View metadata, citation and similar papers at core.ac.uk

brought to you by CORE

provided by AIS Electronic Library (AISeL)

Association for Information Systems
AIS Electronic Library (AISeL)

ACIS 2004 Proceedings | Australasian (ACIS)

December 2004

# Threats Classification with Ishikawa's diagram

Elena Ivankina
*Université Paris*

Follow this and additional works at: http://aisel.aisnet.org/acis2004

Recommended Citation
Ivankina, Elena, "Threats Classification with Ishikawa's diagram" (2004). *ACIS 2004 Proceedings*. 101.
http://aisel.aisnet.org/acis2004/101

This material is brought to you by the Australasian (ACIS) at AIS Electronic Library (AISeL). It has been accepted for inclusion in ACIS 2004 Proceedings by an authorized administrator of AIS Electronic Library (AISeL). For more information, please contact elibrary@aisnet.org.

# Threats Classification with Ishikawa's diagram

Elena Ivankina
Département Systèmes d'Information et de Décision - ESSEC
Avenue Bernard Hirsch B.P. 105 95021
Cergy-Pontoise Cedex France
CRI, Université Paris 1 - Panthéon Sorbonne
90, rue de Tolbiac, 75013 Paris, France
Email: elena.ivankina@malix.univ-paris1.fr

## Abstract

*Threats identification plays an important role in information system design. This paper proposes an approach to analyse causes of threats using extended Ishikawa's diagram. Several classes of causes are identified, introduced and put in example; each class containing several sub-classes. This causes analysis is part of a global threat-based approach which consists of three steps: identification, analysis, and treatment. Causes classification is used at the analysis step.*

**Keywords**

Threats, requirements engineering, Ishikawa's diagram, classification, cause analysis

## INTRODUCTION

The purpose of requirement engineering (RE) is to provide a complete and coherent set of system requirements. This objective is difficult to reach. The set of requirements is often heterogenous, incomplete or contains conflicting requirements. The functioning of the system could be altered by these problems as well as other problems induced by the execution of the system. RE uses and discusses formally both existing and new information to identify new requirements. The main objective of RE is to prevent potential system malfunctioning rather than to repair them during system usage (Lutz 1993). Indeed, identifying problems early in the system development project allows to simple and cheap updates of the system requirements and avoids complex and costly revisions that become inevitable once problems are detected late.

In the domain dealing with malfunctioning of system, several researches are made. These researches handle several notions. Worthy of note: obstacles (Lamsweerde et al. 1998) and conflicts (Robinson1996). Both concepts relate to the goals that users have in mind when they use a system. *Obstacles* are defined as phenomena that occur within the system environment and prevent achievement of a user goal. A *conflict* occurs when two goals on the way to be achieved obstruct each other. In Use Case (UC) based requirements elicitation approaches, cases of malfunctioning of the system are specified with *exceptional scenarios* or using *Misuse Cases* (Alexander, I. 2002) , (Potts C. 1994). The notion of risk (Kontio J, 1997) from System Engineering domain is very close from those described below. The *risk* is generally defined as the potential loss of the system's ability to achieve a given requirement. So far no approach has proposed (i) a unified analysis for any kinds of threats, nor (ii) the systematization of threat analysis in UC modeling. We consider these kinds of threats as similar, especially in the obstruction sense. Each of these notions provides an obstruction to the goal which they affect. Even if these obstructions are of different kinds or degrees, they all lead to the same result: the goal is obstructed. Therefore we propose to unify their analysis within a single process.

In this paper an approach is proposed for treating the malfunctioning of the system. Each barrier to the functioning of the system is considered as a threat. We propose to analyze threats by studying their causes. In order to see more accurately thought the whole link of causes for each requirement. The article presents an approach introduced by Ivankina (2004a, 2004b) and Salinesi (2004) that permits to identify and analyse threats starting from a given requirement. In order to discover new requirements we use the goal-scenario support. The approach is designed as a goal discovery rule composed of three stages: (i) one to systematise the identification of threats, (ii) one to analyse their causes and (iii) one to evaluate their consequences on the system and treat them. The discovered goals are intended to be integrated into the requirements documents as new goal-scenario or its components. This paper is

focused on the analysis stage of our approach. An Ishikawa diagram extension is proposed with a detail of each class. Possible causes of threats are classified in five classes. Parts of this classification are supported by different specialised classifications, for example a part of CREWS-SAVRE user errors centred classification, (Maiden N.A.M. 1998).The classes allow more accurate threat analysis; they are presented and developed in this paper. Each class is supported by a model.

