CREWS Report Series 98 - 4

DESCRIBING BUSINESS PROCESSES WITH A GUIDED USE CASE APPROACH

Selmin Nurcan, Georges Grosz, Carine Souveyet

CRI, Université de Paris I Panthéon-Sorbonne, 90 rue de Tolbiac, 75013 Paris, France

{nurcan, grosz, souveyet}@univ-paris1.fr

Appeared in the proceedings of the 10th International Conference CAiSE'98, B. Lecture Notes in Computer Science 1413, Pernici, C. Thanos (Eds), Springer. Pisa, Italy, June 1998

Describing Business Processes with a Guided Use Case Approach

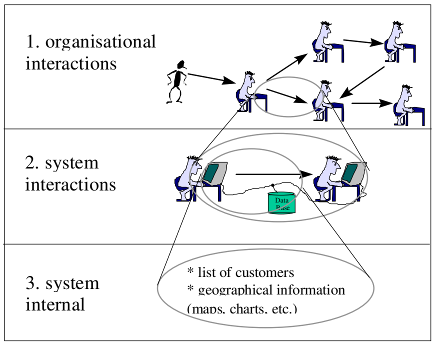
Selmin Nurcan, Georges Grosz, Carine Souveyet
CRI, Université de Paris I Panthéon-Sorbonne, 90 rue de Tolbiac, 75013 Paris, France
email : {nurcan, grosz, souveyet}@univ-paris1.fr
Phone : +33 (0) 1 40 77 46 34, Fax : +33 (0) 1 40 77 19 54

Abstract *: Business Process (BP) improvement and alike require accurate descriptions of the BPs. We suggest to describe BPs as use case specifications. A use case specification comprises a description of the context of the BP, the interactions between the agents involved in the BP, the interactions of these agents with an automated system supporting the BP and attached system internal requirements. Constructing such specifications remains a difficult task. Our proposal is to use textual scenarios as inputs, describing fragments of the BP, and to guide, using a set of rules, their incremental production and integration in a use case specification also presented in a textual form. The paper presents the structure of a use case, the linguistic approach adopted for textual scenarios analysis and the guided process for constructing use case specifications from scenarios along with the guidelines and support rules grounding the process. The process is illustrated with a real case study borrowed to an Electricity Company.*
Keywords *: Business Process Description, Use Case Specification, Textual Scenario Analysis.*

1 Introduction

A Business Process (BP) is defined by Hammer and Champy in [8] as a set of activities which produces (from one or several inputs) an output valuable for the customer. For the sake of improving or re-engineering or simply understanding BPs, Hammer and Champy consider essential to start to describe them as accurately as possible. A BP can be described at different levels, each level corresponding to different types of BP requirements. First, a BP can be described with a set of interactions between agents involved in the BP, we call these interactions «organizational interactions». Agents can be either internal or external (e.g. customer, supplier) to the organisation where the BP takes place. Such a description can be completed by describing how an Information System (IS) supports, or shall support if the IS does not exist, the BP through the description of what we call « system interactions ». In such a description, the IS is considered as an agent. The BP description can be further refined and completed by adding the requirements of the « system internal ». These levels of description are summarised in figure 1.

Similarly to Jacobson [9], we believe essential that *"A tight, seamless relationship is required between the process that develops the business model and the process that develops the information system"*. Therefore, we consider that the development of the three levels of description must be considered seamlessly. Finally, the modelling technique to be used for describing BPs shall be forceful in that it should be possible both to show easy-to-understand surveys of the BP and to describe parts of the BP in details [10], this complies with the different levels of description we propose.



* The work presented in this paper is partly supported by the European Community within the framework of the ESPRIT LTR project CREWS (*Co-operative Requirements Engineering With Scenarios*) n°21903.

Fig. 1. Three levels for describing business processes

On the one hand, we propose to describe BPs as use cases. As mentioned by Jacobson [10], use cases are a simple, natural way to identify and describe BP. Indeed, use case models are interaction oriented models that focus on the communications between the agents of an organisation. They are therefore very well adapted to the two first levels of description presented in figure 1. Complementary, use case driven approaches [2, 3, 9, 10, 19] have proved useful for requirements elicitation and validation. A use case is a description of one or more end to end transactions involving the required system and its environment [17]. The basic idea is to specify use cases that cover all possible pathways through the system functions [3, 19].

On the other hand, it is not sensible to imagine that a description of a BP can be obtained in one shot because BP are complex and involve many agents. Therefore, our proposal advocates for an incremental process for the description of BP. The incremental process guides, using a set of rules, the description of fragments of the BP and their integration into a single use case specification that includes all levels presented in figure 1. Therefore, a use case specification describes the context of the BP, the structured description of all interactions between agents involved in the BP (including the IS supporting the BP) and requirements about the IS.

Finally, we propose to use scenarios as a means for describing the fragments of BPs. In both the Software Engineering (SE) and the Human Computer Interaction (HCI) communities scenarios are widely used as 'engine of design' [1, 15]. In the HCI community, they are used to elaborate on usability issues [7, 16, 26] and to help the expression and understanding of the goals of the design [12, 14, 22]. In the SE community, scenarios serve mainly as a front end to object oriented design [3, 20, 23, 24, 27]. Scenarios can be expressed in various ways : text, animation, prototypes, etc. but textual ones are recommended by several authors [5, 10, 13, 16, 18]. Natural language provides a simple means for describing a problem [12]. However, the lack of guidelines to support the process of constructing use cases is certainly one of the major drawbacks of use case driven approaches for requirements engineering [25].

In this paper, we propose an *approach to guide textual use case authoring for the description of business processes*. Our approach is an extension of the work described in [21] where the use case specification process was limited to the study of system interactions. We enlarge the scope of a use case for the description of the interactions between the agents of an organisation participating to a BP and therefore to describe what we call a "rich" use case. The input of the guided process is a set of textual scenarios describing the BP. The output is a use case specification of the BP, including organisational interactions, system interactions and system internal requirements expressed in a structured and non-ambiguous natural language text. Indeed, part of our approach relies on natural language analysis. Between the two extreme of using too constraining clauses templates (e.g. [2]) and completely free mode of expression (that increases the risks of ambiguity, inconsistency and incompleteness and makes automatic interpretation difficult), we chose a middle one. We propose to combine the use of narrative prose to express scenarios with structured natural language for the construction of "rich" use case specifications.

The remaining of the paper is organised as follows. The use case model is briefly presented in section 2 along with its semantics, its associated linguistic patterns structures and an overview of the guided process. In section 3, we illustrate the process and the use of the rules with a real case study of a business process called "*Electricity Application Fulfilment"* (EAF) borrowed to an Electricity Company (EC). Finally we draw some conclusions and identify future work in section 4.

2 Overview of the Approach

Central to our approach are the structure of a use case, the linguistic approach and the rules which ground the process of use case authoring. These three elements are described in turn. More details and examples about this approach can be found in [21].

2.1 The Structure of a Use Case

Because of the complexity of the use case structure, our approach proposes to construct use case specification incrementally taking into account partial stories called scenarios. These scenarios are the inputs provided by the use case author according to the structure depicted in figure 2.

