An Overview of Memory: Some Issues on Structures and Organization in the Legal Domain

Fabiano M. Hasegawa
Bráulio C. Ávila
Emerson L. dos Santos
Celso A. A. Kaestner

Pontifical Catholic University of Paraná — PUCPR
R. Imaculada Conceição, 1155 — 80.215-901 — Curitiba — PR — Brazil
{fmitsuo, emerson, avila, kaestner}@ppgia.pucpr.br

Abstract

Lawyers often need to look for previous similar legal cases when analysing new ones. The more previous cases, the more time is spent. Classical search engines execute term-based retrieval, which may miss relevant documents as well as fetch several irrelevant ones, causing lack of useful information and waste of time. Ideally, retrieval should be meaning-based. Humans beings are able to do efficient searches due to their knowledge. Therefore, semantic search requires knowledge. This paper presents a semantic search engine. Along the paper, several issues concerning specially knowledge representation and memory are discussed. A formalism based on models of comprehension is introduced, as well as its motivation. Examples of representation of sentences in natural language from the Legal Domain are provided. The search engine and its architecture, based on domain knowledge, are briefly commented. The main goal is to give legal offices the opportunity to save time by providing a more suitable document retrieval.

Keywords: Semantic Representation of Legal Documents, Semantic Document Retrieval.

CACIC 2003 - IV Workshop on Agents and Intelligent Systems.

1 Introduction

In legal offices, document analysis is a common task. New cases are analysed according to similar previous ones. Whatever is the goal, a new case needs to be compared to old ones. Therefore, in legal offices, daily tasks involve searching in a case base. The conventional approach is based on manual search and analysis, where semantic is highly required. Semi-automatic alternatives can be offered by classical search tools, where only a set of documents containing some terms of a query — provided by the user — are manually analysed. However, classical search tools are based on statistic and mathematical approaches: the set of documents retrieved is, thus, based on terms rather than meaning.

The manual approach provides, indeed, the highest accuracy. Nonetheless, resources such as time and money make it expensive and, consequently, a bad idea whenever effectiveness is required. On the other hand, semi-automatic approaches may miss interesting documents and/or retrieve lots of unuseful ones. A lawyer might fail to achieve her/his goal due to missing information. Unuseful information might jeopardise the efficiency of the tool, whereas the user might be required to analyse manually a large amount of documents — what would be almost the same as the trivial manual search.

As fully effective automatic methods are not available yet, a feasible alternative would be to try to improve the results obtained with current approaches. Nonetheless, since classical techniques are doomed to fail [9], it doesn't seem reasonable to try to improve those techniques. Instead, new approaches should be devised. Those ones should be knowledge-oriented, giving a computer the opportunity to reason in a human-like fashion [3]. Since humans use their knowledge to select the desired documents from a whole collection, a computer might be able to retrieve documents through the use of human knowledge.

Considering a static base, there are three major issues to be designed: how knowledge is to be represented, how it is to be retrieved and what will be represented. Representation concerns how structures are defined and how they are organized globally. Retrieval must find the structures available most similars to an arbitrary input. Finally, there must be some coherence about content: models of comprehension of the real world are required. Thereby, a semantic search engine depends greatly on its knowledge base, or memory.

In this work, some issues concerning a semantic search engine are presented. This paper focuses mostly on memory, presenting formalisms as well as examples of knowledge representation. All the knowledge represented concerns natural language. Also, there is a short explanation about the reasoning involved in the engine itself.

In Section 2, a formalism for knowledge representation is introduced, as well as influences from which it was derived and motivation for its use. Section 3 concentrates on discussing how to use the models introduced along with classical ones to understand the real world and to represent natural language. Next, Section 4 presents some sentences from the Legal Domain and their respective representations; issues that motivate those representations are also discussed. The search engine and its architecture are briefly described in Section 5. Last, Section 6 shows conclusions and points out some suggestions for future work.

2 Memory Structures and their Organization

Knowledge is stored in memory. Humans are endowed with dynamic memories [8]; it is very difficult to provide dynamic memories to computers, though — specially if natural language is involved. An alternative, though not as suitable as a dynamic memory, is to provide static

memories to computers[1]. Even though knowledge acquisition is not a trivial task even when made manually, once knowledge has been acquired several sorts of inferences can be triggered as well as tasks accomplished. A static memory is, in this view, a repository of information which, associated to processes for retrieval and use, provides knowledge.