The next section provides a global overview of the method. Threat discovery is achieved by heuristic rules. Then the Ishikawa's diagram, inspired from Ishikawa K. (1976) is used for analyzing threats. Finally an extensible typology of threat treatment strategies (Ivankina E. 2004) is proposed. The third section describes a threat analysis rule and shows how it can be extended to cause class refinement and their models representation. Future work is discussed in the concluding section.

## APROACH OVERVIEW

The approach is a rule based method. Its purpose being the discovery of new goals, we refer to the goal-scenario representation in order to reach it. It uses the original goal and scenario couple (i) to properly formalise a set of possible threats, (ii) to analyse one or more threats of this set using Ishikawa's diagram and finally (iii) to treat it with a set of treatment strategies. The result is a set of new goals that can be analysed in they turn. Figure 1 presents the three main stages of the approach.

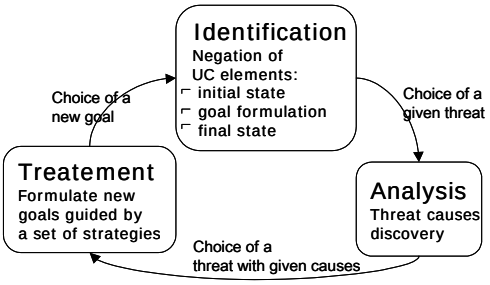


Figure 1 Approach stages

**Identification strategy**

The negation strategy is used in order to identify threats. The idea of this strategy is to find various what-if situations, in which the goal is not reached. The purpose is to find which elements in goal-scenario couple could be at the source of a threat. The approach is applied to the basic form of the goal-scenario which is composed of goal formulation, initial and final states and actions of the scenario. The analysis is especially applied to its initial and final states, and the goal formulation. The three elements strongly contribute to the goal obstruction:

- Goals may be totally obstructed if their initial state cannot be reached. In this case the required conditions for starting the goal achievement cannot be reached.
- If the final state cannot be reached the goal execution is not achieved totally or even partially.
- Goal formulation is important for the requirements accuracy, the potential threats could be found in the goal formulation if it is not exactly representative of stakeholders' needs, or is not sufficiently accurate

The result of the negation strategy is a first set of threats that could be analysed and used in several ways. This set is raw and can contain redundant or different abstraction levels threats. It contains all possible threats that may be formulated with the grammatical and meaning negation. Consequently the new formulation could be insignificant. Consequently one or several threats must be chosen for analysis. The choice could depend on its risk, its frequency, its duration or its strategic importance. We propose to choose one threat and to apply to it a cause analysis. Ishikawa's diagram analysis is suggested to analyse its causes.

**Threat Analysis**

Even if it is possible to identify several causes for each threat without particular technics, this set is still incomplete. It is suggested to apply a formal identification method to systematise and represent threat causes. Thus the Ishikawa's diagram analysis is proposed.

The original Ishikawa's diagram has been developed for the industrial environment; consequently it was necessary to adapt this concept to the RE domain. Besides, we propose five classes of causes adapted to the domain of Information Systems. These classes where defined as a mix between Ishikawa's original classes, the four worlds framework on Information Systems developed in the $F^3$ project (Rolland at al 1998), HAZOP, (Kletz 1999, 2001) and others requirement identification methods.

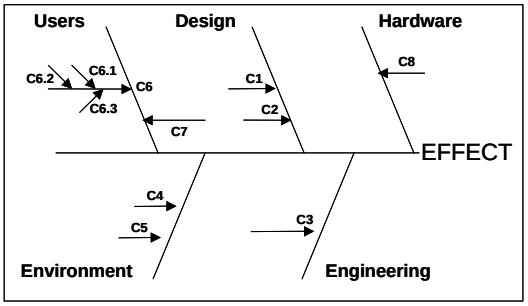


Figure 1: Ishikawa's cause and effect diagram extension

As Figure 2 shows, in order to construct a cause and effect diagram, it is necessary to first identify a problem. Once the problem stated, four to six classes of causes are identified (the number must remain reasonable, but there is no limitation). In the original approach four classes of causes of industrial production problems are proposed: Materials, Machines, Manpower, and Methods. Each of these classes is used as a guide to list the causes of the problem. For example, C4 and C5 are two environment causes of the problem at hand. Once the list of causes completed, each individual cause can itself be taken as a problem and recursively analysed with the same diagram. This is for example the case of C6, which is detailed with C6.1, C6.2 and C6.3. The result is a tree whose root is the initial problem (the "effect"), and whose leaves describe the most detailed causes of this problem. Thus, once the threat and its causes are identified we propose to apply to it one or several treatment strategies.