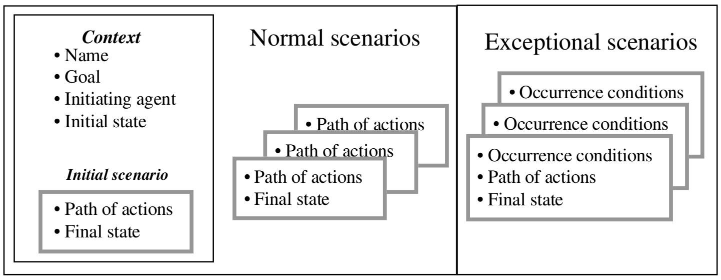


Fig. 2. The structure of the textual inputs of the guided process

In this description, there is first a contextual information which role is to situate the business process in its organisational setting (what Jacobson calls « company's environment » in [10]. The context comprises the *name* of the BP, the description of the *initiating agent* (of the BP) who has a *goal* in mind. For instance, in the case of the EAF process, the initiating agent is the customer of the EC whose goal is to be connected to the EC network. There are also some conditions required for the process to begin. For instance, the interactions of the EAF process does not start if the following condition does not hold : *"the customer is in the EC office with a completed application form"*. Such conditions describe the *initial state* of the use case and are expressed in terms of agent or resource states. A *resource* can be physical (e.g. an identification paper) or abstract (e.g. a customer record).

After having provided the contextual information, the use case author provides all possible scenarios. We distinguish two types of scenarios: normal and exceptional. A normal scenario describes one (possibly conditional) course (path) of actions reaching a final state that fulfil the goal of the initiating agent. An exceptional scenario is also described as a course of actions. However, the final state of an exceptional scenario does not fulfil the goal of the initiating agent. Actions can be either « system interactions » if one of the involved agent is the system supporting the BP or « organisational interactions » if the system is not involved.

An action may be an atomic action or a flow of actions. An atomic action materialises an interaction from an agent to another agent and requires some resources. The sentence *"the customer requests the commercial employee for a connection to the network"* is an example of atomic action from the *customer* agent to the *commercial employee* agent. The parameter of this action is a resource (*"a connection to the network*"). We consider two types of atomic actions: *communication actions* between two different agents and *internal actions* involving a single agent. The previous example of action is an illustration of a communication action whereas the sentence *"a technical employee performs the connection to the network"* is an example of internal action.

We distinguish four types of communication actions: *service request, service provision, information request* and *information provision*. In a service request action from A to B, an agent A asks a service to an agent B. Complementary, in a service provision action from A to B, an agent A provides a service to an agent B. *"The customer requests the commercial employee for a connection to the network"* is an example of service request action which is satisfied by the service provision action *"the customer is informed that the connection to the network is done"*. An information request is a communication action where an agent is asking for some information. *"The commercial employee asks the customer to sign the contract"* is an example of information request which expects the performance of the information provision action *"the customer gives to the commercial employee the signed contract"*.

All scenarios are incrementally integrated in a use case specification. The purpose of the use case is to describe how the initiating agent can interact with other agents to achieve his/her goal. The internal representation of a use case specification is a set of *episodes* (see figure 3). An *episode* comprises a *flow of actions* and the corresponding *final states*. There are several possible final states for an episode and therefore, the flow of actions can include several paths to reach each of the possible final states. A *flow of actions* is a complex action composed of other actions, it is similar to the flow of events as defined by Jacobson [10]. The composition is based on the *sequence, concurrency, iteration* and *alternative* constructors. *"The customer requests for a connection to the network, then the commercial employee asks his identification papers and the location of the house to be connected"* is an example of a flow of actions comprising a sequence of two atomic actions (*request* and *ask*). *"If the meter exists, a technical employee performs the connection to the network"* is an illustration of an *alternative flow of actions*. *Flow conditions* are necessary to integrate several courses of actions in one complex flow of actions. In the last example, the flow of actions integrates the description of what happens when the condition *"if the meter exists"* is true.
We distinguish two types of episode : *normal* and *exceptional*. An episode is said *normal* when each of its possible final states ensures the fulfilment of the user's goal else it is said *exceptional*. An *exceptional episode* describes a « non normal » course of actions reaching a final state which does not fulfil the goal of the initiating agent. In the EAF example, the normal episode corresponds to the flow of actions ending to the connection of the customer to the network. *"If customer record is written-off"* starts the description of a non normal course of actions described in an exceptional episode. An exceptional episode has an *occurrence condition* and includes a reference to an action of the normal episode in which the occurrence condition can become true. A use case specification comprises a single normal episode which is the integration of all normal scenarios, and a set of exceptional episodes that are associated to exceptional scenarios.
A use case specification may be connected to *system internal requirements* as described in the third level of figure 1. System requirements other than the ones which are formalised in a use case specification itself may emerge in the course of the specification process. *"EC must have the land register information"* or *"EC must maintain the list of written-off customers"* are examples of such system internal requirements.
As mentioned earlier, all scenarios are provided in a textual form. Furthermore, a use case specification is also expressed in textual form. Consequently, there is a relationship between the text structure and the use case structure.
The text *structure* can be decomposed into more elementary structures which are either *clause structures* or *sentence structures*. For example, the text *"the customer requests for a connection to the network, then the commercial employee asks his identification papers and the location of the house to be connected"* is a sentence structure decomposed into two elementary clause structures corresponding to the clauses : *"the customer requests for a connection to the network"*, and *"the commercial employee asks his identification papers and the location of the house to be connected"*.
Sentence and clause structures correspond to the surface structures of the textual specification. They have a meaning which is respectively provided by *sentence* and *clause patterns*. The deep structure of the use case specification is provided by the sentence and clause patterns. Sentence patterns provide the semantics of sentences expressing sequence, conditional flows, etc. Clause patterns give the semantics of actions such as service provision actions, information request actions, etc. Sentence and clause patterns which are case patterns in a *case grammar* are presented briefly in the following section.

2.2 The Linguistic Approach

The approach of use case specification presented in [21] is based on Natural Language (NL) text. There is thus, a necessity for catching the semantics of text. In order to fill the gap between the informal representation and the formal model of use cases, a Case Grammar [6] is used. It focuses

on the notion of action which is central in the use case model and permits to catch both the semantics of NL and the semantics of the use case model.
Following Fillmore's approach [6], the case grammar introduces a set of semantic cases such as agent, object, destination, etc. Semantic cases define the semantic roles that the different elements of an action clause play with respect to the main verb. Semantic patterns are defined to associate a semantic case to the different elements of clauses and sentences. The purpose of the semantic patterns is to define the semantics of the clauses and of the sentences which are respectively expressed in the atomic actions and the flows of actions of use case specifications.
At the level of clauses, case patterns are *clause semantic patterns* associated to verbs. *Action clause semantic patterns* provide the semantics of the atomic actions of the use case model by associating a semantic role to the related agents and parameter objects. *State clause semantic patterns* provide the semantics of object states on which rely the initial and final states and the flow conditions of flow of actions. Clause semantic patterns are presented in the form N (V) [C], where N is the name of the pattern qualifying the intrinsic semantics of the verb V, and C is the list of cases to be associated to the elements of the analysed clause. To represent the semantics of a clause in a use case specification consists in instantiating a clause semantic pattern.
Identifying the concepts of the use case model from natural language is a two stage process which requires first the semantic analysis of the text and second the mapping of the resulting semantic patterns onto the concepts of the use case model. These two stages are illustrated in section 5 which describes more extensively the process of constructing the use case specification of the EAF example. Details and examples about the approach we use for natural language analysis can be found in [21].

