Research paper

A Framework for Requirements Elicitation through Mixed-Initiative Dialogue

Renaud Lecœuche [*†] renaudl@dai.ed.ac.uk
Chris Mellish[†] chrism@dai.ed.ac.uk
Dave Robertson[†] dr@dai.ed.ac.uk

Abstract

In this paper we present our work on requirements elicitation. The elicitation process is a complex task which necessitates computer support. Elicitation systems should ideally help their users check the correctness of the specifications obtained but also actively guide them in the acquisition of the requirements. We consider hereafter systems that communicate in natural language. We describe a framework that tries to improve the quality of the guidance it provides to its users by taking into account natural language constraints. We discuss the need for a theory of natural language dialogue structure, and we show how we have integrated such a theory within an early prototype of an elicitation system.

Keywords: requirements elicitation, guidance, knowledge-based systems, natural language, mixed-initiative, WWW

1 Introduction

Numerous frameworks, such as goal-oriented strategies (Dardenne *et al.*, 1993), scenario-based strategies (Hsia *et al.*, 1994) and rapid prototyping (Andriole, 1994; Ramesh and Luqi, 1993; Gomaa, 1997), have been suggested as aids to the acquisition of software requirements. Much less attention has been paid to the control of dialogue taking place between the users and the system whilst using such frameworks (Bubenko *et al.*, 1994). Our ability to develop sophisticated formal frameworks makes this deficiency more acutely felt, since increases in formality are often accompanied by greater difficulty in understanding and using the frameworks (Robertson *et al.*, 1989). Frameworks for requirements acquisition are not normally accompanied by theories of the types of dialogue which they support. However, some theories of dialogue structure are beginning to emerge, notably from the natural language processing community. It is interesting to examine how the structure of requirements acquisition frameworks, which attempt to control the development of a requirements specification, may be linked to the structure of theories of dialogue, which attempt to control dialogue according to the conventions of discourse.

Many approaches to the problem of natural language interaction for requirements elicitation have considered it as a translation problem: the users write specifications in natural language and the system expresses them in a formal framework (Mich, 1996; Moreno, 1997; Nanduri and Rugaber, 1996). Given the difficulty of understanding free natural language, most of these approaches rely on the use of constrained languages (Fuchs and Schwitter, 1995, 1996; Schwitter and Fuchs, 1996). Constrained languages define a set of rules that must be observed while writing sentences. The sentences written using such a language can then be automatically translated into the target formal framework thanks to a predefined mapping between the two. As the natural language statements, in their free or constrained forms, are directly and automatically translated into specifications that should ideally be consistent, complete and coherent[1], they should themselves have those properties. If there are conflicts in the natural language requirements, however, these may not be resolved because the system has no way of knowing what the user really meant. As a result, the task of the user

[*]PSI-LIRINSA, Université de Rouen, 76130 Mont-Saint-Aignan, France
[†]Department of Artificial Intelligence, University of Edinburgh, Edinburgh EH1 1HN, Scotland

[1]The properties that requirements should exhibit are still a matter of debate (Lindland *et al.*, 1994). The ones we cite are just used as examples. The point is that, whatever the properties, the specifications are constrained.

remains difficult. Although the formal notation of the target framework is hidden by the natural language, its constraints are still fully present and the user must consider them to write correct specifications. Vague natural language statements will not be translated into correct specifications. In some approaches, the system tries to limit this problem by offering immediate feedback to the user, i.e., the translation process is carried out incrementally. While of help, this does not change the fundamental problem of the approach, i.e., that the system does not provide guidance to its users in writing the specifications.

In order to actively help users in the process of writing the requirements, the elicitation system must interact with them. The emphasis here is no longer on translating requirements but on actively extracting them through dialogue with the users. This task is complicated by the fact that there are numerous ways of carrying out the requirements elicitation. In other words the requirement frameworks alone are not providing enough constraints to ensure a unique elicitation process. An arbitrary choice could be made, but forcing the users to adopt a predefined method is usually not possible as it would make the elicitation process very difficult to follow and understand. The system must therefore be able to adapt itself to various elicitation methods. On the other hand it is necessary for the system to make choices in order to provide active guidance. A "least-commitment" strategy, such as asking the user at every choice point what to do next, is not a useful approach (Ferguson *et al.*, 1996). One way of offering guidance without restricting the users too much is by using natural language constraints (NLC) and mixed-initiative. NLCs ensure that the system adopts a strategy that will guide the users in a natural and understandable manner. Mixed-initiative ensures that the users can redirect the process and are not limited to a single elicitation method. This is presented in figure 1. The space of ways-of-working defined by the requirements framework is reduced to the "acceptable" ways-of-working from a natural language communication point of view. The system then navigates that space, making choices to direct the elicitation process. The user is still able to intervene by influencing the system decisions at the different choice points if needed, e.g., re-orienting the path followed on figure 1 from 3 to 5. Using such an approach reduces some of the problems discussed above. The specifications do not need to be immediately correct as they will be checked and reworked by the system. The formal framework is hidden from the user but is still there to ensure the correctness of the specifications. Guidance is continuously offered through dialogue, which is influenced by but does not directly follow the steps of construction of the specification. Moreover, because of the mixed-initiative, the user is still free to divert the course of dialogue by volunteering strategic information.