**To treat the threat**

A set of strategies is proposed to apply treatments to the identified threat. These strategies set is actually classified according to the type of action exercised to the threat, such as ignoring threat, reducing or killing it. Several strategies may be used for the same threat. In the other way, it is also possible that no strategy corresponds to it. Anyway the main objective of these strategies is to formulate new goals in order to enrich and make the requirements set more complete. Once the new requirement is defined, a scenario could be attached to it. The new goal scenario couple can be analysed to find new threats.

## CLASSES OF THREAT CAUSES

Threat is considered as something that hinders the satisfaction of a goal in the framework of the information system usage. Several classifications for causes are always developed, and it is difficult to make a general classification. We propose to make a classification based on Ishikawa's original four classes; refined into sub-classes and enriched using the experience about other requirement elicitation methods.

The most important domains exploited for this paper are the System Engineering and Requirement Engineering. Several methods and approaches were used for our classification, such as HAZOP (Kletz 1999, 2001), CREWS-SAVRE, (Maiden N.A.M. 1998), SW-MM and SA-MM, (Higuera R. P et al. 1996) and (Williams R C et al. 1999), and so on. These methods are specialised in several domains of Information System design, and contain different points of view concerning our research. Thus we study each of these methods to adapt them to our approach.

The SW-MM and SA-MM methods propose a classification based on Software Risk Evaluation, Continuous Risk Management, Team Risk Management that is centred on the project risk management and propose three groups of risks: Product Engineering, Development Environment, Program Constraints.

CREWS-SARVE classification is user oriented. This method proposes different user faults typology. Typically it differentiates user slips and user faults. The environment plays an important role in this approach especially for an identification of threats linked to user. This method is supported by a scenario - goal couple.

HAZOP is a hazard identification method that comes from chemical industry. It is adapted to the Information System

design domain. The analysis approach used in this method is qualitative, it is also called the “what if” approach.

Thus an extension of the four original Ishikawa’s classes is suggested. The original approach proposes “Materials”, “Machines”, “Manpower”, and “Methods” classes that cover user problems, equipment, method, and material domains. Nevertheless it is not sufficient for a rich description on Information System design.

Using these results a classification proposition could be presented. We propose five causes’ classes: “User”, “Environment”, “Design”, “Hardware” and ”Engineering” classes. Each class characterise a special type of cause. Each class contain several sub-classes. The number of sub-classes is not in not fixed in each class; it depends from the model of class. These models are based on existing literature and the experience in requirements modelling.

**User class**

The user class is related to the user’s behaviour. This behaviour can cause several problems in the system execution. (Kletz 1999, 2001) (Maiden N.A.M. 1998).User interacts with a system. He/she perceives the information presented by the system, by using system interface; he/she reacts depending on his/her perception of this information. Thus threats could be caused by both actors: the system and the user, depending on their interaction. For example the system presents some kind of information to the user, user’s perception is based on the presentation and the content of this information, the action of user depends on it. If the user doesn’t understand the information his/her feedback will be wrong, what could causes a threat. This is the reason why action is viewed as a general notion which can corresponds both to user or system. User perception depends also on his/her capacities. These capacities could be physical, mental, or technical. It also could correspond to his/her concentration capacities, or his/her experience with the system use. In the same way system presentation depends on its interface, its performances and so on.

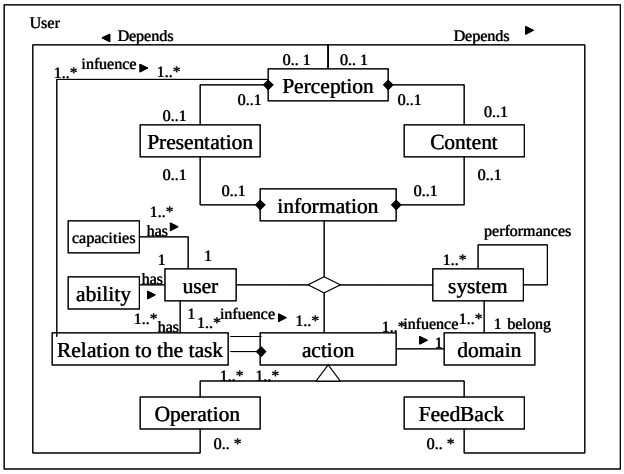


Figure 3: User's class

Figure 3 describes relations between user and system and its components. To synthetise this interaction several sub-classes are suggested. In these sub-classes the interaction is user oriented.

