CREWS Report Series 98 - 26

WRITING AND CORRECTING TEXTUAL SCENARIOS FOR SYSTEM DESIGN

Camille Ben Achour

CRI - Université Paris 1 - Sorbonne
Université Paris 6 - Pierre et Marie Curie
90, rue de Tolbiac. 75013 Paris - France

camille@univ-paris1.fr

Appeared in the proceedings of the Natural Language and Information Systems (NLIS'98) Workshop 28th August 1998, Vienna, Austria, 1998.

Writing and Correcting Textual Scenarios for System Design[1]

Camille Ben Achour

CRI - Université Paris 1 - Sorbonne
Université Paris 6 - Pierre et Marie Curie
90, rue de Tolbiac. 75013 Paris - France

camille@univ-paris1.fr

Abstract

Since a few years, scenarios have gained in popularity in Requirements Engineering. Textual scenarios are narrative descriptions of flows of actions between agents. They are often proposed to elicit, validate or document requirements. The CREWS experience has shown that the advantage of scenarios is their easiness of use, and that their disadvantage stands in the lack of guidelines for 'quality' authoring. In this article, we propose guidance for the authoring of scenarios. The guided scenario authoring process is divided into two main stages : the writing of scenarios, and the correcting of scenarios. To guide the writing of scenarios, we provide style and contents guidelines referring to a conceptual and a linguistic model of scenarios. Our assumption is that scenarios written in conformance to these guidelines can be semi-automatically analysed. Else, to guide the correcting of scenarios, we propose a set of enactable rules. These rules aim at the clarification, completion and conceptualisation of scenarios, and help the scenario author to improve his scenarios until acceptable quality in the terms of the former scenario models.

1. Introduction

Scenarios have recently gained attention in the field of Requirements Engineering (RE). Scenarios may have different forms, contents, coverage, and purposes. However, our investigations show that in the literature, most of the approaches lie on textual scenarios [6], and that in the practice, in about 80% of scenario-based projects, scenarios are described under the textual form [4]. The growing number of practitioners demanding for more 'informality' in the requirements engineering process seems to confirm this trend [1]. However, as show our enquiries at several major European companies, requirement engineers still need help in the process of authoring scenarios [4].

In this paper, we are interested in the guidance of the so called 'scenario authoring' process [8]. By scenario authoring, we mean to write, clarify, complete and conceptualise scenarios. In addition, we focus on *textual scenarios*, that is to say descriptions of stories of system usage in (more or less structured) natural language. Authoring is part of a two-step process of <scenario authoring - goal discovery>. On the one side, scenarios are used to discover new goals, and on the other side, goals are the input for scenario authoring. In order to guide the scenario author (SA), we provide writing guidelines and verification rules. The writing guidelines aim at conducting the SA towards a quality writing of textual scenarios. The quality of textual scenarios refers to a conceptual and a linguistic model of scenarios. While he / she writes a scenario, the SA can evaluate its quality by applying enactable rules. These rules guide the SA in structuring, clarifying and completing scenarios.

The next section describes the conceptual model of scenarios referred to along this paper. Section 3 tackles the linguistic aspect of textual scenarios. Section 4 overviews the process of scenario authoring. and provides respectively the guidelines for writing textual scenarios and the rules for correcting them. The last section discusses the results presented in this paper.

2. The conceptual definition of scenarios

A scenario expresses an example of behaviour of a system. Thus, a scenario is *« one possible behaviour limited to a set of purposeful interactions taking place among several agents »* [7]. The CREWS approach defends a tight coupling of goals and scenarios into *requirement chunks* [7, 8]. To any scenario is attached a *goal*. Goals express in part the requirements for the designed system. They

[1] The work presented in this paper is funded by the European Community within the framework of the ESPRIT LTR project CREWS (*Cooperative Requirements Engineering With Scenarios*) n°21903.

are written as a verb together with one or several parameters, and they are illustrated by scenarios. In the following, we call a scenario by the name of the goal it illustrates. The figure 1 presents the model of scenarios, and figure 2 provides two examples of scenarios.

