Creating and Querying a Domain dependent Know-How Knowledge Base of Advices and Warnings

Lionel Fontan
IRIT - UPS,
118 route de Narbonne,
31062 Toulouse Cedex, France.
antonin_follet@hotmail.fr

Patrick Saint-Dizier
IRIT - CNRS,
118 route de Narbonne,
31062 Toulouse Cedex, France.
stdizier@irit.fr

Abstract

In this paper, we present the explanation structure of procedural texts, that supports and motivates the goal-instruction structure. We focus in particular on arguments, and show how arguments of type warnings and advices can be extracted. Finally, we show how a domain dependent know-how textual knowledge base can be constructed and queried.

1 Introduction

Procedural texts consist of a sequence of instructions, designed with some accuracy in order to reach a goal (e.g. assemble a computer). Procedural texts may also include subgoals. These are most of the time realized by means of titles and subtitles. The user must carefully follow step by step the given instructions in order to reach the goal.

The main goal of our project is to analyse the structure of procedural texts in order to efficiently and accurately respond to How-to ? questions. This means identifying titles (which convey the main goals of the text), sequences of instructions serving these goals, and a number of additional structures such as prerequisites, warnings, advices, illustrations, etc.

In our perspective, procedural texts range from apparently simple cooking recipes to large maintenance manuals. They also include documents as diverse as teaching texts, medical notices, social behavior recommendations, directions for use, assembly notices, do-it-yourself notices, itinerary guides, advice texts, savoir-faire guides etc. (Aouladomar et al., 2005). Procedural texts follow a number of structural criteria, whose realization may depend on the author's writing abilities, on the target user, and on traditions associated with a given domain. Procedural texts can be regulatory, procedural, programmatory, prescriptive or injunctive. The work we report here was carried out on a development corpus of French texts taken from the Web from most of the various domains cited above.

Argument extraction is not yet a very active area, althought it has obvious uses in question answering, in decision theory, etc. For example, extracting arguments from legal texts (ICAIL 2005) or for answering opinion questions is a major challenge of primary use.

We have developed a quite detailed analysis of procedural texts, identifying their main basic components as well as their global structure. For that purpose, we have defined two levels: a segmentation level that basically tags structures considered as terminal structures (titles, instructions, advices, prerequisites, etc.) and a grammar level that binds these terminal structures to give a global structure to procedural texts (Delpech et al. 2008). This structure is textual and dedicated only to elements relevant to procedurality.

Procedural texts are complex structures, they often exhibit a quite complex rational (the instructions) and 'irrational' structure which is mainly composed of advices, conditions, preferences, evaluations, user stimulations, etc. They form what is called the explanation structure, which motivates and justifies the goal-instructions structure, which is the backbone of procedural texts. A number of these elements are forms of argumentation, they provide a strong and essential internal cohesion and coherence to procedural texts.

An important aspect of this project is the accurate identification of the explanation structure as found in procedural texts in order (1) to better understand explanation strategies deployed by humans in precise, concrete and operational situations and (2) to build a knowledge base of advices and warnings related to an application domain, that reflects several forms of know-how on this domain. Such repositories exist, but they have been build completely manually, by various users, often in a wiki fashion. Our goal is then to allow users not only to query procedural texts via How to questions, but also to create and to access to a repository of advices and warnings (basically Why questions and some How-to questions to a lesser extent) about a certain task.

We have already studied the instructional aspects of procedural texts and implemented a quite efficient prototype within the TextCoop project (Delpech et al. 2008) that tags text with dedicated XML tags. In this paper, after a brief categorization of explanation structure as found in our corpus of procedural texts, we focus on the argumentation structure via the recognition of warnings and advices. Then, we show how a textual knowledge base of advices and warnings can be produced and how it can be queried.

2 The explanation structure in procedural texts

We first present, in this section, the general organization of the explanation structure as it emerged from corpus analysis. Then we develop the major component of procedural texts: the instructional compound.

