The KOMODO System: getting Recommendations on how to realize an action via Question-Answering

Marc Canitrot, Thomas de Filippo, Pierre-Yves Roger
Prometil
42 Avenue du Gal Decroute
31100 Toulouse, France
m.canitrot@prometil.com

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

Abstract

In this paper, we present the KOMODO system which is designed to provide tips (advice and warnings) on the way to realize a task from user queries. Information is extracted from web services. We present the different steps of the system: web page selection, ranking and cleaning, extraction of warnings and advice, relevance analysis and contextualization, and production of a response. The different language processing steps are presented together with an evaluation of the results. The system is fully tested and a demonstration will be made if possible during the presentation.

1 Introduction

Given a few key-words, such as *put up wall paper*, Komodo is more than a question-answering system: it provides a series of recommendations or hints to realize this task. these are given under the form of advice and warnings, together with a few explanations. These recommendations are extracted from various web pages which are in general procedures describing how to realize that task, selected as relevant and reliable. Therefore, Komodo explains how to do something, but it offers more by compiling advice and warnings from various candidate pages.

Getting advice, hints and warnings is of much importance for different types of unexperimented users who have a task to realize but want to know more about it before really starting. Obviously, it is often possible to find a web page that explains, on the basis of a procedure, how to realize this task. However, quite frequently, these are not so rich in recommendations, they basically describe the different steps of the work under the form of instructions to follow. Psychological experiments have in fact shown that, besides instructions given in procedures, 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 or to guess. Komodo is aimed to fill in this kind of gap.

Procedures are designed to guide people step by step to realize precise tasks (Delin et ali. 1994) (Takechi et ali. 2003). They 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 (Rosner et ali. 1992). The How-to question answering aspect was developed in (Yin 2004), Aouladomar et al. 2005) and (Delpech et al. 2008).

Procedures abound in a number of domains, from apparently simple cooking recipes to large maintenance manuals. They include documents as diverse as teaching texts, medical notices, social behavior recommendations, directions for use, assembly notices, do-it-yourself notices, itinerary guides, 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 targeted user, and on traditions associated with a given domain. Procedural texts can be regulatory, procedural, programmatory, prescriptive or injunctive.

We have developed a quite detailed analysis of procedural texts, identifying their main basic components as well as their global structure. 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 back-

Proceedings of the KRAQ11 Workshop, pages 1–9,
Chiang Mai, Thailand, November 12, 2011.

bone 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 (Anscombre et al. 1981).

An important aspect of this project at a cognitive level 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 (VanderLinden 1003) and (2) then to be able to provide a set recommendations that would guide users when the perform a task, more or less independently of the precise procedure they follow.

We have already studied the instructional aspects of procedural texts and implemented quite an efficient prototype within the <TextCoop> project that tags texts with dedicated XML tags. The Dislog language (Discourse in Logic) allows the specification of rules that describe the various forms discourse structures can take (Anonymous 2011). In this paper, after a general presentation of the explanation structure in procedures, we focus on the form warnings and advice take in procedures and how these can be extracted using Dislog. We then survey the main steps of the Komodo system from the query to the production of a series of recommendations (advice and warnings) meant to inform a user who wants to realize a certain taks. The Komodo system is now operational, if accepted a demo will be given during the talk. At this moment, it basically works for French, a transposition to English is planned. Some elements of procedural text processing, in particular various forms of explanations have been also tested for Thai.

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.

From our development corpus (1700 web texts of 1 to 3 pages of raw text from 24 different domains, large public and professional), we established a classification of the different forms explanations may take. Basically, the explanation structure is meant to guide the user in two ways: (1) by making sure that he will effectively realize actions as they are specified, via arguments (Amgoud et ali. 2005), (Amgoud et ali. 2001)) such as threats, rewards, advices and warnings which are 'coercitive' in a certain sense, and (2) help considerations such as evaluation of work realized so far and encouragements of different kinds.

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 (Moschler 1985) (Bourse et al 2011). This structure has a strong causal structure (Talmy 2001). 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. 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. Roughly, we have about 25% of instructions which have recommendations in do-it-yourself texts, and up to 60% in social procedural texts. Argumentation structures are relatively general to an applications domain, while facilitation structures are much more specific to the text and the targeted audiences.