The integration of NLCs with requirements elicitation rules is a relatively complex task. It requires two major steps. The first step consists in the definition of the NLCs. The second step consists in the actual integration of these constraints in the requirements elicitation process.

NLCs related to mixed-initiative dialogue have been studied by the natural language processing community since the '70s. Different theories have been proposed. We will present them and explain how they can direct the dialogue between the users and the elicitation system.

The integration of these constraints with the requirements elicitation constraints is a recent domain of research. The explanation community for knowledge-based systems has realised that in some domains explanation and reasoning are interleaved. This has led to the notion of "explanation-driven reasoning" (Wick and Thompson, 1993). In that approach, the reasoning of the system is directed by domain problem solving rules *and* by communication constraints. The same approach can be taken here since the elicitation process requires an on-going communication with the users during the reasoning. However, the interrelations between these two sets of rules are still not well understood. We will present in detail our way of integrating the NLCs with the requirements elicitation process.

This paper is organised as follows. In section 2, we present current dialogue theories and discuss how they can influence the dialogue between the users and the elicitation system. In section 3 we describe our system and how we have integrated the NLCs with the elicitation process. We then show in section 4 how the system works on an example. In section 5 we compare our system with related work. Finally we conclude in section 6 with some shortcomings of our approach and future work.

2 Dialogue theory

To study the structure of dialogues during the elicitation task we have recorded several dialogues in the domain

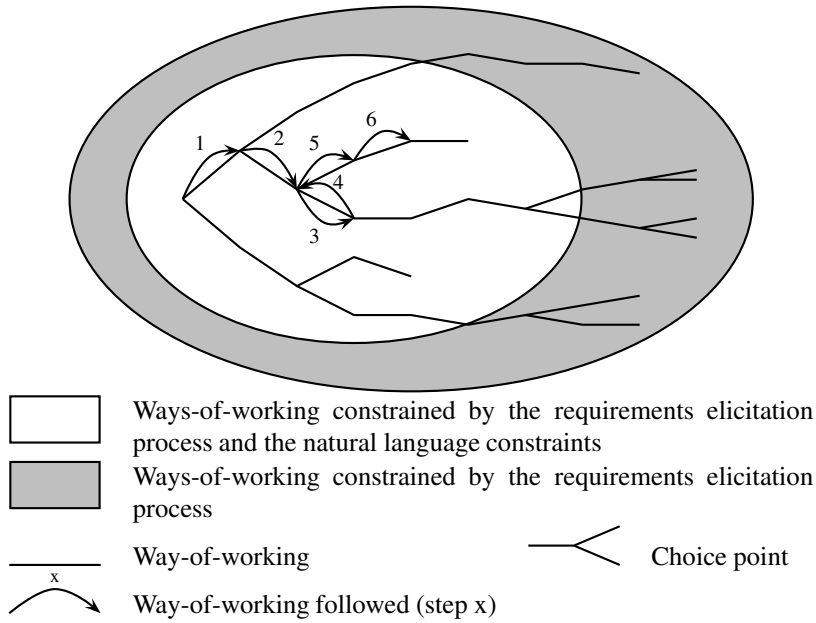

Figure 1: Mixed-initiative guidance

of research group WWW site construction. The participants in the dialogue are a WWW site expert acting as the elicitation system and a research group member acting as the user of the system. The aim of the dialogue is to elicit the structure[2] of the WWW site the user wants to create.

This study has provided two interesting results:

- There are numerous ways of describing the requirements. Although the information conveyed in the different dialogues was basically the same (need for certain information such as the publications of the group, its members, the projects carried out; need for certain navigation properties, such as a link on all the pages to the front page; the need for access restrictions) the way it was elicited varied a lot. Every dialogue is organised differently from the others. This confirms the fact that numerous elicitation methods are possible.

- There are constraints on the structure of the dialogues. Although the dialogues were very different from one another, each of them presents a clear internal structure. The information is acquired in an ordered way. A summary of one of the dialogues is shown in table 1. This table presents the concepts mentioned in a dialogue, in the order they were introduced or re-introduced. Except for a short diversion marked by italic text, the structure of the dialogue is clear: the dialogue first focused on the front page, then on pages mainly for internal use, then on pages for internal and external use, and finally on navigational constraints.

  A system oblivious of the dialogue structure, for example jumping from Publications to Current Researchers, would make the elicitation process difficult for humans to follow.

