

A Conceptual and Operational Model for Procedural Texts and its Use in Textual Integration

Isabelle DAUTRICHE (1,2), Patrick SAINT-DIZIER (1)
(1) IRIT-CNRS, 118, route de Narbonne, 31062 TOULOUSE France
(2) INSA, 135, av. de Rangeuil, 31077 TOULOUSE France
isabelle.dautriche@gmail.com, stdizier@irit.fr

Abstract

In this paper, we propose a conceptual and operational model for procedural texts, within Action theory. We propose some elements to model the complexity and explicitness of instructions, showing, how, for a given execution of the procedure, the success of the goal can be measured. Next, we show how a procedure can be enriched via textual integration in order to improve the success rate of the goal.

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. Goals and subgoals 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. A procedure is in general a finite sequence of subactions, terminal elements being instructions. Instruction execution may be conditional, associated with preferences or advices. Instructions may contain continuous processes, loops and may be organized as a cluster in a loop, till some result is reached. This is presented in (Delpech et al. 2008). Not surprisingly, procedures have a structure that resembles what is usually found in imperative programming.

The main goal of our project is to analyze the structure of procedural texts in order to efficiently and accurately respond to How-to? questions. This means identifying titles (which basically convey the main goals of a procedure in a number of domains), sequences of instructions serving these goals, and a number of additional structures such as prerequisites, warnings, advices, illustrations, etc. (Takechi et al. 2003), (Adam, 2001). A response to an How-to question is then the well-formed text portion within the scope of the title that matches the question (Delpech et al. 2007, 2008). Quite often, we have several texts candidates for the response. It is often difficult to make a choice a priori, therefore, we offer the user two approaches which may complement each other:

- the possibility to get a few relevant documents, undergoing a few constraints (e.g. good typography), associated with inter-document navigation tools based on contents (experiments related to user navigation strategies have been carried out using the Navitexte software (<http://panini.u-paris10.fr/jlm/?Start:projets:NaviTexte>)),
- the selection of a main document and the integration into this document of additional information such as missing manners, instruments, durations, etc., coming from the other candidate procedures, possibly in an interactive way, upon user's request. Obviously, this is very challenging, and it may be necessary e.g. to indicate integration situations to the user in case some incoherences arise.

In this paper, besides developing the foundational aspects of procedural texts within the perspective of Action Theory (originating from (Davidson 80), see also: <http://actiontheory.free.fr/>), we develop a first experiment we carried out for textual integration. Text fusion or integration is in general extremely difficult, however, procedural texts being strongly restricted in form and contents make such an experiment possible and of much interest. What is presented here is basically a feasibility study, identifying difficulties, deadlocks, and the linguistic and reasoning resources which may be needed. Evaluation results are preliminary and therefore indicative.

2 Dealing with Procedural Texts

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, programmatic, prescriptive or injunctive.

We have developed a quite detailed analysis of procedural texts from a manual corpus analysis, identifying their main basic components as well as their global structure. For that purpose, we have defined two levels: a segmentation level that basically identifies structures considered as terminal structures (titles, instructions, prerequisites, connectors, etc.) and a grammar level that binds these terminal structures to give a global structure to procedural texts (Delpech et al. 2008) (Fontan et al. 2008). This structure is textual and dedicated only to elements relevant to procedurality. Our study was carried out on a development corpus of 1700 French texts taken from the Web from most of the areas cited above, and extracted from our more global corpus of 8000 texts.

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 we call the explanation structure, which motivates and justifies the goal-instructions structure, viewed as the backbone of procedural texts. A number of these elements are forms of argumentation, they appear to be very useful, sometimes as important as instructions: they provide a strong and essential internal cohesion and coherence to procedural texts. They also indicate, among other things, the consequences on the target goal of an incorrect or incomplete execution of the associated instruction.

A conceptual and operational semantics for procedural texts can be derived from these considerations, where, in fact, what is essential is to model the complexity of a procedure, the risks, and a series of indicators of the difficulty to reach the goal.

As an illustration, consider the cooking recipe extract in Figure 1 (our work is realized on French, with English glosses), describing the preparation of a pizza, arguments are in italics. As the reader can note, warnings and advices operate at various levels, some are general recommendations, while others clearly indicate the risks of a failure, complete or partial, if the instruction is not correctly carried out.