Scenario
  Composed of : Action
Flow of Action : IsA Action
  Composed of : Action
Sequence : IsA Flow of Action
Constraint : IsA Flow of Action
Concurrency : IsA Flow of Action
  Based on : Flow condition
Repetition : IsA Flow of Action
  Based on : Flow condition
Atomic Action : IsA Action
  Name
  From : Agent
  To : Agent
  Parameter : Object
Agent : IsA Object
Resource : IsA Object

Figure 1 : The conceptual model of scenarios

The scenario model defines a scenario as composed of one or several *actions*. Actions are of two types : atomic actions and flows of actions.

An a*tomic action* is going '*from'* one and only one agent '*to'* one and only one agent, and affects some '*parameter*' '*object*'. The agents may be the designed system, its users, its component objects, etc, and the resources the passive objects that they manipulate. For example *'The CAD system sends the mobilisation order to the relevant ambulance crew'* is an atomic action. Its agents are 'the system' and 'the relevant ambulance crew', and its parameter is the resource 'the mobilisation order'. Agents and objects may be involved into several atomic actions.

The composition of actions is expressed in *flows of actions* which are refined into *sequence*, *concurrency*, *iteration* and *constraint*, that refer to the connectors of strict sequence, co-beginning parallelism, loop, and condition. For example, the sentence *'The CAD system locates the incident place, and then decides which division is going to deal with the incident'* expresses a sequence between two atomic actions. Contrarily to the concurrency and sequence flows of actions, the iteration and constraints are 'based on' flow conditions. *Flow conditions* express the assumptions for which a meaningful behaviour of the scenario agents is defined. In the sentence *'If the call is not in duplicate, then the CAD system locates the incident place'*, the flow condition '*If the call is not in duplicate'* identifies a case of CAD usage. The cases of usage of the CAD system with a call already received for the same incident should thus be described in a separate scenario. As defined in Figure 1, flows of actions can be themselves composed of other flows of actions. Thus, a sequence of actions can be iterated, concurrent actions constrained, several constraints embedded, etc.

Provide the ambulance service to London patients	A caller sends an urgent call to the central ambulance control. Then, the central ambulance control allocates the incident to an ambulance crew, and the ambulance crew delivers the medical service to the patient.

Allocate incidents to an ambulance crew in a normal way	1. The details of the call are entered in the CAD system.
	2. If the call is not in duplicate, then
	  3. The CAD system locates the incident place.
	  4. It decides which division is going to deal with the incident.
	  5. The resource status and incident details are displayed to the division resource allocator.
	  6. If an ambulance and a crew are available at the division, then
	    7. The resource allocator identifies the resource to be mobilised.
	    8. He must enter the mobilisation decision within two minutes.
	    9. The system sends rapidly the mobilisation order to the relevant ambulance crew.

Figure 2 : Unstructured and semi-structured textual representations of scenarios.

3. The linguistic definition of scenarios

A scenario should be written correctly in reference to the scenario model. Thus, the scenario model provides the structure of scenarios. Scenarios are themselves written in natural language. Consequently, there is a relationship between the text structure and the scenario model structure which is highlighted in figure 3.

Flow of Action : IsA Action
Composed of : Action
Expressed by : Sent. St.
Atomic Action : IsA Action
Expressed by : Clause St.
Sentence Structure : IsA Structure
Composed of : Structure
Has for Semantics : Sent. Sem. P.
Clause Structure : IsA Structure
Has for Semantics : Cla. Sem. P.
Sentence Semantic Pattern : IsA Semantic Pattern
Composed of : Semantic Pattern
Clause Semantic Pattern : IsA Semantic Pattern

Figure 3 : Relationships between text structure and the action structure.

