Toward cooperatively-built knowledge repositories

Dr Philippe Martin, Dr Michael Blumenstein, A.Prof. Peter Deer

Griffith University - School of I.T. - PMB 50 Gold Coast MC - QLD 9726 Australia
(The 1st author began this article at the LOA, Italy) pm@phmartin.info

Abstract. After noting that informal documents and formal knowledge bases are far from ideal for discussing or retrieving technical knowledge, we propose mechanisms to support the sharing, re-use and cooperative update of semi-formal semantic networks, assign values to contributions and credits to the contributors. We then propose ontological elements to guide and normalize the construction of such knowledge repositories, and an approach to permit the comparison of tools or techniques.

1 Introduction

The majority of technical information is currently published in mostly *unstructured forms* within documents such as articles, e-mails and user manuals. Thence, finding and comparing tools or techniques to solve a problem is a lengthy process (with often sub-optimal results) that involves reading many documents partly redundant with each other. This process heavily relies on memory and manual cross-checking, and its outcomes, even if published, are lost to many people with similar goals. Writing documents is also a lengthy process that involves summarizing what has been described elsewhere and making choices on which ideas or techniques to describe and how: level of detail, order, etc.

To sum up, whatever the field of study, there is *currently no well structured semantic network of techniques or ideas* that a Web user could (i) navigate to get a synthetic view of a subject or, as in a decision tree, quickly find its path to relevant information, and (ii) easily update to publish a new idea (or a new way to explain an idea) and link it to other ideas via semantic relations. Such a system is indeed difficult to build and initialize. However, it is part of a vision for a semi-formal "standardized online archive of computer science knowledge" [8] and dwarfed by the much more ambitious visions of a "Digital Aristotle" which would be capable of teaching much of the world's scientific knowledge by (i) adapting to its students' knowledge and preferences [3], and (ii) preparing and answering (with explanations) test questions for them [9]. The current approaches that are related to the above-cited problems can be divided into three groups.

First, the approaches *indexing (parts of) documents by metadata* (generated or manually inputed) such as Dublin Core metadata, DocBook metadata, topics, categories from ontologies, and formal summaries. These approaches are useful for retrieving or exploiting a large base of documents but do not lead to any browsable/updatable semantic network organizing facts or ideas. The same can be said about most document-related query answering systems.

Second, the approaches aiming to represent elements of a domain into formal or semi-formal *knowledge bases* (KBs). Some KBs essentially contain *term definitions*, formal ones as in Open GALEN (an ontology of medical knowledge) and semi-formal ones as in Fact Guru [7]; they are interesting to re-use for representing or indexing parts of documents but are *insufficient* to learn about a domain or compare techniques to solve a problem. Other KBs are mainly intended to *support problem solving*, e.g. the KBs of the QED Project (which aims to build a formal KB of all important established mathematical knowledge) and those of the Halo project [9] (which has for long term goal the design of a "Digital Aristotle"), and hence are *not meant to be directly read and browsed*.

Third, the *hypertext-based Web sites* describing and organizing resources (researchers, discussion lists, journals, concepts, theories, tools, etc.) of a domain, e.g. MathWorld. Some sites permit their users to collaborate or discuss by adding or updating documents, e.g. via wiki systems or annotation systems. Because these systems *do not use semantic relations*, the resulting information is often as poorly structured as in mailing lists and hence includes many redundancies, and arguments and counter-arguments for an idea are hard to find and compare. However, Wikipedia, an on-line hypertext encyclopedia which is also a wiki, albeit a very controlled one, has good quality articles on a wide variety of domains. These articles are well connected and permit their readers to get an overview of a subject and explore it to find information. Yet, Wikipedia's content structure and support for collaboration and IR could be improved. An easy-to-use and easy-to-implement feature would be a support for some semantic relations (e.g. subtypeOf, instanceOf, and partOf) and especially argumentation relations (e.g. proof, example, hypothesis, argument, correction; pre-Web hypertext systems like AAA [6] supported predefined sets of such relations). A semi-formal English-like syntax such as ClearTalk (the notation used in Fact Guru and CODE4 [7]) would support more knowledge processing while still being user-friendly.

It would be utopic to think that even motivated knowledge engineers would be (in the near future) able/willing to represent their research ideas completely into a formal, shared, well-structured readable semantic network that can be explored like a decision tree: there are too many things to enter, too many ways to describe or represent a same thing, and too many ways to group and compare these things. On the other hand, representing the most important structures into such a semantic network and interconnecting them with informal representations seems achievable and extremely interesting for education and IR purposes. Section 2 proposes some mechanisms to support the sharing, re-use and cooperative update of such semantic networks, including some mechanisms to assign values to the contributions and credits to the contributors. Section 3 proposes some ontological elements to guide and normalize the construction of these knowledge repositories. Section 4 shows an approach to permit the comparison of tools or techniques.