HOW TO MAKE PIZZA

Making Pizza Dough

Pour the warm water in a large mixing bowl. Add the sugar and stir until dissolved *otherwise it may burn when baking.*

Add the yeast and gently stir the mixture until the yeast is dissolved.

Let the mixture sit for 10 minutes to allow the yeast to become "active", *do not wait for more otherwise its effect will decrease.*

The mixture should become foamy at the surface and appear cloudy, and, it will begin to release its familiar, "yeasty" aroma.

Add the salt and olive oil and stir again to combine the ingredients, *make sure they are well mixed otherwise the oil will burn.*

Add 1 cup of flour to the mixture and whisk in until fully dissolved, *otherwise you'll get lumps later .*

Add the second cup of flour and whisk it in until the mixture is smooth, *alternatively, you can add the flour gradually.*

You may need to add a dusting of flour from time to time to reduce the stickiness of the dough as you work it with your hands. Be patient, folding the dough mixture in on itself, over and over again.

Figure 1: Arguments in a procedure

3 A Formal Model for Procedural Texts within Action Theory

Action Theory is a general framework that deals with various facets of actions, including e.g. planning issues, modelling causality and psychological investigations related to the notion of goal. Action theory seems to be a convenient framework to give a conceptual semantics to procedurality, and to one of its main language realizations: procedural texts.

Since argumentation is a major feature of procedural texts it will play a major role in our modelling. We then develop the main facets of a conceptual semantics for procedural texts, with a focus on the measure of the success in reaching a goal, based on arguments. This leads us to investigate measures of instructions complexity and explicitness. The second part of the paper is an experiment showing how complexity and explicitness can be dealt with in order to guarantee a higher success rate to the procedure using textual integration.

3.1 Structure of Arguments

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 usually appear before the support, but they may also appear after, to stress the support. A conclusion may receive several supports, possibly of different natures (advices and warnings): *don't add natural fertilizer, this may attract insects, which will damage your young plants*. Arguments may be more or less strong, they bear in general a certain weight, mostly induced from the words they contain or from their syntactic construction (Anscombe et al. 1981), (Moeschler 1985), (Amgoud et al. 2001). In natural contexts, this weight is somewhat vague and therefore quite difficult to precisely evaluate.

In procedural texts, arguments are associated with instructions or groups of related instructions (that we call instructional compounds). Contrary to works in artificial intelligence, arguments appear here in isolation, they do not attack each other, they simply contribute to the success of the goal the procedure encodes.

3.2 Modelling Procedural Texts

Let G be a goal which can be reached by the sequence of instructions A_i , $i \in [1, n]$. We will not go here into the details of their exact temporal structure, which is often basically sequential. A correct execution of all the instructions guarantees the success of the goal G .

Let us then assume that any A_i is associated with a support S_i (possibly not realized). The absence of an explicit support does not mean that the instruction is not important, but simply that failing to realize it has obvious consequences on the goal G . A procedure is then a temporally ordered sequence of $A_i S_i$.

Let us next associate with every instruction $A_i S_i$ (or, instructional compound, no distinction is made in this paper) a vector (p_i, g_i, d_i, t_i) where:

- p_i is the associated penalty when the user, for a given execution, partly or fully fails to realize A_i ,
- g_i is the gain associated with A_i : there is a gain only in case where A_i is an advice, aimed at improving the quality of G ,
- d_i is the intrinsic difficulty of the instruction A_i ,
- t_i is the degree of explicitness of A_i .

These parameters are obviously quite difficult to elaborate. This is developed in the next subsections.

Let us concentrate here on the realization of the goal G , and on measuring its success, given a certain execution U by a user. It is important to note that failing to properly realize an instruction does not necessarily mean that the goal cannot be reached, but the result of the procedure won't be as nice as it could have been. 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 indeed modulate those 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 if the instruction is not correctly realized (the instruction at stake will be assigned a high penalty), whereas in the former, the goal 'cleaning your leathers' will just be less successful in a certain sense, where there is an explicit gain associated if professional products are used. These parameters and their language realizations are studied in (Fontan et al. 2008).

