Reusing Scenario Based Approaches in Requirement Engineering Methods: CREWS Method Base

Jolita Ralyté

CRI, Université Paris1- Sorbonne
90, rue de Tolbiac, 75013 Paris
ralyte@univ-paris1.fr

Abstract

Scenarios have proven useful to elicit, validate and document requirements but the development of new methods and tools for Requirements Engineering integrating scenario based approaches has been limited. The view developed in this paper is that scenario based approaches should be looked upon as reusable components. Our concern is therefore twofold : first, to represent scenario based approaches in a modular way which eases their reusability and second, to specify the design context in which these approaches can be reused in order to facilitate their integration in existing methods. The paper presents also an implementation of our proposal using SGML-HTML to store scenario based approaches in the multimedia hypertext documents and illustrates the retrieval of components meeting the requirements of the user by the means of SGMLQL queries.

1. Introduction

In the CREWS[1] project four different scenario-based approaches have been developed with the aim of supporting system requirements acquisition and vali-dation in a systematic way. Two approaches deal with the requirements acquisition from real world scenes [1] and from natural language scenario descriptions [2]. The two other approaches deal with the requirements validation through systematic scenario generation [3] and scenario animation [4]. The project hypothesis is that each of the approaches might be useful in specific project situations which are not well tackled by existing analysis methods and therefore, that it is worth looking for the integration of such approaches in current methods. This shall lead to an enhancement of the existing methods with scenario-based techniques.

[1] The work described in this paper is support by the European ESPRIT project CREWS standing for "Co-operative Requirements Engineering With Scenarios".

We situate our work in the Situational Method Engineering (SME) domain. The SME aims at defining information systems development methods by reusing and assembling different existing method fragments. This approach allows to construct modular methods which can be modified and augmented to meet the requirements of a given situation. Following this approach, a method is viewed as a collection of method fragments [5], [6]. New methods can be constructed by selecting fragments from different methods which are the more appropriate to a given situation [7], [8]. Thus, method fragments are the basic building blocks which allow to define methods in a modular way. In our work we are interested in specific method fragments, namely scenario based approaches, that we call *scenario method chunks.*

The objective of our work is to develop an approach for integrating different kinds of scenarios as method components into usual RE methods. To achieve this goal we propose to represent scenario based approaches in a method base as method components called scenario method chunks. We define also an approach to guide the selection of the must appropriate scenario method chunk to the situation at hand. Finally, we need to define an approach supporting the integration of the selected component in the existing RE method.

This paper is organised as follows. In section 2 we present the structure of the CREWS scenario method base. Section 3 explains the architecture of this base and finally, in section 4 draws some conclusions and discussions around our future work.

2. Structure of the CREWS Method Base

The CREWS Method Base stores the chunks of the methods based on the scenario use. The base is organised in two levels: method knowledge level and method meta knowledge level. Method knowledge level stores the content of the scenario method chunks, whereas the meta-knowledge level describes the reuse context of every chunk.

2.1 Method knowledge level

Figure 1 presents the meta model of the chunk. Every chunk comprises a product model and a process model. The product model represents the class of products obtained as outputs from the use of a chunk, whereas the process model represents the product development process and is supported by a guideline.

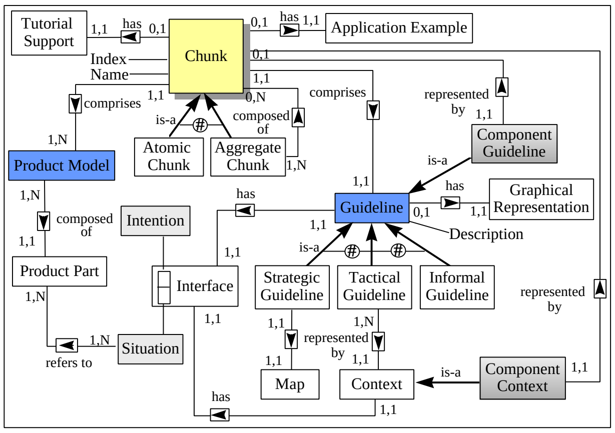
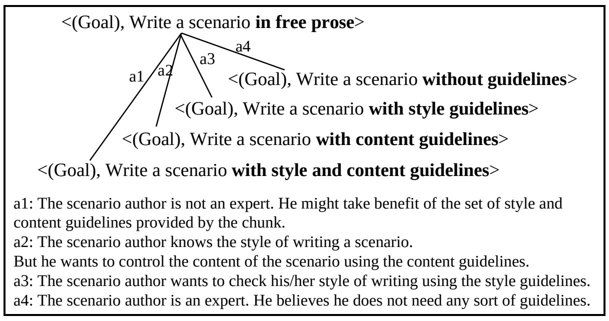


Figure 1: The meta model of the chunk.

