Building and Exploiting Ontologies for an Automobile Project Memory

Joanna Golebiowska [1 2], Rose Dieng-Kuntz [1], Olivier Corby [1], Didier Mousseau [2]

1 INRIA, ACACIA Project, 2004 route des Lucioles, BP 93, 06902 Sophia-Antipolis Cedex, France

2 RENAULT, TPZ D12 138, DTSI/DTPU/KMPD, sce 18820 860 quai de Stalingrad, 92109 Boulogne, France

E-mail: {Joanna.Golebiowska, Rose.Dieng, Olivier.Corby}@sophia.inria.fr

Abstract

This paper describes SAMOVAR (Systems Analysis of Modelling and Validation of Renault Automobiles), aiming at preserving and exploiting the memory of past projects in automobile design (in particular the memory of the problems encountered during a project) so as to exploit them in new projects. SAMOVAR relies on (1) the building of ontologies (in particular, thanks to the use of a linguistic tool on a textual corpus in order to enrich a core ontology in a semi-automatic way), (2) the «semantic» annotations of the descriptions of problems relatively to these ontologies, (3) the formalisation of the ontologies and annotations in RDF(S) so as to integrate in SAMOVAR the tool CORESE that enables an ontology-guided search in the base of the problem descriptions.

1 Introduction

How to preserve and exploit the memory of past projects in automobile design (in particular the memory of the problems encountered during a project) so as to exploit them in new projects? The role of ontologies for knowledge management is more and more. They can play an important role for building a project memory, that is a specific kind of corporate memory [9,10]. Several researchers aim at proposing a methodology for building such ontologies, possibly from textual information sources [2]. Such a methodological framework is interesting for us, as there are several heterogeneous sources of information inside the company: different databases, official references, problem management systems and other specific bases in the departments; moreover, in addition to basic data which can be processed by traditional means, some bases contain important textual data.

After detailing our problematic and the concrete problem to be solved at Renault, we will present the approach adopted for SAMOVAR. Then we will detail our techniques for building the SAMOVAR ontologies, relying on both manual construction and semi-automatic construction thanks to the application of heuristic rules on the output of a linguistic tool applied on a textual corpus stemming from textual comments of a database. Then we will explain their exploitation and the use of the CORESE (Conceptual Resource Search Engine) tool [8] for information retrieval about the descriptions of past problems encountered in vehicle projects. We will generalize our approach so as to propose a method for building a project memory in the framework of any complex system design. In our conclusion, we will compare SAMOVAR to related work.

K-CAP'01, October 22-23, 2001, Victoria, British Columbia, Canada.

2 The problematic

The field of SAMOVAR is the process of prototype validation during a vehicle project. This process is intrinsically complex and raises many problems. These problems frequently slow down the cycle due to the necessity of repeating validations: so, it increases both the delays and the costs of such projects.

A close observation of validation shows that part of the failure is due to loss of information and of experience gained. The objective of SAMOVAR is to improve the exploitation of this information and make it available for future projects. Useful data exist in the form of text. Therefore it is necessary to find suitable techniques and tools, such as for example linguistic techniques for exploiting the knowledge underlying such texts.

2.1 Context

The product development cycle of an automobile is made of numerous repetitive sub-cycles (design/ development / validation) - of short or long duration. The whole cycle is punctuated by milestones and prototype waves which mark the production of successive models and prototypes, more or less complex. During a vehicle project, validations are carried out: the testing department checks that the component-parts or the functions satisfy the requirements of the product specifications.

Thus, the quality of smoothness of the dashboard, the noise of a car door being shut, the behaviour of the car on cobble stones, or even its resistance to high or low temperatures are tested. These validations are spread throughout the vehicle project and done successively by the testing department, starting from the most elementary functions till the final synthesis test. The project begins with tests related to the engineering center according to the parts validated and ends with tests on performance, speed and crash.

These project validation phases often reveal discrepancies with respect to the specifications. From detection of a problem to its resolution, such problems are documented in a unique data management system called Problem Management System (PMS). This system uses a database including the information needed for the process of problem management: especially information on the actors involved in the project and above all, the descriptions and comments on the problems that arose.

2.2 Interest of exploiting the Problem Management System

The appearance of problems increases the additional costs and the project duration. Therefore solutions have been thought out. One possible solution would be to exploit the information contained in the PMS in order to use the PMS not only as a problem management system but also as a source of information.

The PMS can be considered as a huge source of information, thanks to the textual fields of the base which are particularly rich and under-exploited. The actors involved in the automobile design project express themselves freely for describing the problems detected, as well as the various solutions proposed, or the constraints for carrying out such or such solution. This base can therefore be considered as archives or even as constituting (a part of) the memory of a project, more precisely the memory of the problems encountered during the project.

Furthermore, in the company, there are other information sources, such as the official corporate referential or the numerous local bases of the testing department. It would be useful to exploit this information with the contents of the PMS.

Therefore our aim is to propose a means of retrieving, structuring and making reusable this wide quantity of information for the same project or for the other projects. The participants of current projects have expressed needs related to information search and retrieval useful during the validation phases. Their needs concerned especially the retrieval of similar incidents, detection of any correlation or dependency with other incidents and so the reuse of existing solutions within the same or even a different project.