2.1 A global view of the explanation structure

From our development corpus, we established a classification of the different forms explanations may take. Basically, the explanation structure is meant to guide the user by making sure that he will effectively realize actions as they are specified, via e.g. threats, rewards, evaluations, advices and warnings. The main structures are facilitation and argumentation structures; they are either global (they are adjoined to goals, and have scope over the whole procedure) or local, included into instructional compounds, with a scope local to the instructional compound. This latter case is by far the most frequently encountered. These structures are summarized as follows (the terms we use are either borrowed from works on rhetorical relations or are just ours if none exist):

- **facilitation structures**, which are rhetorical in essence (Kosseim et al 2000) (Van der Linden 1993), correspond to *How to do X ?* questions, these include two subcategories:
  (1) user help, with: hints, evaluations and encouragements and
  (2) controls on instruction realization, with two cases: (2.1) controls on actions: guidance, focusing, expected result and elaboration and (2.2) controls on user interpretations: definitions, reformulations, illustrations and also elaborations.

- **argumentation structures**, corresponding to *why do X ?* questions.

  These have either:
  (1) a positive orientation with the author involvement (promises) or not (advices and justifications) or
  (2) a negative orientation with the author involvement (threats) or not (warnings).

In what follows, we will mainly concentrate on this second point, and in particular on warnings and advices which are the most frequently encountered (since there are rarely involvements from the author). These will be used to construct the know-how knowledge base. Argumentation structures are relatively general to an applications domain, while facilitation structures are much more specific to the text and the targeted audiences. There are several ways of defining and approaching argumentation. Without entering any debate, we consider here the approach where an argument is composed of one or more supports associated with a certain statement, as in the following warning: *carefully plug in your mother card vertically, otherwise you will most likely damage its connectors.* where if the intruction (carefully plug in...) is not correctly realized, the user know the consequences.

2.2 From instructions to instructional compounds

In most types of texts, we do not find just sequences of simple instructions but much more complex compounds composed of clusters of instructions, that exhibit a number of semantic dependencies between each other, that we call **instructional compounds**. These are organized around a few main instructions, to which a number of subordinate instructions, warnings, arguments, and explanations of various sorts may possibly be adjoined. All these elements are, in fact, essential in a compound for a good understanding of the procedure at stake.

An instructional compound has a relatively well organized discourse structure, composed of several layers, which are:

- The **goal and justification** level, which has wider scope over the remainder of the compound, indicates motivations for doing actions that follow in the compound (e.g. *in your bedroom, you must clean regularly the curtains...*, which here motivates actions to undertake). It gives the fundamental motivation of the compound.

- The **instruction kernel structure**, which contains the main instructions. These can be organized temporally or be just sets of actions. Actions are identified most frequently via the presence of action verbs (in relation to the domain) in the imperative form, or in the infinitive form introduced by a modal. We observed also a number of subordinated instructions forms adjoined to the main instructions. These are in general organized within the compound by means of rhetorical relations, introduced below.

- The **deontic and illocutionary force structures**: consist of marks that operate over instructions, outlining different parameters. These linguistic structures play a major role in argumentation:

  - deontic: obligatory, optional, forbidden or impossible, alternates (or),
  - illocutionary and related aspects: stresses on actions: necessary, advised, recommended, to be avoided, etc. These marks are crucial to identify the weight of an argument.

- a **temporal structure** that organizes sequences of instructions (and, at a higher level, instructional compounds). In general, the temporal structure is very simple, with sequences of actions to carry out. In some cases, parallel actions are specified, which partially overlap.

- The **conditional structure**: introduces conditions over instructions within the compound or even over the whole instructional compound. We encounter quite a lot of structures organizing mutually exclusive cases.

- the **causal structure** that indicates the goal of an action. We identify four types of causal relations, following (Talmy 2001): intend-to (direct objective of an action: *push the button to start the engine*), Instrumented (*use a 2 inch key to dismount the door*), Facilitation (*enlarge the hole to better empty the tank*) and Continue (*keep the liquid warm till its colour changes*).

- The **rhetorical structure** whose goal is to enrich the kernel structure by means of a number of subordinated aspects (realized as propositions, possibly instructions) among which, most notably: enablement, motivation, circumstance, elaboration, instrument, precaution, manner. A group of relations of particular interest in this paper are arguments, developed hereafter.

Explanations and arguments help the user understand why an instruction must be realized and what are the risks or the drawbacks if he does not do it properly. An example of an instructional compound is:

$[_{instructional\ compound}$
$[_{Goal}$ To clean leather armchairs,]
$[_{argument:advice}$
$[_{instruction}$ choose specialized products dedicated to furniture,
$[_{instruction}$ and prefer them colourless]],
$[_{support}$ they will play a protection role, add beauty, and repair some small damages.]]]