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. The following example is typical of what is usually found:

[$_{instructional\ compound}$
[$_{Goal}$ To clean leather armchairs,]
[$_{argument:advice}$
[$_{instruction}$ choose specialized products dedicated to furniture,
[$_{instruction}$ and prefer them colorless]],
[$_{advicesupport}$ 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

Argument detection and analysis has been developed in the <TextCoop> project and presented in previous papers. Lets us summarize the main results here for the sake of understanding.

3.1 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. 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 262 arguments. 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.

3.2 Processing Advice

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 preceeding a support of type advice is a conclusion.

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 (favor, 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 of 66 texts containing 240 manually identified advices. 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.

The structures of English are quite similar. A short example of an annotated text is given in Fig. 1 below.

4 Constructing an repository of advice and warning for a task: capturing the domain know-how

Besides studying the textual structure of procedural texts and responding to How-to questions from the analysis of these texts, a major challenge of this work is the construction of a **domain know-how knowledge base**, which is probably quite basic, but which could be subject to interesting generalizations. This domain know-how is essentially composed of recommendations under the form of advice and warnings related to the execution of the task at stake, possibly coupled with a few additional explanations (e.g. illustrations, reformulations, etc.)

There are repositories of advice organized by sector of activity available on the Web (e.g. http://www.conseils-gratuit.com). These are realized manually: most of these advice come from hints sent by readers of these pages. These repositories contain in general simple advice and also small procedures which are hints to better realize a certain task. Automatically constructing such repositories is of much interest but also a major challenge in advanced question-answering. We will focus here on textual information, but it is clear that images and possibly videos should complement the text.

Let us now present the different steps of the task.

4.1 Overview of the main steps

The main steps are:

- getting the user query and submitting it to a search engine (Exalead in our case),
- processing the returned links: elimination of irrelevant links, ads, etc.
- downloading the first 50 pages returned by Exalead which have not been filtered out at the previous stage,
- sorting the returned pages by decreasing procedural quality in order to eliminate ill-formed pages, ads, poorly realized pages, etc., approximately the 20 first ones are kept. This decision is based on a relevance metrics we have elaborated.
- cleaning those 20 web pages: keeping only the textual information and some typography elements,
- parsing these pages at discourse level using the <TextCoop> system, the result is an XML tagging of the instructional aspects, prerequisites, advice and warnings based on the patterns given above and a few explanation forms.
- in order to get the best and the most relevant advice and warnings, only advice and warnings from the 'best' paragraphs (according to our relevance metrics) are extracted possibly with their context.
- construction of a response web page and a know-how repository (query - set of related advice and warnings) for future similar queries.

The following domains have been investigated and are addressed in the Komodo system: house, cooking, garden, computer, do it yourself, animals, beauty. In the next subsections the above steps are developed, in particular those related to language processing and question-answering. The

$[_{procedure}$
$[_{title}$ 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

work has been realized on French: English glosses are given here for the sake of readability. The investigations reported below have been realized from a development corpus of 1700 procedural texts, from 24 different domains.

The kernel of the system, <TextCoop> and the language resources are realized in SWI Prolog. Interfaces, web page collection and result construction are realized in Java, returned pages are processed 'in parallel' via a multithread implementation to enhance efficiency.

4.2 Downloading relevant procedural texts

The user query, which refers to a task to be realized, is submitted to the Exalead search engine, which handles query enrichment if needed. Contrary to Google, Exalead returns links which directly points to web pages. These links are analyzed in order to keep only those which correspond to professional sites. Redundant addresses are also eliminated. This first step of filtering does depend on the domain. For do-it-yourself, it is easy to access the main platforms that explain how to realize an action. This is not so easy e.g. for social recommendations where blogs are the richest sources in terms of advice. In this latter case, a list of 'hot' links is construcxted and constantly updated.

A total of 50 links are kept after this first filtering. Then, those pages which are directly refered to by these links are downloaded in parallel in order to limit loading delays.

