace Applications*. NASA, 2002.

[20] A. van Lamsweerde, S. Brohez, R. D. Landtsheer, and D. Janssens. From System Goals to Intruder Anti-Goals: Attack Generation and Resolution for Security Requirements Engineering. In *Proc. of RHAS'03*, 2003.

[21] Z. Zhang, H. Shen, X. Defago, and Y. Sang. A Brief Comparative Study on Analytical Models of Computer System Dependability and Security. In *Proceedings of the 6th Int. Conf. on PDCAT*, 2005.