In order to build up a memory, some issues must be well defined. Firstly, one must consider how that memory will be structured — how a single unit of memory is to be represented. In this work, knowledge is associated with natural language, so the formalism for representation should be able to take into account natural language. Whatever was the formalism, inferences would be further triggered; thus, ease, feasibility and experience leaded this work to Conceptual Dependency (CD) [6, 9]. Some advantages of CD are:

- Two different sentences in natural language which share the same meaning have both a unique representation in CD;
- The availability of Conceptual Analysers (CAs) in the literature [2] which can be rapidly developed[2].

As will be seen in Section 3, CD did not provide enough semantic for the content that should be represented. However, the structure of a unit of memory remained the same, as exposed in (1), where:

- *predicate* identifies a type of structure, which packages a list of *Slots*[3];
- A *slot* has the form $role(filler)$ and expresses a perceptible item in a structure along with its value;
- $role_i$ identifies the perceptible item i in a structure;
- $filler_i$ is the value of the perceptible item i — which can itself be another structure — in a structure.

$$predicate([role_1(filler_1),\ \ldots,\ role_n(filler_n)]) \tag{1}$$

Since the value of a perceptible item i in a structure s_1 can be another structure s_2 , some structures will obviously be aggregating others; therefore, structures can link to each other. Thus, basic structures can be packaged by top-level ones. Moreover, *fillers* can contain patterns, what allows the use of matching constraints: if a *filler* contains a pattern, only instances which match that pattern can fill the respective slot; if a *filler* is unbound, no constraint is applied.

So far, the syntax of memory structures has been presented — a common syntax in the AI domain. Much of the organization issues are managed by the structures themselves, once top level structures can package secondary — but not less important — ones[4]. Nonetheless, there is still one missing organization issue: hierarchy. Hierarchies provide an ontology for structures which can be used to, amongst other things, provide more abstract matching constraints. A set of structures which share some semantic behaviour can be grouped under an unique abstract

[1]It is not an easy task either, but not as hard as in the dynamic case.

[2]CAs are programs which translate sentences expressed in natural language into sentences expressed in CDs

[3]Empty lists are allowed.

[4]Secondary, in this point of view, means a structure whose *fillers* are not complex structures. Nevertheless, there is no reason to think simple structures are not important, since complex structures would not exist without them.

class. Besides, a single unit can be an instance of more than one abstract class. In this work, the only purpose of such a representation — also very familiar to the computer science community — was to provide abstracts patterns for *fillers*.

Hierarchies were not explicit in the formalism presented. In this work, they were expressed as pairs $(s,\ [a_1,\ \ldots,\ a_n),\ n\ >\ 1$, where s is either a specialization or an abstraction and a_i is an abstraction. The use of abstractions can be better understood in Subsection 5.1, where some *slots* are filled only with objects which respect constraints.

This framework was intended to be both flexible and clear. Abstractions are not directly defined within a structure because once predicates are known to be actually types of structures, it would be wasteful to define the same abstractions for every single structure. As a table, the correspondence is made only once for each type of structure — since one assumes all structures of the same type share always the same behaviours.

3 Memory Modeling

In Section 2, the formalism for memory representation was presented, where units of memory are represented as structures that can link to each other. Also, an hierarchic ontology provided generazilation capabilities. Nonetheless, this framework does not provide any hint about how the world must be represented. Knowledge representation, specially concerning natural language, is in fact a memory modeling problem. Natural language and the orientation towards a goal constitute two major problems of a memory model [11]. Fortunately, it is possible to identify some typical and recurring problems within a domain and through domains [10].

The more accurate a memory model reflects real world pictures (observations), the cleaner and the more useful the representation will be. Unfortunately, classic knowledge representation formalisms try to adapt the world view to their modeling heuristics, while the sensible way would be to adapt the modeling heuristics to world views instead.