The guideline has an interface and a body. The interface is defined by a couple <*situation, intention*> which characterises the conditions of its applicability: the current situation which is the input to the chunk process and the intention or the goal that the chunk achieves. The body of the guideline details how to apply the chunk to achieve the intention. It can be represented graphically or described informally according to the type of the guideline. There are three types of guidelines: strategic, tactical and informal.

Informal guidelines provide general assumptions and informal explanation describing how to proceed to obtain the target product.

Tactical guidelines propose a step-wise process to produce the corresponding product. They are represented by a tree of contexts following the NATURE process modelling formalism [9]. The informal description is provided to facilitate the understanding and the application of the chunk. Figure 2 shows an example of a tactical guideline which is represented by a choice context. This context proposes four different alternatives to write a scenario describing how to achieve a given goal.


<(Goal), Write a scenario **in free prose**>
a1
a2
a3
a4
<(Goal), Write a scenario **without guidelines**>
<(Goal), Write a scenario **with style guidelines**>
<(Goal), Write a scenario **with content guidelines**>
<(Goal), Write a scenario **with style and content guidelines**>

a1: The scenario author is not an expert. He might take benefit of the set of style and content guidelines provided by the chunk.
a2: The scenario author knows the style of writing a scenario.
But he wants to control the content of the scenario using the content guidelines.
a3: The scenario author wants to check his/her style of writing using the style guidelines.
a4: The scenario author is an expert. He believes he does not need any sort of guidelines.

Figure 2: The example of the tactical guideline.

The contexts tree representing the tactical guideline may have "leave" contexts which are represented in the method base as other chunks and are called *"component chunks"*. Each leave context in Figure 2 is represented in the method base as an other chunk.

Strategic guidelines provide a strategic view of the development process telling what intention can be achieved following which strategy. It is represented by a map and a set of guidelines [10]. A map is a labelled directed graph in which the nodes are the *intentions* and the edges between intentions are the *strategies*. The map permits to represent a process allowing several different ways to develop the product. As the requirements engineer progress in the map the process model is constructed dynamically.

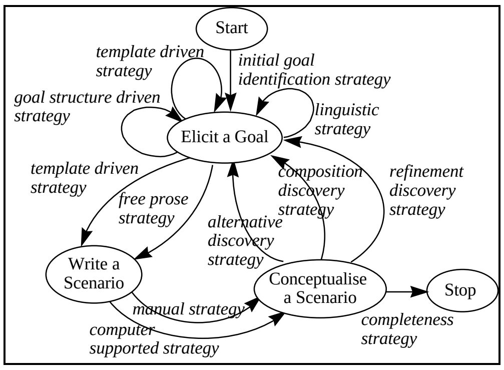


Figure 3: The example of the map.

Figure 3 depicts an example of the strategic guideline represented by a map. The interface of this guideline is <*(Problem statement), Elicit goal / scenario couples following CREWS-L'Ecritoire approach>*[2].

Three kinds of guidelines are used in the map to guide the requirements engineer in the construction of the intention driven process. The *Intention Achievement Guideline (IAG)* defines the way in which an intention can be achieved. For every triplet <source intention, target intention, strategy> in the map there exists one IAG. The

[2] The CREWS-L'Ecritoire approach is a scenario based approach developed by Paris 1 university in the CREWS project

IAG can be represented in the method base as an other chunk. Two other types of guidelines help the requirements engineer to progress in the map. The *Strategy Selection Guideline (SSG)* determines the strategies connecting two intentions and guides the selection of a strategy, whereas the *Intention Selection Guideline (ISG)* determines all succeeding intentions for a given one. Neither SSG nor ISG can not be a chunk.

As shown before, every tactical and strategic guideline can be recursively composed of other guidelines which can be represented in the method base as other chunks. Therefore, there are two types of chunks in the method base : atomic and aggregate ones. The aggregate chunks are composed of several atomic chunks. Each atomic chunk can participate in one or several aggregate chunks. Thus, the method base proposes chunks with different level of granularity.