Some pieces of information are relatively simple to retrieve. However, this is not the case for the textual data of PMS. The vocabulary used by the project participants in such comments is broad and varied: a given term (existing in the corporate official referential) frequently has different designations according to the department or even the phase reached in the project. Therefore, our objective was to detect a suitable semantic term, to classify it according to the validation process and to link it with all the variations encountered. So, we needed to extract the main terms of the domain (and the relations between them if possible) and to structure them in our ontology.

2.3 SAMOVAR's approach

A synthesis of tools dedicated to the extraction of terms and of relations from textual corpora is proposed in [*3*]. Several linguistic tools exist to extract candidate terms: Lexter [5], Nomino[1] , Ana [11] [12]. With regard to the acquisition of semantic relations, several approaches enable to acquire them (based on the exploitation of syntactical contexts : [17], or the use of the lexical-syntactical patterns : [18], [19]). Few tools are offered such as Coatis [14] for causal relationships, Cameleon [28] [27] for hyponymy and meronymy relations.

The approach of SAMOVAR consists of structuring the knowledge contained in the PMS textual fields describing problems, and of enabling the user to carry out searches with the aim of finding similar problem-descriptions.

[1] http://www.ling.uqam.ca/nomino

As a starting point, we took directly the exploitable sources (i.e. the different databases of the company), and then we built up several ontologies offering different viewpoints on the validation process: problems, projects, services, components (i.e. parts). After having primed our base manually, we completed it progressively, with the elements from the PMS textual data using Natural Language Processing (NLP) tools – in particular, Nomino that was chosen as term extractor for availability reasons. This stage is automatic, however the support of an expert is necessary throughout the process. Then we annotated the problem descriptions automatically with instances of concepts of the ontologies. Finally we facilitated the access to the base of problem-descriptions thanks to the formalization in RDF(S) of the ontologies and of the annotations, enabling the use of the CORESE tool [8] to carry out ontology-guided searches through the such annotated base of problem-descriptions. The whole SAMOVAR approach is summarized in figure 6.

3 SAMOVAR ontologies

The SAMOVAR base is a multicomponent ontology composed of 4 ontologies, each dedicated to the description of a precise field :

- *Component* Ontology: it is based on the official company referential, corresponding to the functional segmentation of a vehicle into sub-components;
- *Problem* Ontology: it contains the problem types and it is built up semi-automatically from a manually-activated core from textual fields taken from the problem management system;
- *Service* Ontology: it corresponds to the services cross-referenced with the company organization (management and profession) and it is supplemented by PMS information. This ontology gives an additional overall point of view on the problems;
- *Project* Ontology: it reflects the structure of a project and it is made up of knowledge acquired during a project vehicle, according to the interviews carried out with different actors on the project.

Each ontology is a n-leveled hierarchy of concepts linked by the specialization link.

All the ontologies (or Samovar ontology components), apart from the Problem ontology, were built automatically, by an extraction of the PMS data base.

Remark: Instead of building several interconnected ontologies, we could have built one single ontology organized through several sub-ontologies. We chose to distinguish the different ontologies in order to enable their possible reuse independently from one another. The various constituents of our ontology correspond to the possible points of view concerning the the validations process. Even though, in fact, they constitute a single object, it is important to protect the possibility of various points of approach for validations.

3.1 Construction of the ontologies

The ontologies were built through two phases according to the data type and the means involved:

- a first extraction of the information contained in data bases,
- a second extraction, with specific techniques and tools for discovering the information « hidden » in texts.

The core of our ontology was primed manually, thanks to elements stemming from existing bases (see figure 1).

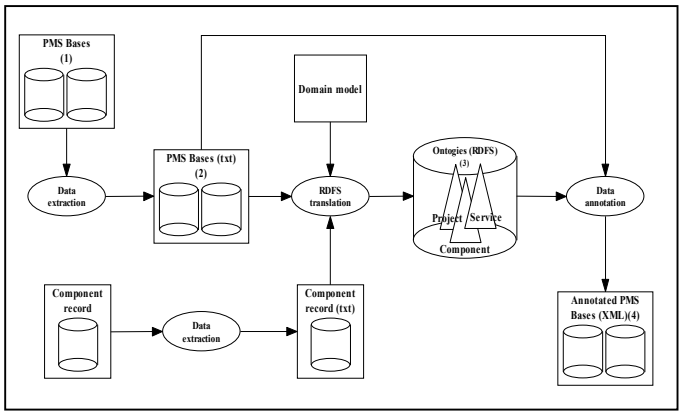

Figure 1: *Construction of ontologies for SAMOVAR – first data extraction*

A first extraction of the initial data (1) supplied a textual format (2) which was then translated in the form of an ontology, by respecting the RDFS format (as expected by CORESE). In parallel, another extraction was made from the Component referential in order to complete the previous data with additional information. In this way Component, Service and Project Ontologies are obtained, our ontological base (3). Then this base was used to annotate the data with the terms designating concepts of the ontologies. Thus we obtained the initial base annotated with annotations related to the concepts of the ontologies (4).