We have here an argument of type advice which is composed of 2 instructions (later called a conclusion) and a conjunction of three supports which motivate the 2 instructions.

3 Identifying arguments in procedures

In this section let us first give a quite informal definition of what an argument is, and how it interacts with the goal-instructions structure. Let us then focus on warnings and advices which are, by far, the most frequently encountered structures. Most warnings and advices are included into instructional compounds.

3.1 Argumentation and Action theories

Roughly, argumentation is a process that allows speakers to construct statements for or against another statement called the conclusion. These former statements are called supports. The general form of an argument is : **Conclusion 'because' Support** (noted as $C\ because\ S$). In natural language, conclusions often appear before the support, but they may also appear after. A conclusion may receive several supports, possibly of different natures (advices and warnings). Arguments may be more or less strong, they bear in general a certain weight, induced from the words they contain (Anscombre et al. 1981), (Moeschler 1985), (Amgoud et ali. 2001). In natural contexts, this weight is somewhat vague, and only general classes can be produced, e.g. from light to strong.

In the case of procedural texts, the representation and the role of arguments in a text can be modelled roughly as follows. Let G be a goal which can be reached by the sequence of instructions $A_i,\ i\ \in\ [1,n]$, whatever their exact temporal structure is. A subset of those instructions is interpreted as arguments where each instruction (A_j, viewed as a conclusion) is paired with a support S_j that stresses the importance of A_j (*Carefully plug in your mother card vertically, otherwise you will damage the connectors*). Their general form is: $A_j\ because\ S_j$ (we use here the term 'because' which is more vague than the implication symbol used in formal argumentation, because natural language is not so radical). Supports S_k which are negatively oriented are warnings whereas those which are positively oriented are advices. Neutral supports simply introduce basic explanations.

Similarly to the principles of argument theory, but within the framework of action theory (e.g. Davidson 2003), if A_j is associated with a support of type warning S_j then if A_j is not realized correctly, the warning S_j is 'active' and attacks the goal G, i.e. it makes its realization more difficult, if not impossible. Conversely, if S_j is an advice, it supports the goal G, making its full realization easier, or providing better results if A_j is executed. Note however that there is an implicit gradability in the realization of an action, which may be more or less accurately and completely realized. In that case, negative or positive consequences on the main goal evolve accordingly.

Supports can themselves receive supports : *don't add natural fertilizer, this may attract insects, which will damage your young plants*. In the same range of ideas, instructions A_j which are advices or warnings have a different status than 'normal', unsupported instructions (although one can say that most of them could be associated with an implicit support such as *otherwise you will fail*). Advices are often optional instructions: they are a kind of invitation to do the associated action for better results, whereas warnings are an incitation to be more careful. Therefore, instructions in a procedure do not have all the same operational strength and status.

As can be noted, our definition includes terms which are gradual: 'more difficult', 'easier', because in practice, failing to realize an instruction properly does not necessarily means that the goal cannot be reached, but the user will just be less successful, for various reasons. In the natural language expressions of conclusions (the A_j) as well as of supports, there are many modals or classes of verbs (like risk verbs) that modulate the consequences on G, contrast for example:
use professional products to clean your leathers, they will give them a brighter aspect. with:
carefully plug in your mother card vertically, otherwise you will most likely damage its connectors.
In the latter case, the goal 'mounting your own PC' is likely to fail, whereas in the former, the goal 'cleaning your leathers' will just be less successful.

3.2 Processing arguments

From the above observations, we have defined a set of patterns that recognize instructions which are conclusions and their related supports. We defined those patterns from a development corpus of about 1700 texts from various domains (cooking, do it yourself, gardening, video games, social advices, etc.). The study is made on French, English glosses are given here for ease of reading. The recognition problem is twofold: identifying propositions as conclusions or supports by means of specific linguistic marks (sometimes we also found a few typographic marks), and then delimiting these elements. In general, boundaries are either sentences or, by default, instructional compound boundaries. In procedural texts, roughly, the proportion of advices and warnings is almost equivalent.

