Large-scale cooperatively-built KBs

Philippe Martin and Peter Eklund

Distributed System Technology Centre
KVO Laboratory - Griffith University
PMB 50 Gold Coast MC, QLD 9726 Australia
{philippe.martin,p.eklund}@gu.edu.au

Abstract. We describe a knowledge server that permits Web users to retrieve and add knowledge in a shared knowledge base. The following features distinguish WebKB-2 from other ontology servers or KBMSs:
(i) the ontology is large (at present, 69,000 categories and 87,800 links mostly coming from WordNet) and extendible at any time by any user,
(ii) asynchronous cooperation between users is supported and encouraged (users are encouraged to re-use, complement or correct the knowledge of other users but do not have to agree with each other and may add new names to categories) while the knowledge base is kept unique to maximize knowledge interconnection, retrieval and inconsistency detection,
(iii) the proposed knowledge representation languages are designed to be both expressive and readable to permit and encourage the users to enter all the knowledge they want (though that still requires motivation).
WebKB-2 is ultimately intended to permit cooperatively-built Yellow-Page like catalogs, that is, permit Web users to publish their information in a way that is automatically retrievable and comparable with other users' knowledge (as opposed to publishing information in plain text documents or even RDF documents). For example, database developpers or car dealers could describe and compare their products in a precise way, supporting precise queries.

1 Introduction

Current Web search engines can retrieve documents that include some given keywords but cannot extract and therefore retrieve and inter-link precise information (or knowledge) from them. The problem is that *knowledge retrieval and interlinking* is (re)done by each individual using his/her own memory. For example, each person trying to find a good database system for a project has to search and read the documentation of many systems, find comparative criteria and, from what he has read, try to classify each system against these criteria. With some luck, existing up-to-date comparisons may be found and feedback from users of some of these systems compiled. Nevertheless, *each* search is likely to be long, have sub-optimal results and remain unknown to (other) database system seekers or providers. This example is typical of many other searches: car, insurance, employer, employee, software, hardware, etc. In each case, the solution implies *knowledge representation and centralization.*

A **first step** is to develop languages and systems permitting Web users to create or re-use Web documents containing knowledge representations. To this aim, in 1997, we created WebKB-1 [3], a knowledge-based private annotation tool ("annotation" implies "representation", "indexation" and "retrieval"). Many XML extensions were conceived for this purpose, e.g. RDF[1] in 1998. In [4], we discussed the inadequacies of these XML-based languages for precision-based knowledge indexation and retrieval. Nevertheless, in [5], to permit better knowledge exchange/retrieval via RDF, we proposed extensions to RDF and general conventions for knowledge representation (in RDF, CG, ...). However, even if these conventions were adopted, in this distributed (document based) approach, there would still be a lot of small competing and loosely inter-linked ontologies and hence automatic comparison of representations would remain limited and based on lexical matching.

The **next step** toward centralization - and hence better automatic/manual knowledge retrieval, comparison, inter-linking, cross-checking and cooperation between knowledge providers - is to develop knowledge base servers permitting Web users to add representations into a shared repository while reusing or extending a large ontology. For efficiency and commercial reasons, all Web-users would not use the same knowledge server but rather a few general knowledge servers (e.g. managed by portal companies) and more specialized knowledge servers. By (partly) mirroring one another, general servers would probably share a similar general WordNet-like or CYC-like ontology, and competing specialized knowledge servers would also share some similar content[2]. Thus, it would not really matter where a Web user publishes information first, and this centralized approach would keep the advantages of the current distributed approach.

Early 2000, we stopped adding features to WebKB-1 and began the design and implementation of WebKB-2, a *large-scale knowledge server permitting each users to re-use, complement or annotate the knowledge of other users*[3]. This type of tool is still new. Close relatives are "ontology servers" (e.g. Ontolingua[4] and Ontosaurus[5]) but they mostly have small ontologies (a few hundred categories) and do not support/encourage a tight interlinking/annotation of knowledge from various users. Large-scale KBMSs also rarely support a large "dynamic" ontology (users cannot change the ontology interactively, a re-compilation or re-indexation phase is necessary). In this article, we first present a few elements that we think necessary to enable a large-scale cooperatively-built knowledge base, then we describe the cooperation mechanisms in WebKB-2, followed by necessary extensions to classic "searches for specializations of graphs". Finally, we compare our approach to others.

---

[1] http://www.w3.org/RDF

[2] The processes of mirroring and answering queries involving several knowledge bases is by itself permitted by the similarity or interconnection of various used ontologies.

[3] WebKB-2 also inherits the features of WebKB-1, i.e. it can exploit Web documents mixing text and representations and use them as private knowledge bases.