2.2 Meta-knowledge level

The knowledge on the reuse context of the chunk is captured in the chunk descriptor. Figure 4 represents the structure of the descriptor.

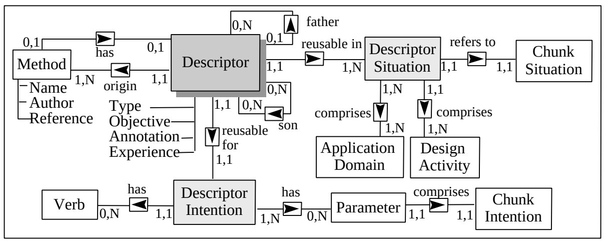


Figure 4: The structure of the descriptor.

The descriptor defines the design situation in which the chunk can be reused and the design intention which can be fulfilled by the chunk. The situation of the descriptor includes two aspects: the *application domains* in which the chunk can be applied and the *design activities* in which the scenario chunk is relevant. The intention of the chunk descriptor expresses how the scenario approach encapsulated in the chunk participates to the achievement of the design activity. *Information Systems, Business Processes, Socio-Technical Systems, Human Computer Interfaces* are the examples of the application domains in which a chunk may be applied. The design activities supported by the chunk may be *Requirements Capture, Requirements Documentation, Requirements Validation etc.*. The origin of the chunk: the name of the method in which the chunk has been identified, its author and the references to the literature, is also captured in its descriptor. The descriptor specifies the type of the chunk (i.e. atomic or aggregate). If the chunk is an aggregate, its descriptor is linked to the descriptors of its components (to its sons). If the chunk is a component of one or several aggregate chunks, its descriptor is connected with the descriptors of the corresponding aggregates (its fathers). The intention of the descriptor incorporate recursively the intention of the corresponding chunk. *"Discover system requirements by eliciting an alternative goal in a goal structure driven manner"* in which the manner *"by eliciting an alternative goal in a goal structure driven manner"* expresses the intention of the corresponding chunk.

3. Architecture of the CREWS method base

The CREWS method base is divided into two parts. One part deals with the knowledge necessary to the chunk selection and retrieval from the method base. This knowledge is captured in the chunk descriptor and represented using the SGML (Standard Generalized Markup Language) document. The SGMLQL queries deal with this part of the base and allow us to retrieve the chunk from the method base. The second part deals with the representation of the reusable knowledge to the method base user. This part describes in the HTML (Hyper Text Markup Language) documents the body of the chunk: graphical representation and informal explanation of the guidelines, links to component chunks and to the product model description, its context of reuse etc. All chunks must have the same SGML structure presented below. We also propose a template for the HTML structure, but it can be adapted for every chunk.

3.1 SGML part of the method base

The SGML [11] is an international standard language to describe a document using a set of mark ups defined in a grammar. SGML documents are structured as trees. SGML's query language, SGMLQL [12] enables a user to query the method base. Besides SGMLQL is available to query a base of SGML documents, we found the language adequate for representing the descriptors of our chunks.

The SGML part of the chunk captures the information necessary for the chunk retrieval. This information is represented in the chunk descriptor. The SGML structure of the CREWS method base is presented by the tree in Figure 5. The root is the element CREWS-BASE which represents a collection of CHUNKs. The element CHUNK is itself characterised by the attribute *kind* and the tags: INDEX, PRODUCT-MODEL, DESCRIPTOR, HTML-BODY etc. The INDEX is an identification of the chunk. The PRODUCT-MODEL contains the name of the corresponding product model. The DESCRIPTOR is composed of DESCRIPTOR-SITUATION and DESCRIPTOR-INTENTION.