These pages are then 'cleaned'. By this term, we mean keeping only the elements which are useful for our purpose: the text and a limited number of typographic marks. Our cleaning programme has a set of parameters that describe the typographic symbols (and possibly their context) that we wish to keep. The others are eliminated. This operations is crucial to eliminate ads, summaries, etc. We also have a list of 'stop-terms' which provoke the elimination of the sentence in which they occur (e.g. click here to get a free coupon). This process eliminates in general between 20 and 60page contents.

The next step aims at identifying among those 50 pages those which are really procedural. Indeed a number of these pages often turn out to be of little or no interest. In addition, texts which are really short (less than 80 words) are excluded a priori: it is unlikely that they contain any advice or warnings. From the inspection of 280 texts in our corpus, we defined a simple metrics that can detect the procedural level of of a text. This metrics has been elaborated by contrasting regular texts with procedural ones. Procedural texts are much richer in terms of action verbs, in the infinitive or imperative form (these are morphologically different in French). They also have a large number of typographic which is not often encountered in regular texts:

mark	procedure	regular text
action verbs	85%	52%
imperative forms	44%	17%
infinitive forms	40%	25%
typographic marks	17%	2%

Verb ratios are computed as follows: number of imperative verbs w.r.t. total number of verbs found. Typographic marks rate: number of marks w.r.t. total number of words in the text. Html marks count for one mark per html tag.

As can be seen the contrast is quite high between regular and procedural texts. Furthermore, we are interested in collecting the best texts, i.e. those with a rich typographic mark, with very standard verb forms in imperative or infirnitive forms. We can then use this metrics to sort those texts considering the most procedural ones first. The metrics is defined as follows:
$rate = NV/TV + 2x(TM/NW)$.
where NV is the total number of verbs in the imperative or infinitive form, TV is the total number

of verbs found in the text. TM is the total number of typographic marks found (which are related to procedures) and NW is the total number of words in the text. To have a better taking into account of the importance of typography, a weight of 2 is introduced for that parameter. This metrics is very simple, but seems to be sufficient for the task at stake.

Analysing the results of applying the metrics to gardening, do-it-yourself, health care and cooking, we get the following results:

text rank	metrics ratio	evaluation
1-5	> 0.80	highly procedural
6 - 10	0.80 to 0.70	very procedural
10-15	0.69 to 0.65	good
15-20	0.64 to 0.62	good
20-30	0.61 to 0.54	average

It seems therefore that a threshold of 0.62 for a text would guarantee that the text considered is of a good procedural quality. In the above experiment, this means keeping about 20 texts, but this number may be higher or smaller depending on the query and the domain.

4.3 Processing relevant procedures with <TextCoop>

Our aim is to provide users with relevant advice and warnings related to their query. In fact, a closer look at the selected procedural texts shows that they indeed contain warnings and advice but a number of of them turn out to be irrelevant w.r.t. the user query. For example, a text may contain an introduction with general purpose advice. Providing these advice in the response would mean some analysis and sorting work for the reader which may not feel so confident about the overall result.

The relevant advice and warnings are all in the text sections or paragraphs which indeed describe the actions to undertake to realize the procedure. To select these text portions, we use another simple metrics that computes the number of verb domains w.r.t. the total number of words. Text portions above a certain threshold (which is a domain dependent parameter) are kept and processed by <TextCoop>. The set of verbs typical of the domain is constructed via the analysis of quite a large number of procedures in that domain. The number of verbs associated with a domain turn out to be much higher than expected. For the gardening domain, we identified about 500 verbs, which is very high. To get this set of verbs, about 1250 procedures have been inspected. This figure is very high considering the number of available texts in gardening in French. For the gardening domain, the threshold above which paragraphs are relevant for warnings and advice extraction is O.O5. In that domain relevant verb frequency ranges from O.O2 to O.20. The threshold of O.O5 allows the extraction of about 64% of the total text, which is a really efficient filtering.

These text portions are then submitted for discourse processing to <TextCoop>. This system identifies: titles and subtitles, prerequisites, instructions, illustrations, goal expressions and warnings and advice. Assuming that 20 texts are selected and keeping only the relevant paragraphs, between 1 and 7 warnings or advice are extracted by text, with an average of 3 per text. This means about 60 warnings or advice (almost in equal proportion) are extracted over the 20 texts.