2.3 Guiding the Use Case Specification of Business Processes

The process of use case specification of business processes is a stepwise process which guides the progressive transformation of input prose texts (starting with the initial scenario) into refined and structured texts and their integration in the use case specification. It comprises four main steps to :

1. define the context of the use case,
2. describe and complete the initial scenario,
3. integrate the scenario into the use case specification, and
4. prompt the need for new scenarios and guide their description and integration.

During step 1, the use case is situated in its context by defining its name (the business process it describes), the initiating agent, the goal of the initiating agent and the initial state.
Step 2 is an iterative one. It starts with the capture of the initial scenario. It is a text describing a course of actions that can be incomplete and ambiguous. It proceeds with the check and possible interactive completion of the initial scenario. The result of step 2 is a complete description of a pathway in an episode expressed unambiguously according to our semantic patterns. It corresponds to one path in the graph of episodes of the use case.
During step 3, the completed scenario is integrated into the episode structure of the use case. Positioning the scenario in the use case specification is inferred from the study of the flow conditions.
Performing step 2 and 3 may prompt the need for new scenarios (step 4) leading to iterate sequences of steps 2, 3 and 4. At this stage, a shift in the levels presented in section 1 (see figure 1) can be performed. For instance, as it will be shown in the next section while presenting the example, it is possible to shift from the description of "organisational interactions" to the description of "system interactions". The same applies to "system internal" requirements. The shift could also be the other way around (i.e. from a "system interactions" to "organisational interactions").
Guidelines and *rules* are defined to support the performance of each step. Guidelines help the author to perform the requested action. Rules define how to map textual scenarios onto the internal use case structure and to reason about it. The overall architecture of the process is sketched in figure

3. The front end is a guided communication with the user. The back end is based on rules working on the use case structure.

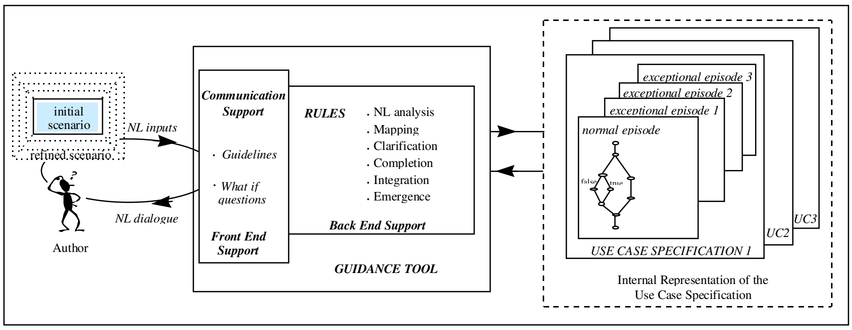


Fig. 3. Overall architecture of the process.

Guidelines have the form of plain text which can be prompted to the use case author on demand while writing down a scenario. They provide recommendation on the style of writing narrative prose. They suggest, for example, to avoid the use of anaphoric references, of synonyms and homonyms, ways for expressing an action, a conditional action, etc. They also advise the author on the expected contents of his/her prose. For instance, a guideline states that "*The expected scenario prose is a description of a single course of actions. Alternative scenarios or exceptional treatments are described separately*", etc.. There exists guidelines specific to levels 1 and 2 mentioned in section 1 (see figure1). For example, all interactions described at level 2 involve the computerised system supporting the business process as one of the agents.
Rules are of five types : (1) *analysis rules* analyse the semantic contents of each sentence against the linguistic patterns; (2) *mapping rules* map the elements of instantiated patterns into elements of the use case model, (3) *refinement rules* include the clarification and completion (3) *integration rules* help in positioning the scenario in the use case specification and (4) *emergence rules* prompt the need for other scenarios. Below, we exemplify the different types of *rules* mentioned above. More will be introduced in the next section with a walk through the EAF example guided process.
All rules have a premise and a body separated by a «→ ». They are described by a first order logical formula. The premise defines the precondition for executing the body. This precondition is expressed on elements of the use case specification and defines when the rule is applicable. The body is either an action required from the author (using the ASK predicate) or the generation of new elements of the use case specification (using the GENERATE predicate). In the prototype tool under development, rules are implemented as PROLOG clauses. The enactment mechanism of the guided process is therefore, built on top of the PROLOG inference engine. More details about rules can be found in [21].

3 Guiding the Description of the Electricity Application Fulfilment Business Process

This section describes the guided process of use case specification with the EAF (Electricity Application Fulfilment) example. The presentation is organised in five steps which show the progressive transformation, completion and integration of input scenarios describing the process of applying for electricity into a use case specification. With regard to the levels presented in section 1, in this example, we start with scenarios describing « organisational interactions » and end with the description of « system interactions ». Due to space limitation, the « system internal » level is not tackled in this paper.
Let us assume that the use case has been situated and its contextual information provided and stored in the use case structure (see in appendix the corresponding specification). The first step that we consider is therefore, the capture of the initial textual scenario which is intended to describe a normal course of actions.

3.1 The Initial Scenario Capture

Let us assume that after having studied the guidelines, the use case author writes down his view of the normal course of interactions between a user who applies for electricity and the Electricity Company (EC) as follows:

> The customer requests for a connection to the network. The commercial employee asks his identification papers and the location of the house to be connected. If the customer is not written-off and the installation exists the commercial employee asks to the customer to sign the contract and to pay the deposit. If the meter exists, a technical employee performs the connection to the network, but before the commercial employee sends a service order for connection. Then, the customer is informed that the connection to the network is done.
>
> *The final states are :* The customer is connected to the network. EC has a contract for this customer. EC has the network extended with a meter connection.

The text is then scanned with the analysis rules and mapped onto elements of the use case specification using mapping rules.

3.2 Linguistic Analysis and Mapping on the Episode Structure

As part of the linguistic approach sketched in section 2, *analysis rules* are used to identify the text structure, and to map the text onto instances of sentence and clause semantic patterns. For example the analysis rule AN1 aims at generating the action pattern from a clause having a subject and a verb expressed in the active form.