[4] http://WWW-KSL-SVC.stanford.edu:5915/

[5] http://www.isi.edu/isd/ontosaurus.html

2 Elements for a large-scale cooperatively-built KB

2.1 Notations

We believe a general knowledge base server should support a knowledge representation language that is (i) expressive enough to permit the user to be exact in his/her representations, and (ii) limiting the number of different (and not automatically comparable) ways to express the same information. If the language is not expressive enough, users will either enter incorrect information, develop various incomparable ways to represent it, or simply not use the server. Any of these cases hinder knowledge comparison, cross-checking, inter-linking and re-use. This does not mean that the server should have inference capabilities that exploit all the subtleties of the language. Implementing such an inference engine would actually often involve application-dependant choices. Instead, the server should perform minimal consistency checks and help filter the knowledge relevant to answer a query (i.e. the knowledge relevant for an application).
We designed two notations – Frame-CG (FCG) and Formalized English (FE) – to improve on the readability and expressivity of the Conceptual Graph linear notation (see [6] for details). A translator could be built to transform CGLF or CGIF into FE or FCG, and conversely for simple cases [6].

2.2 Lexical/syntactic/semantic/ontological recommendations

To reduce the number of ways, information can be expressed (and therefore increase the chance of automatic comparison), the WebKB-2 user is asked to follow conventions (most of which were described in [6]): lexical recommendations (use English singular nouns as category names), semantic recommendations (be precise, contextualize statements, re-use and complement existing knowledge), syntactic recommendations (e.g. how to represent various kinds of quantifiers, collections, intervals, contexts, 2nd order types/relations), ontological recommendations (e.g. how to represent states and processes, descriptions, indexations, characteristics, measures, numbers, collections, temporal/spatial/logical entities/relations). These recommendations and the associated "patterns" are also a guide. Most are more or less enforced by the use of our notations and of the existing types in the shared knowledge base.

2.3 Structures for a multi-users KB

The elements of the KB of WebKB-2 are: *categories* (concept/relation types, individuals), *links between categories* (e.g. specialization and exclusion links), category *names* and *links between categories and names* (each category has a unique *identifier* but may have many names; conversely, each name/word may belong/refer to various categories - as many categories as the word has different meanings), *concept* nodes (a graph is itself a concept node), *relation* nodes, and *users* (each of the previous elements is connected to (an object representing) its creator, and each user is represented by an individual in the ontology). All connections in WebKB-2 can be traversed in both ways (e.g. each concept node with a certain type is accessible from that type) programmatically and from browsing interfaces.

The connections from/to creators are necessary for handling updates by multiple users and permitting each user to filter or focus on knowledge from certain users by referring to them with an identifier, one or several type or supertypes, or even a graph description. The alternative choice of storing knowledge from each creator in a different module would not permit as much flexibility in the management and filtering of knowledge from multiple creators (or it would be harder to implement).

The connections between categories and words are necessary to permit lexical freedom (any user can add new names to existing categories) and permit the use of words instead of category identifiers within graphs. This last feature spares the users the tedious work of looking for the identifiers of categories to build graphs (statements or queries). If the word used in a concept node refers to only one category, or if the other categories can be eliminated given the signature of the relations connected to the concept node, the relevant category can be found. Otherwise, the list of candidate categories can be proposed for the user to select. For a query graph, there is no harm in making an automatic choice and let the user refine the query if a wrong category has been selected.

2.4 Re-use of a natural language ontology

Links from a natural language ontology (e.g. WordNet[6]) form the backbone of a large shared KB. Such links permit WebKB-2 to relate, compare and retrieve knowledge representations. They also provide the user with various meanings for a word or various distinctions for a notion, most of which s/he would not have thought about, thence leading the user to enter more precise and comparable representations. For the same reason, provided semantic constraints are associated to the top level categories of the ontology, it permits some automatic checking on the users' statements and extensions to the ontology.

We initialized the current knowledge base of WebKB-2 with the content of the lexical database WordNet 1.6: 94,500 nouns and 66,000 categories referred by nouns (in accordance to our lexical conventions, we ignored information regarding verbs, adverbs and adjectives). Various kinds of links connect these categories: `specialization`, `exclusion`, `similar`, `member`, `part`, `substance`, and their reverse links. The interpretation of the links other than `specialization`, `exclusion` and `similar` is not always clear nor consistent within Wordnet. For example, a `part` link from the category `airplane` to the category `wing` could mean that "any airplane has for part at least 1 wing" or "all airplanes have for part the same wing", "any wing is part of a plane", etc. We assumed the first interpretation was correct for direct links (e.g. `part`, `substance`, etc.) and therefore opposite for their inverse links (`part of`, `substance of`, etc.). This interpretation is exploited in our graph comparison/retrieval algorithms.

[6] http://www.cogsci.princeton.edu/~wn/

We distinguished the Wordnet `specialization` links into `subtype` links and `instance` links by isolating 2900 individuals. We also made a few other structural corrections (e.g. we removed 3 redundant subtype links, 4 links with same source and destination, and introduced `location` links to replace some unfortunate usages of the links `subtype` and `part`). Finally, to permit knowledge representation and automatic checks, we inserted the WordNet top-level categories into a top-level ontology of about 100 concept types, and complemented it by 140 basic relation types signed on these top-level concept types.