*Ability:* This sub-class corresponds to user knowledge of the system, for example if he/she is an expert or just a user.

*Domain:* It corresponds to the user knowledge of system application domain. System characteristics and processes change depending on domain. This system characteristic influences its use. Thus domain influences actions.

*Capacities:* It concerns physical, mental, technical, or concentration capacities of user. Two types of capacities are considered: default and acquired.

- Default capacities represent physical properties like a possible handicap of user.
- Acquired capacities are linked to the environment of work of user, such as concentration.

*Relation to the task:* This element of schema depends on user perception, the presentation on the computer interface, and the action that user has to achieve. The relation to the task could be altered if the computer interface is not clear, so user could have some difficulties to understand it. User could make mistakes in a case of repetitive or physically intensive tasks. The relation to the task composes the user’s action. So the action or the type of action influences it.

*Perception:* It refers to the comprehension of the information, such as presentation and content. It puts together presentation, content, feedback and operation. All these elements contribute to the user's perception. The sub-class is mainly system dependent. Is considered here the perception that user has of the presented information regardless of his abilities or capacities. The presentation is a possible information visualisation.

**Environment class**

The environment of system is a set of physical, virtual, organisational considerations and so on. Each system acts in a given type of environment that usually depends on the system domain (Higuera R. P et al. 1996) and (Williams R C et al. 1999). At the source of threats, different environment dimensions could be found. A system could be unable to achieve perfectly its goals because of an incompatibility between the system and the organization in which it works. For example it could be inappropriate to the organisational processes or hierarchy, typically for tasks validation. The organization perspective is linked its geographical dimension. System constraints are very different if the system is local or global. In the same way virtual and physical environment are differentiated. Physical conditions of system use must be defined as well as its interacting with other systems.

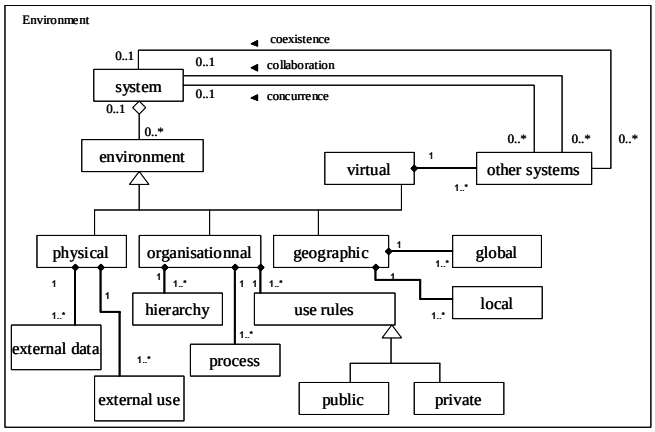


Figure 4: Environment class

Figure 4 presents this interpretation of system environment. It is organised in four sub-classes.

*Virtual:* This sub-class concerns other systems, and interaction with them. Several relationships types are defined between systems.

- Coexistence - in this case systems are aware about each other, but there are no real relations between them.
- Collaboration – it supposes that systems help each other in order to achieve their goals.
- Concurrencies - each system wishes to achieve its own goal to the detriment of other, or that they have to achieve the same goal in a competitive way.

*Physical:* It concerns physical conditions of system execution. For example, a system does not have the same behaviour if it works in difficult natural conditions, such as climatic conditions, or specificities of use. Consequently there are two sub-sections to this sub-class:

- *External data*, which defines physical environmental conditions,
- *External use*, which specifies the type of system use.

*Organisational:* It concerns enterprise processes, tasks, standards, policies, and hierarchy. This sub-class plays an important role in the user's behaviour. It depends on enterprise organization and processes that are applied, such as team work, hierarchy, processes. The system's behaviour is hardly affected by rulers of system use. This rulers restricts user rights and access permissions.

*Geographical:* It refers to system localization. Geographical dimension is very important in the system design. If the system is *global* special characteristics must be defined, like localization that is not necessary if the system is *local*.

**Design class**

Design threats could be due to the system development. For example the presentation of the information can be incorrect, if the system development is not perfect. Typically processes could change during the life cycle of the system if the development methods are not sufficiently dynamic - changes will be not updated. This class reflects design choices too. In this class the development team is considered, typically different changes and choices of development team could have an influence on technical choices.