> *AN1 : $\forall$NG, V : [[NG](Subject)$_{Object}$ [V] (Main Verb)$_{Action}$] (VG active)$_{Action}$ $\rightarrow$ GENERATE(Action(V)[Agent:? ; Object:NG])*

Applying all necessary rules results in our example in the following set of instantiated patterns :

```
Communication (request) [Agent:'the customer' ; Object:'connection to the network' ; Source:'the customer' ; Destination: ?]
Communication (ask) [Agent:'the commercial employee' ; Object:'his identification papers and the location of the house to be
connected' ; Source:'the commercial employee' ; Destination: ?]
Constraint [ Condition : [State [Object:'the customer' ; State:'not written-off'], State[Object:'installation' ; State:'exists']] ;
        Constrained : Sequence
        [Before : Communication (ask) [Agent: 'commercial employee' ; Object:'to sign the
        contract', Source: 'commercial employee'; Destination: 'the customer']
         After : Communication (ask) [Agent: 'commercial employee' ; Object:' to pay the deposit' ; Source: 'commercial
        employee'; Destination: 'the customer']]];
Sequence[Before : Constraint [ Condition : State [Object:'the meter' ; State:'exists']; Constrained : Sequence
              [Before : Communication (send) [Agent: 'commercial employee'; Object:'a service order for connection' ;
                   Source: 'commercial employee' ; Destination: ?] ;
              After : Action (perform) [Agent:'a technical employee' ; Object:'the connection to the network']]]
        After : Communication (inform) [Agent: ?; Object: 'the connection to the network is done' ; Source : ?; Destination: 'the
        customer']]
```

The final states are : State [Object:'the customer' ; State:'connected to the network']
Ownership [Owner:'EC IS' ; Owned:'contract for this customer']
Ownership [Owner:'EC IS' ; Owned:'network extended with a meter connection']

Fig. 4. Instantiated patterns

The analysis of the initial scenario leads to the instantiation of seven action clause patterns within which six are communication action clause patterns. The instantiation of the action and communication action clause patterns from the input text provides values to the agent, source, object and destination cases. Question marks characterise missing elements in the input text ; they will be used for completion. The analysis rules identify the agents of the scenario out of the agent, source and destination cases : *'the customer', 'the commercial employee'* and *'the technical employee'*. Then, the object case identifies the resources used in each atomic action of the flow of actions *: 'connection to the network', 'his identification papers and the location of the house to be connected', 'to sign the contract and to pay the deposit', 'a service order for connection'*, and *'the connection to the network is done'*. Moreover, the analysis of the initial scenario shows that the actions are related through two sequence sentence patterns and two constraint sentence patterns.

Based on these pattern instances, *mapping rules* are automatically used to produce a natural language specification of the flow of actions of the initial scenario (consequently, it is a single course of actions).

For sake of clarity, we use an indented presentation of the flow of actions, and we associate a unique identification number to each action.

1. The customer requests for a connection to the network.
2. The commercial employee asks his identification papers and the location of the house to be connected.
3. If the customer is not written-off and the installation exists
    4. Then
    5. The commercial employee asks to the customer to sign the contract
    6. The commercial employee asks to the customer to pay the deposit.
    7. If the meter exists
        8. Then
        9. The commercial employee sends a service order for connection.
        10. A technical employee performs the connection to the network.
        11. The customer is informed that the connection to the network is done

Final states : The customer is connected to the network. EC has a contact for this customer. EC has the network extended with a meter connection.

Fig. 5. Flow of actions and final states after the mapping of the initial scenario

Let us comment this mapping. First, based on the action and communication action clause patterns instantiated during the initial text analysis, atomic actions are identified through rules MA1, MA2 and MA3.

MA1 : ∀ V, A, O, S, D : Communication (V) [Agent:A ; Object:O ; Source:S ; Destination:D] ∧ (Unify (A, S) ∨ Unify (A, D)) → GENERATE(Atomic Action (Name : V, From Agent : S, To Agent : D, Parameter : O))
MA2 : ∀ V, A, ∃ O : Action (V) [Agent:A ; Object:O] ∧ Agent (O) → GENERATE(Atomic Action (Name : V, From Agent : A, To Agent : O))
MA3 : ∀ V, A, ¬∃ O : Action (V) [Agent:A ; Object:O] ∧ Agent (O) → GENERATE(Atomic Action (Name : V, From Agent : A, To Agent : A, Parameter : O))

As stated in these rules, the atomic actions are analysed separately, even if they occur in the same sentence. Communication action pattern instances lead to the mapping of an atomic action. Our purpose being not to rephrase the use case author scenario, the expression of the atomic actions identified in the initial text is not modified.
Based on the sequence patterns, atomic actions have been organised in the right sequence. For example, the sentence *"a technical employee performs the connection to the network, but before the commercial employee sends a service order for connection"* has been split into *"(9) The commercial employee sends a service order for connection. (10) A technical employee performs the connection to the network"*. When no explicit sequencing of the actions is expressed, the ordering of the sentences in the initial scenario is respected.
Flow conditions such as *"if the customer is not written-off"* or *"if the meter exists"*, are identified from constraint patterns. Once identified, the alternative flows of actions are isolated, and the corresponding flow conditions are put in the order provided by the constraint pattern instances. For example the sentence *"If the meter exists, a technical employee performs the connection to the network, but before the commercial employee sends a service order for connection"* becomes *"(7) If the meter exists (8) Then (9) The commercial employee sends a service order for connection (10) A technical employee performs the connection to the network"* .

3.3 Clarification and Completion of Actions

The linguistic analysis provides a baseline for linguistic based clarification and completion of the identified atomic actions. Both rules rely on situations identifying possible linguistic ambiguities in the expression of the atomic actions.
The *clarification rules* are used to change the wording and to remove possible ambiguities. Even if the first guideline recommends to *"avoid the use of anaphoric references such as he, she, it, his and him"*, it is necessary to check systematically the text provided by the author. The grammatical analysis performed as a pre-requisite for analysis rules provides the information required for these checks. Clarification rule CL1 uses this information and proposes to replace anaphoric references by nouns.

CL1 : ∀ A : (Action [Agent:A ; Object:_] ∨ Action [Agent:_ ; Object:A] ∨ Communication [Agent:_ ; Object:_ ; Source:A ; Destination:_] ∨
Communication [Agent: _ ;Object: _ ; Source: _ ; Destination:A] ∨ State [Object: _ ; State:A] ∨ Ownership [Owner:A ; Owned: _] ∨
Ownership [Owner: _ ; Owned:A] ∨ Localisation [Location:A]) ∧ Anaphoric Reference (A) → ASK(« Clarify A by replacing this anaphoric reference by a noun »)
Note : « _ » is used to denote an anonymous variable which value is of no importance. The predicate 'Anaphoric Reference (A)' identifies if the term A includes a pronoun (he, his, him, etc.).

The use case author has been using the pronoun *"his"* in the second sentence of his scenario. The clarification rule suggests to *"clarify* his *by replacing this anaphoric reference by a noun"*. Taking this suggestion into account, he/she now decides to modify the flow of action 2 and replace *"his"* by *"the customer's"* which is a more explicit resource name. The action (2) thus becomes *"The commercial employee asks the customer's identification papers and the location of the house to be connected"*.
As explained in the previous section, the instantiated patterns may highlight missing parameters through question marks. Some of the c*ompletion* rules (e.g. CO5) help avoiding this form of incompleteness. In the EAF example, several communication action pattern instances are in the situation of one or several missing elements. For example, as shown in the analysis section, analysing the atomic action *"the customer is informed that the connection to the network is done"* instantiates the pattern *Communication (inform) [Agent: ?; Object: 'the connection to the network is done' ; Source : ?; Destination: 'the customer']* where the agent of the action and the source of the communicated information are missing.