3.2.1 Processing warnings

Warnings are basically organized around a unique structure composed of an 'avoid expression' combined with a proposition. The variations around the 'avoid expressions' capture the illocutionary force of the argument via several devices, ordered here by increasing force :
(1) 'prevention verbs like avoid' NP / to VP (*avoid hot water*)
(2) do not / never / ... VP(infinitive) ... (*never put this cloth in the sun*)
(3) it is essential, vital, ... to never VP(infinitive).
In cases where the conclusion is relatively weak in terms of consequences, it may not have any specific mark, its recognition is then based on the observation that it is the instruction that immediately precedes an already identified support.

Supports are propositions which are identified from various marks:
(1) via connectors such as: *sinon, car, sous peine de, au risque de* (otherwise, under the risk of), etc. or via verbs expressing consequence,
(2) via negative expressions of the form: *in order not to, in order to avoid, etc.*
(3) via specific verbs such as risk verbs introducing an event (*you risk to break*). In general the embedded verb has a negative polarity.
(4) via the presence of very negative terms, such as: nouns: *death, disease, etc.*, adjectives, and some verbs and adverbs. We have a lexicon of about 200 negative terms found in our corpora.

Some supports have a more neutral formulation: they may be a portion of a sentence where a conclusion has been identified. For example, a proposition in the future tense or conditional following a conclusion is identified as a support. However, as will be seen below, some supports may be empty, because they can easily be inferred by the reader. In that case, the argument is said to be truncated.

Patterns are implemented in Perl and are included into the TextCoop software. From the above observations, with some generalizations and the construction of lexicons of marks, we have summarized the extraction process in only 8 patterns for supports and 3 patterns for conclusions. Pattrens are basically morpho-lexical, with the need to recognize a few local structures, treated by means of local automata. A pattern in Perl has the following form:

```
 (PRO:PER--Modalite +)?--
evit(ez|er)--(\w+ )*--##
```

with modalite = devoir, veiller a, etre essentiel, etc. Some local automata are associated with most patterns in order to make them as generic as possible. In our programme, Perl scripts are treated one after the other, in sequence. We do not have any efficieny requirement since these treatments are realized in batch mode. However, for the whole processing, we tag abouit 200 Mo of text per hour on a standard 3GhZ Pentium machine.

3.2.2 Evaluation

In procedural texts, arguments are tagged by XML tags. We carried out an indicative evaluation (e.g. to get improvement directions) on a corpus of 66 texts over various domains, containing 302 arguments, including 140 advices and 162 warnings. This test corpus was collected from a large collection of texts from our study corpus. Domains are in 2 categories: cooking, gardening and do it yourself, which are very prototypical, and 2 other domains, far less stable: social recommendations and video games solutions. Arguments were manually tagged in these texts, and a comparison was made with the output of the system. Therefore, we report below the recall, the precision being almost 100% (very little noise).

We get the following results for warnings:

conclusion recognition	support recognition	(3)	(4)
88%	91%	95%	95%

(3) conclusions well delimited (4) supports well delimited, with respect to warnings correctly identified.

As far as warnings are concerned, results are really good. Errors are very diverse, some of them involve uses of the verb *pouvoir* (to be able to) and the auxiliary *être* (to be).

3.2.3 Processing Advices

Conclusions of type advice are identified essentially by means of two types of patterns (in French):
(1) advice or preference expressions followed by an instruction. The expressions may be a verb or a more complex expression: *is advised to, prefer, it is better, preferable to, etc.,*
(2) expression of optionality or of preference followed by an instruction: *our suggestions: ...*, or expression of optionality within the instruction (*use preferably a sharp knife*).
In addition, as for warnings, any instruction preceding a support of type advice is a conclusion.

The first pattern above is recogned by the following script:

```
 ceci|cela|NOM|PRO:
PER+--tre?--ADV?--Verb/
advice exporession--(\w+ )*--##
```

Supports of type advice are identified on the basis of 3 distinct types of patterns:
(1) Goal exp + (adverb) + positively oriented term. Goal expressions are e.g.: in order to, for, whereas adverb includes: better (in French: mieux, plus, davantage), and positively oriented term includes: nouns (savings, perfection, gain, etc.), adjectives (efficient, easy, useful, etc.), or adverbs (well, simply, etc.). For this latter class of positively oriented terms we constructed a lexicon that contains about 50 terms. terms.
(2) goal expression with a positive consequence verb (favour, encourage, save, etc.), or a facilitation verb (improve, optimize, facilitate, embellish, help, contribute, etc.),
(3) the goal expression in (1) and (2) above can be replaced by the verb 'to be' in the future: *it will be easier to locate your keys*.