A second process deals with the textual data (the final goal being to enrich the result of the first extraction with the information stemming from the texts). To be able to deal with a text we needed a minimun of tools adapted to this type of data – the Natural Language Processing tools. We wanted to avoid heavy treatments requiring building the entire chain of treatment, for this reason we've reduced NLP treatments to the candidates terms.

This process exploits the output obtained after application of the linguistic tool Nomino on the textual corpus stemming from the textual comments contained in the problem management system (PMS). Nomino is a tool for extraction of nominal groups from a representative corpus in a domain. Nomino takes as input a textual corpus and produces as output a set of « lexicons » - lists of nouns, nominal complex units (NCU), additional nominal complex units (ANCU), verbs, adjectives, adverbs. The (A)NCU corresponds to the prepositional groups (PG) or the nominal groups (NG). The lexicons of the NCU are accessible in the form of graphs which illustrate the existing dependencies for a PG or a NG.

Then, we exploited the lexicons and the graphs produced by Nomino, in order to :

- detect the significant terms (i.e. corresponding to important validation points in the automobile design validation process),
- enrich the *Problem* ontology by means of the Nomino graphs, by exploiting the regularity of their structures.

3.1.1 Detection of significant terms

Firstly, we analysed the lexicons produced by Nomino in order to discover the most frequent terms, likely to be the most representative terms of the domain : *wiring, assembling, pipe, attachment, centring, component, installation, conformity, branch, hole, clip, screw, contact, maintains, tightening, paw, position, geometry, connecting*.

These structured terms allowed us to set up the *Problem* ontology. The initial structuring of this ontology was based on discussions with the experts. Figure 2 shows an extract of this *Problem* ontology.

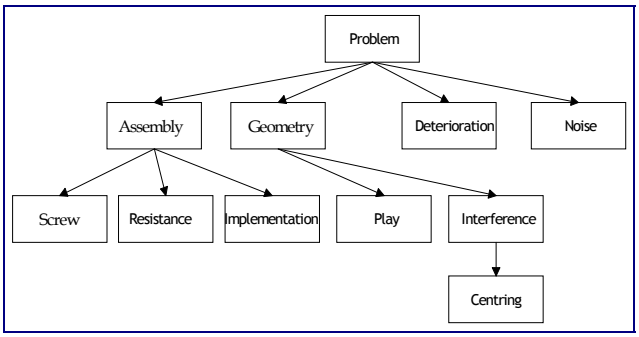

Figure 2: ***Extract of the Problem Ontology***

The terms selected for the bootstrap were those which are exploitable as semantic clues for a problem type: for example, a problem of `Centring` can be discovered thanks to the presence of such clues as `«indexage», coaxiality, «entraxe»`, etc.

Indeed the Nomino outputs can be sorted by frequence numbers. The most frequent words can be considered as relevant fr the processed domain and we exploit them as clues for the *Problem* ontology bootstrap.

The validity of the terms (i.e. the candidate terms for the bootstrap, and the clues exploited to find them) was confirmed with support of the experts.

Once the bootstrap of ontology was constituted, it needed to be enriched. For this purpose, we used the prepositional groups stemming from Nomino.

The extraction process implemented so far was applied to the enrichement of Problem Ontology. The other ontologies were constructed automatically from different data base fields, with help of interviews information. That is why most examples presented below concern only Problem Ontology. In the second phase we intend to reuse this method to enrich the Component ontology, notably to extract supplementary terminologie (synonyms, etc.)

3.1.2 Enrichment of the Problem ontology

Besides nouns, Nomino produces nominal and prepositional groups. We exploited the structures of the most frequent cases produced by Nomino.

The manual analysis of these NCU was performed by studying each Nomino output carefully so as to find some regularities in the NCU obtained by Nomino. This manual analysis, carried out with the support of the expert, supplied the structures which we exploited to build the SAMOVAR heuristic rules. For instance, we could find cases such as:

- (*DIFFICULTY EFFORT PROBLEM HARDNESS LACK RISK*) OF PROBLEM
- *DISCOMFORT* FOR PROBLEM OF PART
- *IMPOSSIBILITY* OF PROBLEM OF PART
- PROBLEM(*INCORRECT IMPOSSIBLE INSUFFICIENT DIFFICULT*)
- (*DAMAGE DISPLACEMENT LACK BREAK BREAKAGE*) OF PART

We exploited these structural regularities of Nomino outputs to build manually heuristics rules validated by the expert, heuristic rules which would enable the feeding of the ontology in a semi-automatic way.

These rules that reflected the existing structures in the corpus were determined manually, but once implemented and activated, they helped us to enrich the *Problem* ontology automatically by suggesting to attach a relevant new concept corresponding to a new term, at the right position in the ontology. Figure 4 shows examples of heuristic rules.