As scenario actions are refined into atomic actions and flows of actions, the text *structures* can be decomposed into more elementary structures which are either *clause structures* or *sentence structures*. For example the scenario extract *'The central ambulance control allocates the incident to an ambulance crew, then the ambulance crew delivers the medical service to the patient'* has a sentence structure which can be decomposed into two elementary clause structures. Structures correspond to the surface representation of the textual specification, also referred to as the syntax. The semantics is provided by *semantic patterns*. *Sentence* and *clause semantic patterns* reflect respectively the deep aspect of sentence and clause structures. Thus, clause semantic patterns provide the semantics of atomic actions and sentence semantic patterns provide the semantics of flows of actions : the sequence, the concurrency, the repetition and the constraint.

Our approach of scenario authoring uses natural language. As presented in section 2, the scenario model focuses on the notion of action. However, in textual scenarios, actions are expressed informally. Thus, there is a need to fill the gap between the textual representation of actions and their formal counterpart in the scenario model.

As shown in [9], the use of free natural language in RE raises difficult problems. For example ambiguous and implicit statements requiring contextual or domain knowledge are especially dangerous in the framework of requirements engineering where the least misunderstanding may have dreadful consequences. In order to avoid such problems, we restrict the language of scenario expression to : a restricted set of terms, the syntax defined by a finite set of structures, the semantics defined by the scenario model, and the statements that do not require contextual or domain knowledge to be interpreted. We use patterns of a Case Grammar [3] to fill the gap between the informal and the formal representation of scenarios. We justify this choice by the fact that it is consistent with these assumptions, and that it focuses on the notion of action, as shows our ontology of semantic patterns presented in [8].

Each structure has a semantics defined in a semantic pattern. The semantics of sentence structures is in sentence semantic patterns, and the semantics of clause is in clause semantics patterns. The other way round, one semantic pattern can be expressed following several different surface level structures.

In addition, each concept of the scenario model has a semantic identified in a semantic pattern. For example atomic actions correspond to the communication and action clause semantic patterns. The patterns of sequence, constraint, concurrency and repetition identify the flows of actions of the scenario model ; that is to say respectively the sequence, constraint, concurrency, and iteration. The semantic patterns being expressed using pre-defined surface structures, each concept of the scenario model can be expressed in a restricted natural language. As an example, the following piece of scenario specification :

Atomic action :
Name : 'enter'
From Agent : 'the control assistant'
To Agent : 'the CAD system'
Parameter : 'the details of the call'

corresponds to the pattern instance :

Communication ('enter') [Agent : 'the control assistant' ; Object : 'the details of the call' ; Source : 'the control assistant' ; Destination : 'the CAD system']

This communication action clause semantic pattern can be expressed using the structure :

[['The details of the call'](Subject)$_{Obj}$ *['are entered'](Main Verb)*$_{Communication}$ *['by the control assistant'](Complement)*$_{Ag+So}$ *['in the CAD system'](Complement)*$_{Dest}$*](VG passive)*$_{Communication}$

which corresponds to the clause :

'The details of the call are entered by the control assistant in the CAD system'.

Identifying the concepts of the scenario model from natural language is a two stage process which requires first the semantic analysis of the text and then the mapping of the resulting semantic patterns onto the concepts of the scenario model. However, the conceptualisation of scenarios is possible only if the scenarios are well written. These two aspects are dealt with in the following sections.

4. The scenario authoring process

In the CREWS goal-scenario approach, the activity of scenario authoring is complementary to the goal discovery. Both can be performed iteratively, until full completeness, validity and agreement. The goal discovery is described in [8].

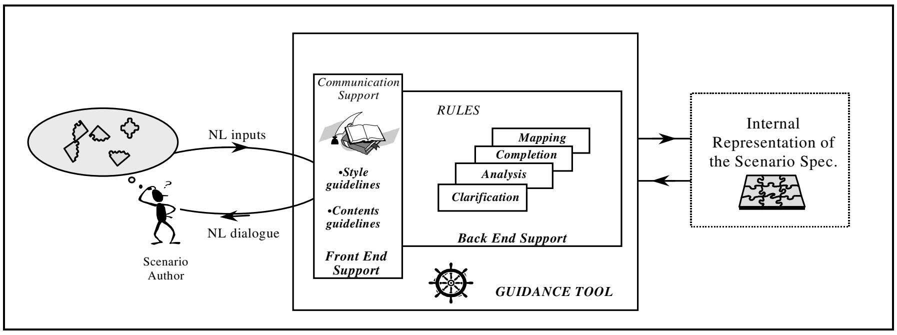

Figure 4 : Overall architecture of the scenario authoring process.