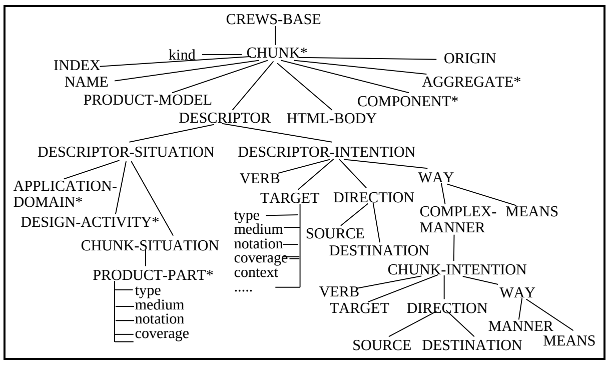


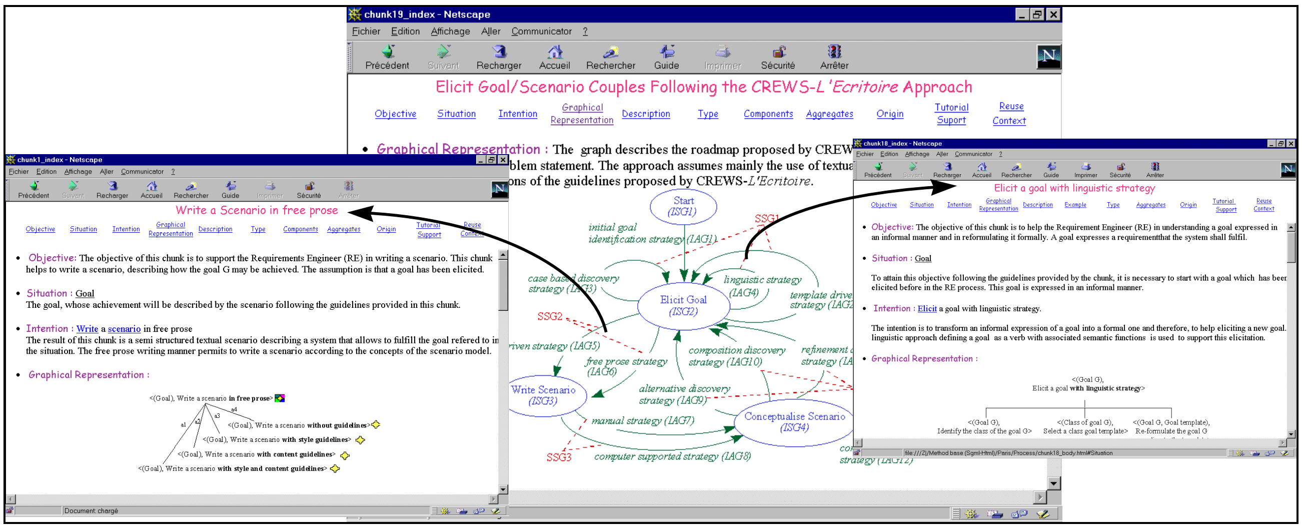
Figure 5: The structure of SGML part of the method base.

As shown in Figure 5, the DESCRIPTOR-SITUATION has three parts: APPLICATION-DOMAIN, DESIGN-ACTIVITY and CHUNK-SITUATION. Every chunk can be applied in one or several application domains and supports one or several design activities. The CHUNK-SITUATION establishes what are the required PRODUCT-PARTs allowing to apply the chunk. If the type of the required product is *"scenario based"*, this product must be characterised by providing values to the scenario classification attributes. These attributes are defined in the scenario classification framework [13]. They permits to specify what is the required scenario medium (text, table, graphic, image, etc.), notation (informal, formal, semi-formal), coverage etc. DESCRIPTOR-INTENTION is decomposed into a VERB and its parameters TARGET, DIRECTION and WAY according to the goal template [14]. The parameter TARGET is mandatory in the intention description, whereas the DIRECTION is optional. The COMPLEX-MANNER of the parameter WAY is also mandatory because it recursively describes the intention of the corresponding chunk whereas the MEANS is optional. For example, given the chunk intention *"Write scenario in free prose"*, the intention of the corresponding descriptor is *"Describe system requirements with write scenario in free prose strategy "*. The HTML-BODY contains the name of the corresponding HTML file. The COMPONENT and AGGREGATE tags capture the indexes of the component chunks and the aggregate chunks respectively. The ORIGIN permits to identify the chunk which represents the overall method in which the corresponding chunk take part.