R1 : Noun [type=Problem,n=i] Prep[« of »] **Noun[type=Problem,n=i+1]** ;

R2 :(difficulty||effort||hardness|||ack||risk) Prep[« of »] **Noun[type=Problem]**

R3 : impossibility Prep[« of »] **Noun[type=Problem]** Prep[« of »] Noun[type=Component]

R4 : **Noun[type=Problem]** Prep[« of »||« on »||« under »] Noun[type=Component]

Figure 3: ***Examples of heuristic rules***

These rules represent the possible combinations between the elements of the *Component* and *Problem* ontologies as attested in the texts. A rule is presented as a series of categories, each one possibly decorated with a set of features (for example type=Problem to indicate that the element is part of the *Problem* ontology, type=Component for an element of *Component* ontology, etc.).

For example, the rule R1 authorizes a succession of terms consisted of noun, preposition and noun, where the first is a noun of Problem type, it is followed by a preposition "of" aanother noun of Problem type, which becames the son of the first noun.

The second rule R2, authorizes a succession of terms consisted of noun, preposition and noun, where the first can be "difficulty" ("effort", "hardness" or "lack"), followed by a preposition "of" and another noun of Problem type.

These rules were implemented in PERL.

3.1.3 Kinematic of the process

We enriched the *Problem* ontology gradually (see Figure 4). For that, the SAMOVAR system takes in entry the Nomino outputs, the *Component* ontology, *Problem* ontology bootstrap and the heuristic rule base. Then it analyses the nominal groups to see with which rule each of them can match.

Example of a Nominal Group and the corresponding rule:

NOISE OF RUBBING OF THE WHEEL DURING ITS HEIGHT ADJUSTMENT

Noun[type=Problem,n=i] Prep[« of »] Nom[type=Problem,n=i+1]

The rule matches the nominal group, recognises the first term as a noise (that corresponds to an existing concept in the *Problem* ontology) and proposes to build a concept for the second noun and to insert it in the *Problem* ontology, as a son of the *Noise* concept. In the following case, the rule matches the name of the part and proposes to link the first term as a *Problem* :

JUDDERING OF THE REAR SWEEP ARM ON PPP3

Noun[type=Problem] Prep[« of »||« on »||« under »] **Noun [type=Component]**

The output provides the candidate terms to insert in the *Problem* ontology. The knowledge engineer (possibly with the support of the expert) validates each candidate and decides if the position proposed for insertion in the existing *Problem* hierarchy is correct. If yes, a concept corresponding to the term is inserted in the ontology. Such a concept – that was attested in the textual corpus - can be compared to a «terminological concept» if we use the terminology of Terminae [4].

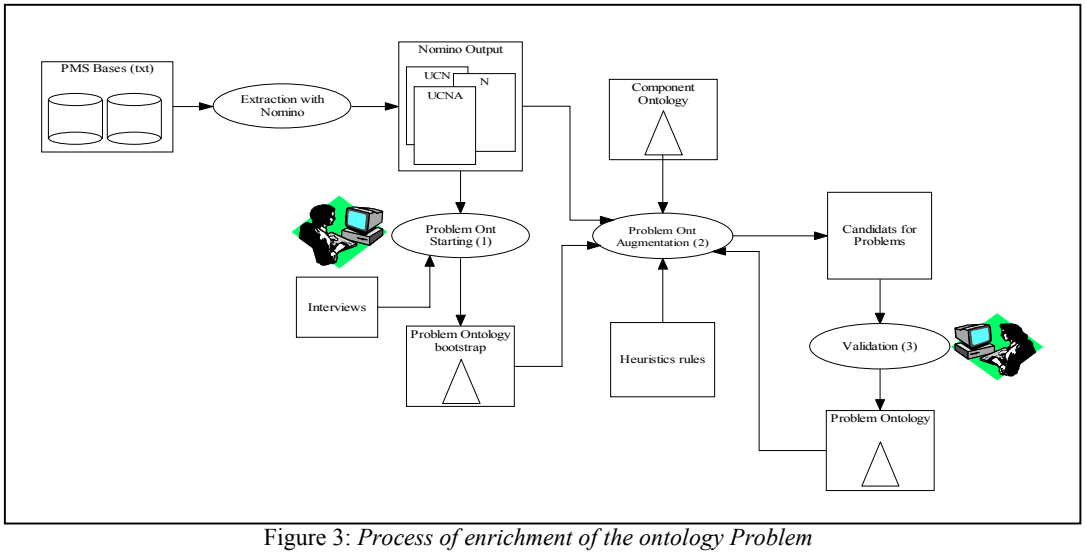

Figure 3: *Process of enrichment of the ontology Problem*

To formalize our ontologies, we chose the RDF Schema (RDFS) language, which is recommended by W3C for description of resources accessible by the Web. RDFS allows to simply describe the ontology to which RDF annotations will be relative to. Such RDF annotations are quite relevant to describe resources within a company. We can consider the descriptions of the problems met in a vehicle project (i.e. problem descriptions contained in PMS) as resources being a part of the memory of this project.

Therefore, we developed a parser which, at the end of the process, generates a version of the ontology in RDF Schema (which is also the formalism required by the CORESE software). After RDF(S) generation, the annotations of the PMS problem-descriptions are automatically updated by SAMOVAR in the form of RDF statements.

4 Exploitation of the Ontologies

4.1 Use of the CORESE Tool

The ontologies set up were used to make annotations on the problem-descriptions from the PMS, considered as document elements. Their formalization in RDF Schema and the formalization of the annotations in RDF enabled to use the CORESE tool for information retrieval guided by such RDF(S) ontologies and annotations [8].