CO5 : ∀ V , ∃ O: Communication (V) [Agent : ? ; Object : O, Source : ? ; Destination : ?] / Atomic Action (V,_) → ASK(« Complete : V by ... (agent of the communication) from... (source of the communication) to... (destination of the communication) »)

Applying the completion rule CO5 displays the following template and asks the use case author to fill it in : "the customer is informed that the connection to the network is done *by... (agent initiating the communication) from... (source of the communication) "*. Making use of the template, the use case author completes the sentence which becomes *"the customer is informed by the commercial employee that the connection to the network is done"* in which the commercial employee is the agent and the source.
The systematic application of linguistic completion rules leads to the completion of four actions. The new version of the current flow of actions specification is shown in figure 6 where the supplementary elements are in bold, and the elements that have been clarified in bold and italic.

1. The customer requests **the commercial employee** for a connection to the network.
2. The commercial employee asks **to the customer** ***the customer's*** identification papers and the location of the house to be connected.
3. If the customer is not written-off and the installation exists
    4. Then
    5. The commercial employee asks to the customer to sign the contract.
    6. The commercial employee asks to the customer to pay the deposit.
    7. If the meter exists
        8. Then
        9. The commercial employee sends **to the technical employee** a service order for connection.
        10. A technical employee performs the connection to the network.
        11. The customer is informed **by the commercial employee** that the connection to the network is done.

Final states : The customer is connected to the network. EC has a contact for this customer. EC has the network extended with a meter connection.

Fig. 6. Flow of actions after linguistic clarification and completion

3.4 Completing Action Dependencies

In the use case model, atomic actions are refined by several sub-types: service request, service provision, information provision, information request, internal action, etc. Based on this typology, we defined *action dependency patterns* which state dependencies between several types of atomic actions. The non respect of these dependencies is captured in the situations of the completion rules. Rule CO8, for example, states that the provision of a service S from an agent B to an agent A should be preceded in the flow of actions by the request of the service S from A to B.

CO8 : ∀ V1, S, A, B : (Atomic Action(Name : V1, From Agent : B, To Agent : A, Parameter : S) ∧ (Type(V1) = 'Service Provision')) ∧ ¬(∃ V2 : Atomic Action (Name : V2, From Agent : A, To Agent : B, Parameter : S) ∧ (Type(V2) = 'Service Request') ∧ Follow(V1, V2)) → ASK(« Complete service provision V1 with the anterior action :... (service request S from A to B) »)

Similarly, any service request should be followed by a service provision, this also applies to information requests and provisions. These patterns are exploited in other completion rules which are similar to CO8. Standalone requests or provisions of services and information can thus be identified and completed if necessary with their counterpart.

Another action dependency pattern establishes the dependency between an alternative action and the action of verification of the corresponding flow condition. The corresponding rule is triggered when this dependency pattern is not respected in the flow of actions allowing to associate a verification action to the condition.

A systematic application of the associated rules on our flow of actions of figure 7 leads to the following specification, in which the new elements are emphasised in bold.

1. The customer requests the commercial employee for a connection to the network.
2. The commercial employee asks to the customer the customer's identification papers and the location of the house to be connected.
3. **The customer gives to the commercial employee the customer's identification papers and the location of the house to be connected**.
4. **The commercial employee checks if the customer *record* is not written-off *and the customer record exists* and the installation exists**.
5. If the customer ***record*** is not written-off ***and the customer record exists*** and the installation exists
    6. Then
    7. The commercial employee asks to the customer to sign the contract.
    8. The commercial employee asks to the customer to pay the deposit.
    9. **The customer gives to the commercial employee the signed contract**
    10. **The customer gives to the commercial employee the money for deposit**.
    11. **The commercial employee gives back to the customer the customer's identification papers.**
    12. **The commercial employee checks if the meter exists**.
    13. If the meter exists
        14. Then
        15. The commercial employee sends to the technical employee a service order for connection.
        16. A technical employee performs the connection to the network.
        17. **The technical employee informs the commercial employee that the connection is done**.
        18. **The commercial employee sends to the customer a copy of the contract.**
        19. The customer is informed by the commercial employee that the connection to the network is done.

Final states : The customer is connected to the network. **The customer has the customer's identification papers. The customer has a copy of the contract**. EC has a contact for this customer. EC has the network extended with a meter connection.

Fig. 7. Flow of actions after the action dependencies completion.

Let us comment the incremental changes occurred between figures 6 and 7. As completion rules are based on types of actions, the use case author is first asked to provide atomic action typing. He/she thus identifies that :

- the *"request connection to the network"* action is a *service request *,*
- the *"ask the customer's identification papers and the location of the house to be connected"* action is an *information request,*
- the *"ask to sign the contract "* action is an *information request,*
- the *"ask to pay the deposit"* action is a *service request,*
- the *"send a service order for connection"* action is a *service request,*
- the *"perform the connection to the network"* is an *internal action*,
- the *"inform that the connection to the network is done"* action is a *service provision *.*

The *"ask the customer's identification papers and the location of the house to be connected"* information request action, does not have a corresponding information provision. At this point, the use case author thinks about describing the interactions between the customer and the actors of the EAF in a more detailed way. Thus, he/she inserts the following sentence in the flow of actions *"The customer gives his identification papers and the location of the house to be connected"*. Using the linguistic analysis and mapping rules, the sentence is then converted into the atomic action 3 of figure 7. Indeed, the linguistic completion and clarification are also performed, as presented in previous section. This has led to replace *"his"* by *"the customer's"*, and to complete with a destination : *"to the commercial employee"*. In the same way, the *"ask to sign the contract"* information request action is completed by the use case author inserting the following sentence *"The customer gives to the commercial employee the signed contract"*. The same applies to *"ask to pay the deposit"*. Finally, the *"send a service order for connection"* service request action from the commercial employee to the technical employee is completed by the corresponding service provision action *"The technical employee informs the commercial employee that the connection is done"*.
The application of the completion rules leads also to acknowledge that some parts of the specification are complete with respect to action dependency patterns. For example, the *"request connection to the network"* action is a service request from the customer to the commercial employee with the corresponding provision of the requested service (*connection to the network*). This correspondence is described by the use case author (see the two * in the list of typed actions provided above) at the same time than the types of actions. Thus, the description is already complete with respect to the satisfaction of a requested service. Note also that the use case author may decide not to apply the suggested completion.
There are two flow conditions in the specification : (a*) "the customer is not written-off and the installation exists",* (b) *"the meter exists"*. They do not have a preceding action for checking the flow condition. Asking the use case author to verify the need for two new actions for the condition checking leads to complete the episode specification with *"(4) The commercial employee checks if the customer is not written-off and the installation exists"* and *(12) "The commercial employee checks if the meter exists"*.
Using the corresponding completion rule, the use case author is asked to verify that the if these two conditions are complete. This leads him to insert another condition in the fourth sentence giving thus *"(4) The commercial employee checks if the customer is not written-off and the customer exists and the installation exists"*.
Final states have also to be completed, using the associated rule. The use case author is asked to verify their completeness and if needed provides the missing elements. With regard to the added elements, the use case author is asked to provide the necessary action for the final states to be reached.
Similarly, the use case author is asked to verify that the conditions referring to agent names are either dealing with the agent itself or an abstract representation of the agent. For example, in action n° 4 of figure 7, the condition "*the customer exists*" may mean that either the customer is in the EC office or that he is known in the record of the company. This leads to replace *"customer"* by *"customer record"* giving thus*" (4) The commercial employee checks if the customer record is not written-off and the customer record exists and the installation exists"*.