2 Support of Cooperation Between Knowledge Providers

2.1 Making Knowledge Explicit and Sharing It

We only consider asynchronous cooperation since it both underlies and is more scalable than exchanges of information between co-temporal users of a system.

The most decentralized knowledge sharing strategy is the one the W3C envisages for the "Semantic Web": many small ontologies, more or less independently developed, thus partially redundant, competing and very loosely interconnected. Hence, these ontologies have problems similar to those we listed for documents: (i) finding the relevant ontologies, choosing between them and combining them is difficult and sub-optimal even for a knowledge engineer (let alone for a machine), (ii) a knowledge provider cannot simply add one concept or statement "at the right place", and is not guided by a large ontology (and a system exploiting it) into providing precise concepts and statements that complement existing ones and are more easily re-used, and (iii) the result is not only more or less lost to others but increases the amount of "data" to search.

A more knowledge-oriented strategy is to have a knowledge server permitting registered users to access and update a single large ontology on a domain and upload files mixing natural language sentences with knowledge representations (e.g. in a controlled language). WebKB-1, WebKB-2, OntoWeb/Ontobroker and Fact Guru are examples of servers allowing this. This was also the strategy used in the well publicized KA2 project [1] which re-used Ontobroker and aimed to let Knowledge Acquisition researchers index their resources, but (i) the provided ontology was extremely small and could not be directly updated by users, and (ii) the formal statements had to be stored within an invented attribute (named "onto") of HTML hyperlink tags via a poorly expressive language. Thus, this approach was limiting and frustrating, and this project was not followed.

We know of only two knowledge servers having special protocols to support users' cooperation[1]: Co4 [2] and WebKB-2 [4]. The approach of Co4 is based on peer-reviewing; the result is a hierarchy of KBs, the uppermost ones containing the most consensual knowledge while the lowermost ones are the private KBs of contributing users. We believe the approach of WebKB-2 which has a KB shared by all its users leads to more relations between categories (types or individuals) or statements from the different users and is easier to handle (by the system and the users) for a large amount of knowledge and large number of users. Details can be found in [4] but here is a summary of its principles.

To avoid lexical problems, each category identifier is prefixed by a short identifier of its creator (who is also represented by a category). Each statement also has an associated creator and hence, if it is not a definition, may be considered as a belief. Any object (category or statement) may be re-used by any user within his/her statements. The removal of an object can only be done by

[1] Most servers, including WebKB-2, support concurrency control (e.g. via KB locking) and several, like Ontolingua, support users' permissions on files/KBs. Cooperation support is not so basic: it is about helping knowledge re-use, conflict prevention and the solving of each conflict once it has been detected by the system or a user.

its creator but a user may "correct" a belief by connecting it to another belief by a "corrective relation" (e.g. `pm#corrective_specialization`). (Definitions cannot be corrected since they cannot be false). If entering a new belief introduces a redundancy or an inconsistency that is detected by the system, it is rejected. The user may then either correct his/her belief or re-enter it again but connected by a "corrective relation" to each belief it is redundant or inconsistent with: this allows and makes explicit the disagreement of one user with (his/her interpretation of) the belief of another user. This also technically removes the cause of the problem: a proposition A may be inconsistent with a proposition B but a belief that "A is a correction of B" is not technically inconsistent with a belief in B. (Definitions from different users cannot be inconsistent with each other, they simply define different categories/meanings). Choices between beliefs may have to be made by people re-using the KB for an application, but then they can exploit the explicit relations between beliefs, e.g. by always selecting the most specialized ones. WebKB-2 displays a statement *with* its meta-statements, hence with the associated corrective relations. Finally, to avoid seeing the objects of certain creators during browsing or within query results, a user may set filters on these creators, based on their identifiers, types or descriptions.

For the construction of knowledge repositories, an interesting aspect of this approach to encourage re-use, precision and object connectivity is that it also works for semi-formal KBs. Here, regarding a statement, semi-formal means that if it is written in a natural language (whether it uses formal terms or not) it must *at least* be related to another statement by a formal relation, e.g. a generalization relation (`pm#corrective_generalization`, `pm#summary`, etc.) or an argumentation relation. Thus, to minimize redundancies and to help information retrieval within information repositories, this minimal semantic structure (which in many situations is the only one bearable by people) could be used to organize ideas that are otherwise repeated in many documents. For instance, for a Web site that centralizes and organizes/represents in a formal, semi-formal and informal way resources (tools, techniques, publications, mailing list, teams, etc.) related to a domain (e.g. CGs), it would be very interesting to have some space where discussions could be conducted in this minimal semi-formal way, and hence index or partly replace the mailing list: this would avoid recurring discussions or presentations of arguments, show the tree of arguments and counter-arguments for an idea, permit incremental additions, encourage deeper or more systematic explorations of each idea, and record the various reached status-quos.