The CORESE tool implements a RDF(S) processor based on the conceptual graph (CG) formalism [30]. CORESE relies on RDF(S) to express and exchange metadata about documents. CORESE offers a query and inference mechanism based on the conceptual graph (CG) formalism. It may be compared to a search engine which enables inferences on the RDF statements by translating them into CGs.

CORESE translates the classes and properties of RDFS towards CG concept types and relation. CORESE also translates the base of RDF annotations into a base of CGs. This enables the user to ask queries to the RDF/CG base. A query is presented in the form of an RDF statement which is translated by CORESE into a query graph which is then projected on the CG base (using the projection operator available in CG formalism). The graphs results of this projection are then translated back into RDF for providing the user with the answers to his query. The projection mechanism takes into account the concept type hierarchy and the relation type hierarchy (obtained by translation of the RDF schemas).

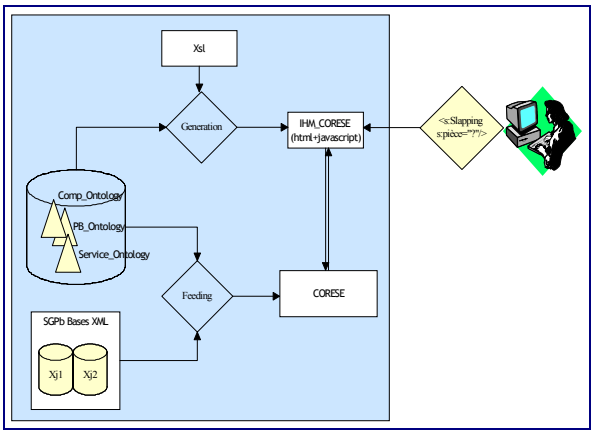

Figure 4: ***Architecture of SAMOVAR***

To exploit CORESE, we formalised the SAMOVAR ontologies into RDFS. Then, we indexed the problem-descriptions of the PMS base with instances of concepts from these ontologies, while respecting the XML-based RDF syntax. After these two stages, the user could carry out information retrieval from the annotated problem-description base. The results of the user's query take into account not only the initial terms of the query but the links modeled in the different ontologies.

4.2 Examples of queries

Here are two examples in which we show that the problems extracted from texts and structured with hierarchical links allows us to find duplications of problem descriptions:

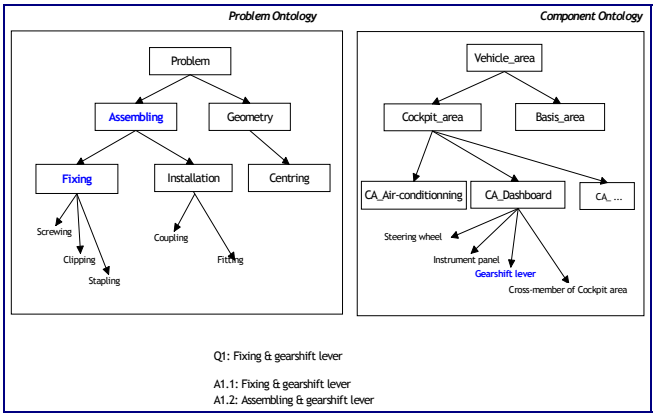

Figure 5: *Pathway for the ontologies to retrieve information*

In the first example, the user is looking for the problems of *fixing on the gearshift lever bellows*. A single answer is obtained:

T_Fixation rdf:about=http://coco.tpz.tot.fr:8080/SAMOVARXML/MOXj1-02057.xml

libelle DIAMETRE DU SOUFFLET AU NIVEAU DU BOUTON PRESSION NON EN CONCORDANCE AVEC LE DIAMETRE DU POMMEAU DU SELECTEUR DE VITESSE (**VOIR PSXj2-00193**)

piece SOUFFLET_DE_LEVIER_DE_VITESSE

On the other hand, if the user extends her query to take into account more general concepts, following the ontological links (in our case - *assembling*), she will find a second case, which is effectively a similar problem-description.

Following a successive route through the ontologies thanks to the generalization and specialization links, the user can expand the query to find the subsuming concepts (cf. the fathers of the elements of the query) and the sibling concepts. In the example, the user can explore the *problems on gearshift lever*, level by level: from problems of fixing /connecting, she can go up to the father of this last concept (i.e. *Assembling*), and then go down to the other children concepts (e.g. *Installation*). The second case thus found is a similar problem-description to the first answer :

T_Montage rdf:about=http://coco.tpz.tot.fr:8080/SAMOVARXML/PSXj2-00193.xml

libelle BOUTON PRESSION DU SOUFFLET DE LEVIER DE VITESSE IMMONTABLE (**GEREE PAR MOXj1-02057**)

piece SOUFFLET_DE_LEVIER_DE_VITESSE

In the second example, the user would like to find the *problems of centring on crossbar of cockpit area*. The system

returns three cases among which two turn out to be problem-descriptions pointing mutually:

T_Centrage rdf:about= http://coco.tpz.tot.fr:8080/SAMOVARXML/MOXj1-00403.xm

libelle FIXATIONS PDB : FIXATIONS LATERALE G ET COMPTEUR DECENTRE SUR TRAVERSE.

piece TRAVERSE_DE_POSTE_DE_CONDUITE

T_Centrage rdf:about=http://coco.tpz.tot.fr:8080/SAMOVARXML/MOXj1-02071.xml

libelle FIXATION : SUPPORT CARMINAT SUR TRAVERSE DECENTREE. (**VOIR PSXj2-00023**)

piece TRAVERSE_DE_POSTE_DE_CONDUITE

T_Centrage rdf:about=http://coco.tpz.tot.fr:8080/SAMOVARXML/PSXj2-00023.xml

libelle NON COAXIALITE DES TROUS DE FIXATION SUPPORT CALCULATEUR CARMINAT SUR TRAVERSE.(**GEREE PAR MOXj1-02071**)

piece TRAVERSE_DE_POSTE_DE_CONDUITE

The browsing through the ontology lets the user browse the whole base of problem-descriptions, following the semantic axes modeled through links in the ontologies. This browsing helps the user to find similar problem-descriptions.

4.3 Evaluation of the ontologies for the search of similar problem-descriptions

The tests were made on the *Component* and *Problem* ontologies covering the corpus corresponding to an extract of the PMS base of a vehicle-project:

- a first step was concerning a specific perimeter (*Dashboard*) for 2 milestones,
- a second step processed the entire base of the project.

We created these ontologies taking the different information sources into account (official references cross-checked with items from the problem base). In professional terms the domain corresponds to the process of *assembling*. At present the *Dashboard* perimeter contains 118 concepts and 3 relations among which 22 components within 6 architectural areas, 12 sections and 3 levels reflecting the official *Component* referential. The *Problem* ontology contains about 43 types of problems. The *Service* ontology comprises 9 services extracted automatically from the base. These ontologies have been used to annotate around 351 problem-descriptions.

The whole base contains 792 concepts and 4 relations among which 467 components are structured in the same way, but updated with a typology of 39 component managers. The *Problem* ontology contains about 75 types of problems. The *Service* ontology contains about 38 types of services retrieved from base. These ontologies have been used to annotate around *4483* problem-descriptions.

4.3.1 Discussion

The first exploratory investigations on search of similar problem-descriptions have been proved to be interesting. All problem-descriptions mutually pointing have been found (in the case where problem-descriptions belong to the covered perimeter). Furthermore, there were less answers, but only the relevant ones.

So, we can conclude that good results are obtained thanks to the annotations of problem-descriptions with the instances of the problem types discovered from texts and structured in an ontology.

We can also notice that the modeling of the ontology is essential in this method. Test modifications in the *Problem* ontology had more or less positive repercussions on the results. It is important to make sure of the validity of the ontology with the experts' support.

More generally, the method strongly depends on the corpus of the handled domain : if we reuse it for another domain, it will probably be necessary to update the heuristic rules allowing extraction of new concepts in order to cover the structures not processed. Indeed, the heuristic rules depend on the regularities found among the candidate terms extracted from the corpus.

Other « adjustments » were necessary during the process. For example, annotations with problems are at present performed by pattern matching : an annotation with a specific problem is activated as soon as the presence of some clues (for example `Centring` will be detected thanks to the presence of such clues as `indexage`, `coaxiality`, `entraxe`). According to the order of triggering of the rules, a problem-description can be annotated with instances of different ontology concepts. It would be interesting to order the rule triggering.

Besides, some other NLP tools (such as relation extractors [14] [28]) could help to refine furthermore the results of the *Problem* ontology construction.

As a further work, we intend to apply the same approach for building a *Solution* ontology (that would be connected to the *Problem* ontology). The same approach can be adopted: i.e. write heuristic rules from the manual analysis of the regularities of the candidate terms produced by Nomino and expressing possible solutions to the problems.

It would enable to index the problem-descriptions not only with instances of the concepts of the ontologies *Problem*, *Project*, *Service* and *Component*, but also with adequate instances of concepts of this *Solution* ontology.

5 Conclusions

5.1 Related Work

We have previously evoked several linguistic tools, dedicated to the extraction of terms and of relations from textual corpora. Among such tools, the choice of Nomino was due to both its relevance for our purposes and its availability. SAMOVAR can be compared to several approaches or tools integrating linguistic tools for extraction of candidate terms from a textual corpus.

Terminae [4] offers a methodology and an environment for building ontologies thanks to linguistic-based techniques of textual corpus analysis. The method is based on a study of the occurrences of terms in a corpus in order to extract the conceptual definitions and the environment helps the user in her modeling task by checking the characteristics of a new concept and by proposing potential family knot. Lexiclass [1] offers an interesting approach for building a regional ontology from technical documents. This tool enables the classification of syntagms extracted from a corpus, in order to help the knowledge engineer to discover important conceptual fields in the domain. Lexiclass coupled with Lexter, carries out a syntagm classification from Lexter according to the terminological context of the terms.