Firstly, the world is often represented as a set of objects linked to each other. This could be perfectly suitable for geoprocessing systems, for example, but do not provide enough knowledge for intelligent agents which aim to interact in a real environment and to be able to make decisions over time. Here a distinction needs to be made: a static memory is not a memory which cannot account for time evolutions. A static memory means a memory which does not change over time; consequently, it is not able to learn[5] Therefore, a static memory can — actually should — account for time evolutions. As long as there are structures in memory which can follow the course of some events over time, those structures are able to recognize sequences of events and, thus, able to understand what goes on in an arbitrary scenario. Most modeling methods do not believe a sequence of events is worth storing. Having no structures which account for world changes makes a memory not able to recognize potential time-dependent relationships and patterns; concerning interaction, what one gets is only meaningless reactions.

Thereby, situations do occur within time windows which must be well expressed in memory structures which are meant to understand those situations. Even though a memory is not able to learn, it can certainly understand world changes if it is properly prepared to. Understanding time evolutions makes a memory able to provide causes and predictions about a known event.

Whereas a memory needs to account for time evolutions if it is meant to provide meaningful structures for an agent interaction, for instance, there is a great change in the world view: world

[5] A memory which does change over time and is, then, able to learn is called a dynamic memory, as seen in Section 2.

is not a set of objects any longer, but a sequence of observations or pictures. Some sequences may provide the information needed to the identification of events, while others may provide the possibility to identify states. In this view, objects are only a specific set of states which are, for the sake of simplicity, referred the way they are. Notice, it could be postulated objects do not actually exist in nature[6], but do in human minds, considering conscious understanding. Thus, the so-called objects are better understood as states, or a more abstract type of state.

Recall that there is not objection to the way people see the world; actually, what must be undestood is that the way people see it consciously does not need to be the way human memory structures *de facto* represent it and, consequently, does not need to be the way intelligent systems must represent knowledge. Hence, an object is just they way people consciously understand a state, which may aggregate other states. Again, for the sake of simplicity, some time down this hierarchy a state may aggregate structures which could be named objects. In this case, it is only a granularity threshold, instead of a major modeling issue.

The result of this orientation is a set of tangled structures dominated by complex ones which account for changes over time in the representation of situations. The world is not any longer viewed as a set of objects but as an environment where observations are taken from, which somehow may depend on previous observations and provide predictions for future ones [8]. Under this view, the main issues for memory modeling are transferred from elements to events and this is the way a knowledge engineer should consider the world in her/his task of modeling knowledge.

In this work, some of the issues discussed so far were applied, though in a restrict manner. In Figure 1, time is clearly represented: the abstract concept `implication` packages the concepts `state` and `state transition`. It is perfectly clear there was a state transition. If, otherwise, `implication` packaged only the final state of the transition in its *consequence* slot, that transition would be only meaningful for human beings, not for computers. A common error in memory modeling is to forget computers do not know what people know, mostly when quite trivial stuff is dealt with.

Another serious mistake is to believe objects contain states. States can be applied to other states and, if one assumes some structures to be objects in order to simplify a model, to objects. In this view, if a state is determined by a rule, deactivating the rule makes the dependent state disappears, but nothing happens to whatever it was applied to — either a state or object. In both Figure 1 and Figure 2, states are concepts which package what they are applied to and a description of the state itself.

There are two more issues worth cited in Figure 1 and Figure 2. Firstly, some concepts are quite intuitive, but are worth representing, such as `critic`, `decision` and `hypothesis`. There is a great load of knowledge in each of them, what makes them hard to be represented properly. A single concept can account for some basic needs though. Secondly, specially when natural language is involved, concepts such as `not` and `or` are needed. They are not regular concepts, as they do not offer a meaning themselves; actually, they evaluate for concepts in the domain of the concepts they are packaging. They are better understood as operators.

Shortly, the main issue in memory modeling is to try to see the world the way it is, instead of they way one consciously sees it. Besides, the world is not a set of objects, but an environment where people collect observations (perceptions) which often have time relationships. This view is able provide notions of past and future, besides taking into account classical ones, such as objects and their parts. The gain is verified in the potential knowledge obtained with predictions and explanations upon either an event or a sequence of events. In addition, the flexibility of this

[6]Perhaps they do in the quantum level.

technique allows the representation of much more information, making it possible to provide much more knowledge in an specific context than classical techniques used to.