Below is an example of what the top-levels of a semi-formal discussion could look like when displayed in an indented linear form (we do not expect this organization to be the direct result of a discussion but it may be the result of a semi-automatic re-organization of this discussion and then be refined by further semi-formal discussions). To save space, we have replaced (counter-)argument relations by '+' and '-', used comments to give an idea of lower levels, and not represented the authors of the statements. The (counter-)arguments for a statement are valid for its specializations and the (counter-)arguments of the specializations are (counter-)examples for their generalizations.

Addendum to this article: this example does not distinguish between a relation from a statement, a relation on a relation, and a conjunctive set of statements; for better representations of that discussion, see http://www.webkb.org/doc/papers/iccs05/iccs05.html#argumExample

```
A KRL should (also) have an XML-based notation to ease knowledge sharing
 +: this permits to use URIs and Unicode
     -: most syntaxes can be adapted for that  //+: as noted by Berners-Lee
 +: this permits to re-use XML tools (parsers, XSLT, ...)
     -: useless additional step since KBSs do not use XML internally
     > classic XML tools re-usable even if a graph-based model is used
        > classic XML tools work on RDF/XML
           -: some XML tools expect a classic XML tree to be followed
           -: with difficulties since RDF/XML has multiple serialisations
 -: XML-based KRLs are hard to read without XML tools
     +: this is acknowledged by about everyone  //including by the W3C
     -: this does not matter since XML tools exist
         -: this is impossible or inconvenient in many tools or situations
         -: a good notation and a good text editor is often more convenient
 +: other notations may still be used  //-: but they are not standards ...
 +: Not using XML would require a separate plug-in for each syntax
     -: installing a plug-in takes less time than always loading XML files
 > The Semantic Web (SW) KRL should have an XML notation
    > The SW KRL should have an XML syntax but a graph-based model
       +: for flexibility and normalization reasons  //+: TBL's arguments
       +: classic XML tools re-usable even if a graph-based model is used
       -: then there are partially redundant standards (e.g. with RDF/XML)
       > RDF should have the current RDF/XML syntax
          -: it is particularly arbitrary and hard to understand
          -: it leads to errors //+:cannot use schema validation languages
          -: it is inefficient //+:parsers 5-20 times slower than with XML
```

2.2 Valuating Contributions and Contributors

The above described knowledge sharing mechanism of WebKB-2 records and exploits annotations by individual users on statements but does not record and exploit any measure of the "usefulness" of each statement, a value representing its "global interest", popularity, originality, etc. Yet, this is interesting for a knowledge repository and especially for semi-formal discussions: statements that are obvious, un-argued, or for which each argument has been counter-argued, should be marked as such (e.g. via darker colors or smaller fonts) in order to make them less visible (or invisible, depending on the selected display options) and discourage the entering of such statements. More generally, the presentation of the combined efforts from the various contributors may then take into account the usefulness of each statement. Given that the creator of each statement is recorded, (i) a value of usefulness may also be calculated for each creator (and displayed), and (ii) in return, this value may be used when calculating the usefulness of the creator's contributions; these are two additional refinements to detect, encourage and regulate argued and interesting contributions.

Ideally, the system would allow user-defined measures of usefulness for a statement or a creator, and adapt its presentation of the repository accordingly. Below we present the *default* measures that we shall implement in WebKB-2 (or more exactly, its successor and open-source version, AnyKB). We may try to support *user-defined* measures but since each step of the user's browsing would imply re-calculating the usefulness of all statements (except those from WordNet) and all creators, the result is likely to be very slow. For now, we only consider beliefs: we have not yet defined the usefulness of a definition.