[3] describes a general method for building an ontology, method based on analysis of textual corpus using linguistic tools. The authors give the example of the Th(IC)2 project where they combine several tools for processing the textual corpus, each tool dedicated to a specific task (Lexter for terms extraction, Cameleon for relations, Terminae - for concept hierarchy construction) Our

method is situated in such a methodological framework: we use various specific tools in every step of the process, but with a corpus stemming from different origins (i.e. both interviews and textual data retrieved from existing databases). This variety characterizes the originality of our approach. [22], [23], [20] also present a general architecture for building an ontology from a textual corpus. [22], [23] exploit different linguistic tools so as to build a concept taxonomy and exploit a learning algorithm for mining non-taxonomic relations from texts.

The integration of CORESE in SAMOVAR and its ability to enable information retrieval thanks to annotations linked to the concepts of the ontologies thus build in a semi-automatic way is one originality of SAMOVAR. We must notice that SAMOVAR thus implements an approach for finding similar problems among past problem descriptions, which is a typical capability of case-based reasoning systems [26].

5.2 Further work

As noticed earlier, we will study heuristic rules for extraction of the *Solution* ontology from the textual corpus. Moreover, making explicit the links between the *Problem* and the *Solution* ontologies would enable to refine the indexing of the problem descriptions. Therefore, we will exploit a linguistic tool enabling the extraction of domain-dependent semantic relations, adapted to the automobile domain.

5.3 Towards a Method for Building a Project Memory

By finding information about similar problems processed during a given project, SAMOVAR began the process of capitalization in the company. It will be possible henceforth to spread it to wider scale - to exploit the incidents and the existing solutions between the various vehicle projects, to study problems and solutions within the same range or the same project. And in the longer term, exploit this capitalization to discover recurring problems in a company by re-showing weak spots "problems generators " to the engineering centres.

So SAMOVAR could enhance information sharing among the teams involved in the same or different vehicle projects.

We could exploit the SAMOVAR principles for other projects, provided that the right adaptations are carried out, especially at the level of the ontologies. We can thus generalize our approach to other domains than automobile design, for example to build and exploit a memory of the project of design or construction of any complex system, particularly regarding the memory of the problems encountered in such projects (e.g. incidents met during the design of a plane, a satellite, even a power plant, etc.). We propose a method relying on the following steps:

1. If there exists a database or a referential describing the components of this complex system, exploit it to build semi-automatically a *Component* ontology. Otherwise, use linguistic tools and method such as the ones described in [3] in order to build this Component ontology.
2. If there exists a description of a project characteristics in the considered company, exploit it to build a *Project* ontology. Otherwise, rely on interviews of the experts.
3. Establish a corpus of texts describing the problems met during one or several existing projects. It can involve texts resulting from textual documents or from textual comments in databases.
4. Exploit some existing linguistic tools allowing the extraction of candidate terms (e.g. Lexter [5, 6] or Nomino).
5. Analyse manually (with the support of an expert) the regularities among the candidate terms which are liable to describe types of problems (resp. solutions). Then thanks to the regularities observed, write heuristic rules exploiting both these regularities and the *Component* and *Project* ontologies in order to suggest terms to include as concepts into the *Problem* (resp. *Solution*) ontology and even more to propose their position in this ontology. Validate such heuristic rules by the expert.
6. Use these heuristic rules and let an expert validate the propositions of the system obtained thanks to these heuristic rules.
7. Use the concepts of the *Problem, Solution, Component* and *Project* ontologies, so as to index automatically the elementary problem-descriptions (in the textual corpus) with instances of these concepts.
8. Exploit an RDFS generator for the ontologies and an RDF generator for the annotations, in order to be able to use the search engine CORESE to query the base annotated by the instances of problems.

The proposed methodology is generic. However the rules are constructed relying on the corpus: they reflect the existing structures of the corpus and are strongly connected to it. So, to apply the methodology for another domain it will be necessary to rebuild the heuristic rule base, so as to make it reflect the regularities observed in the corpus. This is typical of a methodology based on corpus analysis.

5.3.1 Acknowledgments

We wish to thank our colleagues for their precious advices on our work and for their contribution in reading over this article.

6 References

[1] Assadi H, Construction of a regional ontology form text and its use with a documentary system. In N. Guarino, ed. Proc. of the 1st Int. Conf. On Formal Ontology and Information Systems (FOIS'98), IOS Press, 1998.

[2] Aussenac-Gilles N, Biébow B, Szulman S, Corpus analysis for conceptual modelling, EKAW'2000 Workshop Ontologies and Texts, Juan-les-Pins, October 2-6, 2000 pages 13-20

[3] Aussenac-Gilles N, Biébow B, Szulman S, Revisiting Ontology Design : a Method Based on Corpus Analysis, In R. Dieng and O. Corby eds, Knowledge Engineering and Knowledge Management: Methods, Models and Tools, EKAW 2000, Juan-les-Pins, French Riviera, October 2-6, 2000, p. 172-188.

[4] Biébow B, Szulman S, Terminae : a linguistics-based tool for building of a domain ontology, In D. Fensel and R. Studer, eds, Knowledge Acquisition, Modeling and Management, Proc. of the 11th European Workshop (EKAW'99), LNAI 1621,. Springer-Verlag, 1999.