Given a certain realization by a user, the success of a goal G can be evaluated by the sum of the gains on the one hand, and the sum of the penalties on the other hand. Gains and penalties a priori do not compensate each other: they operate at different levels. Since any A_i is in fact realized successfully to a certain degree by the user for a given an execution U , gains and penalties need to be weighted for that execution, i.e. paired with a success measure for each instruction. Let us introduce respectively μ_i and τ_i for gains and penalties on A_i , each of these weights being included in $[0, 1]$. Then, for a given execution of the goal G , we have:

$$\text{gain}(G) = \sum_{i=1}^n g_i \times \mu_i$$

$$\text{penalty}(G) = \sum_{i=1}^n p_i \times \tau_i$$

3.3 Elaborating penalties and gains

A penalty is equal to 0 when the action is correctly realized. Otherwise it has a positive value. Gains are also positive values and are only associated with advices (optional instructions), therefore, gains are positive only when advice are executed, otherwise they are null.

The difficulty is to elaborate a model that can describe in a simple way the form penalties may have. For that purpose, and as a first experimentation, we introduce a three place vector representing three types of execution levels, to which penalty costs can be associated. The vector represents penalties according to three levels of success for an instructions: (good execution, average, failure). To make it representative of the various importance of the instructions in a procedure, we introduce four levels of importance:

- Essential action, with vector: $(0, N, \text{infinite})$,
- Important action : $(0, 1, N)$,
- Useful action : $(0,0,1)$,
- Optionnal action : $(0,0,0)$.

The value of N remains a parameter that needs to be adjusted experimentally. In case of an essential action, failure entails the goal failure, since the value of the penalty is infinite. The four levels of importance of actions is a parameter quite frequently encountered in experimental psychology, didactics and cognitive ergonomy when users have to evaluate the importance of a task, e.g. in emergency situations.

For each instruction, the level of importance can be estimated from the terms in the instruction (illocutionary force captured in warnings) or be specified by the author of the text, e.g. via appropriate devices like icons, which abound in large public procedural texts.

We can have a similar characterization for gains:

- Important advice: $(0, 1, M)$,
- Useful if done completely: $(0, 0, 1)$,
- No advice $(0, 0, 0)$.

The value of N needs also to be experimentally elaborated.

3.4 Measuring the intrinsic difficulty rate d of an instruction

It is of much interest to be able to measure the inherent complexity or difficulty of an instruction. This notion obviously depends on the reader, and may be somewhat dependent on the other actions around it. Nevertheless, we think that some features introduce in any situation some inherent difficulties. We give here some elements found in procedures identified as introducing some complexity, independently of the domain at stake. Those elements are essentially structures like verbs, PPs and adverbs.

Considering that procedural texts must limit as much as possible the distance between the text and the action (they are oriented towards action, not inferencing), we have identified elements inducing complexity by asking users to indicate and possibly comment every indication in instructions for which they had to make an elaboration (imagine a gesture, identify a tool among a set of closely related tools, evaluate a manner (slowly, cautiously), etc.). The protocol is simple and informal, but nevertheless gives interesting indications for parameters inducing complexity.

The most frequently encountered parameters are, informally:

- presence of 'complex' manners (e. g. *very slowly*), by complex we mean either a manner which is inherently difficult to realize or a manner reinforced by an adverb of intensity,

- technical complexity of the verb or the verb compound used: if most instructions include a verb which is quite simple, some exhibit quite technical verbs, metaphorical uses, or verbs applied to unexpected situations, for which an elaboration is needed. This is for example relatively frequent in cooking (with specialised verbs like 'réserver' which have a different meaning than the standard one), or in do-it-yourself texts written by technicians of the domain.
- duration of execution as specified in the instruction (the longer the more difficult),
- synchronization between actions, in particular in instructional compounds,
- uncommon tools, or uncommon uses of basic tools (*open the box with a sharp knife*),
- presence of evaluation statements or resulting states, for example to indicate the termination of the action (*as soon as the sauce turns brown add flour*).

At the moment, we have lists of a priori complex verbs and manners for each domain collected from our development corpus. The task is to organize these terms by increasing complexity, via the design of adequate scales. Manners are relatively limited and have been ordered manually quite easily. For larger groups of terms, we can also simply form groups of terms classified according to complexity, without introducing scales and accurate order analysis.

Obviously, these observations allow us to introduce a very preliminary measure of complexity, since more empirical measures need to be realized. To be able to have an indicative evaluation, each of the points above counts for 1, independently of its importance or strength in the text. Complexity c therefore ranges from 0 to 6. The complexity rate d_i of instruction i is $c/6$ to keep it in the $[0,1]$ range.