To calculate a belief usefulness, we first associate two more basic attributes to the belief: 1) its "state of confirmation" and 2) its "global interest".
1) The first is equal to 0 if the belief has no (counter-)argument linked to it (examples of counter-argument relation names: "counter-example", "counter-fact", "corrective-specialization"). It is equal to 1 (i.e. the belief is "confirmed") if (i) each argument has a state of confirmation of 0 or 1, and (ii) there exists no confirmed counter-argument. It is equal to -1 if the belief has at least one confirmed counter-argument. It is also equal to 0 in the remaining case: no confirmed counter-argument but each of the arguments has a state of confirmation of -1. All this is independent of whom authored the (counter-)arguments.
2) Each user may give a value to the interest of a belief, say between -5 and 5 (the maximum value that the creator of the belief may give is, say, 2). Multiplied by the usefulness of the valuating user, this gives an "individual interest" (thus, this may be seen as a particular multi-valued vote). The *global interest* of a belief is defined as the average of its individual interests (thus, this is a voting system where more competent people in the domain of interest are given more weight). A belief that does not deserve to be visible, e.g. because it is clearly a particular case of a more general belief, is likely to receive a negative global interest. We prefer letting each user explicitly give an interest value rather than taking into account the way the belief is generalized by or connected to (or included in) other beliefs because interpreting an interest from such relations is difficult. For example, a belief that is used as a counter-example may be a particular case of another belief but is nevertheless very interesting as a counter-example.
Finally, the *usefulness of a belief* is equal to the value of the global interest if the state of confirmation is equal to 1, and otherwise is equal to the value of the state of confirmation (i.e. -1 or 0: a belief without argument has no usefulness, whether it is itself an argument or not).
In argumentation systems, it is traditional to give a type to each statement, e.g. fact, hypothesis, question, affirmation, argument, proof. This could be used in our repositories too (even though the connected relations often already give that information) and we could have used it as a factor to calculate the usefulness (e.g. by considering that an affirmation is worth more than an argument) but we prefer a simpler measure only based on explicit valuations by the users.

Our formula for a user's usefulness is: `sum of the usefulness of each belief from the user + square root (number of times he/she voted on the interest of beliefs)` . The second part of this equation acknowledges the participation of the user in votes while decreasing its weight as the number of votes increases. (Functions decreasing more rapidly than `square root` may perhaps better balance originality and participation effort).

These measures are simple but should incite the users to be careful and precise in their contributions (affirmation, arguments, counter-arguments, etc.) and give arguments for them: unlike in anonymous reviews, careless statements here penalise their authors. Thus, this should lead users not to make statements outside their domain of expertise or without verifying their facts. (Using a different pseudo when providing low quality statements does not seem to be an helpful strategy to escape the above approach since this reduces the number of authored statements for the first pseudo). On the other hand, the above measures should hopefully not lead "correct but outside-the-main-stream contributions" to be under-rated since counter-arguments must be justified. Finally, when a belief is counter-argued, the usefulness of its author decreases and hence he/she is incited to remove it or deepen the discussion.

In his description of a "Digital Aristotle" [3], Hillis describes a "Knowledge Web" to which researchers could add ideas or explanations of ideas "at the right place", and suggests that it should "include the mechanisms for credit assignment, usage tracking, and annotation that the [current] Web lacks", thus supporting a much better re-use and evaluation of the work of a researcher than via the current system of article publishing and reviewing. However, Hillis does not give any indication on such mechanisms. Although the mechanisms we proposed in this sub-section and the previous one were intended for one knowledge server,they seem usable for the Knowledge Web too. To complement the approach with respect to the Knowledge Web, the next sub-section proposes a strategy to achieve knowledge sharing between knowledge servers.

2.3 Combining the Advantages of Centralization and Distribution

One knowledge server cannot support the knowledge sharing of all researchers. It has to be specialized and/or act as a broker for more specialized servers. If competing servers had an equivalent content (today, Web search engines already have "similar" content), a Web user could query or update any server and, if necessary, be redirected to use a more specialized server, and so on recursively (at each level, only one of the competing servers has to be tried since they mirror each other). If a Web user directly tried a specialized server, it could redirect him/her to use a more appropriate server or directly forward him/her query to other servers.

To permit this, our idea is that each server periodically checks *related servers* (more general servers, competing servers and slightly more specialized servers) and 1) integrates (and hence mirrors) all the objects (categories and statements) generalizing the objects in a *reference set* that it uses to define its "domain" (if this is a general server, this set is reduced to `pm#thing`, the uppermost concept type), 2) integrates either all the objects that are more specialized than the objects in the reference set, or if a certain depth of specialization is fixed, associates to its most specialized objects the URLs of the servers that can provide specializations for these objects (note: classifying servers according to domains is far too coarse to index/retrieve knowledge from distributed knowledge servers, e.g. knowledge about "neurons" or "hands" can be relevant to many domains; thus, a classification by objects is necessary), and 3) also associates the URLs of more general servers to the direct specializations of the generalizations of the objects in the reference set (because the specializations of some of these specializations do not generalize nor specialize the objects in the reference set).