This class presents very changing part of the system. Indeed even if some choices were made during the beginning of the project, these choices may be changed during the information system live cycle. Thus the following classification (Figure 5) is proposed that is centred on the perspective of change of the system:

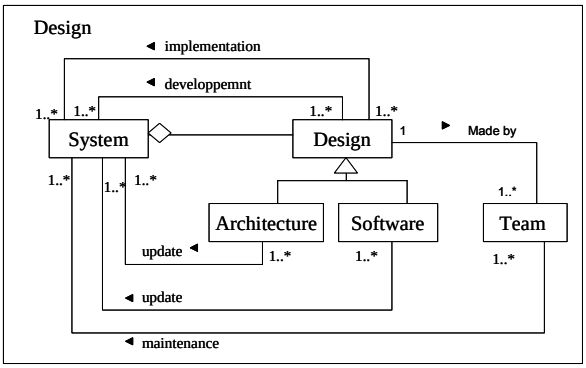


Figure 5: Design class

*Implementation:* this sub-class concerns implementation problems, for example code execution in real-time conditions. This kind of problems is very likely to the system execution just after service record, or during testing period, but it is also possible during the real execution of the system, such as bugs, impossible operations etc.

*Development*: this sub-class concerns development choices, and team. Different causes that could be classified to this sub-class could concerns difficulties dues to frequent changes in the development team, or development methods.

*Update:* this sub-class relate to the updating of the system, correction of identified bugs, or improvement. Causes that could be found in this sub-class could concerns inaccuracy in updating, like inconsistency between content and presentation of given information.

**Hardware class**

The hardware class represents possible causes related to the physical material supporting a system. Typical causes in this class are due to choices of system devices, the wear off, etc. Figure 6 suggests relations between its components.

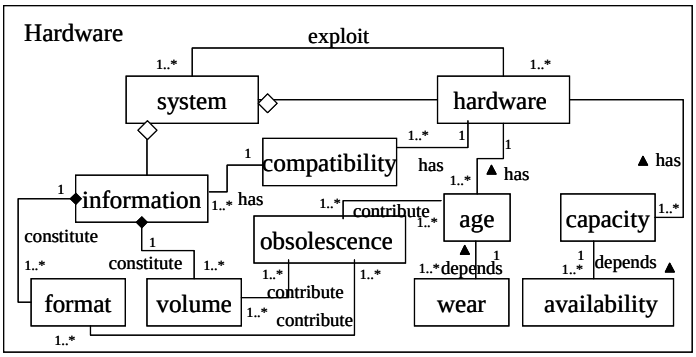


Figure 6: Hardware class

This class is composed of following sub-classes:

*Capacity:* this sub-class describes physical capacities of system devices, for example processors or memory capacities, which could be insufficient for the volume of information to be treated. The capacity determines the availability of the hardware

*Obsolescence*: is linked to the ageing of devices, but it is not the single cause of obsolescence. The obsolescence can be induced by the changing of requirements or information volume or format. The obsolescence of the system devices is defined when a given device can no longer satisfy the requirement for system usability.

*Compatibility*: this sub-class describes threat causes due to problems of compatibility between the type of device and, the type of information to be treated, as well as compatibility between devices, typically in networked systems.

*Wear:* this sub-class represents losses due to the system kind of use, for example if the system is used in the public place such as a bank machine could be physically wearing off quickly that a bank inside system.

**Engineering class**

This class represents possible threats due to the engineer's approach of system design and also project management problems. A similar classification is presented in (Higuera R. P et al. 1996) and (Williams R C et al. 1999). In the information system design process, engineering and management choices identify requirement set and project progress. Figure 7 describes main elements of information design process, which contain development standards, contracts, means, or requirement set quality.

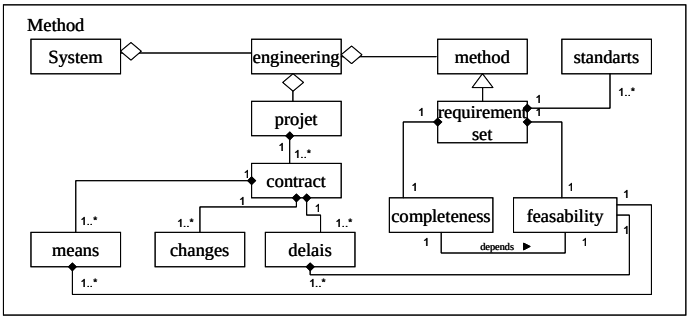


Figure 7: Method class

*Standards:* concerns the choices of standard definitions. This element influence mainly the requirement set and can depends on domain of system.