4 Case Studies: Representing Legal Documents

In a legal office new cases are guided by previous cases. Thus, an Information Retrieval (IR) system is required to increase the effectiveness in the analysis of previous cases. In this work, a conceptual IR tool based on Conceptual Analysis and Conceptual Dependence [5, 7, 4, 9] was built and an ontology[7] was created with terms of the domain as a knowledge source. The following examples demonstrate how a legal document can be modeled — only a significant part of the legal document was chosen to represent the whole document:

1. Sentence: "Hypothesis in which the faltering accord does not suffer from the referred omission." — Figure 1 for graphical modeling. The concept hypothesis aggregates the content "the faltering accord does not suffer from the referred omission". The concept `hypothesis` has an object `accord` in the state `faltering`. In this case, the state is just a form under which the object presents itself. The concept `suffer` was represented by a state upon which the operator `not` acts. This state, in turn, is acting upon a concept representing an act of omission. In this example, the following concepts were used:

   - `hypothesis`: a concept which has an object as the content of the hypothesis;
   - `not`: an operator which produces the negation of a concept;
   - `state`: an observation of an object or another state in a certain moment in time, providing the notion of a static behaviour;
   - `omission`: a concept which represents the notion of an act of omission;
   - `accord`: a concept which represents a type of legal document;
   - `faltering`: this concept represents and describes the state faltering.

2. Sentence: "Embargos of declaration against decision which does not contain obscurity, contradiction or omission are inconsistent." — Figure 2.

   - `implication`: this concept represents a relation between two other concepts where one is the cause for the ocurrence of the other;
   - `embargo of declaration`: a concept which represents a special type of embargo;
   - `critic`: represents the notion of criticism, where an argument criticizes a theory;
   - `or`: operator of disjunction;
   - `obscurity, omission, contradiction`: represent their respective notions;
   - `inconsistent`: a concept which describes the state of the concept `declaration embargo`.

The graphical models facilitate the creation of the respective entries in the ontology — which are sometimes fairly complex, as seen in Figure 2 —. The model helps to define what

[7] In the original paper, the term ontology is referred as memory.

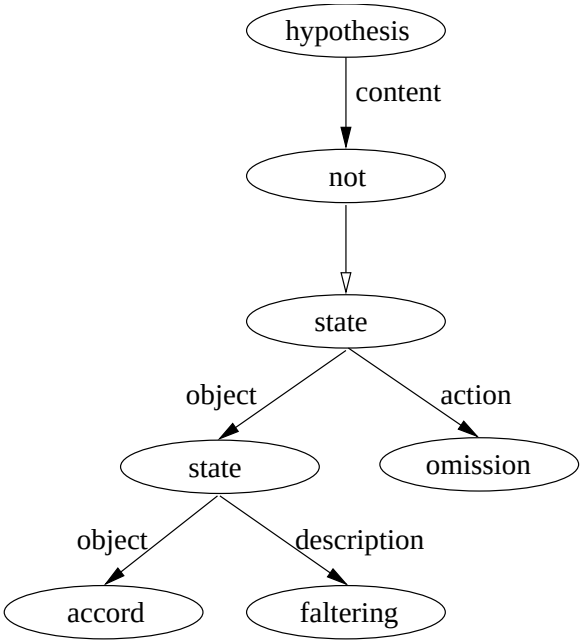

Figure 1: First sentence's conceptual model.