Integrating knowledge from other servers is certainly not obvious but this is a more scalable and exploitable approach than letting people and machines select and re-use or integrate dozens or hundreds of (semi-)independently designed small ontologies. A more fundamental obstacle to the widespread use of this approach is that many industry-related servers are likely to make it difficult or illegal to mirror their KBs. However, other approaches will suffer from that too.

3 Some Ontological Elements

By default, the shared KB of WebKB-2 includes an ontology derived from the noun-related part of WordNet and various top-level ontologies [5]. A large general ontology like this is necessary to ease and normalize the cooperative construction of knowledge repositories but is still insufficient: an initial ontology on the domain of the repository is also necessary. As a proof of concept for our tools to support a cooperatively-built knowledge repository, we initially chose to model two related domains: (i) Conceptual Graphs (CGs), since this domain is the most well known to us, and (ii) ontology related tools, since Michael Denny's "Ontology editor survey"[2] attracted interest despite being purposefully shallow and poorly structured.

Modelling these two domains implies partially modelling related domains, and we soon had the problem of modularizing the information into several files to support readability, search, checking and systematic input[3]. These files are also expected to be updatable by users when our knowledge-oriented wiki is completed. In order to be generic, we have created six files[4]: "Fields of study", "Systems of logic", "Information Sciences", "Knowledge Management", "Conceptual Graph" and "Formal Concept Analysis". The last three files specialize the others. Each of the last four files is divided into sections, the uppermost ones being "Domains and Theories", "Tasks and Methodologies", "Structures and Languages", "Tools", "Journals, Conferences, Publishers and Mailing Lists", "Articles, Books and other Documents" and "People: Researchers, Specialists, Teams/Projects, ...". This is a work in progress: the content and number of files will increase but the sections seem stable. We now give examples of their content.

[2] http://www.xml.com/pub/a/2004/07/14/onto.html

[3] Although the users of WebKB-2 can direcly update the KB one statement at a time, the documentation discourages them to do so because this is not a scalable way to represent a domain (as an analogy a line command interface is not a scalable way to develop a program). Instead, they are encouraged to create files mixing formal and informal statements and ask WebKB-2 to parse these files, and in the end when the modelling is complete and if the users wish to, integrate them to the shared KB.

[4] See http://www.webkb.org/kb/domain/

3.1 Domains and Theories

Names used for domains ("fields of study") are very often also names for tasks. Task categories are more convenient for representing knowledge than domain categories because (i) organizing them is easier and less arbitrary, and (ii) many relations (e.g. case relations) can then be used. Since for simplicity and normalization purposes a choice must be made, whenever suitable we have represented tasks instead of domains. When names are shared by domain categories and task categories (in WebKB-2, categories can share names but not identifiers), we advise the use of the task categories for indexing or representing resources.

When studying how to represent and relate document subjects/topics (e.g. technical domains), [10] concluded that representing them as types was not semantically correct but that mereo-topological relations between individuals were appropriate. Our own analysis confirmed this and we opted for (i) an interpretation of theories and fields of study as large "propositions" composed of many sub-propositions (this seems the simplest, most precise and most flexible way to represent these notions), and (ii) a particular part relation that we named ">part" (instead of "subdomain") for several reasons: to be generic, to remind that it can be used in WebKB-2 as if it was a specialization relation (e.g. the destination category needs not be already declared) and to mak clear that our replacement of WordNet hyponym relations between synsets about fields of study by ">part" relations refines WordNet without contradicting it. Our file on "Fields of study" details these choices. Our file on "Systems of logics" illustrates how for some categories the represented field of study *is* a theory (it does not *refer* to it) thus simplifying and normalizing the categorization. Below is an example (in the FT notation) of relations from WordNet category `#computer_science`, followed by an example about logical *domains/theories.* When introducing general categories in Information Sciences and Knowledge Management, we used the generic users "is" and "km". In WebKB-2, a generic user is a special kind of user that has no password: anyone can create or connect categories in its name but then cannot remove them.

```
#computer_science__computational_science  (^engineering science that ...^)
  >part:    #artificial_intelligence,  //in WordNet, AI is ">part:" of CS
  >part:    is#software_engineering_science (is), //link created by "is"
  >part:    is#database_management_science (is),
  >part of: #engineering_science__engineering__applied_science__technology,
  part:     #information_theory,  //link from WordNet: "(wn)" is implicit
  part of:  #information_science;

km#substructural_logic (^system of propositional calculus weaker ...^)
 >part of:  km#non-classical_logic__intuitionist_logic,
 >part:  km#relevance_logic  km#linear_logic,
 url: http://en.wikipedia.org/wiki/Intuitionistic_logic;

km#CG_domain__Conceptual_Graphs__Conceptual_Structures
 >part of: km#knowledge_management_science,
 object: km#CG_task  km#CG_structure  km#CG_tool  km#CG_mailing_list,
 url: http://www.cs.uah.edu/~delugach/CG/  http://www.jfsowa.com/cg/;
```