*Contract:* this sub-class shows constraints imposed by the contract on the project. This sub-class is detailed on delays, means, or changes that influence on contract execution, importance given to each stage of the project.

*Feasibility:* concerns the requirement definition and its accuracy. This element is considered as a sub-class, contrary to the completeness that is not represented is a sub-class because it is at the source of feasibility, which is also defined by means given to realise a requirement set, and the delay allocated to it.

All cause classes presented in this section are linked to each other. Some cause types contribute to the worsening; or reduction of others. Some of these cause types could help to discover others. For example if a threat cause is found in environment class and physical sub-class, it could be supposed that additional causes can exist in design and hardware classes. Furthermore the classification and relations between cause classes could help to elaborate identification patterns in future work.

In the following section is showed a general model of classes and sub-classes presented above. Some possible relations are deducted.

## INTERCLASSES RELATIONSHIPS

The classes of causes are connected to each other. These connections could help in their identification, and later, in the choice of treatment strategies. In this section a general model that represents these connections is presented.

Within this model are grouped together each class that was developed before. Then links between classes and sub-classes are defined that go beyond Ishikawa's diagram presentation. In each class of causes, there are one or several elements that could be linked to other classes. This relations complete classes' descriptions. These links are represented in red (Figure 8). In this section a brief description of some of these links is given.

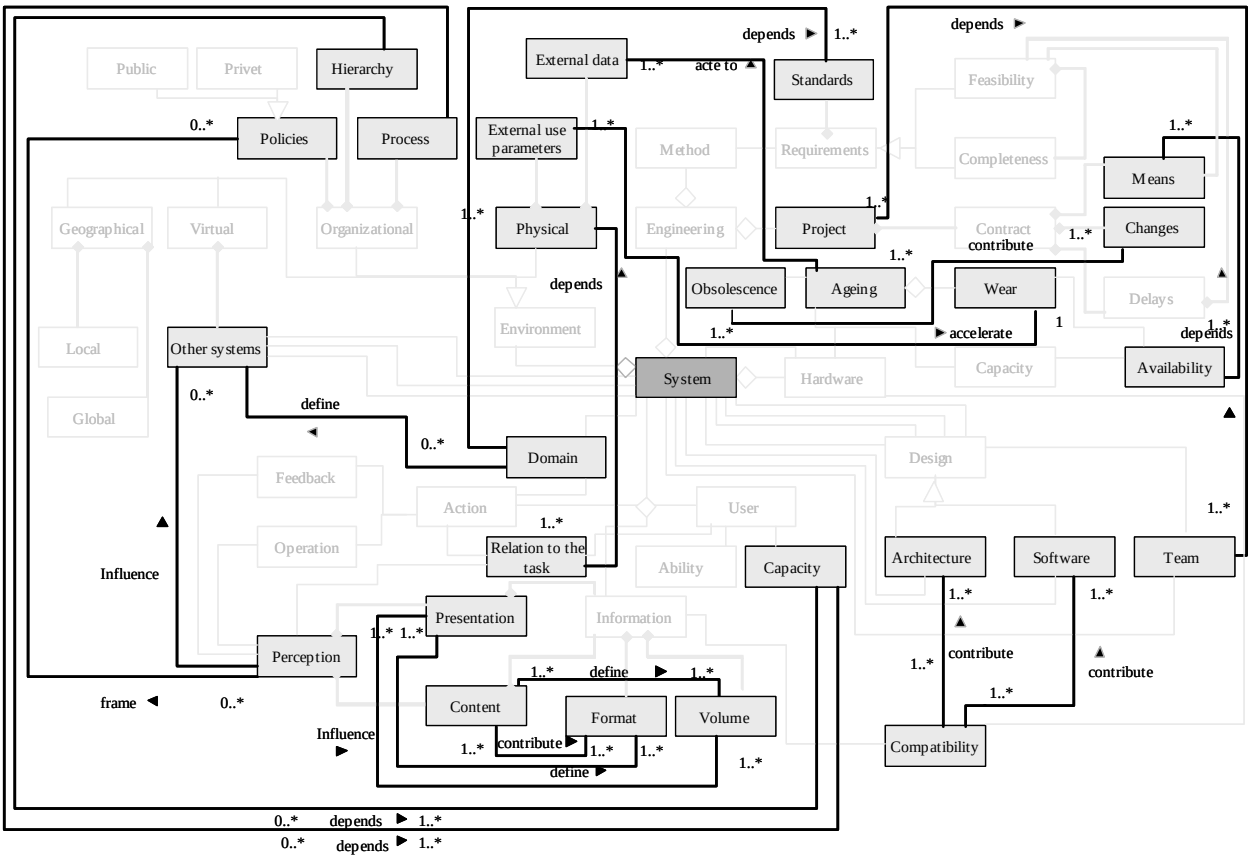


Figure 8: Causes classes relationships

The user class is linked to the Environment class, the Hardware class, and those of Engineering.