To each WordNet category, we associated a unique key name using the first of the category names (in WordNet, the first name is the most common name used for referring to the category). When various categories shared a same first name, suffixes such as "/2" and "/3" were used to create unique key names. Then, we manually modified some of the key names to simplify the knowledge representation task (the name order in WordNet is based on name frequencies in some manually tagged corpora and therefore this order may not be adequate).

2.5 Flexible ways to refer to a category

Flexible identifiers and multiple interpretation modes are necessary for taking advantage of the connections between users, categories and names.

In WebKB-2, a category identifier is either an URL, an e-mail address or the concatenation of the creator identifier and the key name, e.g. `wn#domestic_dog`, `wn#time`, `wn#time/2`, `wn#time/3`, `pm#IR_system`. (Category identifiers with same key names but different creators refer to different categories and hence should represent different objects). A category identifier may also show all the names given by the creator, e.g. `pm#IR_system__information_retrieval_system` and `wn#dog__domestic_dog__Canis_familiaris` (to represent names given by other users, the "`name`" link – abbreviated by the character '`_`' – must be used). Given 95% of current categories in WebKB-2 comes from WordNet, the "wn" prefix may be left implicit, e.g. `#time`. More precisely, this is the case except *within graphs* when a list of default creators has been specified (e.g. with the command "`default creators: pm wn;`" in input files). For instance, if `pm` and `wn` are the default creators, the graph `[a #car]` is accepted if `pm#car` or `wn#car` have been declared. The order of the creators in the list is important (the first candidate category is preferred).

Words (i.e. category names) are simply entered as such, e.g. `domestic_dog` and `time`. Category names, instead of category identifiers, are accepted within graphs only if this option has been selected (e.g. via the command "use names;" in input files). Signatures are exploited for eliminating candidate categories. If there is more than one candidate, the parsing stops or issues a warning depending on an internal ambiguity acceptation level (for our main purpose, ambiguities should not be allowed but an application of WebKB-2 that requires an automated agent to be used as a knowledge provider will probably need to accept ambiguities). If ambiguities are accepted and a list of default creators specified, WebKB-2 exploits this list to select the best candidate category.
Apart from signatures, type constraints explicitly associated to categories within

a graph may be used for guessing categories. For instance, the graph `[a transformation \\pm#process]` means: there exists an individual instance of a concept type that has "transformation" as one of its names and that is a subtype of `pm#process`. The type constraints permit WebKB-2 to eliminate the two other senses proposed by WordNet for "transformation": the mathematical function and the transmutation. Top-level types such as `pm#process` are proposed in WebKB-2 menus to help construct graphs.
For better readability, we will often use names instead of category identifiers in the example graphs of this article.

3 Mechanisms for cooperative editing of the KB

The WebKB-2 user is asked to be as precise as possible when making statements (to avoid conflicts and permit to answer queries more adequately). For instance, a user (lets say "user1") should not simply represent that "birds fly" (in FCG: "`[user1#birdsFly [any bird, agent of: a flight]]`") since this is not always true. If this happens, other users are encouraged to "correct" the information. In WebKB-2, any user can do this by connecting the "faulty" graph to a more precise version using a relation of type `pm#corrective_specialization` (then, depending on display options, the first version may or may not be filtered by WebKB-2 when answering queries). Similarly, if a user thinks a statement from another user can be generalized, s/he can use a relation of type `pm#corrective_generalization`. For example, if "user1" stated that "birds fly" and "user2" wants to correct and specialize that by "a study made by Dr Foo found that in 1999, 93% of healthy birds could fly", s/he can write:

```
[user1#birdsFly, corrective_specialization:
   [user2#93pcHealthyBirdsCanFlyAccordingToFoo
      [[93% of (bird, experiencer of: a good health),
          agent of #: a flying  //'#' means ''can''
       ], time: 1999], source: (a study, author: Foo@bird.org)]
   ]]
```

(Note: if a graph is not explicitly named, WebKB-2 generates a name for it).

We believe a scalable approach for cooperation between users of a knowledge base server implies that two seemingly incompatible goals are reached:
(i) each user should be able to represent what s/he considers true, and correct or complement other users' knowledge in a non-destructive manner, use the categories and names s/he wants (providing that lexical recommendations are respected and existing categories re-used or specialized), and should not have to discuss and find an agreement with other users each time a conflict arises,
(ii) knowledge from different users should remain consistent and tightly interconnected to permit comparison, search, cross-checking and optimal unification.

We have partly shown how these points can be achieved and that they are not incompatible, providing users connect their categories and graphs to other existing ones. **Removal/modification/addition protocols are also required for semantic conflicts to be managed asynchronously and**

without person-to-person agreement. The following four points describe our approach.