This figure, however, varies greatly from a domain to another. For 20 texts, we have:

domain	nb of words	nb of A/W
DIY	1100	60
cooking	800	22
gardening	1450	54
health care	1950	38
computer	1550	33

nb of words is the total number of words of all the 20 procedures.

As can be seen, the DIY domain is very rich in advice and warnings, health care is very verbose and advice are not always very easy to identify. The computer domain is for specialists: it therefore contains less advice or warnings.

4.4 Response construction

At this stage, there are still three main problems to resolve:

- redundancy elimination: the best advice or warning extracted among those which are almost synonyms should be kept,
- contextualisation: quite frequently an advice or a warning cannot be understood in isolation: it is therefore necessary to introduce some form of contextual information, e.g. for an easy pronominal or event reference resolution,

- relevance: all advice and warnings are a priori relevant, however some are more crucial than others: sorting them by decreasing order of importance would be helpful for the user, in particular when ther eare many.

Redundancy elimination is a very difficult to resolve efficiently, since it requires a lot of domain knowledge, and lexical and textual inference to detect equivalent information. To resolve this difficulty, we organize warnings and advice per web page. In that case we have between one and 7 advice or warning per page, which is not so much, and redundancy is less visible because it occurs over different pages. In fact, then, this redundancy may be useful as a way to insist e.g. on a precaution to take.

This approach of displaying the response per web page also avoids the crucial problem of sorting advice and warnings by decreasing importance. However, to be cooperative with the user, we evaluate the strength of those statements, via the analysis of adverbs and injunctive forms, as described in the patterns above, and display each advice or warning with a logo that indicates its importance a priori.

Finally, contextualization is dealt with as follows. Warnings and advice which do not contain any kind of reference are displayed alone. The others are displayed with the instruction that immediately precedes them. Our observations show that this adequately resolves the context problem in about 90% of the cases.

An example is given in a screenshot in fig 2 (last page of this document).

5 Conclusion

The work presented here describes the different stpes of the Komodo project, which is designed to provide users with a set of recommendations under the form of advice and warnings useful to know before starting any task, a priori described by a procedure.

The system is interactive (a demo will be given if accepted): from his query a user gets a series of recommendations and links to the relevant pages in case he wants to know more about them or access the whole procedure. The kernel of the system, <TextCoop> and the language resources are realized in SWI Prolog. Interfaces, web page collection and result construction are realized in Java, returned pages are processed 'in parallel' via a multithread implementation to enhance efficiency. In our still experimental version a query is fully processed in an average of 4 seconds, which is still a bit high. Collecting web pages takes about 1.5 seconds and linguistic processing about 2 seconds.

Pairs query - responses are stored a database, called know-how database. This is useful for frequently asked questions. However, to avoid having outdated data, a robot updates reponses regularly so that at least once a week each query has an updated set of recommendations.

Acknowledgements This work is supported by the French ANR project LELIE and by the Franco-Indian cooperation framework IFCPAR under ref. ICT 4200-1.

References

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

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

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

Bourse, S., Saint-Dizier, P., The grammar of explanation structures in technical texts, Sintagma, vol. 27, 2011.

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

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

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

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, pp. 61-70, Maine, USA, 1994.

Delpech, E., Murguia, E., Saint-Dizier, P., *A Two-Level Strategy for Parsing Procedural Texts*, VSST07, Marrakech, October 2007.

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

Gardent, C., Discourse tree adjoining grammars, report nb. 89, Univ. Saarlandes, Saarbrucken, 1997.

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

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

Rosner, D., Stede, M., *Customizing RST for the Automatic Production of Technical Manuals*, in R. Dale, E. Hovy, D. Rosner and O. Stock eds., Aspects of Automated Natural Language Generation, Lecture Notes in Artificial Intelligence, pp. 199-214, Springler-Verlag, 1992.

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.

Talmy, L., Towards a Cognitive Semantics, vol. 1 and 2, MIT Press, 2001.

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

Webber, B., D-LTAG: extending lexicalized TAGs to Discourse, Cognitive Science 28, pp. 751-779, Elsevier, 2004.