The major theory of focus in task-oriented dialogues has been proposed by Grosz for dialogue understanding (Grosz, 1977; Grosz and Sidner, 1986). In this theory, the dialogue focus is modelled by a stack. The dialogue is restricted to the concepts contained in the stack. Each time a new task is undertaken, a new focus is pushed on the stack. This focus contains the information needed to realise that particular task. Once the task has been achieved, the stack is popped, removing the concepts that were used for the task just terminated. The dia-

[2]By structure, we mean the organisation of the site and the navigation constraints. This corresponds to the third part of the usual decomposition of WWW sites in content, i.e., what is the site about, presentation, i.e., how is the information presented, and navigation, i.e., how users move between pages, topics.

logue of table 1 does mostly fit this model. Each time a new part of the requirements is elicited, a new focus is pushed on the stack. It is popped when the elicitation of that particular part is completed. The process then begins again for another part. For instance, the concepts used for describing the Useful Links are popped at the end of the elicitation of that topic and thus are not referred to when the Publications page is elicited.

The Grosz and Sidner theory has been criticised by McCoy and Cheng (McCoy and Cheng, 1991) on two points:

- Certain phenomena can not be explained by the stack model. For example, the research group member realised in our dialogue that he had forgotten an important piece of information about the useful links page. He therefore put that particular subject back into focus, completed the elicitation process, and switched back to the previous topic. This cannot be done if the focus is modelled by a stack since the focus history is lost when the stack is popped. Therefore McCoy and Cheng propose to use a focus tree rather than a focus stack. Pushes create new branches in the focus tree and pops move the focus back up in the tree.

- The Grosz and Sidner theory does not explain why certain focus shifts occur. For example, when describing an object, people tend to shift to the properties of the object, e.g., its colour, its weight, while they tend to shift to the general setting when describing a particular event. These different topic shifts are summarised in table 2. Since our system has to be able to direct the dialogue it must understand what focus shifts are appropriate to the situation it is in. Therefore having some generic information about the type of objects present in the domain and the possible shift of focus between them is necessary.

Site
Front Page
Pages for Internal Use
Useful Links
Publications
Pages for Internal and External Use
Talk Schedule
Links to Related Sites
Conferences
Useful Links
Links to Related Sites
Researchers
current
past
Projects
current
past
Navigation
Researchers - Project
All Pages - Front Page

Table 1: Example of dialogue structure
This table represents the structure of one dialogue. Concepts are listed from top to bottom in the order they appeared in the dialogue. Pushes are represented by tabulation, i.e., concepts that are "dominated" (see Grosz and Sidner, 1986) are on the right of the dominating concept. Italic text represents shifts back to previously introduced concepts.

An example of focus shift predicted by this latter theory is the Pages for Internal Use – Useful Links shift corresponding to a Attribute – Objects with that attribute shift. Indeed, the targeted audience of a page (people working in the research group or external people) is an attribute of the object `page`. By speaking about "Pages for Internal Use", the user puts into focus all the pages that have this value for the attribute. These pages are then focused on individually and are described in detail.

This theory does not however explain three important aspects of our problem:

- No rules are proposed for determining the shifts to previous topics, i.e., backward shifts. Although having a focus tree allows topics to be referred back to, the theory does not indicate how this is done. It is therefore unclear what the elicitation system should do in the case such a shift is needed, e.g., when the user realises that a piece of information is missing. Nothing is said on "unusual" forward shifts either, i.e., shifts to topics not yet discussed and not related to the current focus. Although the theory recommends to adopt a depth-first method, i.e., preferably pushing related topics in focus as shown in table 2, it does allow for other methods, i.e., creating another and distinct

Type	Focus shift candidates
Object	attributes of the object, actions the object plays a prominent role in (e.g., is actor of)
Attribute	objects which have the attribute, more specific attribute
Setting	objects involved in the setting actions which typically occur in the setting
Action	actor, object, etc., of the action – any participant role purpose (goal) of action, next action in some sequence, sub-actions, specialisations of the action
Event	actions which can be grouped together into the event

Table 2: Candidate focus shift (from McCoy and Cheng, 1991)

branch of the focus tree. The rules that constrain these kinds of shifts are not clear, making the use of methods other than depth-first difficult.

- Nothing is said about the cue words that usually mark the shifts in the dialogue. Grosz proposes cue words, such as "now", "anyway", "the end", to make the shift explicit in the dialogue. McCoy and Cheng do not propose similar words. Without them the dialogue would be difficult for users to understand.

- The shifts in focus also depend on domain related properties. For example, the shift to a link between pages can only be done if there are at least two pages that already have been considered[3].

We propose some solutions to these problems:

- We have incorporated domain specific shift rules in our system[4]. As will be explained in section 3, this is achieved by mapping shift rules with domain specific elicitation rules. For the example given above, a focus rule requires the concept of navigation to be in focus to elicit a link and a domain specific elicitation rule requires that two different pages must already exist. The resulting rule that will effectively be applied combines the two separate rules and requires the two set of constraints to be satisfied.