1) A user may **remove a category, link or graph** only if s/he has created it and unless this induces an inconsistency in the user's knowledge. If the category, link or graph being removed is used by other users or is necessary for their knowledge to remain consistent, it is actually not removed but its creator is changed to one of the users relying on its existence. In WebKB-2, inconsistency detection currently only exploits relation signatures and exclusion links. However, we plan to exploit inconsistencies detected by users and signaled with a relation of type `pm#contradiction` between two graphs.

2) The creator of a category may **modify a link** connected to this category – so that the link uses an alternate category – unless this modification induces an inconsistency. The creator of a relation type may modify its signature unless such a change induces an inconsistency (in which case, s/he must first modify the ontology or related graphs so that the inconsistency disappear). A user may **not modify a graph** that s/he has not created but s/he can connect it to another graph via a relation of type `pm#corrective_specialization`, `pm#overriding_specialization`, `pm#corrective_generalization` or `pm#correction`. This last relation type should only be used when the ontology cannot be modified (or another relation type used) for correcting the first graph. Since graphs can be used for representing links these three relation types may also be used to state alternate links. Depending on display/filtering options, corrected graphs or links are displayed/used for inference or not.

3) A user may **add a graph or a link** (even if s/he is not the creator of the linked categories) unless that induces an inconsistency or a redundancy (for consistency and re-use purposes, WebKB-2 does not accept a graph that already has a specialization or a generalization in the KB; please see Annex 1 for details). If this happens, the user must either refine his/her graph before re-trying to add it, modify the ontology or use one of the four "corrective" relations cited above.

4) In any of these previous cases, when the knowledge of a user is modified by another user, the change should automatically be e-mailed to the first user or presented to him/her the next time s/he logs on to WebKB-2.

An alternative approach would be to always allow the creator of a category to add, modify or remove the categories or links s/he has created *even* when that induces an inconsistency in other users' knowledge. Then, the inconsistency would have to be repaired automatically. Since the update means a change of interpretation of a category (at least from the viewpoint of the other users), a way to repair the inconsistency would be to "duplicate" the categories and links that should not be modified in order to avoid the inconsistency (i.e. the modified category and some of its subtypes from the same user). The "duplicates" would be attributed to other users. Although this approach would allow each user to ignore how his/her categories are used by other users, it is far less optimal than manual corrections, reduces cooperation between users and the tight interlinking of their knowledge. It is also complex to implement and cannot be extended to handle graph modifications in a similar manner.

4 Search mechanisms

4.1 Searching categories and links

Fig. 1 shows a WebKB-2 interface for searching categories or links according to a category identifier or name, and/or a link connected to the result categories and an optional destination. The parameters shown in Fig. 1 specify a display of the category `pm#thing` (the uppermost concept type) and all its direct or indirect subtypes created by the user `rdf` or by users that are members of the KVO group (`M pm#KVO_group`) but not from `f_modave` and any Australian (`^ #Aussie`). In the current KB, these filtering constraints resolve to the users `rdf` and `pm`. Subtype links and categories that do not belong to these users are explored but not shown (though increases in the indentation show the user how many intermediary categories have not been displayed). Fig. 2 shows the result in the default format. The characters '`!`', '`^`' and '`>`' respectively represent links of type `exclusion, instance of` and `subtype`.

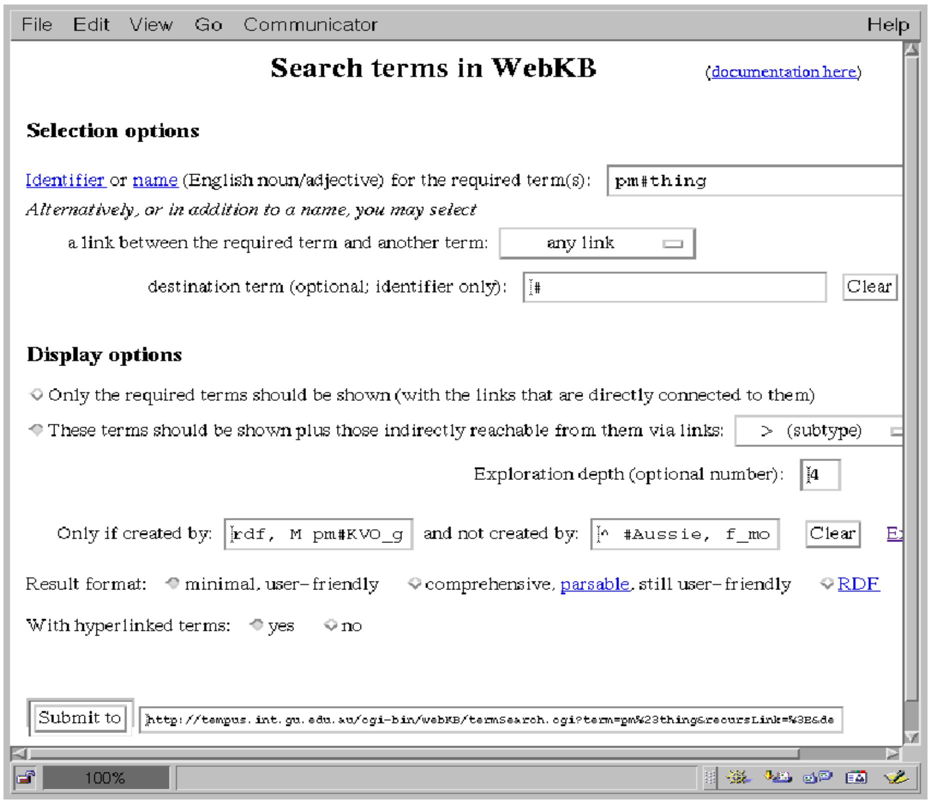

Fig. 1. Query for the subtypes of pm#thing that belong to the user "pm"