Owing to the SGMLQL query language, the query represented in Figure 6 selects the chunks which support the discovering of the system requirements. The search is based on the descriptor intention verb *"Discover"* and target *"System requirements"*. The result of this query is a list of selected chunk names linked to the corresponding HTML documents.

```
global $myfile = file"MethodBase.sgml";
global $chunks=select "<LI><A HREF=".text($hb).">". text($n-
>NAME)."<A></LI>"
from $c in every CHUNK within $myfile,
          $di in every DESCRIPTOR-INTENTION within $c,
          $v in first VERB within $d,
          $t in first TARGET within $d,
          $hb in HTML-BODY within $c
where text($v) match "Discover" and  text($t) match  "System
(functional, intentional, non-functional) etc. The requirements";
```

Figure 6: The example of the query.


chunk19_index - Netscape
Fichier Edition Affichage Aller Communicator ?
Précédent Suivant Recharger Accueil Rechercher Guide Imprimer Sécurité Arrêter
Objective Situation Intention Graphical Representation Description Type Components Aggregates Origin Tutorial Suport Reuse Context
Start (ISG1)
initial goal identification strategy (IAG1)
SSG1
case based discovery strategy (IAG3)
linguistic strategy (IAG4)
template driven strategy (IAG2)
Elicit Goal (ISG2)
SSG2
driven strategy (IAG5)
free prose strategy (IAG6)
composition discovery strategy (IAG10)
refinement discovery strategy (IAG11)
Write Scenario (ISG3)
alternative discovery strategy (IAG9)
manual strategy (IAG7)
Conceptualise Scenario (ISG4)
SSG3
computer supported strategy (IAG8)
strategy (IAG12)
chunk1_index - Netscape
Objective Situation Intention Graphical Representation Description Type Components Aggregates Origin Tutorial Support Reuse Context
a1
a2
a3
a4
Document: chargé
chunk18_index - Netscape
Objective Situation Intention Graphical Representation Description Example Type Aggregates Origin Tutorial Support Reuse Context
file:///Z|/Method base (Sgml-Html)/Paris/Process/chunk18_body.html#Situation

Elicit Goal/Scenario Couples Following the CREWS-*L'Ecritoire* Approach

- Graphical Representation : The graph describes the roadmap proposed by CREWS… …blem statement. The approach assumes mainly the use of textua… …ons of the guidelines proposed by CREWS-*L'Ecritoire*.

Write a Scenario in free prose

- Objective: The objective of this chunk is to support the Requirements Engineer (RE) in writing a scenario. This chunk helps to write a scenario, describing how the goal G may be achieved. The assumption is that a goal has been elicited.
- Situation : Goal
  The goal, whose achievement will be described by the scenario following the guidelines provided in this chunk.
- Intention : Write a scenario in free prose
  The result of this chunk is a semi structured textual scenario describing a system that allows to fulfill the goal refered to in the situation. The free prose writing manner permits to write a scenario according to the concepts of the scenario model.
- Graphical Representation :

<(Goal), Write a scenario **in free prose**>
<(Goal), Write a scenario **without guidelines**>
<(Goal), Write a scenario **with style guidelines**>
<(Goal), Write a scenario **with content guidelines**>
<(Goal), Write a scenario **with style and content guidelines**>

Elicit a goal with linguistic strategy

- Objective: The objective of this chunk is to help the Requirement Engineer (RE) in understanding a goal expressed in an informal manner and in reformulating it formally. A goal expresses a requirementthat the system shall fulfil.
- Situation : Goal
  To attain this objective following the guidelines provided by the chunk, it is necessary to start with a goal which has been elicited before in the RE process. This goal is expressed in an informal manner.
- Intention : Elicit a goal with linguistic strategy.
  The intention is to transform an informal expression of a goal into a formal one and therefore, to help eliciting a new goal. linguistic approach defining a goal as a verb with associated semantic functions is used to support this elicitation.
- Graphical Representation :

<(Goal G), Elicit a goal **with linguistic strategy**>
<(Goal G), Identify the class of the goal G>
<(Class of goal G), Select a class goal template>
<(Goal G, Goal template), Re-formulate the goal G

Figure 7: The example of HTML pages representing the chunks from CREWS method base.

3.2 HTML part of the chunk

The HTML part of a chunk represents its body and the context of its applicability. Every chunk is represented by an HTML document having the following structure: on the top of the page we find links corresponding to the objective, situation, intention, graphical representation etc. of the chunk at hand. Clicking on the link "intention" for example allows us to visualise the section specifying what is the intention of the chunk. Because the intention of the chunk has a predefined structure, each of its component may be defined as a link to other HTML documents as glossary providing definition of the intention verb or the document including the details about the target product etc. If the chunk is an aggregate we can directly access the HTML pages of its components by clicking on its graphical representation. Similarly, if the chunk is a component of an aggregate we can visualise the HTML document of the corresponding chunk by clicking on the link to this document. Figure 7 shows the HTML representations of three chunks. The chunk represented in the top window is an aggregate. We can brows the content of this chunk by clicking on the links in the top of this HTML document or display the HTML pages of its components (represented in the two other windows) by clicking on the map elements or by clicking on the component names listed in the section *"Components"*.