- The cue words used in the dialogues we have collected are different from one dialogue to the other. Therefore it seems more practical to use a standard cue word rather than debating on the best possible cue word for each particular shift. In our current system very simple cue words are used, such as "Let's talk about" and "Let's go back to" for push and pop shift respectively. More complex combinations will certainly be necessary to provide more natural sounding dialogues but the current dialogues are already much clearer than those without cues.

- Rules for determining unusual forward and backward focus shifts are still an open problem. If a concept is introduced only once in the dialogue, a backward shift to that concept is easy to do: we just need to put it back into focus with the part of the focus tree leading to it. However, if a concept is introduced more than once, therefore having multiple paths in the focus tree leading to it, choosing the right path remains an open question. Fortunately this latter case does not seem to appear often in the elicitation process, as far as we can judge from the sample dialogues we have collected. In our system, the shift to a previous focus is supported by jumping back to its latest occurrence in the dialogue. While certainly not fully satisfactory, this approach performs reasonably well. Forward shifts are even more difficult to allow since there may be numerous potential branches in the focus tree leading to a particular topic. Therefore we only allow a one-step forward shift. In other words, we enable the users to choose the next topic to push, without modifying the existing branch of the tree. This very restricted approach allows the system to keep track of the focus correctly and enables the users to adjust the focus.

[3] We consider here links internal to the site being elicited.

[4] Our system is a very early prototype working in the domain of WWW site elicitation. Using the system the users can specify the site organisation and navigation. HTML pages can be automatically generated from the specifications.

Our current system contains a little more than 10 focus rules for around 15 concepts and 15 attributes in the knowledge base.

3 System

A major point of this paper is to show how the integration of NLCs with a usual requirements elicitation system can be achieved. We especially tried to minimise the impact of adding the NLCs on the elicitation system. As a result our system can run with or without NLCs. This enables us to develop semi-independently the set of elicitation rules and the set of focus rules. This feature should also help us evaluate the usefulness of the NLCs by turning on and off the NLCs and comparing the resulting dialogues.

The general architecture of our system is described in figure 2.

The system is composed of 5 main components:

Specification model describes the entities that should be elicited and their relations. It is currently based on a domain-specific object-oriented framework[5] but could be replaced by an existing requirements representation framework such as KAOS (Dardenne *et al.*, 1993; Darimont, 1995). Elicited specifications are incrementally added to the model as the elicitation process progresses.

Dialogue model describes the entities that can be spoken about during the dialogue and their relations. It is currently based on an object-oriented framework[6]. Its separation from the specification model allows it to be linked with an upper model (Bateman, 1990), facilitating natural language communication. The dialogue model also contains the focus tree.

Elicitation rules define how particular pieces of information are acquired. They are based on a precondition-action template. The preconditions can test for specific structures in the specification model.

Dialogue rules define how concepts are created and how the focus can evolve. They are based on a precondition-action template. The preconditions can test for specific structures in the dialogue model.

Mapping rules define how the concepts used in the dialogue are related to the entities manipulated by the elicitation system. The mapping is done between a subset of the specification model and a subset of the dialogue model. Therefore some entities, part of the *unmapped region* of the specification model, may not be considered in the dialogue. This allows a form of implicit requirements (Hagopian and Maxwell, 1996) where the system takes in charge the creation of specifications hidden from the user.

Concepts that are not directly used in the specifications can also be spoken about. This is for instance the case of the navigation concept (see table 1). These concepts form the *unmapped region* of the dialogue model.

The mapping rules are the link between the elicitation process and the dialogue process that are otherwise independent.

The mapping rules define a relation of compatibility between the elicitation rules and the dialogue rules. An elicitation rule and a dialogue rule are compatible if and only if:

- the rules act locally on the unmapped region of the specification model and on the unmapped region of the dialogue model respectively. These rules represent a special case of compatibility that do not depend on the mapping. They are represented in separate boxes in figure 2.
- the elicitation rule creates a class or relation instance that is mapped to a concept or link instance created by the dialogue rule and conversely.
- the elicitation rule sets the value of an attribute of a specification instance that is mapped to a dialogue instance attribute whose value is set by the dialogue rules and the two values are compatible. Two values are compatible if and only if:
  - the two attributes whose values are set are a class attribute and a concept attribute and the values are equal.

[5] The specification model is composed of classes such as site, subsite, page, and of relations such as part-of or link. Classes and relations can have attributes.

[6] In order to distinguish the dialogue model from the specification model, a class in the dialogue model is called a concept and a relation a link.