Fig. 2. Result of the previous query

4.2 Searching graphs

Classic searches for graph specializations permit searches "by the content" but need to be extended for more flexibility in the formulation of the query graph and to increase the number of relevant answers. WebKB-2 uses four extensions.

1) Let us assume the KB includes the graphs `[John, owner of: a car]` and `[John, owner of: an appartment]`. A classic search for graphs specializing the query graph `[a man, owner of: a car, owner of: a lodging]` would not retrieve the previous graphs since only the union of these specializes the query graph. When WebKB-2 looks for specializations, it also looks for other graphs including coreferent categories: identical individuals, identical types universally quantified or using the same coreference variable. If they permit to answer the query graph, these different graphs are displayed separately – joining them would often not produce a meaningful graph (e.g. their embedding graphs could not be joined). E.g. 2 other graphs that could be presented in answer to the previous query are:

```
[ [[Tom \\IBM_employee, owner of: an apartment], time: 2000], author: Tom]
[ [any IBM_employee, owner of: a car], author: IBM]
```

2) Searches should also take into account knowledge represented via links instead of graphs. For instance, let us assume the categories representing the geographical areas "Gold Coast" and "Southport" are connected via a `part` link and the knowledge base includes the following graph.

```
[philippe.martin@gu.edu.au, agent of: (the renting,
         object: (an apartment, part: 1 bedroom, location: Southport),
         instrument: 140 Australian_dollars, period: a week,
         beneficiary: Spirit_Of_Finance)]
```

WebKB-2 exploits the ontology to present this graph in answer to the query graph `[an apartment, location: (a district, part of: Gold_Coast)]`.

3) Let us assume the graph `[John, owner of: a lodging]` is in the knowledge base and a query graph is `[a man, owner of: an apartment]`. The first graph is not a specialization of the query graph since `wn#housing/2__lodging` is a supertype of `wn#apartment__flat` not the reverse. However, a user may want such a graph to be provided. This is why WebKB-2 provides two graph search commands: "spec" to search specializations of the graph given in parameter, and "?" to search graphs comparable to the one given in parameter. With the second command, supertypes of categories in the query graph are also used. The first graph would not answer the query "`? [a man, owner of: a bike]`" since `wn#housing/2` is neither a subtype nor a supertype of `wn#bicycle__bike`.

4) Structural flexibility should be permitted in query graph specification. We believe the simplest way (both for the user and from an implementation perspective) is to allow the specification of path sequences with common regular expression operators ("*" for "0, 1 or many times", "+" for at "at least 1 time", "?" for "0 or 1 time"). Let us assume the following graph is in the KB.

```
[philippe.martin@gu.edu.au, agent of:(a research, within_group: KVO_group)]
```

Users looking for a person conducting research at "Griffith Uni., Gold Coast campus" are unlikely to find this graph via classic searches for specialization only. However, since `pm#School_of_IT_at_Griffith_Uni_Gold_Coast_Campus` is connected via a `part` link to `pm#KVO_group` and via a `location` link to `QLD#GCcGU__Gold_Coast_campus_of_Griffith_Uni`, and since `pm#relation` is the uppermost relation type, it should be possible to find this graph with:

```
     spec [a person, agent of: (a research, relation+: GCcGU)
or:  spec [a research, (relation: a thing)+ location: GCcGU)
or:  spec [a research, relation 3+ (part of: a group)3+ location:GCcGU)
```

("3+" means that at most 3 relations of the specified type should be traversed).

Fig. 3 shows one of WebKB-2's interfaces for searching graphs. Names, instead of category identifiers, have been used and "pm" has been specified as the creator of the graphs to retrieve. Fig. 4 shows the result. It first indicates that 2 categories share the name "Gold_Coast" and that the first has been selected. Then, a graph ("with hyperlinked categories") answering the query is presented.