To provide a core ontology that will guide the sharing, indexation or representation of techniques in Knowledge Management, hundreds of categories will need to be represented. We have only begun this work. On ontological issues too our approach departs from the one of the KA2 project [1] since a good part of its small predefined ontology was a specialization hierarchy of 37 Knowledge Acquisition (KA) domains, the names of which could have been used for tasks, structures, methods (PSMs) and experiments. E.g., this hierarchy included:
`reuse_in_KA > ontologies  PSMs;      PSMs > Sysiphus-III_experiment;`

3.2 Tasks and Methodologies

In most model libraries for KA, each non-primitive task is linked to techniques that can be used for achieving it, and conversely, each technique combines the results of more primitive tasks. We tried this organization but at the level of generality of our current modelling it turned out to be inadequate: it led (i) to arbitrary choices between representing sometimes as a task (a kind of process) or a technique (a kind of process description), or (ii) to the representation of both notions and thus to introduce categories with names such as `KA_by_classification_from_people`; both cases are problematic for readability and normalization. Similarly, instead of representing methodologies directly, i.e. as another kind of process description, it seems better to represent *the tasks* advocated by a methodology (including their supertask: following the methodology). Furthermore, with tasks, many relations can then be used directly: similar relations do not have to be introduced for techniques or methodologies. Hence, we represented all these things as tasks and used multi-inheritance. This considerably simplified the ontology and the source files. Here are some extracts.

```
km#KM_task__knowledge_management_task__KM   < is#information_sciences_task,
 > km#knowledge_representation  km#knowledge_extraction_and_modelling
   km#knowledge_comparison  km#knowledge_retrieval_task
   km#knowledge_creation  km#classification  km#KB_sharing_management
   km#mapping/merging/federation_of_KBs  km#knowledge_translation
   km#knowledge_validation
   {km#monotonic_reasoning  km#non_monotonic_reasoning}
   {km#consistent_inferencing km#inconsistent_inferencing}
   {km#complete_inferencing km#incomplete_inferencing}
   {km#structure-only_based_inferencing km#rule_based_inferencing}
   km#language/structure_specific_task  //e.g. km#CG_task and km#FCA_task
   km#teaching_a_KM_related_subject  km#KM_methodology_task,
 object of: km#knowledge_management_science,
 object: km#KM_structure; //between types, the default cardinality is 0..N
 //"object" has different meanings depending on the connected categories

   km#knowledge_retrieval_task  < is#IR_task,
    > {km#specialization_retrieval  km#generalization_retrieval}
      km#analogy_retrieval  km#structure_only_based_retrieval
      {km#complete_retrieval km#incomplete_retrieval}
      {km#consistent_retrieval km#inconsistent_retrieval};
```

3.3 Structures and Languages

In our top-level ontology [5], `pm#description_medium` (supertype for languages, data structures, ...) and `pm#description_content` (supertype for fields of studies, theories, document contents, softwares, ...) have for supertype `pm#description` because (i) such a general type grouping both notions is needed for the signatures of many basic relations, and (ii) classifying WordNet categories according to the two notions would have often led to arbitrary choices. However, we represented the default ontology of WebKB-2 as a part of WebKB-2 and hence allowed part relations between any kind of information. To ease knowledge entering and certain exploitations of it, we allow the use of generic relations such as part, object and support when, given the types of the connected objects, the relevant relations (e.g. `pm#subtask` or `pm#physical_part`) can automatically be found.

For similar reasons, to represent "sub-versions" of ontologies, softwares, or more generally, documents, we use types connected by subtype relations. Thus, for example, `km#WebKB-2` is a type and can be used with quantifiers.

```
km#KM_structure  < is#symbolic_structure,
 > {km#base_of_facts/beliefs  km#ontology  km#KB_category  km#KB_statement}
   km#KB  km#KA_model  km#KR_language  km#language_specific_structure;

   km#ontology__set_of_category_definitions/constraints
    > km#lexical_ontology  km#language_ontology  km#domain_ontology
      km#top_level_ontology  km#concept_ontology  km#relation_ontology
      km#multi_source_ontology__MSO,
    part: 1..* km#KB_category  1..* km#category_definition;

   km#KR_language__KRL__KR_model_or_notation
    > {km#KR_model/structure  km#KR_notation}
      km#frame_oriented_language  km#predicate_logic_oriented_language
      km#graph_oriented_language  km#KR_language_with_query_commands
      km#KR_language_with_scripting_capabilities,
    attribute: km#semantics;

km#CG_structure  < km#language_specific_structure,
 > km#CG_statement  km#CG_language  km#CG_ontology;
```