The scenario authoring is in two stages : the writing of the scenarios, and the correcting of scenarios. Once written and corrected, scenarios can be used for discovering new goals or for negotiating the requirements concerning the designed system.

To guide the Scenario Author (SA) in the writing of scenarios, we propose *writing guidelines* (Figure 4). When he / she wants to write a scenario, the SA has already defined a goal that the scenario will illustrate. Thus, the guidelines drive the SA in illustrating a given goal by a scenario. The guidelines consist in a list of advice on how to write a scenario and what to put in the scenario. We expect that the quality of the scenarios produced improves when the guidelines are correctly applied. However, they are mandatory : they can be followed or not. The writing guidelines are of two complementary types : style and contents guidelines. *Style guidelines* provide recommendations on the style of the expected prose, in conformance with the expected text structure presented in section 3. *Contents guidelines* advise the SA on the expected contents of his / her prose. Contents guidelines are adapted to the scenario model of the section 2 and to the expected text semantics presented in section 3. Both guidelines are illustrated with the example presented in figure 5.

At the moment of writing a scenario, the scenario author can use the help provided by the guidelines. The scenarios produced following the guidelines should be of acceptable quality ; in other word, their syntax and their semantics should correspond to the ones expected through the scenario model, the structures, and the semantic patterns. If so, the textual scenario can be conceptualised, to support automated reasoning on the system requirements. However, it is necessary to verify that the scenarios are sufficiently well-written to be conceptualised, and to correct them if they are not. The *correcting rules* aim at identifying conflicts between a scenario, and a guideline. They are described as sequences of steps aiming at :

- identifying sus pect situations,
- proposing solu tions to modify the scenarios,
- acquiring complementary info- rmation about the scenario contents, and

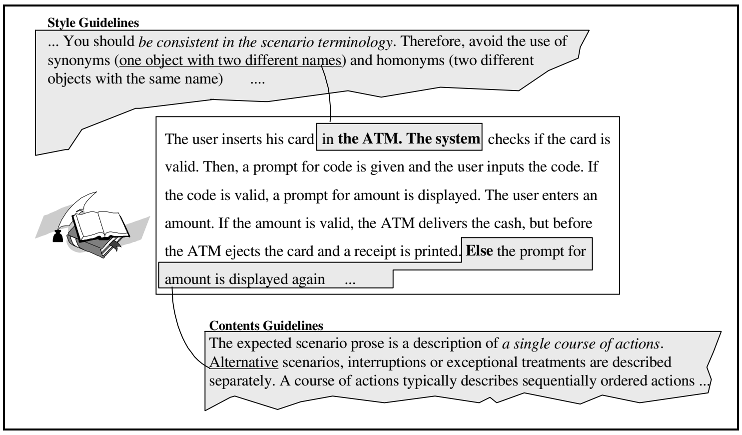

Figure 5 : Example of application of writing guidelines and correction rules

- improving the scenarios.

The correcting of scenarios involves : conceptualisation, clarification and completion. The *conceptualisation* of scenarios is performed according to the relationships between the scenario structure, its semantics, and the model of scenarios, respectively presented in the sections 3 and 2. Conceptualisation is (at least in part) necessary to enact other correction rules. It involves structure analysis, semantics analysis, and model instantiation. The *clarification* rules help the SA identifying and correcting ambiguous actions. The *completion* rules guide the SA in the search of missing elements in the scenario, for example atomic actions with unprecise parameters, etc.