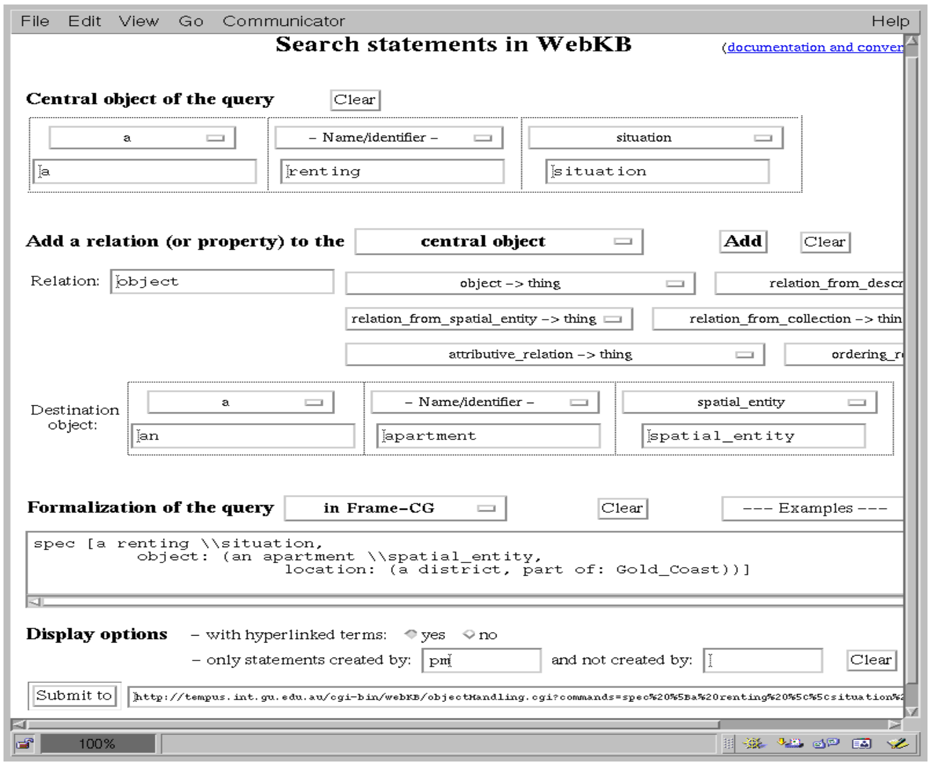

Fig. 3. Query for the specializations of a graph

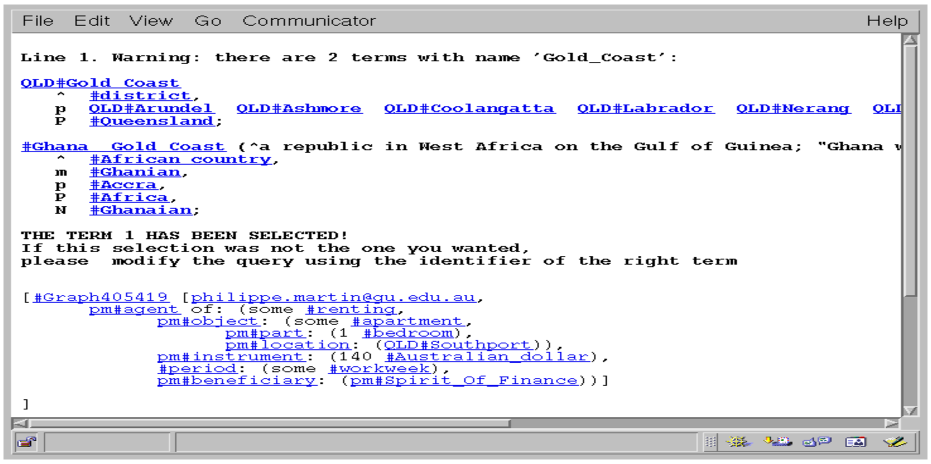

Fig. 4. Result of the previous query

5 Comparison with other tools

Guarino *et al.* [2] developed an information retrieval system called Ontoseek that exploits the WordNet lexical database and simple existential conceptual graphs to store the content of Yellow-Pages like catalogs and permit their access in a flexible way. They show that structured content representations coupled with linguistic ontologies increase both the recall and precision of content-based retrieval. More exactly, Ontoseek re-uses Sensus[7] which mostly includes WordNet and the Penman top-level ontology[8]. It is unclear from [2] whether or not users can modify this ontology but they apparently can enter simple existential conceptual graphs via the interface or ask/tell communication protocols. Classic searches for specializations are performed and queries may use names instead of categories. It is unclear whether structural constraints in the ontology are exploited to guess adequate categories and if there are actual relation types. WordNet types which can heuristically be identified as "role types" (or types for "relational nouns") may be used as relation types (this is also the case in WebKB-2).

Thus, WebKB-2 has similarities in intent and approach with Ontoseek. However, we believe the notation proposed in Ontoseek is insufficient for a precise or adequate representation of Yellow-Pages like catalogs with detailed descriptions of products or services. Precision or correctness in the representations may not be that important for Ontoseek since the knowledge is only intended to be used as an index for products in a catalog (not for re-use or unification with knowledge from many users) but WebKB-2 requires expressive notations, the handling of multiple users, and knowledge representation conventions. We have also shown in the previous section the insufficiency of classic searches for specializations.