It is important to note that the higher the difficulty d_i is, the more risky the instruction is in terms of failure. Since it is not in general easy to decompose a difficult action into several simpler ones, a strategy to limit risks is to enrich a difficult instruction as much as possible so that all the details are given: this can be measured by the explicitness criteria.

3.5 Measuring the explicitness rate t of an instruction

Explicitness characterizes the degree of accuracy of an instruction. Several marks, independently of the domain, contribute to making more explicit an instruction:

- when appropriate: existence of means or instruments,
- pronominal references as minimal as possible, and predicate argument constructions as comprehensive as possible,
- length of action explicit when appropriate (*stir during 10 minutes*),
- list of items to consider as explicit and low level as possible (*mix the flour with the sugar, eggs and oil*),
- presence of an argument, advice or warning,
- presence of some help elements like images, diagrams, or elaborations, variants, etc.

Those criteria may be dependent on the domain, for example length of an action is very relevant in cooking, somewhat in do-it-yourself, and much less in the society domain. Similarly as for d , each item counts for 1 at the moment, explicitness e therefore ranges from 0 to 6. The explicitness rate is $t_i = e/6$ to keep it in the $[0,1]$ range.

Note also that the higher t_i is, the more chances the instruction has to succeed since it is very explicit and has a lot of details.

Now, if we consider the product $d_i \times (1 - t_i)$, the more it tends towards 1, the higher the risk is for the action to fail. Therefore, when d_i is high, it is also necessary that t_i is high to compensate the difficulty. Given that d_i remains unchanged (if the instruction cannot be simplified), the strategy is then to increase t_i as much as possible.

4 Enhancing explicitness: towards procedure integration

An approach to enhancing t is to add information to those instructions which are judged difficult, wherever possible. This technique is called document integration. It consists in considering a 'reference' document that is gradually enriched from information contained in other closely related documents (i.e. procedures with the same goal). A difficulty is to keep the coherence and cohesion of the document. Information may be added gradually, by instruction or by theme (e.g. instrument, duration), over the whole procedure depending e.g. on the user's needs or requirements.

Integrating information into an already existing document is a very difficult task. The work presented here is essentially a feasibility study, applied to a very restricted type of text: procedural texts, where the text can be segmented into small items (instructions), which have a quite homogeneous and simple style. This allows us to evaluate feasibility, difficulties, needs and to propose some elements of a method. We view document integration as a kind of inference mechanism that includes several aspects, among which:

- evaluation of the relevance of adding or revising a certain type of information, w.r.t. explicitness requirements, e.g. based on Grice's maxims,
- maintenance of the coherence of the text, while keeping integration a monotonic process.

Besides being basically a monotonic process, integration is also modular since it can operate by themes, possibly following different strategies.

Two main types of structures may be integrated: (1) additional information within an instruction or (2) additional instructions between two instructions. We will concentrate here on the first, which is probably the less complex. By adding information, we mean in particular adding a role not specified so far (e.g. instrument, duration) or updating an already existing data (e.g. cheese + mozzarella \rightarrow 'cheese, e.g. mozzarella', the symbol '+' being in this paper the integration operator).

In this section, we focus on texts which are prototypically procedural such as cooking recipes, do-it-yourself, gardening, etc... We propose some criteria to select a reference procedure among a choice of procedures related to the same goal. Then we propose criteria to align instructions judged to be closely related, based on a semantic tagging of instructions. We propose a simple form of similarity ranking and, finally, show how information can be integrated into the reference document, generating well-formed instructions.

4.1 A task-oriented tagging of instruction contents

Let us first introduce the tags we consider, inspired from thematic roles, but which may apply to a variety of syntactic constructs, besides NPs and PPs, such as adverbs or adjectives. These roles are basically action-oriented, they form the prototypical pattern one may associate with an instruction. We defined them from thematic roles and stabilized their definition and scope from a number observations over manual annotations. We have the following main roles: **Themes:** are basically ingredients, tools or objects on which an action is realized (*preheat the oven*) or shapes (*form a ball of dough*). They can also be persons in the social relation domain. **Manner:** indicates how the action is carried out, it does not modify the theme, but indicates the way the instruction must be realized (*serrez la poigne d'un demi tour.*, turn the handle of half a tour).