3.5 Reasoning on Flow Conditions

A flow of actions issued from a scenario description involves several flow conditions. For instance, in the current version of the normal episode of the EAF use case, there are four flow conditions :

- if the customer record is not written-off
- if the customer record exists
- if the installation exists
- if the meter exists

As a consequence of the scenario definition, this flow of actions together with the flow conditions constitute a pathway that permits the user to reach successfully one of the possible final states of the episode. In the case above, the flow of actions leads to an happy EC customer with the connection to the network. As presented in section 2, the normal episode may comprise several pathways, and be complemented by exceptional episodes. Each of them permits to reach one of the possible final states. We believe that reasoning on the flow conditions of the initial scenario description can help to discover exceptional episodes, and new pathways of the normal episode. The emergence rules based on flow conditions support this reasoning. These rules raise the questions of what happens if the flow conditions do not hold.

Based on the flow conditions of a given pathway all combinations of negative conditions are calculated. Thanks to emergence rules, the use case author is first asked to characterise all of them being either normal, exceptional or meaningless. Second, if two combination of conditions lead to describe the same scenario, then it should be clarified by the use case author. The result of applying these rules leads to the following table.

(1)	*customer is written-off*	**: exceptional**
(2)	*(customer is not written-off) & (customer does not exist) & (installation exists) & (meter exists)*	**: normal**
(3)	*(customer is not written-off) & (customer exists) & (installation exists) & (meter does not exist)*	**: normal**
(4)	*(customer is not written-off) & (customer does not exist) & (installation exists) & (meter does not exist)*	**: normal**
(5)	*(customer is not written-off) & (customer exists) & (installation does not exist) & (meter does not exist)*	**: normal**
(6)	*(customer is not written-off) & (customer does not exist) & (installation does not exist) & (meter does not exist)*	**: normal**
(7)	*(customer is not written-off) & (customer exists) & (installation does not exist) & (meter exists)*	**: meaningless**
(8)	*(customer is not written-off) & (customer does not exist) & (installation does not exist) & (meter exists)*	**: meaningless**

For each normal and exceptional scenario, a new iteration in the specification process starts. This includes the activities 2, 3 and 4 presented in section 2.3 : the textual scenario description provided by the use case author, its analysis and completion and its integration in the use case specification. There are two kinds of integration : the integration of an exceptional episode, and the integration of a new course of actions in the normal episode. The integration of an exceptional episode is simple and consists in relating the exception to an action of the normal episode. The integration of a new normal course of actions in the normal episode requires to embed the new flow of actions in this episode.

The *guidelines* proposed to the use case author during the capture of these new scenarios are the same as the ones used to capture the initial scenario. In addition, we offer a *copy & paste* facility which enables to duplicate in the scenario under writing, a course of actions which already exists in the specification. This facility is also used by the tool to provide to the author a partially written scenario description and ask him to complete it. The following text illustrates this step of the process for case (1). The use case author's writing is in bold whereas the text automatically generated by the guidance tool is in italics, the use of the copy and paste functionality is mentioned as a comment between the signs " /* " and " */ ".

*/*copy & paste action(s) 1 to 2 of the normal episode */*
The customer requests the commercial employee for a connection to the network.
The commercial employee asks to the customer the customer's identification papers and the location of the house to be connected.
The customer gives to the commercial employee the customer's identification papers and the location of the house to be connected.
The commercial employee checks if the customer record is not written-off and the customer record exists and the installation exists.
If ¬(the customer record is not written-off)
Then **the commercial employee informs the customer that he is written-off and the connection to the network can not be done.**
Final states **: The customer is not connected to the network.**

The NL text is analysed and completed in a similar way as the one illustrated for the initial scenario. The resulting exceptional episode specification is the following.

Exceptional episode of the use case *Electricity Application Fulfilment*
Name **: WrittenoffCustomer**
Occurrence condition **: When ⌉(the customer record is not written-off).**
Where **: the action 5 of the NormalCase episode.**
Action :
1. The commercial employee informs the customer that the customer record is written-off and the connection to the network can not be done.
2. The commercial employee gives back to the customer the customer's identification papers.

Final states : The customer is not connected to the network. The customer has the customer's identification papers.

Proceeding in the same way with the scenario number 2 leads the use case author to describe the flow of actions when the customer does not exist, the installation exists and the meter exists as shown in figure 8.

*/*copy & paste action(s) 1 to 4 of the normal episode */*
The customer requests the commercial employee for a connection to the network.
...
If the customer record is not written-off
Then If ⌉(the customer record exists)
  Then **The commercial employee creates the customer record**
  /*copy & paste actions 7 to 10 of the normal episode */
  If (the installation exists) Then
    The commercial employee asks to the customer to sign the contract
    The commercial employee asks to the customer to pay the deposit.
    The customer gives to the commercial employee the signed contract.
    The customer gives to the commercial employee the money for deposit
    The commercial employee gives back to the customer the customer's identification papers.
    The commercial employee checks if the meter exists.
    If (the meter exists)
      Then
      /*copy & paste actions 13 to 17 of the normal episode */
      The commercial employee sends to the technical employee a service order for connection.
      A technical employee performs the connection to the network.
      The technical employee informs the commercial employee that the meter connection is done.
      The commercial employee sends to the customer a copy of the contract.
      The customer is informed by the commercial employee that the connection to the network is done.

Final states : **The customer is connected to the network. The customer has the customer's identification papers. The customer has a copy of the contract. EC has a new customer. EC has a contact for this customer. EC has the network extended with a meter connection.**

Fig. 8. The scenario number 2 as provided by the use case author

Note that the *copy & paste* functionality is used in this case by the use case author to introduce actions 7, 8, 9, 10, 11, 12, 15, 16, 17, 18 and 19 of the normal episode in the current scenario.
The analysis and completion steps are performed to obtain a specification that can be integrated in the use case specification. The integration step stands in two parts : the integration of the final states and the integration of the actions. The integration of the final states leads to add the keyword *"sometimes"* when a final state exists in the normal episode but not in the new pathway and vice versa. This results in adding *"sometimes"* before the final state *"EC has the network extended with a meter installation"* in the normal episode during the integration of scenario number 5. Then the integration rules are applied to re-organise the flow of actions of the normal episode.
The integration of all the normal flows of actions, namely cases (2), (3), (4), (5) and (6) leads to the normal episode of figure 9. An asterisk marks a flow condition related to an exceptional episode.
The same reasoning can be recursively applied to the new flow conditions (33) and (39). This leads to the emergence and specification of two exceptional episodes, namely *"NetworkConnectionAborted"* and *"InstallationOnly"* that are described in the appendix. The appendix presents all exceptional episodes of the EAF use case.