WebKB-1 and WebKB-2 can be called "ontology servers", i.e. Web servers that permit users to build and publish ontologies. Most ontology servers also permit the construction of existential graphs and therefore could be called "knowledge base servers" but the possibility of modifying the ontology is a rarer feature. WebKB-1 and WebKB-2 are two opposite extremes in the handling of cooperation between users: while most other ontology servers (e.g. the Ontolingua ontology server[9], Ontosaurus[10], Ikarus[11], Tadzebao and WebOnto[12]) store the knowledge of users in independant modules/files on the server disk, WebKB-1 uses Web-accessible files stored by users on their own disks and WebKB-2 stores the knowledge of users in a single knowledge base on the server disk. Some ontology servers, e.g. the Ontolingua server or Ontosaurus, permit any user or a group of users to edit the module but, apart from locking/session mechanisms, no particular support for asynchronous cooperation is generally provided: no record of creators for categories/links/graphs, no conventions or protocols, etc.

[7] http://www.isi.edu/natural-language/projects/SENSUS-demo.html
[8] http://www.darmstadt.gmd.de/publish/komet/gen-um/newUM.html
[9] http://WWW-KSL-SVC.stanford.edu:5915/
[10] http://www.isi.edu/isd/ontosaurus.html
[11] http://www.csi.uottawa.ca/ kavanagh/Ikarus/IkarusInfo.html
[12] http://ksi.cpsc.ucalgary.ca:80/KAW/KAW98/domingue/

An exception we know of is the Co4 system[13] which has protocols modeled on submission procedures for academic journals, i.e. on peer-reviewing, resulting in a hierarchy of knowledge bases, the uppermost containing the most consensual pieces of knowledge while the lowermost ones are the knowledge bases of each user. This approach certainly leverages some problems of module-based approaches but would doubtly scale to large knowledge bases or a large number of users. The Ontoloom/Powerloom authors mainly rely on knowledge comparison procedures and the pre-existence on a large ontology to guide and check users in their extension of a unique knowledge base.

Modules are an easy way to delimit knowledge about a particular subject and handle competing formalizations, but since categories between modules are generally not inter-connected, automatic comparisons of knowledge representations from/re-using different modules is unlikely to succeed. For the same reason, even when general descriptions of the content of modules are made using graphs, the selection of adequate modules to re-use or search is a difficult task. From a knowledge retrieval point of view, the indexation of knowledge according to some knowledge domains or other characteristics is a coarse-grained approach. In WebKB-2, this selection problem does not exist: categories are tightly interlinked, and each link or relation in the knowledge base may be used as an index for retrieving a relevant piece of knowledge, thus permitting to take into account any combination of characteristics specified in a query not just combinations given by knowledge providers in their general indexations.

Compared to other large scale KBMSs, a notable feature of WebKB-2 is that the ontology is large and can be dynamically/interactively modified by the users (no lengthy re-compilation phase or graph re-indexation is necessary). This feature is shared by the Parka-DB system[14] which we considered to re-use for implementing WebKB-2. We also considered the SHORE deductive database[15] as well as standard relational databases. However, we found more flexibility and programming ease was provided by the free-to-use object-oriented main-memory database system called FastDB[16] (in case the database grows larger than 4Gb (on a 32 bit system), FastDB can be replaced with a disk-based version called GigaBASE[17]). In Parka-DB the ontology is also entirely loaded in memory but the graphs remain on disk. Large-scale CG systems also begin to appear. Santiago[18] is based on the MG database system[19] and requires a re-indexation phase each time a type is added. In Bernd Groh's relational database based CG system[1], no re-indexation phase is necessary.

[13] http://ksi.cpsc.ucalgary.ca/KAW/KAW96/euzenat/euzenat96b.htm
[14] http://www.cs.umd.edu/projects/plus/Parka/parka-db.html
[15] http://www.cs.wisc.edu/shore/
[16] http://www.ispras.ru/ knizhnik/fastdb.html
[17] http://www.ispras.ru/ knizhnik/gigabase.html
[18] Gerard Ellis' system. See CG mailing list.
[19] http://www.mds.rmit.edu.au/mg/

6 Conclusion

We have presented an approach permitting Web users to search and cooperatively build a shared knowledge base, and engineered a system supporting this approach[20]. The approach permits and relies on knowledge re-use and interconnections at a local level, e.g. categories are connected to names, creators and other categories, while concept nodes and graphs are interconnected via relations or the categories they re-use. In coarser-grained approaches, these connections are often not represented (and, we believe, more difficult to represent in a manageable way) and therefore cannot be automatically combined to permit knowledge comparison or more relevant and complete knowledge retrieval. We also described our approach to permit asynchronous cooperation, and necessary extensions to classic searches for specializations.