[5] Bourigault D, Lexter, un Logiciel d'Extraction de TERminologie, Application à l'acquisition des connaissances à partir de textes, PhD thesis, E.H.E.S.S, Paris, France, 1994

[6] Bourigault D, Lexter, a natural language processing tool for terminology extraction. Proc. of the 7th EURALEX Int. Congress, Goteborg, 1996.

[7] Brickley D. and Guha R.V. eds. Resource Description Framework (RDF) Schema Specification 1.0, W3C Candidate Recommendation 27 March 2000, http://www.w3.org/TR/rdf-schema

[8] Corby O, Dieng R , Hébert C, A Conceptual Graph Model for W3C Resource Description Framework, ICCS'2000, Springer-Verlag, Darmstadt, August 2000.

[9] Dieng R., Corby O., Giboin A. and Ribière M. Methods and Tools for Corporate Knowledge Management. In S. Decker and F. Maurer eds, International Journal of Human-Computer Studies, Special issue on Knowledge Management, 51:567-598, September 1999.

[10] Dieng R., Corby O., Giboin A., Golebiowska J., Matta N. and Ribière M. Méthodes et Outils pour la Gestion des Connaissances, Dunod, 2000.

[11] Enguehard C, ANA, Apprentissage Naturel Automatique d'un réseau sémantique, thèse de doctorat, UTC, 1992

[12] Enguehard C, and Pantera L. Automatic natural acquisition of terminology. Journal of Quantitative Linguistics, 2/1:27-32, 1995.

[13] D. Faure and C. Nédellec., In D. Fensel and R. Studer, editors, Proc. of the 11th European Workshop (EKAW'99), LNAI 1621,. Springer-Verlag, 1999.

[14] Garcia D, Analyse automatique des textes pour l'organisation causale des actions. Réalisation du système informatique COATIS, PhD thesis, Université de PARIS IV, Paris 1998

[15] Golebiowska J, SAMOVAR - Knowledge Capitalization in the Automobile Industry aided by Ontologies, PKAW 2000, Sydney, December 11-13, 2000.

[16] Golebiowska J, SAMOVAR - Setting up and Exploitation of Ontologies for capitalising on Vehicle Project Knowledge. In Aussenac-Gilles N., Biébow B., Szulman S., eds, Proc. of EKAW'2000 Workshop Ontologies and Texts, Juan-les-Pins, October 2000 pages 79-90

[17] Grefenstette G, Explorations in automatic thesaurus discovery, Kluwer Academic Publishers, Boston, 1994

[18] Hearst M, Automatic Acquisition of Hyponyms from Large Text Corpora, ICCL, COLING 92, Nantes July 25-28, 1992

[19] Jouis C, Contribution à la conceptualisation et à la Modélisation des connaissances à partir d'un analyse linguistique de textes. Réalisation d'un prototype : le système SEEK. Thèse de doctorat, 1993, EHESS.

[20] Kietz J.-U., Maedche A. and Volz R. A Method for Semi-Automatic Ontology Acquisition from a Corporate Intranet. In Aussenac-Gilles N., Biébow B., Szulman S., EKAW'2000 Workshop Ontologies and Texts, Juan-les-Pins, October 2-6, 2000 pages 37-50.

[21] O. Lassila and R. R. Swick eds. Resource Description Framework (RDF) Model and Syntax Specification, W3C Recommendation 22 February 1999, http ://www.w3.org/TR/REC-rdf-syntax

[22] Maedche A. and Staab S., Mining Ontologies from Texts. In Dieng R. and Corby O. eds, Knowledge Engineering and Knowledge Management: Methods, Models and Tools, EKAW 2000, Juan-les-Pins, French Riviera, October 2-6, 2000, p. 189-202.

[23] Maedche A. and Staab S., Discovering conceptual relations from text. Proc. of ECAI'2000, IOS Press, August 2000.

[24] Morin E. Acquisition de patrons lexico-syntaxiques caractéristiques d'une relation sémantique, TAL (Traitement Automatique des Langues), 1999

[25] Morin E Automatic acquisition of semantic relations between terms from technical corpora. Proc. of the 5th Int. Congress on Terminology and Knowledge Engineering (TKE'99), 1999.

[26] Moussavi M. - A Case-Based Approach to Knowledge Management, in Aha D.W. (Ed). Proc. of the AAAI'99 Workshop on "Exploring Synergies of Knowledge Management and Case-Based Reasoning". Juillet 1999; Orlando, FL. AAAI Press Technical Report WS-99-10.

[27] Séguéla P. and Aussenac-Gilles N.. Extraction de relations sémantiques entre termes et enrichissement de modèles du domaine. IC'99, pages 79-88, Paris, 1999.

[28] Séguela P, Adaptation semi-automatique d'une base de marqueurs de relations sémantiques sur des corpus spécialisés. Terminologies Nouvelles, 19:52-60, 1999.

[29] Séguéla P. Construction de modèles de connaissances par analyse linguistique de relations lexicales dans les documents techniques. PhD Thesis, Université de Toulouse, March 2001.

[30] Sowa J. F. Conceptual Graphs : Information Processing in Mind and Machine. Reading, Addison Wesley, 1984.