Similarly as above, we carried out an indicative evaluation on the same corpus as above, with the same experimental conditions. We get the following results for advices:

conclusion recognition	support recognition	(3)	(4)	(5)
79%	84%	92%	91%	91%

(3) conclusions well delimited, (4) supports well delimited, both with respect to advices correctly identified. (5) support and conclusion correctly related.

A short example of an annotated text is given in Fig. 1 below.

4 Constructing and Querying a know-how textual database

Besides studying the textual structure of procedural texts and responding to How-to questions (Delpech et al. 2007) from the analysis of these texts, a major application of this work is the construction of **domain know-how knowledge base**, which is probably quite basic, but which could be subject to interesting generalizations. Obviously, to make this knowledge optimal, it would be useful to associate with every statement a formal representation that supports inference, data fusion, etc.

This domain know-how knowledge base of advices, hints and warnings is of much importance for different types of users who have a procedure to realize a task but who want to know more before starting. Some psychological experiments have in fact shown that, besides instructions given in procedural texts, users are very much interested in what remains implicit in those texts: what you are supposed to know or care about (but have no means to ask). This know-how textual database is aimed to fill in this kind of gap.

The work presented hereafter is still exploratory, since the task is quite complex. The domain know-how textual database is planned to be either directly consulted by users, or queried by means of requests in natural language or keywords.

4.1 Constructing a text database of domain know-how

There are repositories of advices organized by sector of activity available on the Web (e.g. http://www.conseils-gratuit.com). These are realized manually: most of these advices come from hints sent by readers of these pages. These repositories contain in general simple advices and also small procedures which are hints to better realize a certain task.

In our approach, the text units that we have access to are either (1) procedural texts decomposed into subgoals when they are large (e.g. the different phases of assembling a computer), or (2) instructional compounds. Compounds roughly correspond to the various advice forms found in man-

$[_{procedure}$
$[_{title}$ How to embellish your balcony
$[_{Prerequisites}$ 1 lattice, window boxes, etc.]
....
$[_{instructional-compound}$ In order to train a plant to grow up a wall, select first a sunny area, clean the floor and make sure it is flat......
$[_{Argument}$ $[_{Conclusion:Advice}$ You should better let a 10 cm interval between the wall and the lattice.]
$[_{Support:Advice}$ This space will allow the air to move around, which is beneficial for the health of your plant.]
......]]]]

Figure 1: An annotated procedure

ually realized repositories of advices. Advices and warnings mainly appear within these instructional compounds. However, compounds being inserted into a larger procedure may be somewhat elliptical in some cases. Therefore, the textual database we are constructing will contain titles (to settle context) and compounds.

Let us now present the construction of the domain know-how textual database of advices and warnings. At this stage, this is an experimental tentative that needs further improvements and evaluation. We first process texts by domain, according to our corpus (about 8000 texts). The main functions of this processing are:
(1) cleaning web pages from irrelevant data (adds, forums, summaries, links, etc.),
(2) XML tagging the instructional aspects, with dedicated tags: tagging titles (and reconstructing the numerous titles which are incomplete, with missing verb or object, and tagging instructional compounds and prerequisites, and
(3) tagging within instructional compounds advices and warnings based on the patterns given above.

In the textual database, the first level of structure is domains: `house, coocking, administration, health, garden, computer, do it yourself, animals, beauty, society`.

Next, below each of these domain top nodes, we have a list of items that correspond to procedures main titles (e.g. *boucher un trou avec du platre (fill up a hole with plaster)*. Since, for most domains we have several hundreds of documents, we need to organize those titles and abstract over them. This is being organized around two axis:
(1) task oriented: where action verbs are grouped on the basis of closely related terms to form a single title (for that purpose we use our verb lexical base (Volem)). A second level of generalization is carried out by skipping adjuncts, therefore we have items like: 'repairing walls' independently of the material or the technique used, e.g. with plaster. mastic, cement.
(2) object oriented: where we only keep track of the objects, viewed as a theme: wall, wood, plaster, etc. so that the user can access the different operations these objects may undergo.
These revised titles form a second level in the structure of the know-how textual knowledge base.