User's perception, behaviour and capacities are linked to the social environment. The environment acts mainly on the acquired capacities. User capacities are influenced by the enterprise organisation.

- *Capacities – Process:* The capacity element is directly linked to the process element. If the organisational processes is too complex or don't corresponds to the user vision of his tasks in the organization, it could distort his capacities. Processes link and redirect user actions.
- *Capacities – Policies:* Policies restrict access and limit the user power. The matching between these elements was resumed in (Crook 2002), where is underlined the importance of the clearly defined relation between organization policies and security efficiency.
- *Capacities – Hierarchy:* this relation express the dependency of user capacities from organisational hierarchy. For example, when the user don't know who is his/her hierarchical superior it could be difficult for him/her to achieve his/her task. The Environment acts also to the user relation to the task.
- *Relation to the task – physical environment:* if user works in a very noisy environment he/she could be distracted during his/her task achievement. User's relation to the task could be also altered by external data, for example if user works outside or inside.
- *Domain – other systems:* The definition of the domain depends often of the cohabitation with other systems. For example a given type of system exists in large network; this fact is a part of his domain definition.
- *Presentation, Content/Perception – Format, Volume/Information:* contribute to the information representation and comprehension. In the user class, the presentation and the content of information play an important role. Both depend on the elements of the hardware class. The format of the information determines its content and presentation, as well as its volume. The volume of the information constraints the presentation and the format can restrict the possibilities of the presentation.
- Standards – Domain: System use depends of domain of its use. Each domain has a system development standards set. The domain description and the system's adaptation to this domain depend on the quality

of the standards it's adaptation by the engineering team.

The hardware class is linked to the software class and the environment class.

- *Wear off – Usage parameters* - wearing depends on several things; for example does a device is used by one or several persons, does a usage is made outside or inside. If system is used outside it risks to be altered by natural phenomenon. The ageing element regroups the obsolescence notion, which depends on time, and on the evolution of requirements. To illustrate this evolution the notion of the obsolescence is linked to the project change in the engineering class.

The *method class* defines several elements of the system. Thus *project means* define the *availability* of devices. By definition of means is defined the budget for the project, the importance given to devices, and the availability of the devices.

In the hardware class, *compatibility* is linked to *the software and architectural design*. The design of the system and its architecture contribute to the compatibility of the hardware devices, typically in design's choices.

*The team* sub-class is linked to *the project*. The project characteristics define human resources in a project, and consequently the change within team depends of project organization.

A deeper analysis of links between classes and its components will be developed in future works.

## CONCLUSION

This paper presents a requirement identification approach guided by an analysis of threat causes. The proposed approach is presented under the form of a methodological pattern decomposed into three parts: one to identify threats, one to refine their definition, and one to identify how the system can treat the identified threats. Each part of the pattern is guided with a specific approach. The paper focuses on threat definition refinement by cause analysis. The cause analysis is supported by the Ishikawa diagram where causes are classified into five classes. Those five classes of causes represent user, environment, design, hardware and engineering sources of threats. Each of these classes is extended to sub-classes. Those sub-classes detail each class and provide a deeper threat identification approach. Possible relationships between classes and sub-classes are presented in a general model that regroups a particular model of each class. The relationships must facilitate and enrich threat identification. These relationships will be developed in future work.

The future work will be centred on cause classification.

- A detailed analysis of each class will be proposed with a description of relationships with other classes.
- Treatment strategies have to be developed. Specific strategies relating to each class must be identified.
- Further relationships between threat causes and treatment strategies will be achieved by a proposition of a set of patterns. The set of patterns is supposed to permit a ruler based semi-automatic threat identification and analysis method, aiming to the identification of new goals
- An evaluation of the method is planed, as well as a prototype development.