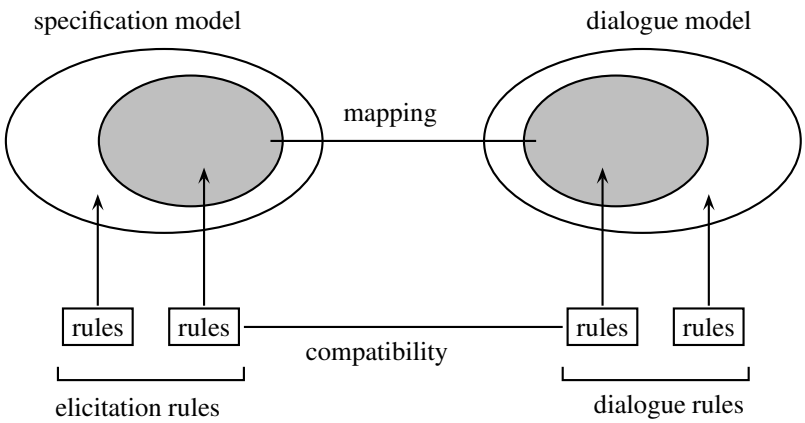

Figure 2: System architecture

- the two attributes whose values are set are a relation attribute and a link attribute and the values correspond to identifiers of mapped instances.
- one of the values is a question while the other is grounded, in which case the question is not asked and its answer is calculated from the grounded value.
- the two values are questions, in which case only the dialogue rule question is asked while the elicitation rule question answer is calculated from the answered value.

We can see how these rules work on three small examples.

The focus rule used to go up in the dialogue tree, i.e., the rule that would correspond to popping the stack in Grosz's theory, acts on the focus tree which is in the unmapped region of the dialogue model. Indeed there is no equivalent structure in the specification model. Therefore this rule is compatible with any specification rules that would act locally on the unmapped region of the specification model (first rule). This rule fires for example at the end of a page elicitation, putting the notion of sub-site back into focus.

Let's suppose that the class `subsite` is mapped to the concept `information cluster`, i.e., each sub-site corresponds to an information cluster users talk about. A specification rule creating an instance of a sub-site must then be matched by a dialogue rule creating an information cluster instance in order to fire (second compatibility rule).

Finally suppose that the classes `site` and `subsite` are mapped to the concepts `WWW site` and `information cluster` respectively. Suppose also that a relation and a link called `part of` are mapped. Then, an instance of the relation that links an instance of `subsite` with an instance of `site` will have to be matched by an instance of the link that links the instance of the concept `information cluster` *mapped to the subsite instance* with the instance of concept `WWW site` *mapped to the site instance* (third compatibility rule, point 2). This is represented in figure 3.

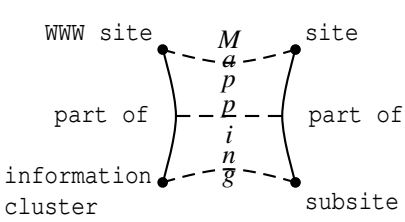

Figure 3: Mapping between relations and links

The compatibility rules enable the system to synchronise the requirements elicitation process with the dialogue. The specifications and dialogue rules are written independently but the mapping between the specification and dialogue models ensure that they are executed coherently. The compatibility rules also enable

the system to reduce the number of questions asked by using the mapping between specifications and dialogue entities to avoid redundant questions (third compatibility rule, points 3 and 4).

The system then runs as shown in the following figure.

```
While some rules can be applied
    find a dialogue rule
    find an elicitation rule
    check their compatibility
    If compatible
        apply dialogue rule
        apply elicitation rule
    EndIf
EndWhile
```

Checking the compatibility of the rules ensures that they abide by the domain specific elicitation constraints as well as the dialogue constraints. It also avoids putting any overhead on the user as similar questions are eliminated. Two special void rules have been added. Those rules do not have any preconditions and have no action. They enable independent elicitation rules or independent focus rules to fire when no other rules are applicable. For example, when all the possible rules have been fired for a particular topic, a rule making the focus evolve will fire. Its specification counterpart will be the specification void rule, since no actions on the specification are made. Those two rules although trivially compatible are prevented from firing together thus avoiding a potential infinite loop in the program execution.

4 Example

In this section we present a sample of the dialogue conducted by the system. No attention has been paid yet to the presentation of the system utterances. We are mainly interested in the structure of the dialogue and the capability of the system to direct or reset the focus as the user wants. The dialogue is presented in table 3 while comments are given below.

1. `ignore` is a keyword indicating that the user does not know the answer to that question at that moment. The system tries to elicit a maximum of information before shifting the focus of dialogue, thus minimising the complexity of the dialogue. However, the users can indicate that they would rather come back to a question later by using the `ignore` keyword. The system then searches for another question which may involve changing the dialogue focus.