Means: identifies the entity via which the action is realized, this entity is in general an element that interacts with the theme in some way, e.g. to form a whole with it (*arroser les endives avec jus*, put sauce on the endives).

Instrument: the object used to realize an action, controlled by the agent, and that participates at various degrees to the success of the action.

Goal: refers to the abstract or concrete objective towards which the action is directed.

Result: refers to the expression that characterizes the consequence of the action (*make notes of your expenses to be able to evaluate how much you spend over a week*).

Condition: expression that indicates the constraints under which the action can be realized or should be considered.

Localization: source, fixed or destination applied to time or space, possibly to abstract entities.

We also observed a number of rhetorical relations within the instruction (or instructional compound), but this is not dealt with at the moment. For example, we will not integrate advices or elaborations.

In terms of parsing and tagging, we use the last version of Cordial (http://www.synapse-fr.com/correcteur_orthographe_grammaire.htm), that produces a parse which is appropriate for our needs: constituents are correctly identified and are paired with semantic annotations that we use to assign roles, in conjunction with the lexicon of the predicates of the domain that indicates the roles of arguments with their semantic type. Adjuncts are labelled on the basis of the preposition that introduces them (via our PrepNet database) and the semantic type given by Cordial. For cooking recipe and gardening texts the accuracy is at the moment about 72% which is a moderately good result. Most errors come from incorrect style in the procedure, and lack of domain specific descriptions for prepositions introducing adjuncts.

4.2 Selection of the reference text

There are different strategies for selecting the text, called the reference text (noted as R), that will be enriched from other procedural texts dealing with the same procedure. We assume at this stage that we have texts dealing exactly with the same procedure, possibly with minor variants. The text with the largest number of instructions is selected as the reference. The idea behind is that it is probably the most detailed one (but the correlation is not absolute), and therefore also the simplest one since, for a given task, its decomposition level into instructions is probably the highest. When we have several procedures with approximately the same number of instructions, the decision may be more difficult to make. Two criteria are considered: the quality of the typography, which is an important feature for users, and secondly, the origin

(professionnal site preferred to a blog).

4.3 Aligning instructions

Given a reference text (noted as R) and another procedure (noted as E) used to enrich the former, the first challenge is to align instructions dealing exactly with the same action. This is not necessarily a one to one relation: several instructions in E may correspond to a single one in R or vice versa. Next, instructions in the two texts may be organized differently: we observed slightly different temporal organizations (in particular for minor operations, e.g.: *add salt and pepper* which may appear at several places).

Let us concentrate here on simple situations where two instructions are comparable. Our starting point are instructions tagged as indicated above. Let $A_{R,i}$ be such an instruction in R and $A_{E,j}$ such an instruction from E. The procedure is roughly the following:

(1) aligning verbs: a first parameter, VA, encodes the alignment quality of the main verb of the two instructions. The situations are the following: (1) identical, (2) quasi-synonym, as stated in our domain verb ontology, (3) related via lexical inference (*arroser* (*Theme: la garniture*) (*manner: de sauce*) versus *verser* (*theme: la sauce*) (*fixed-position: sur la garniture*), gloss: baste/pour sauce on the garnish), (4) support construction synonym of the other construction (*mettre un torchon sur / couvrir*, put a dish towel on /cover), and (5) different. A mark is associated with each of these levels, tentatively 4 for (1), 2 for (2) to (4) which are of a similar complexity and 0 for (5).

(2) aligning and comparing arguments: arguments tagged by the same role (or those shown to be identical via lexical inference) are considered. The objects included into those common roles must be closely related: this is evaluated via a conceptual metrics (that basically considers sisters and immediate ancestors at this stage) applied on the domain ontology, cooking in our experiment. Success or failure is measured via a second parameter AA. Furthermore, roles are distributed into 3 categories, the first one having the highest weight. For the cooking domain, we have: (1) theme and temporal localizations (duration or position), (2) spatial localization (fixed, source, destination), (3) means, instrument. The other roles are quite difficult to compare in the cooking domain and are so far kept in level 3.