3.4 Tools

We first illustrate some specialization relations between tools then we use the FCG notation to give some details on WebKB-2 and Ontolingua. (The FT notation does not yet permit to enter such details. As in FT, relation names in FCG may be used instead of relations identifiers when there is no ambiguity).

```
km#CG_related_tool  < km#language/structure_specific_tool,
 > km#CG-based_KBMS  km#CG_graphical_editor  km#NL_parser_with_CG_output;

   km#CG-based_KBMS < km#KBMS,
    > {km#CGWorld  km#PROLOG\+CG  km#CoGITaNT  km#Notio  km#WebKB};

      km#WebKB  > {km#WebKB-1  km#WebKB-2},  url: http://www.webkb.org;
```

```
km#input_language (*x,*y) = [*x, may be support of: (a km#parsing,
                                     input: (a statement, formalism: *y))];
[any pm#WebKB-2,   //", part:": has for part, "a ": existential quantifier
  part:(a is#user_interface, part:{a is#API, a is#HTML_based_interface,
                                    a is#CGI-accessible_command_interface,
                                    no is#graph_visualization_interface}),
  part: {a is#FastDB, a km#default_MSO_of_WebKB-2},
  input_language: a km#FCG,   output_language: {a km#FCG, a km#RDF},
  support of: a is#regular_expression_based_search,
  support of: a km#specialization_structural_retrieval,
  support of: a km#generalization_structural_retrieval,
  support of: (a km#specialization_structural_retrieval,
               kind: {km#complete_inferencing, km#consistent_inferencing},
               input: (a km#query, expressivity: km#PCEF_logic),
               object: (several km#statement, expressivity: km#PCEF_logic)
              )];       //"PCEF": positive conjunctive existential formula
[any km#Ontolingua,
  part: {a is#HTML_based_interface, no is#graph_visualization_interface,
         no DBMS, a km#ontolingua_library},    input_language: a km#KIF,
  output_language:{a km#KIF, no km#RDF}, support of: a is#lexical_search];
```

3.5 Articles, Books and other Documents

This example shows how a simple document indexation using Dublin Core relations (we have done this for all the articles of ICCS 2002). Representing ideas from the article would be more valuable. For examples of representations of conferences, publishers, mailing lists, researchers and research teams, please access http://www.webkb.org/kb/domain/.

```
[an #article, dc#Coverage: km#knowledge_representation,
  pm#title: "What Is a Knowledge Representation?",
  dc#Creator: "Randall Davis, Howard E. Shrobe and Peter Szolovits",
  pm#object of: (a #publishing, pm#time: 1993,
                    pm#place: (the #object_section "14:1 p17-33",
                                  pm#part of: is#AI_Magazine)),
  pm#url: http://medg.lcs.mit.edu/ftp/psz/k-rep.html];
```

4 Example of Comparison of Two Ontology-related Tools

Fact Guru (which is a frame-based system) permits the comparison of two objects by generating a table with the object identifiers as column headers, the identifiers of all their attributes as row headers, and for each cell either a mark to signal that the attribute does not exist for this object or a description of the destination object. The common generalizations of the two objects (possibly one of them) is also given. However, this strategy is insufficient for comparing tools. Even for people, creating detailed tool comparison tables is often a presentation challenge and involves their knowledge of which features are difficult or important and which are not. A solution could be to propose predefined tables for easing the entering of tool features and then compare them. However, this is restricting. Instead or in complement, we think that a mechanism to generate good comparison tables is necessary and can be found. The following query and generated table illustrates an approach that we propose. The idea is that a specialization hierarchy of features is generated according to (i) the uppermost relations and destination types specified in the query, and (ii) only descriptions used in at least one of the tools and their common generalizations are shown. To that end, some FCG-like descriptions of types can be generated. In the cells, '+' means "yes" (the tool has the feature), '-' means "no", and '.' means that the information has not been represented. We invite the reader to compare the content of this table with the representations given above; then, its meaning and the possibility to generate it automatically should hopefully be clear. A maximum depth of automatic exploration may be given; past this depth, the manual exploration of certain branches (like the opening or closing of sub-folders) should permit the user to give the comparison table a presentation better suited to his/her interest. Any number of tools could be compared, not just two.

```
> compare pm#WebKB-2 km#Ontolingua on
   (support of: a is#IR_task, output_language: a KR_notation), maxdepth 5

                                            WebKB-2  Ontolingua
support of:
is#IR_task                                     +         +
  is#lexical_search                            +         +
    is#regular_expression_based_search         +         .
  km#knowledge_retrieval_task                  +         .
    km#specialization_structural_retrieval     +         .
      (kind: {km#complete_inferencing, km#consistent_inferencing},
       input: (a km#query, expressivity: km#PCEF_logic),
       object: (several statement, expressivity: km#PCEF_logic))
                                               +         .
    km#generalization_structural_retrieval     +         .

output_language:
km#KR_notation                                 +         +
  (expressivity: km#FOL)                       +         +
    km#FCG                                     +         .
    km#KIF                                     .         +
  km#XML-based notation                        +         .
    km#RDF                                     +         -
```