2. The system is usually in charge of setting and resetting the focus appropriately as the dialogue progresses. However, if the users disagree with the automatic setting of the focus, they can indicate it. The system will then ask for the topic the users want to discuss and will reset the focus accordingly. The elicitation process restarts in that new

Let's talk about the site	
—: ok.	
Let's talk about the front_page	
—: ok.	
What is the topic of the page ?	
—: ignore.	(1)
Do you want to create a new subsite ?	
—: yes.	
Let's talk about the subsite	
—: ok.	
(. . .)	
—: no.	(2)
Which topic do you want to discuss ?	(3)
—: front_page.	
What is the topic of the page ?	
—: research_group.	
—: no.	
Which topic do you want to discuss ?	
—: publications.	
(. . .)	
Let's talk about the navigation	
—: ok.	
Which page do you want to consider ?	
—: current_publications.	
—: ok.	
Do you want to create a new internal_link ?	(4)
—: yes.	
What is the destination of that link ?	(5)
—: past_publications.	
—: ok.	
(. . .)	

Table 3: Example of dialogue

situation. The users can only shift the focus to already mentioned topics, i.e., perform a backward shift. Forward shifts are only permitted in very restricted ways (see 5).

3. The system asks to what topic the focus should be reset to and then determines the questions that can be asked with that focus.

4. An `internal link` links two pages of the site. Its anchor is the page actually in focus while...

5. ... its destination is the page indicated in this question. Putting the anchor page in focus before doing the elicitation enables the system not to ask what the anchor of each link is. It is by convention the page focused on. Therefore, the information that is implicit in the focus makes the elicitation process less burdensome. The system allows the user to make a one-step forward shift in focus since in that case the focus tree remains entirely defined. This is the only case in general where the system can do that.

Once the requirements for the site have been elicited the system can create actual WWW pages using presentational rules. These rules define the lay-out of the pages based on the elicited information[7]. The front page created is shown on figure 4.

Figure 4: Site front page

5 Related work

In this section we compare our approach with two pieces of related work.

The requirements engineering process has been extensively studied in the F3 project (Plihon and Rolland, 1995; Rolland, 1993; Rolland *et al.*, 1995). In that project requirements elicitation is seen as a spiral, as shown on figure 5.

A major point in the F3 approach that is related to our work is the shifting from one problem to another through context dependency. Rolland defines context dependency as follows (Rolland, 1994):

> "Context-dependency is the key element of our global process approach. In order to guide the requirements engineer in his progression along the spiral we propose to predefine dependencies. Therefore, within a turn of the spiral the requirements engineer will be supported by the associated *chunk*; from one turn to another he will be supported by a *chunk cluster* grouping and ordering chunks which are dependent one from the other. [...] Criteria for clustering are of two kinds: *intentions* (or goal) and *mode* (or reasoning drive)."

Our approach is similar to the F3 one in that context dependencies are guiding the reasoning of the system. This approach differs from ours on two major aspects:

- The context-dependencies in F3 are predefined. Therefore the users are forced into a particular

[7] Our research group WWW site is currently generated by this sort of rule. The content of the site is stored in the form of prolog terms that are interpreted by the rules to create the HTML pages.

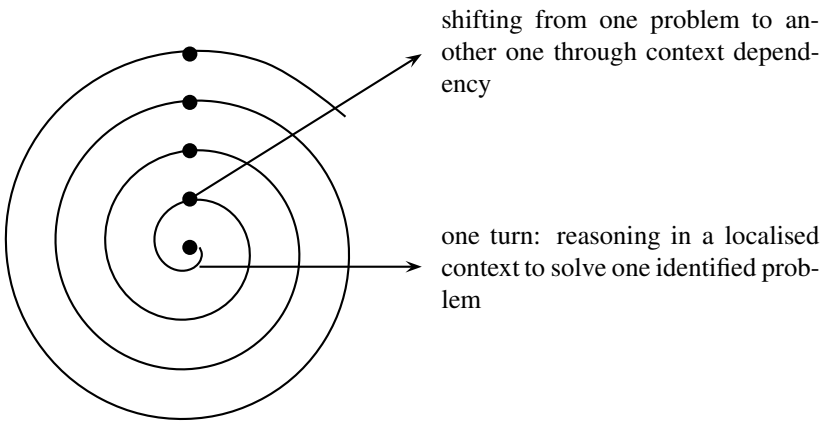

Figure 5: Requirements elicitation spiral (from Rolland, 1994)

method for eliciting the requirements[8]. For example, it is not clear how the users can switch back to a previous topic if they want to.

- The context-dependencies are based on elicitation process properties. NLCs are not taken into account. While being a huge improvement on unguided elicitation approaches, the F3 system may still be difficult to understand for its users since it does not include NLCs.

Another piece of work related to our approach is on topic explanation (Southwick, 1989). In this work, the rules used by an expert system are decomposed into topics. For example the following rule base is decomposed into four topics as shown on figure 6.

```
A if B and C
B if D and E
E if H
C if F and G
```

Each topic is characterised by a landmark defined as follows:

> "We can represent the rule base has a graph, showing the logical relationships of the rules. A problem solver must traverse this graph as it attempts to find a solution. A topic is a set of related rules, constituting a sub-tree in this search space. The landmark for a topic is the topmost designated node of the sub-tree, and the topic set consists of all the children of that node, (possibly) including other landmarks. [...] A topic has a description, consisting of background information necessary or useful to someone wanting to understand that topic."