For each tagged role which is comparable, a mark is given that takes into account the level of the tag (1 to 3 above). The mark we get per aligned argument (k) in instructions $A_{R,i}$ (the instruction from the referent text) and $A_{E,j}$ (the one from the enriching text) is:

$$m_k = f(A_{R,i,k}, A_{E,j,k})$$

where $f(A_{R,i,k}, A_{E,j,k})$ is the metrics that assigns a similarity measure for the argument k, as indicated above. The global rate associated with n arguments successfully aligned between the two instructions is:

$$MA_i = \sum_{i=1}^n \rho_i \times m_i$$

where ρ is the weight that corresponds to the three categories of roles given in the paragraph above. The global alignment rate is then:

$$rate(A_{R,i}, A_{E,j}) = \alpha \times VA_i + \beta \times MA_i/n.$$

where α and β encode respectively the relative importance of verbs and the arguments in the alignment. These parameters need to be adjusted experimentally. At the moment, they are both equal to 1, meaning that the verb and the arguments in total have the same weight in the alignment rate. As can be noted, we do not take into account at this level the possible negative effects of arguments not aligned, since they may contain, in fact, useful information to add to the reference text.

All the instructions in R and E that can potentially align are compared, independently of the temporal relations in the texts. As a result a matrix containing all the alignment rates is produced. For each instruction in R, the best rate, if it is above a certain minimal threshold, is considered and the corresponding instruction in E is considered for information integration. In the case where there are several instructions in E which can be aligned with an instruction in R, then two factors are considered, in the following priority order:

- whether these instructions also have good alignment rates with other instructions in R, in which case, they may be selected for alignment with one of these other instructions,
- whether they temporally coincide in the sequence of instructions in R and E. It should however be noted that, although procedural texts have a strong, but simple, temporal structure, we observed many permutations which make this criteria less strong than it could have been.

At this stage, it is quite difficult to evaluate the accuracy of the alignment strategy since we do not cover all the cases, which may entail errors. However, considering just the cases where we have a one-to-one alignment strategy, over cooking and gardening domains, tested on 14 reference-enriching pairs of texts (with an average of 18 instructions for reference texts), we get a recall of about 74% for a precision of about 90%. Those results are quite encouraging, but we need to analyze the reasons of errors.

We observed more complex cases where alignment must be realized on the basis of patterns, i.e. groups of instructions (possibly discontinuous) and not single, isolated instructions. This is in particular the case with iterative structures (*stir again for one minute*) which are repeated with variants. We have not explored these situations, but it is clear that the detection of such patterns is an important aspect.

4.4 Integrating information

Given two aligned instructions, the next step is to add information to the reference document instruction A_R . The verb in A_R is kept, while the information which is added is basically the tagged elements, which are arguments or adjuncts. All the tagged elements in the enriching document instruction (A_E) are considered, even if A_R already has an element based on the same tag. Let us define an integration operator, noted as '+', composed of a left context element, fragment of A_R , and a right context, fragment of A_E . The result is an enriched instruction. This operation is monotonic: the amount of information resulting from one or more applications of information integration is always increasing.

Adding information follows a strategy with two main aspects:

(1) parameterized adjunction: of missing elements (tags not present in A_R), according to two parameters:

- *information explicitly requested by the user* (if this possibility is offered), such as instruments,
- *information a priori essential to a domain:* for example adding tools, manners and durations is important in cooking to make instructions more explicit, and therefore less difficult to realize. Duration may be less relevant in do-it yourself texts.

(2) integrating already existing information: in A_R , where the three following points are treated so far:

- *disjunctive knowledge integration:* add (soya milk + rice milk) \rightarrow add soya or rice milk.
- *exemplification:* add (cheese + mozzarella) \rightarrow add cheese, for example mozzarella.
- *blocking ill-formed configurations:* for example, in the case of incorporation: (butter the bread + with butter \rightarrow butter the bread) 'with butter' is not included because the noun butter is already incorporated into the verb.

Let us now consider the surface realization of the resulting enriched instruction. For the first aspect above, this process starts from the tagged elements so that a correct linguistic realization can be produced. We do not just copy and paste the additional information to A_R , but, based on the annotated elements, we introduce insertion rules, that indicate where the information must be inserted into the proposition. For example, [stir manner: gently duration: for 4 minutes] + [theme: the sauce] must be realized as:

[stir manner: gently theme: the sauce duration: for 4 minutes]

where the theme is inserted between the manner and the duration. The resulting instruction is then:

Stir gently the sauce during 4 minutes.