Let's take the example of a scenario author who has written the scenario partly shown in figure 5. As underline in the upper grey box, this scenario does not respect the guideline stating that *'the scenario terminology should be consistent'*. Indeed, the 'ATM' is also referred to as the 'system'. Conceptualising such a scenario would lead to erroneous result. Having identified this problem, the correcting rule proposes to the scenario author to rephrase various references to the 'ATM' in the scenario. In addition, the scenario does not respect the definition of scenarios expressed in the contents guidelines by *'The expected scenario prose is a single course of actions ...'*. Indeed the scenario addresses two different cases as expressed by the 'Else' statement in the last sentence. Once raised, this issue is solved by the writing an additional scenario considering the alternative case separately. For the sake of place, the rules and guidelines can not be extensively described here. However, more complete information can be found in [2], and [7].

5. Conclusion

Very few approaches say how to author textual scenarios. Based on a formal conceptual model of scenarios, this paper aims at guiding the process of scenario authoring. This guidance emphasises two stages : writing and correcting.

To help the scenario author *writing* 'quality' textual scenarios, we propose a set of guidelines. Besides, we propose a set of *rules* to guide the *clarification*, *completion* and *conceptualisation* of scenarios. As for the guidelines, the linguistic foundation for these rules is the Case Grammar which is tailored for the scenario conceptual model. The grammar defines linguistic structures and semantic patterns lying on the concepts of the scenario model. The former identify the numerous ways to express atomic actions, flows of actions, conditions in natural language. The latter expresses the semantics of the scenario contents. The grammar permits us to go beyond the simple solution of sentence templates by providing tools to support the writing and correcting of scenarios.

We are currently evaluating these guidelines by empirical evaluations with students and engineers, in collaboration with the City University (UK). In addition, examples of application of the guidelines and of the rules have been already explored with the ATM case study [7], and with a real case study borrowed to an Electricity Company [5]. Future works will concern the evaluation of the linguistic aspects of our approach and the extension of the Visual Basic - PROLOG implementation to support the entire set of rules. In particular, we shall search for evidence of the scalability of the approach through the exploration of numerous and extensive case studies.

6. References

[1] A. Cockburn, *Structuring Use Cases with Goals*. Technical report. Human and Technology, 7691 Dell Rd, Salt Lake City, UT 84121, HaT.TR.95.1, http://members.aol.com/acocburn/papers/usecases.htm, 1995.

[2] C. Ben Achour, *Guiding Scenario Authoring*. Proceedings of the 8th European Japanese Conference on Information Modelling and Knowledge Bases, pp. 181-200, Ellivuori, Finland, May 26-29, 1998.

[3] C. Fillmore, *The case for case*. In E.Bach, R. Harms (eds.) 'Universals in linguistic theory', Holt, Rinehart and Winston Publishing Company, pp. 1-90, 1968.

[4] M. Jarke, K. Pohl, P. Haumer, K. Weidenhaupt, E. Dubois, P. Heymans, C. Rolland, C. Ben Achour, C. Cauvet, J. Ralyte, A. Sutcliffe, N.A.M. Maiden and S. Minocha, *Scenario Use in European Software Organisations - Results from Site Visits and Questionnaires*. CREWS deliverable W1: Industrial problem capture Working Group, 1997.

[5] S. Nurcan, G. Grosz, C. Souveyet, *Describing Business Processes with a Guided Use Case Approach.* Proceeding of the CAiSE'98 Conference on Advanced Information Systems Engineering, to appear in June 1998.

[6] 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, 3 :1, 1998.

[7]. C. Rolland, C. Ben Achour, *Guiding the Construction of Textual Use Case Specifications*. Data and Knowledge Engineering Journal, Vol 25 N° 1,1998.

[8] C. Rolland, C. Souveyet, C. Ben Achour. *Guiding Goal Modelling Using Scenarios.* Submitted to the TSE Journal, special issue on scenarios, 1998.

[9] K. Ryan, *The Role of Natural Language in Requirements Engineering*. Proceedings of the IEEE Int. Symposium on RE, San Diego California, pp. 80-82, 1993.