The system uses these landmarks and the associated descriptions to explain its reasoning to its users. Each time a landmark is encountered during the reasoning, the system stops and explains what is going to be worked on. Topics and landmarks are therefore the main components in the dialogue between the system and its users. The systems also presents with the topic description all the methods it knows to solve the problem at hand. The users can then choose the method they desire. A certain mixed-initiative is thus built in the system.

Our approach differs from the topic explanation one on several accounts:

- We use the notion of topic not only to explain the reasoning of the system but also to guide it. Topics directly influence the firing of focus rules that in turn modify the applicability of the elicitation rules.

- We separate the definition of the topics from the system rule base using a mapping between the two.

[8] In fact, there may be choice points where the users can choose between competing paths. However the choices are limited.

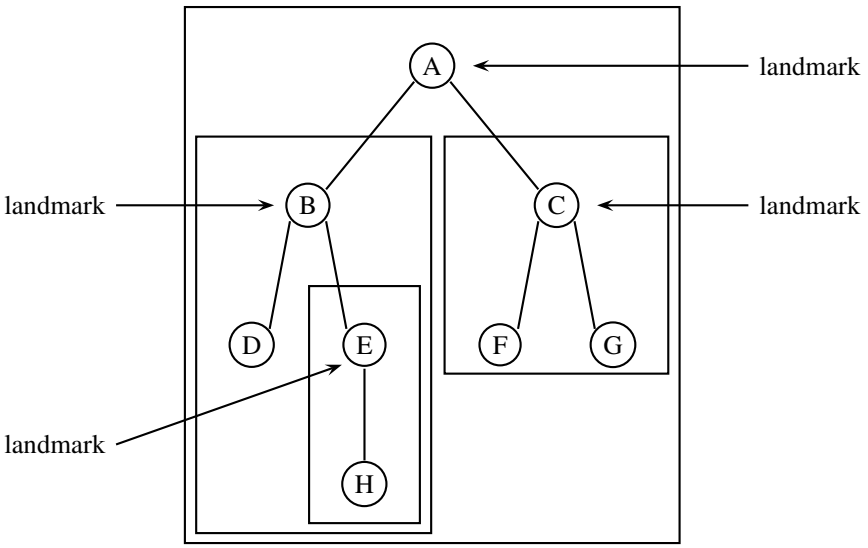

Figure 6: A rule base split into topics (from Southwick, 1989)

This enables us to introduce topics that have no equivalent in the system reasoning. For example, in our system, the notion of navigation is purely a natural language concept that has no mapping with elicited entities.

- The mixed-initiative that we allow in our system is more important than just a choice of methods. The users can switch back to entirely different topics.

In spite of those differences, the idea behind our system and that behind topic explanation are very similar. The mapping between the dialogue model and the specification model creates an implicit division of the latter into topics. It might therefore be interesting to study the influence of designing the dialogue model on the specification model. As already mentioned by Swartout (Swartout and Smoliar, 1987) who uses NLCs to constraint knowledge-base representation, the creation of these two models is not independent and their interactions can lead to more stable design.

Our system improves the two previous approaches by allowing a dialogue based on NLCs and by allowing more freedom in the mixed-initiative.

6 Conclusion and Future Work

In this paper we have presented our framework for requirements elicitation through mixed-initiative dialogue. We first described what the current theories of dialogue are and how we can adapt them for requirements elicitation. We then showed how those theories can be implemented to cooperate with the elicitation rules. We finally compared our approach with some related work.

There are a number of remaining problems:

- The dialogue theory we use is incomplete. The cases of complex backward and forward shifts in focus are not adequately handled. Our focus rules are also domain specific and lack generality. We plan to remedy these two problems by studying in greater depth the dialogues we have collected.

- The system does not have a proper user interface. We need to interface our system with a natural language generation tool. Given the fact that the NLCs are separated from the elicitation process, this should be relatively easy. We are planning to see how we can link our system with existing upper-models, thus facilitating natural language production.

- Since the system is a very early prototype, no tests have been done on its usability for casual users. We plan to study the impact of the NLCs on the quality of dialogue using the capability of the system to use or not use the focus rules to guide the elicitation process.

References

Andriole, S. (1994). Fast, cheap requirements: prototype, or else! *IEEE Software*, **11**(2), 85–87.

Bateman, J. A. (1990). Upper modeling: A general organization of knowledge for natural language processing. In *Proceedings of the Workshop on Standards for Knowledge Representation Systems, Santa Barbara, USA*.

Bubenko, J., Rolland, C., Loucopoulos, P., and DeAntonellis, V. (1994). Facilitating fuzzy to formal requirements modelling. In *Proceedings of the IEEE International Conference on Requirements Engineering, Colorado-Springs, USA*.