Normal episode
name : NormalCase
action :

1. The customer requests the commercial employee for a connection to the network.
2. The commercial employee asks to the customer the customer's identification papers and the location of the house to be connected.
3. The customer gives to the commercial employee the customer's identification papers and the location of the house to be connected.. *
4. The commercial employee checks if the customer record is not written off and the customer record exists and the installation exists
5. If the customer record is not written off *
    6. Then
    7. If ¬(the customer record exists)
        8. Then
        9. The commercial employee creates the customer record
    10. If (the installation exists)
        11. Then
            12. The commercial employee asks to the customer to sign the contract.
            13. The commercial employee asks to the customer to pay the deposit.
            14. The customer gives to the commercial employee the signed contract.
            15. The customer gives to the commercial employee the money for deposit. *
            16. The commercial employee gives back to the customer the customer's identification papers.
            17. The commercial employee checks if the meter exists.
            18. If (the meter exists)
                19. Then
                    20. The commercial employee sends to the technical employee a service order for meter connection.
                    21. A technical employee performs the connection to the network.
                    22. The technical employee informs the commercial employee that the meter connection is done.
                23. Else
                    24. The commercial employee sends to the technical employee a service order for meter installation and a service order for meter connection.
                    25. A technical employee performs the meter installation and the connection to the network.
                    26. The technical employee informs the commercial employee that the meter installation and the meter connection are done.
        27. Else
            28. The commercial employee requests to technical employee to investigate the site.
            29. The technical employee performs investigation
            30. The technical employee informs the commercial employee that the investigation is done.
            31. The commercial employee calculates price
            32. The commercial employee asks to the customer to pay for the installation.
            33. If the customer pays the commercial employee for installation *
                34. Then
                35. The commercial employee sends to technical employee a service order for installation.
                36. The technical employee performs installation
                37. The technical employee informs the commercial employee that the installation is done.
                38. The customer is notified by the commercial employee that the installation is done.
                39. If the customer asks to the commercial employee a connection to the network *
                    40. Then
                    41. The commercial employee asks to the customer to sign the contract.
                    42. The commercial employee asks to the customer to pay the deposit.
                    43. The customer gives to the commercial employee the signed contract.
                    44. The customer gives to the commercial employee the money for deposit. *
                    45. The commercial employee sends to the technical employee a service order for meter installation and a service order for meter connection.
                    46. A technical employee performs the meter installation and the connection to the network.
                    47. The technical employee informs the commercial employee that the meter installation and the meter connection are done.
    48. The commercial employee sends to the customer a copy of the contract.
    49. The customer is informed by the commercial employee that the connection to the network is done.

Final states : The customer is connected to the network. The customer has the customer's identification papers. The customer has a copy of the contract. Sometimes, EC has a new customer. EC has a contact for this customer. EC has the network extended with a meter connection. Sometimes, EC has the network extended with a meter installation. Sometimes, EC has the network extended with an installation.

Fig. 9. Version 4 of the Normal episode

3.6 Completing the Use Case Specification with « System Interactions »

Now that we have integrated all scenarios describing the interactions between the human agents within the use case specification, we shall concentrate on the interactions between the agents of the organization and an automated system that shall support the business process, what is called « system interactions » in section 1, figure 1. To this end, we have defined a set of completion rules which aim at querying the use case author about the requirements for a computerized system that can support the performance of the process. The rules concentrate on the both the communication and internal actions.

For each communication action, the author is asked if the communication is supported by the system. If this is so, the author completes the action accordingly. Similarly, for each internal action, the use case author is asked if the action is supported in some way by the system. This also leads to the emergence of new requirements for the system internal and to the completion of the system interactions. Using these rules, the dialogue leads to complete the normal episode with system interactions. The result is shown in figure 10 (EC IS stands to the Electricity Company information system, it is the information system that supports the EAF business process).

---

1. The customer requests the commercial employee for a connection to the network.
2. The commercial employee asks to the customer the customer's identification papers and the location of the house to be connected.
3. The customer gives to the commercial employee the customer's identification papers and the location of the house to be connected.. *
4. The commercial employee **requests to EC IS** if the customer record is not written-off and the customer record exists and the installation exists
5. **EC IS checks if the customer record is not written-off and the customer record exists and the installation exists**
6. **EC IS informs the commercial employee if the customer record is not written-off and the customer record exists and the installation exists**
7. If the customer record is not written-off *

8\. Then

9\. If ¬(the customer record exists)

10\. Then

11\. The commercial employee **requests to EC IS** to create the customer record

12. EC IS creates customer record

13\. **EC IS acknowledges the creation of customer record to the commercial employee**

14\. If (the installation exists)

15\. Then

16\. The commercial employee asks to the customer to sign the contract.

17\. The commercial employee asks to the customer to sign the contract to pay the deposit.*

18\. The customer gives to the commercial employee the signed contract.

19\. The customer gives to the commercial employee the money for deposit.*

20\. The commercial employee gives back to the customer the customer's identification papers.

21\. The commercial employee **requests to EC IS** if the meter exists.

22. EC IS checks if the meter exists

23\. **EC IS informs the commercial employee if the meter exists**

24\. If (the meter exists)

25\. Then

26\. The commercial employee requests to EC IS to send to the technical employee a service order for meter connection.

27\. EC IS sends to the technical employee a service order for meter connection

28\. A technical employee performs the connection to the network.

29\. The technical employee informs the EC IS that the meter connection is done.

30\. EC IS informs the commercial employee that the meter connection is done

31\. Else

......

Fig. 10. Version 5 of the normal episode completed with system interactions

4. Conclusion

Activities, such as business process engineering, business process re-engineering or business process improvement call for accurate description of business processes. Our proposal is about an approach supporting the construction of BP specifications. A BP specification takes the form of a use case comprising information about the context of the BP, the interactions between the agents involved in the BP, the interactions of these agents with a computerised system supporting the BP and a set of internal system requirements.

We propose to guide the construction of a use case specification for BP using textual scenarios. A scenario describes, in natural language, interactions between different agents (i.e. a service requester and some service providers) and internal actions involving a single agent. An interaction is expressed as a communication action between two agents. A use case specification integrates all normal and exceptional scenarios describing a BP.

We use the case grammar presented in [21] to analyse and to extract the semantics of textual scenarios. On top of the linguistic approach, the construction of use case specification is guided. Guidance is based on use case model knowledge and takes the form of rules which encapsulate knowledge about the use case model concepts in order to facilitate (1) the completion of scenarios, (2) emergence of other normal or exceptional scenarios and (3) integration of scenarios into a complete use case specification. The guided process was illustrated with a real case study dealing with the business process *"Electricity Application Fulfilment"* borrowed to an Electricity Company. In this paper, we focused on the interactions between the organisational agents of the studied business process, and their interactions with the automated system supporting the BP.

Our current work consists in extending the guidance rules to support the emergence of system internal requirements on one hand, and system contextual requirements on the other hand. The former relates to internal system objects and behaviour whereas the latter deals with the context in which the business process takes place (weaknesses, opportunities, non functional requirements, etc.). Meanwhile, we are completing the current PROLOG implementation to handle the entire set of rules presented in this paper.