Entering information in WebKB-2 is more difficult than entering sentences in a document, but information from documents cannot be interconnected to respond to precise queries and is therefore lost to most people. We believe that entering information in WebKB-2 is easier than in most other systems thanks to the adopted notations, the initialisation of the knowledge base with WordNet and our top-level ontology, and the possibility of using everyday words instead of category identifiers. Some information remain difficult to represent precisely but we think that WebKB-2, or an extension of it with nicer interfaces, could be used by Yellow-Pages-like-services or community servers to permit people to advertize products and services or publish information.

References

1. Groh, B., P. Eklund.: Algorithms for creating relational power context families from conceptual graphs, In ICCS'99, 7th International Conference on Conceptual Graphs, Springer Verlag, LNAI 1640, pp. 389–400, 1999.
2. Guarino, N., Masolo, C., Vetere, G.: Ontoseek: Content-based Access to the Web. In: IEEE Intelligent Systems, Vol. 14, No. 3 (1999) 70–80
3. Martin, Ph.: The WebKB set of tools: a common scheme for shared WWW Annotations, shared knowledge bases and information retrieval. In: ICCS'97, 5th International Conference on Conceptual Structures, Springer Verlag, LNAI 1257 (1997), 585–588. URL http://meganesia.int.gu.edu.au/~phmartin/webKB/doc/papers/cgtools97/
4. Martin, Ph., Eklund, P.: Embedding Knowledge in Web Documents: CGs versus XML-based Metadata Languages. In: ICCS'99, 7th International Conference on Conceptual Structures, Springer Verlag, LNAI 1640 (1999) 230–246. URL http://meganesia.int.gu.edu.au/~phmartin/WebKB/doc/papers/iccs99/iccs99.ps
5. Martin, Ph., Eklund, P.: Conventions for Knowledge Representation via RDF. In: WebNet2000, ACCE press 378–383. URL http://meganesia.int.gu.edu.au/~phmartin/WebKB/doc/papers/webnet00/
6. Martin, Ph.: Conventions and Notations for Knowledge Representation and Retrieval. In: ICCS'00, 8th International Conference on Conceptual Structures, Springer Verlag, LNAI 1867 (2000) 41–54. URL http://meganesia.int.gu.edu.au/~phmartin/WebKB/doc/papers/iccs00/iccs00.ps

[20] Accessible at http://meganesia.int.gu.edu.au/~phmartin/WebKB/shared.html

7 Annex 1

Note: this annex was not part of the article published in the ICCS'01 proceedings but is constituted by two of the slides used for presenting the article at this conference.

The WebKB-2 user may not add a graph `g1` if it contradicts, generalizes or specializes an existing graph `g0` without connecting `g1` to `g0` via a relation of type `pm#corrective_generalization`, `pm#corrective_specialization`, `pm#correction` or `pm#overriding_specialization`. There is one exception: when `g1` instantiates `g0`.

For example, consider Fig. 5 where some statements are represented in Formalized English (FE) and exclusion/specialization/instantiation relationships between them are given. A user will not be allowed to enter "no bird can be agent of a flight" or "2 birds can be agent of a flight" if the statement "at least 1 bird can be agent of a flight" already exists in the KB. Assuming that `pm#AtLeast1birdCanBeAgentOfFlight` is the identifier of this statement, the user should enter: `pm#AtLeast1birdCanBeAgentOfFlight has for corrective_specialization 'no bird can be agent of a flight'` or `pm#AtLeast1birdCanBeAgentOfFlight has for correction '2 birds can be agent of a flight'`.

However, a user may enter "Tweety can be agent of a flight" even if the statements "2 birds can be agent of a flight" or "any bird can be agent of a flight" already exist in the KB because this is what we call an "instantiation": the new graph just gives an example or occurence of a more general statement (there is no potential conflict between the authors' respective intentions).

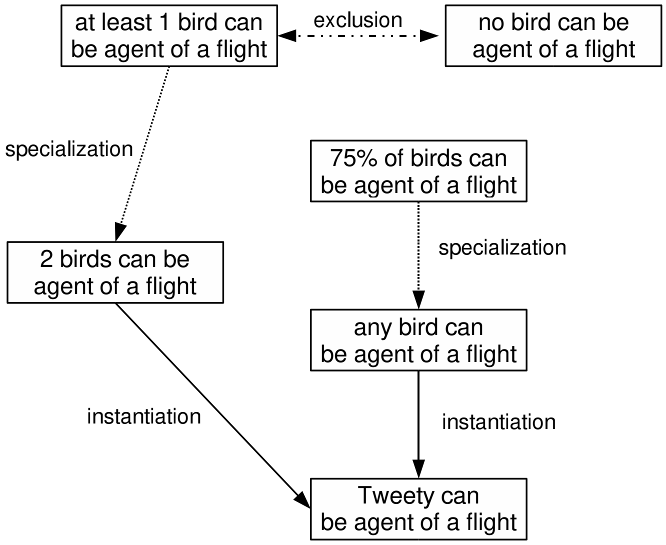

Fig. 5. Explicit connections between graphs are required when exclusion/specialization (but not instantiation) relationships are discovered by WebKB-2