4. Conclusion

This paper have proposed an approach for supporting the reuse of scenario based chunks made available in the CREWS method base. The proposed approach advocates a modular representation of scenario chunks and an intentional description of their reuse context. The former results cohesive chunks which are applicable in specific situations for specific purposes whereas the later provides contextual information identifying in which specific design situations for which specific design intentions the chunks are reusable. The paper also reports on the implementation of a scenario method base in SGML and HTML.

Future work shall concentrate on developing guidelines to integrate scenario method chunks in existing methods. Besides, in order to support the process for retrieving chunks matching specific requirements we are developing a set of SgmlQL macro-queries. At the moment, the CREWS method base contains only the chunks defined by CREWS project partners. In the future, we shall add in our base the chunks coming from different scenario based methods.

5. References

1. P. Haumer, K. Pohl, K. Weidenhaupt, *Requirements Elicitation and Validation with real world scenes.* IEEE Transactions on Software Engineering, Vol. 24, N°. 12, Special Issue on Scenario Management, December. 1998.
2. C. Rolland, C. Souveyet, C. Ben Achour, *Guiding Goal Modelling Using Scenarios*. IEEE Transactions on Software Engineering, special issue on Scenario Management, 1998.
3. A. G. Sutcliffe, *Scenario-based Requirements Analysis*. Requirements Engineering Journal, Vol 3 N° 1, (ed. P. Loucopoulos, C. Potts), Springer Verlag. 1998.
4. E. Dubois, P. Heymans, *Scenario-Based Techniques for supporting the Elaboration and the Validation of Formal Requirements*, Submitted to RE Journal, 1998.
5. C. Rolland, N. Prakash, *A proposal for Context-Specific Method Engineering*, IFIP TC8 Working Conference on Method Engineering, Atlanta, Gerorgie, USA, 1996.
6. F. Harmsen, S. Brinkkemper, H. Oei, *Situational Method Engineering for Information System Projects*. In Olle T. W. and A. A. Verrijn Stuart (Eds.), Mathods and Associated Tools for the Information Systems Life Cycle, Proceedings of the IFIP WG8.1 Working Conference CRIS'94, pp. 169-194, North-Holland, Amsterdam, 1994.
7. S. Brinkkemper, M. Saeki, F. Harmsen, *Assembly Techniques for Method Engineering*. Proceedings of the 10th Conference on Advanced Information Systems Engineering, CAiSE'98. Pisa Italy, 8-12 June, 1998.
8. V. Plihon, J. Ralyté, A. Benjamen, N.A.M. Maiden, A. Sutcliffe, E. Dubois, P. Heymans, *A reuse-oriented approach for the construction of scenario based methods*. Proceedings of the International Software Process Association's 5th International Conference on Software Process (ICSP'98), Chicago, Illinois, USA, 14-17 June 1998.
9. G. Grosz, C. Rolland, S. Schwer, C. Souveyet, V. Plihon, S. Si-Said, C. Ben Achour, C. Gnaho, *Modelling and Engineering the Requirements Engineering Process : an erview of the NATURE approach*. Requirements Engineering Journal 2, pp. 115-131, 1997.
10. C. Rolland, N. Prakash, A. Benjamen, *A multi-model view of process modelling*. To appear in the RE journal, 1999.
11. C. F. Goldfarb, *« The SGML Handbook »*, Oxford Clarendon Press, 1990.
12. J. Lemaitre, E. Murisasco, M. Rolbert, SgmlQL, *"Un langage d'interrogation de documents SGML"*, Proc. of the 11th Conference on Advanced Data Bases, Nancy, France, 1995.
13. C. Rolland, C. Ben Achour, C. Cauvet, J. Ralyté, A. Sutcliffe, N.A.M. Maiden, M. Jarke, P. Haumer, K. Pohl, Dubois, P. Heymans, *A proposal for a scenario classification framework*. Requirements Engineering Journal Vol 3, No 1, Springer Verlag, pp.23-47, 1998.
14. N. Prat, *Goal formalisation and classification for requirements engineering*. Proceedings of the Third International Workshop on Requirements Engineering: Foundations of Software Quality REFSQ'97, Barcelona, June 1997.