This is realized by an insertion rule that says:

Verb (manner:X duration:Y) + (theme:Z) \rightarrow Verb manner:X theme:Z duration:Y.

From the needs in terms of integration and from the observation of the structure of instructions, we have defined at the moment 19 insertion rules. These are now being generalized so that more generic situations can be captured. In particular, we rephrased these rules in terms well-formedness conditions essentially based on thematic linear precedence, a widely used approach in language generation:

instrument < duration, meaning that any instrument always precedes a duration expression. Similarly, we have:

theme < instrument, duration, localization.

In this approach, we keep the 'base' form used in most instructions since there is almost never any syntactic alternation. These precedence specifications are very close to thematic role hierarchies, except that we deal here also with a number of adjuncts (goals, manners, etc.) that may occupy a large number of positions in the instruction. These adjuncts are however subject to constraints, e.g.:

goal, condition < manner, means, result.

Finally, we apply the 'heavy-NP shift' movement when the argument is large, for example a long theme will be moved at the end of the instruction.

At the visualization level, integrated information is underlined so that the user knows that it has been imported, and may consider it with some care or may get the original enriching instruction.

5 Conclusion

In this paper, we have presented a model for the structure of procedural texts based on Action Theory, where the success of a goal is evaluated via several parameters. We have in particular investigated the notions of complexity and explicitness of a procedure, and proposed an approach to improve the success rate of a procedure via the integration of additional knowledge from other procedures into a reference procedure. This latter point has entailed the introduction of a semantic tagging, task-oriented, of instructions, the development of an alignment strategy and a simple language realization strategy.

Obviously what we propose here is basically exploratory, to evaluate complexity and feasibility. Document integration is a very hard task. Although restricting ourselves to very prototypical procedural texts does help to propose solutions, it is clear that this work needs a lots of adjustments and testing.

Acknowledgements This work was supported by the French ANR research programme,

under the TextCoop project.

References

- [1] Adam, J.M., *Types de Textes ou genres de Discours? Comment Classer les Textes qui Disent De et Comment Faire*, Langages, 141, pp. 10-27, 2001.
- [2] Amgoud, L., Bonnefon, J.F., Prade, H., *An Argumentation-based Approach to Multiple Criteria Decision*, in 8th European Conference on Symbolic and Quantitative Approaches to Reasoning with Uncertainty, ECSQARU'2005, Barcelona, 2005.
- [3] Amgoud, L., Parsons, S., Maudet, N., *Arguments, Dialogue, and Negotiation*, in: 14th European Conference on Artificial Intelligence, Berlin, 2001.
- [4] Anscombre, J.-Cl. Ducrot, O., *Interrogation et Argumentation*, in Langue française, no 52, L'interrogation, 5 - 22, 1981.
- [5] Aouladomar, F., Towards Answering Procedural Questions. Dans : KRAQ'05 - IJCAI workshop, Edinburgh, F. Benamara, P. Saint-Dizier (Eds.), p. 24-36, juillet 2005.
- [6] Davidson, D., *Essays on Actions and Events*, Oxford: Oxford University Press, 1980.
- [7] Delin, J., Hartley, A., Paris, C., Scott, D., Vander Linden, K., *Expressing Procedural Relationships in Multilingual Instructions*, Proceedings of the Seventh International Workshop on Natural Language Generation, USA, 1994.
- [8] Delpech, E., Saint-Dizier, P., A Two-Level Strategy for Parsing Procedural Texts, proc. VSST07, Marrackech, 2007.
- [9] Delpech, E., Saint-Dizier, P., Investigating the Structure of Procedural Texts for Answering How-to Questions, LREC 2008, Marrakech.
- [10] Fontan, L., Saint-Dizier, P., Analyzing the explanation structure of procedural texts: dealing with Advices and Warnings, STEP, semantics in text processing, College publications, J. Bos and R. delmonte (edts), 2008.
- [11] Moschler, J., *Argumentation et Conversation, Eléments pour une Analyse Pragmatique du Discours*, Hatier - Crédif, 1985.
- [12] Takechi, M., Tokunaga, T., Matsumoto, Y., Tanaka, H., *Feature Selection in Categorizing Procedural Expressions*, The Sixth International Workshop on Information Retrieval with Asian Languages (IRAL2003), pp.49-56, 2003.