**COPYRIGHT**

[Ivankina Elena] © 2004. The authors assign to ACIS and educational and non-profit institutions a non-exclusive licence to use this document for personal use and in courses of instruction provided that the article is used in full and this copyright statement is reproduced. The authors also grant a non-exclusive licence to ACIS to publish this document in full in the Conference Papers and Proceedings. Those documents may be published on the World Wide Web, CD-ROM, in printed form, and on mirror sites on the World Wide Web. Any other usage is prohibited without the express permission of the authors

## REFERENCES

Alexander, I. (2002) "Modelling the Interplay of Conflicting Goals with Use and Misuse Cases". *Proceedings of REFSQ'02, 8th International Workshop on Requirements Engineering: Foundations of Software Quality*. Essen, Germany, pp. 146-153,

Crook R. Ince D. Lin L. Nuseibeh B., (2002) Security Requirements Engineering: When Anti-requirements Hit the Fan, *IEEE joint Internationnal Conference*, September 9-13, University of Essen, Germany

Ishikawa K. (1976) "Guide to Quality control". *Asian Productivity Organisation*,

Ivankina E, Salinesi C, (2004) An Approach to Guide Requirement Elicitation by Analysing the Causes and Consequences of Threats, *European Japanese Conference on Information Modeling and Knowledge Bases*, Skoevde, Sweden,

Ivankina E, Salinesi C, (2004) Adapting Ishikawa's Cause-Effect Diagrams to Understand Requirements Threats, *Business Information Systems, Proceedings of BIS*, Poznan, Poland

Jarke M, Pohl K. (1992). Information Systems Quality and Quality Information Systems *In Proceedings of the IFIP 8.2 Working Conference on the Impact of Computer Supported Techniques on IS Development.* Minneapoli, USA,

Julio S. Leite, G. Hadad, J. Doorn, G. Kaplan. (2000) A Scenario Construction Process *Requirements Engineering Journal*, Springer-Verlag. Vol.5 N1 38-61.

Higuera R. P., Haimes Y.Y. (1996) Software Risk Management *Technical Report* CMU/SEI-96-TR-012 ESC-TR-96-012 June

Kletz T. (1999). "Hazop and Hazan: Identifying and Assessing Process" Industry Hazards. *Institution of Chemical Engineers*

Kletz, T (2001) "An engineer's view of human error". *Institution of Chemical Engineers* A 3rd ed.

Kontio J, (1997) The Riskit Method for Software Risk Management, version 1.00, Institute for Advanced Computer Studies and Department of Computer Science, *CS-TR-3782, UMIACS-TR-97-38*University of Maryland

Lamsweerde A., Letier E.,(1998) Integrating Obstacles in Goal-Driven Requirements Engineering, *Proc. ICSE'98 – 20th Intl. Conference on Software Engineering*, Kyoto, April, Vol, 53-64

Lutz R R. (1993) Analyzing Software Requirements Errors in Safety-Critical, Embedded Systems *IEEE International Symposium on Requirements Engineering* San Diego, CA, USA.

Maiden N.A.M. (1998) CREWS-SAVRE: Scenarios for Acquiring and Validating Requirements. *Journal of Automated Software Engineering*, 5, pp. 419-446.

Potts C. Takahashi K., Antón A. I. (1994) Inquiry-Based Requirements Analysis. *IEEE Software*, pp. 21-32,

Robinson W.N., Volcov S., (1996) Conflict Oriented Requirements Restructuring *Working Paper* CIS-96-15

Rolland C, Ben Achour - Salinesi C, Cauvet C, Ralyté J., Sutcliffe A,. Maiden NA.M, Jarke M, Haumer P, Pohl K, Dubois E, Heymans P. (1998a)"A Proposal for a Scenario Classification Framework. *Requirements Engineering Journal*, Vol.3, N°1, pp.23-47,. Also available as CREWS report N°96-01 at URL http://sunsite.informatik.rwth-aachen.de/CREWS

Rolland C, Souveyet C, Ben Achour - Salinesi C. (1998b) Guiding Goal Modelling using Scenarios *IEEE Transactions on Software Engineering, Special Issue on Scenario Management*, Vol 24, No 12, p. 1055-1071 December.

Salinesi C, Ivankina E, (2004) Guiding Use case Discovery by analysing threats, *The 16th International Conference on Advanced Information Systems Engineering*, Riga, Latvia,

Tawbi M, Ben Achour – Salinesi C, Vélez F. (1999) Guiding the Process of Requirements Elicitation Through Scenario Analysis: Results of an Empirical Study In Proceedings of REP99, 1rst International Workshop on the Requirements Engineering Process - Innovative Techniques, Models, Tools to support the RE Process, Florence, Italy,

Williams R C. Pandelios G. J. Behrens S. G. (1999) Software Risk Evaluation (SRE) Method Description (Version 2.0) *Technical Report* CMU/SEI-99-TR-029 ESC-TR-99-029 December