In the general case, the above approach where the descriptions are put in the rows and organized in a hierarchy is likely to be more readable, scalable and easier to specify via a command than when the descriptions are put in the cells, e.g. as in Fact Guru. However, this may be envisaged as a complement for simple cases, e.g. to display `{FCG, KIF}` instead of '+' for the `output_language` relation. In addition to generalization relations, "part" relations could also be used, at least the `>part` relation. E.g., if Cogitant was a third entry in the above table, since it has a complete and consistent structure-based and rule-based mechanism to retrieve the specializations of a simple CG in a base of simple CGs and rules using simple CGs, we would expect the entry ending by `km#PCEF_logic` to be specialized by an entry ending by `km#PCEF_with_rules_logic`.

5 Conclusion

Knowledge repositories, as we have presented them, have many of the advantages of the "Knowledge Web" and "Digital Aristotle" but seem much more achievable. To that end, we have proposed some techniques and ontological elements, and we are: (i) implementing a knowledge oriented wiki to complement our current interfaces, (ii) experimenting on how to best support and guide semi-formal discussions, and more generally, organize technical ideas into a semantic network, (iii) implementing and refining our measures of statement/user usefulness, (iv) completing the above presented ontology to permit at least the representation of the information collected in Michael Denny's "Ontology editor survey" (we tend to think that our current ontology on knowledge management will only need to be specialized, even though we have not yet explored the categorization of the basic features of multi-user support such as concurrency control, transactions, CVS, file permissions, file importation, etc.), (v) permitting the comparison of tools as indicated above, and (vi) providing forms or tables to help tool creators represent the features of their tools.

References

1. V.R. Benjamins, D. Fensel, A. Gomez-Perez, S. Decker, M. Erdmann, E. Motta and M. Musen. Knowledge Annotation Initiative of the Knowledge Acquisition Community: (KA)2. *Proc. of KAW98*, Banff, Canada, April 1998.
2. J. Euzenat. Corporate memory through cooperative creation of knowledge bases and hyper-documents. *Proc. of 10th KAW*, (36) 1–18, Banff (CA), Nov. 1996.
3. W.D. Hillis. "Aristotle" (The Knowledge Web). *Edge Foundation, Inc.*, No 138, May 2004. *http://www.edge.org/3rd_culture/hillis04/hillis04_index.html*
4. P. Martin. Knowledge Representation, Sharing and Retrieval on the Web. *Web Intelligence* (Eds.: N. Zhong, J. Liu, Y. Yao), Springer-Verlag, Jan. 2003. *http://www.webkb.org/doc/papers/wi02/*
5. P. Martin. Correction and Extension of WordNet 1.7. *Proc. of ICCS 2003* (Dresden, Germany, July 2003), Springer Verlag, LNAI 2746, 160-173.
6. W. Schuler and J.B. Smith. Author's Argumentation Assistant (AAA): A Hypertext-Based Authoring Tool for Argumentative Texts. *Proc. of ECHT'90*, Cambridge University Press, 137–151.
7. D. Skuce and T.C. Lethbridge. CODE4: A Unified System for Managing Conceptual Knowledge. *Int. Journal of Human-Computer Studies (1995)*, 42, 413–451. Fact Guru, the commercial version of CODE4, is at www.factguru.com
8. D.A. Smith. Computerizing computer science. *Communications of the ACM (1998)*, 41(9), 21–23.
9. Vulcan Inc. Project Halo: Towards a Digital Aristotle. *www.projecthalo.com*
10. C.A. Welty and J. Jenkins. Formal Ontology for Subject. *Journal of Knowledge and Data Engineering (Sept. 1999)*, 31(2), 155-182.