References

1. J.M. Caroll, The Scenario Perspective on System Development, in J.M. Carroll (ed.*), Scenario-Based Design: Envisioning Work and Technology in System Development* (1995).
2. 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).
3. B. Dano, H. Briand, F. Barbier, A Use Case Driven Requirements Engineering Process, In *Third IEEE International Symposium On Requirements Engineering (RE'97)*, Antapolis, Maryland (IEEE Computer Society Press, 1997).
4. E. Dubois, P. Heymans, M. Jarke, K. Phol, K. Weidenhaupt, A. Sutcliffe and N.A.M. Maiden, Integration of Scenario Representations: Formalisation and Examples, ESPRIT Reactive Long Term Research Project 21.903 CREWS, Deliverable W4: Knowledge Representation Working Group (1997).
5. T. Erickson, Notes on Design Practice: Stories and Prototypes as Catalysts for Communication, in John M. Carroll (ed.*), Scenario-Based Design: Envisioning Work and Technology in System Development* (John Wiley and Sons, 1995) 37-59.
6. C. Fillmore, The Case For Case, in Holt, Rinehart and Winston (eds.*), Universals in Linguistic Theory* (Bach & Harms, 1968) 1-90.
7. J.D. Gould, How to design usable systems, in M. Helander (ed.), *Handbook of Human-Computer Interaction* (Elsevier Science, 1988) 757-790.
8. Hammer M. Champy J., "Re-engineering the corporation : a manifesto for business revolution", Harper Collins publishers, inc., New York, 1993.
9. I. Jacobson, The Use Case Construct in Object-Oriented Software Engineering, in John M. Carroll (ed.), *Scenario-Based Design: Envisioning Work and Technology in System Development* (John Wiley and Sons, 1995) 309-336.
10. I. Jacobson, M. Ericsson and A. Jacobson*, The Object Advantage, Business Process Reengineering with Object Technology* (Addison-Wesley Publishing Company, 1995).
11. 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. Monicha, Scenario Use in European Software Organisations - Results from Site Visits and Questionnaires, ESPRIT Reactive Long Term Research Project 21.903 CREWS, Deliverable W1: Industrial problem capture Working Group (1997).

12. J. Karat, Scenario Use in the Design of a Speech Recognition System, in John M. Carroll (ed.*), Scenario-Based Design: Envisioning Work and Technology in System Development* (John Wiley and Sons, 1995) 109-135.
13. K. Koskimies and H. Mossenbock, Scene: Using Scenario Diagrams and Active Text for illustrating Object-Oriented Programs, in *Proc. of ICSE-18* (1995) 366-375.
14. M. Kyng, Creating Contexts for Design, in John M. Carroll (ed.*), Scenario-Based Design: Envisioning Work and Technology in System Development* (John Wiley and Sons, 1995) 85-107.
15. R.L. Mack, Discussion : Scenarios as Engines of Design, in John M. Carroll (ed.*), Scenario-Based Design: Envisioning Work and Technology in System Development* (John Wiley and Sons, 1995) 361-387.
16. J. Nielsen, Scenarios in Discount Usability Engineering, in John M. Carroll (ed.*), Scenario-Based Design: Envisioning Work and Technology in System Development* (John Wiley and Sons, 1995) 59-85.
17. C. Potts, K. Takahashi and A.I. Anton, Inquiry-based Requirements Analysis, *in IEEE Software* 11(2) (1994) 21-32.
18. J.C.S. do Prado Leite, G. Rossi, F. Balaguer, A. Maiorana, G. Kaplan, G. Hadad and A. Oliveros, Enhancing a requirements baseline with scenarios, In *Third IEEE International Symposium On Requirements Engineering (RE'97)*, Antapolis, Maryland (IEEE Computer Society Press, 1997) 44-53.
19. B. Regnell, K. Kimbler and A. Wesslen, Improving the Use Case Driven Approach to Requirements Engineering, in *the Second IEEE International Symposium On Requirements Engineering*, York, England (I.C.S. Press, March 1995) 40-47.
20. S.P. Robertson, Generating Object-Oriented Design Representations via Scenarios Queries, in John M. Carroll (ed.*), Scenario-Based Design: Envisioning Work and Technology in System Development* (John Wiley and Sons, 1995) 279-308.
21. C. Rolland, C. Ben Achour : Guiding the construction of textual use case specifications, in the Data & Knowledge Engineering Journal, 25(1-2) Special Jubilee issue, March 1998, 125-160.
22. M.B. Rosson and J.M. Carroll, Narrowing the Specification-Implementation Gap, in John M. Carroll (ed.*), Scenario-Based Design: Envisioning Work and Technology in System Development* (John Wiley and Sons, 1995), 247-278.
23. J. Rumbaugh, M. Blaha, W. Premerlani, F. Eddy, and W. Lorensen*, Object-Oriented Modeling and Design*, (Prentice Hall, 1991).
24. J. Rumbaugh and G. Booch, Unified Method, Notation Summary Version 0.8 (Rational Software Corporation, 1996).
25. K. Weidenhaupt, K. Pohl, M. Jarke, P. Haumer, Scenario Usage in System Development : a Report on Current Practice, ESPRIT Reactive Long Term Research Project 21.903 CREWS, Deliverable W1-D2, (1997).
26. J. Whiteside, J. Bennett and K. Holtzblatt, Usability Engineering : Our experience and evolution, in M. Helander (ed.), *Handbook of Human-Computer Interaction* (Elsevier Science, Amsterdam, 1988) 791-818.
27. R. Wirfs-Brock, Designing Objects and their Interactions: A Brief Look at Responsibility-driven Design, in John M. Carroll (ed.*), Scenario-Based Design: Envisioning Work and Technology in System Development* (John Wiley and Sons, 1995) 337-360.

Appendix

The use case specification is composed of the contextual information shown below, one normal episode shown in figure 9 and four exceptional episodes.

Contextual information

Use case name	Electricity Application Fulfilment
Initiating agent	A customer of the Electricity Company (EC)
Goal	To connect the customer to the company network
Initial states	Customer is present in the EC office with a completed application form.

Exceptional episode of the use case *Electricity Application Fulfilment*
Name **:** WrittenoffCustomer
Occurrence condition **:** When ⌉(the customer record is not written-off).
Where **:** the action 5 of the NormalCase episode.
Action :

1. The commercial employee informs the customer that the customer record is written-off and the connection to the network can not be done.
2. The commercial employee gives back to the customer the customer's identification papers.

Final states : The customer is not connected to the network. The customer has the customer's identification papers.

Exceptional episode of the use case *Electricity Application Fulfilment*
name : NoIdentificationPaper
occurrence condition : When ⌉(the customer gives to the commercial employee the customer's identification papers and the location of the house to be connected).
Where : the action 3 of the NormalCase episode.
action :

1. The commercial employee informs the customer that the connection to the network can not be done without customers identification paper's.

Final states : The customer is not connected to the network.

Exceptional episode of the use case *Electricity Application Fulfilment*
name : NetworkConnectionAborted
occurrence condition : When ⌉(the customer gives to the commercial employee the signed contract and the money for deposit) OR ⌉(the customer pays the commercial employee for installation).
Where : the action 15 and a year after action 32 of the NormalCase episode.
action :

1. The commercial employee informs the customer that the connection to the network can not be done without payment.

Final states : The customer is not connected to the network. The customer has the customer's identification papers.

Exceptional episode of the use case *Electricity Application Fulfilment*
name : InstallationOnly
occurrence condition : When ⌉(the customer asks to the commercial employee a connection to the network) OR ⌉(the customer gives to the commercial employee the signed contract and the money for deposit).
Where : the action 43 and 6 months after action 38 of the NormalCase episode.
action :

1. The commercial employee informs the customer that the network connection request is aborted.

Final states : The customer is not connected to the network. The customer has the customer's identification papers. , EC has the network extended with an installation.