Below these links, we have the list of relevant web pages. Each of these pages is associated with an index composed of the set of titles it contains and the list of supports identified (reconstructed supports are not yet included). Titles are used to make the procedure context more precise so that the scope of supports is more clear, since some supports are vague. A short example is given in Fig. 2 below. Supports which are too vague to be of any use are filtered out. At the moment we are studying various forms of filters based on the type of words they contain and their relevance.

4.2 Querying the know-how textual database

In general, attempting to match queries directly with supports in order to get the advice, i.e. the associated conclusion does not lead to the best results because supports are often incomplete or they contain a lot of pronominal references. Our matching procedure therefore includes the taking into account of the page title, or subtitles together with support contents. It seems that this leads to better results in terms of accuracy and relevance.

Related to Fig. 2, a query could be: *how to get smooth plaster surfaces on a wall ?*. There is no procedural text that corresponds to this query,

```
domain: do-it-yourself
    topic: repairing walls
        repairing your walls with plaster -[INDEX: Title, list of supports]-[TEXT]
        filling up holes in your walls ]-[INDEX: Title, list of supports]-[TEXT
.....
    topic: painting walls
.....
```

Figure 2: A text database index

which is rather an advice request. Answering this question is realized by the following steps:
(1) based on keywords which appear as objects in the query, select a domain and a topic in the knowledge base.
(2) then, over the topics selected, match the query with one or more supports. Matching is obviously not direct and requires, as in most systems, some flexibility. Of interest here are adjectives, which abound in this type of question, for which we need to develop scales that capture the different language expressions of the properties they characterize. This type of scale, in (Cruse 1986), is called non branching proportional series. For example 'smooth' will appear on a scale of type 'surface granularity' that includes other adjectives such as rough, grainy, etc.

5 Perspectives

The work presented here complements the tagging of titles and instructional compounds in procedural texts of various domains, as reported in (Delpech et al. 2008). We analyzed the forms arguments of type advice and warning may take, and have implemented and tested a system that tags those structures and attempts at reconstructing empty supports. At this level, there is still linguistic and formal work to be carried out, for example to evaluate the illocutionary force of arguments and to better settle this work within action theory. We believe we have a very useful corpus of examples of arguments, of much interest for research in argumentation theory.

In a second stage, we have now established a first version of criteria to construct from these arguments a domain know-how textual database, that users can query to get additional information when realizing a task, often information which remains implicit in a procedure, but that users do need to operate safely and efficiently. The construction of such a repository is a complex task that we will pursue, together with an analysis of how it can be queried accurately.

Credits We thank the French ANR-RNTL research programme for supporting this project. We also thank very warmly Leila Amgoud for discussions on argumentation, Daniel Kayser for comments on this paper, and 3 anonymous reviewers.

References

Amgoud, L., Parsons, S., Maudet, N., 2001, *Arguments, Dialogue, and Negotiation*, in: 14th European Conference on Artificial Intelligence, Berlin.

Anscombre, J.-Cl. Ducrot, O., 1981, *Interrogation et Argumentation*, in Langue francaise, no 52, L'interrogation, 5 - 22.

Aouladomar, F., Saint-dizier, P., 2005, *Towards Answering Procedural Questions*, Workshop KRAQ05, IJCAI05, Edinburgh.

Cruse, A., 1986, *Lexical Semantics*, Cambridge Univ. Press.

Davidson, D., 1963, *Actions, Reasons, and Causes*, Journal of Philosophy, 60.

Delpech, E., Saint-Dizier, P., 2008, Investigating the Structure of Procedural Texts for Answering How-to Questions, LREC 2008, Marrakech.

Kosseim, L., Lapalme, G., 2000, *Choosing Rhetorical Structures to Plan Instructional Texts*, Computational Intelligence, Blackwell, Boston.

Moschler, J., 1985, *Argumentation et Conversation, Eléments pour une Analyse Pragmatique du Discours*, Hatier - Crédif.

ICAIL, 2005, Automatic semantics extraction in law documents, proceedings, C. Biagili et ali. (ed), ACM ICAIL publications, Stanford.

Vander Linden, K., 1993, *Speaking of Actions Choosing Rhetorical Status and Grammatical Form in Instructional Text Generation* Thesis, University of Colorado.