Yin, L., *Topic Analysis and Answering Procedural Questions*, Information Technology Research Institute Technical Report Series, ITRI-04-14, University of Brighton, UK, 2004.

Zuckerman, I., McConachy, R., Korb, K., Using Argumentation Startegies in Automatic Argument Generation, INLG 2000, Israel.

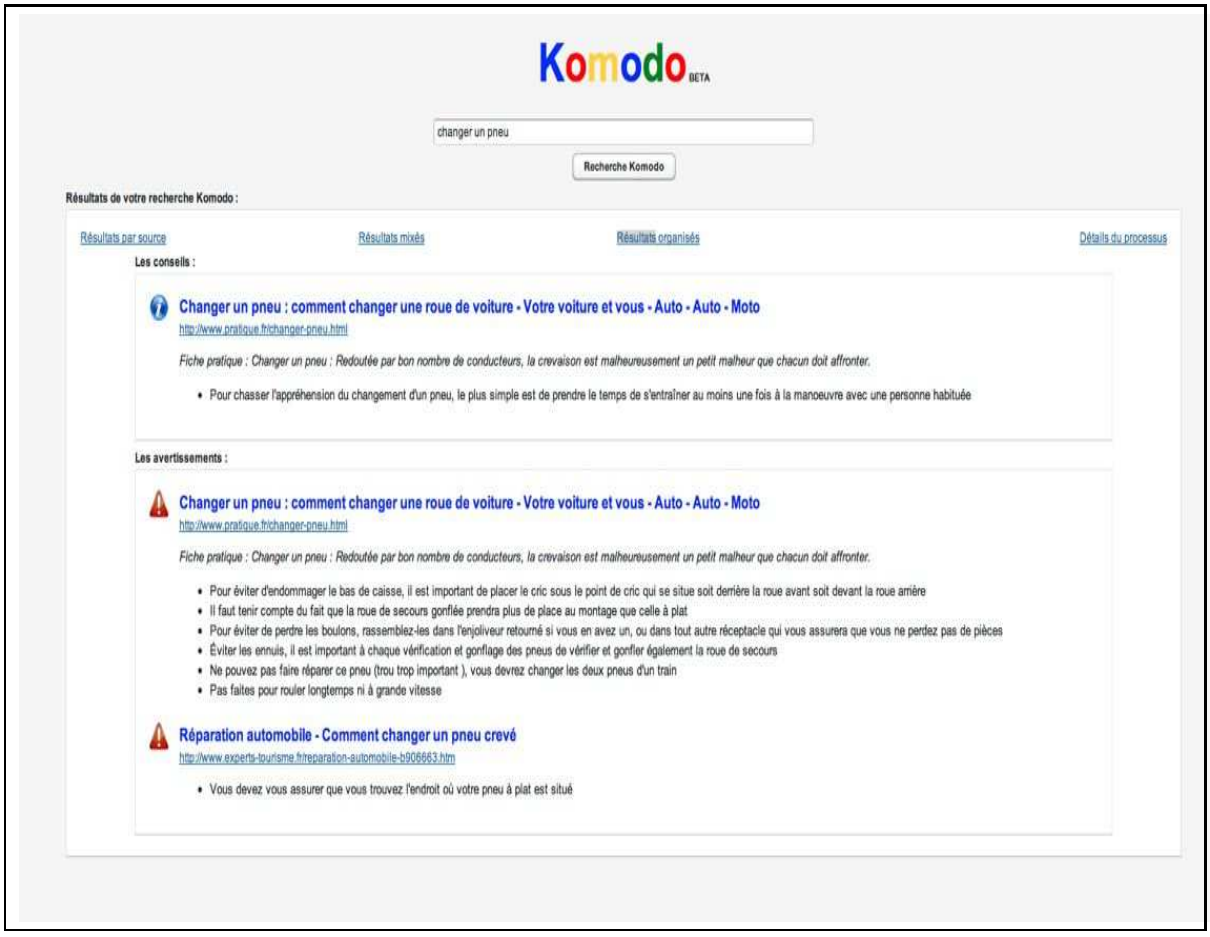

Figure 2: Screenshot of Komodo (experimental)