Dardenne, A., van Lamsweerde, A., and Fickas, S. (1993). Goal-directed requirements acquisition. *Science of Computer Programming*, **20**, 3–50.

Darimont, R. (1995). *Process Support for Requirements Elicitation*. Ph.D. thesis, Université catholique de Louvain, Louvain-la-Neuve, Belgium.

Ferguson, G., Allen, J. F., and Miller, B. (1996). Trains-95: Towards a mixed-initiative planning assistant. In *Proceedings of the third conference on Artificial Intelligence Planning Systems, Edinburgh, Scotland*, pages 70–77.

Fuchs, N. and Schwitter, R. (1995). Attempto - controlled natural language for requirements specifications. In *Proceedings of the seventh ILPS workshop on logic programming environments, Portland, Oregon, USA*.

Fuchs, N. and Schwitter, R. (1996). Attempto controlled english (ace). In *Proceedings of the first international Workshop on Controlled Language Applications, Université catholique de Leuven, Leuven, Belgium*.

Gomaa, H. (1997). The impact of prototyping on sofware system engineering. In R. Thayer and M. Dorfman, editors, *Software Requirements Engineering*, pages 431–440. IEEE Computer Society Press, 2^{nd} edition.

Grosz, B. J. (1977). The representation and use of focus in dialogue understanding. Technical Report 151, SRI International.

Grosz, B. J. and Sidner, C. L. (1986). Attention, intentions, and the structure of discourse. *Computational Linguistics*, **12**(3), 175–204.

Hagopian, J. and Maxwell, T. (1996). Explicit and implicit resources: a simplified approach to user requirements modeling. In *Electronic Proceedings of the Space Ops 96 Symposium (http://www.op.dlr.de/SpaceOps/spops96/misplan/track3.htm)*.

Hsia, P., Samuel, J., Gao, J., Kung, D., Toyoshima, Y., and Chen, C. (1994). Formal approach to scenario analysis. *IEEE Software*, **11**(2), 33–41.

Lindland, O., Sindre, G., and Solvberg, A. (1994). Understanding quality in conceptual modeling. *IEEE Software*, **11**(2), 42–49.

McCoy, K. F. and Cheng, J. (1991). Focus of attention: Constraining what can be said next. In C. L. Paris, W. R. Swartout, and W. C. Mann, editors, *Natural Language Generation in Artificial Intelligence and Linguistics*, chapter 4, pages 103–124. Kluwer.

Mich, L. (1996). NL-OOPS: from natural language to object oriented requirements using the natural language processing system LOLITA. *Natural Language Engineering*, **2**(2), 161–187.

Moreno, A. M. (1997). Object-oriented analysis from textual specifications. In *Proceedings of ninth international conference on Software Engineering and Knowledge Engineering, Madrid, Spain*.

Nanduri, S. and Rugaber, S. (1996). Requirements validation via automated language parsing. *Journal of Management Information Systems*, **12**(2), 9–19.

Plihon, V. and Rolland, C. (1995). Modelling ways-of-working. In *Proceedings of the seventh international Conference on Advanced information Systems Engineering, Jyvaeskylae, Finland*.

Ramesh, B. and Luqi (1993). Process knowledge based rapid prototyping for requirements engineering. In *Proceedings of the first IEEE international symposium on Requirements Engineering, Los Alamitos, USA*, pages 248–255. IEEE Computer Society Press.

Robertson, D., Uschold, M., Bundy, A., and Muetzelfeldt, R. (1989). The ECO program construction system: Ways of increasing its representational

power and their effects on the user interface. *International Journal of Man Machine Studies*, **31**(1), 1–26.

Rolland, C. (1993). Modelling the requirements engineering process. In *Proceedings of the third European-Japanese Seminar on Information Modelling and Knowledge Bases, Budapest, Hungary*.

Rolland, C. (1994). A contextual approach for the requirements engineering process. In *Proceedings of the sixth international conference on Software Engineering and Knowledge Engineering, Jurmala, Latvia*.

Rolland, C., Souveyet, C., and Moreno, M. (1995). An approach for defining ways-of-working. *Information Systems Journal*, **20**(4).

Schwitter, R. and Fuchs, N. (1996). Attempto - from specifications in controlled natural language towards executable specifications. In *Proceedings of the GI EMISA Workshop, Tutzing, Germany*.

Southwick, R. W. (1989). Topic explanation in expert systems. In B. Kelly and A. Rector, editors, *Research and Development in Expert Systems V*, pages 47–57. Cambridge University Press.

Swartout, W. R. and Smoliar, S. W. (1987). Explanation: A source of guidance for knowledge representation. In *Proceedings of the Workshop on Knowledge Representation and Organization in Machine Learning, Eringerfeld, Germany*.

Wick, M. R. and Thompson, W. B. (1993). Reconstructive expert system explanation. *Artificial Intelligence*, **54**, 33–70.