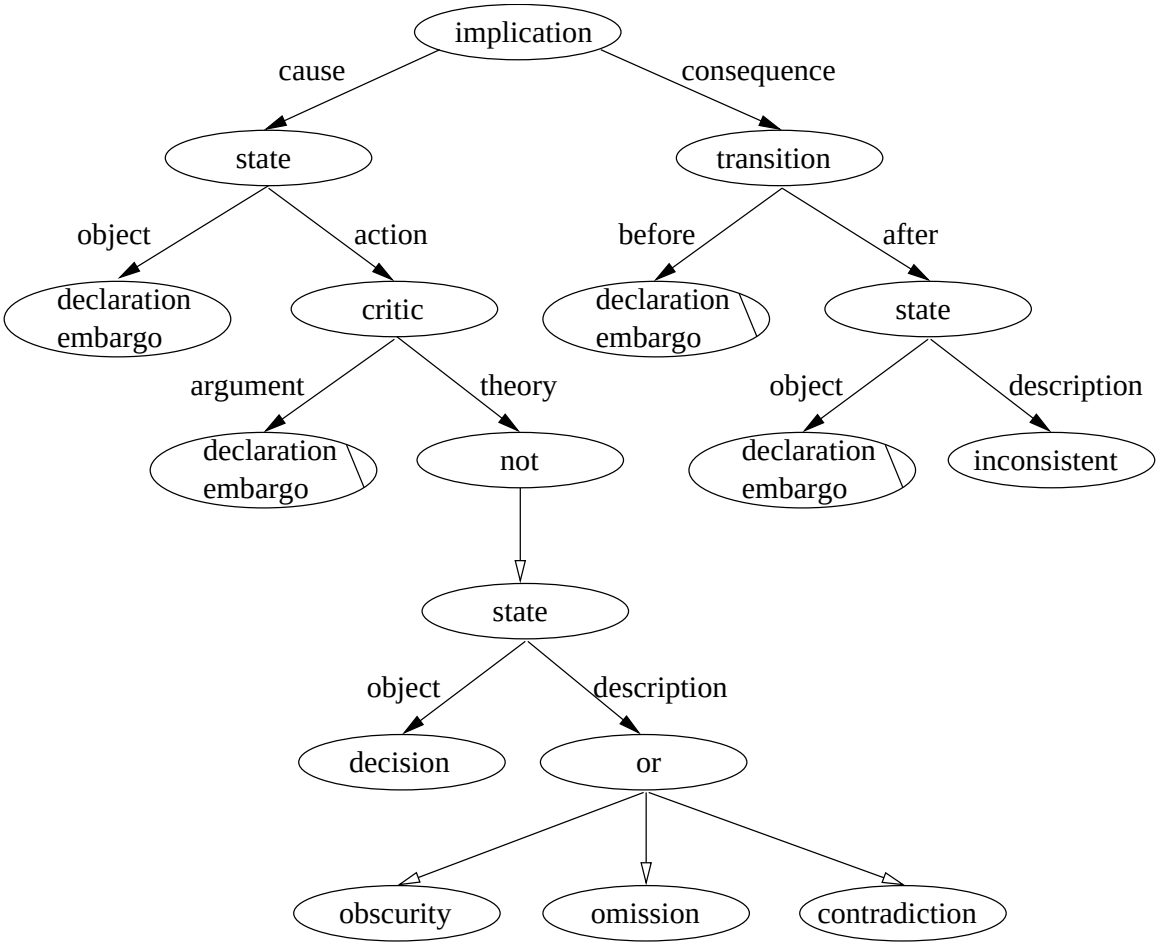

Figure 2: Second sentence's conceptual model.

kind of request[8] will be needed, what the request will do when the term is read and how the term will eventually be represented. The following example demonstrates a piece of an ontology for CA coded in Prolog:

```
word(declaration,
     [request(1,
      [before(mental_object, O)],
      [set_slot(O, typo(declaration([]))),
       set_mode(normal)])]).
word(hypothesis,
     [request(1,
      [],
      [add_cd(hypothesis([]), CD),
       set_mode(noun_group)]),
      request(1,
      [after(lexical_item, which)],
      [activate([request(1,
      [after_destructive(mental_object, C)],
      [set_slot(CD, object(C))])])])]).
```

5 A Semantic Search Engine

As seen in Section 4, an IR tool was built. That tool uses a conceptual search engine based on CD similarity, instead of classical search engines based on mathematical approaches. The theoretical basis and the system architecture will be explained shortly in the following subsections.

5.1 Conceptual Analysis

The Conceptual Analyzer (CA) is the parser that maps sentences in natural language into conceptual representations [2] — each term is read from left to right. Expectations perform an important role in CA: they are used to fill in the slots in the CD representing the meaning of the input sentence. Expectations or requests are triggers which may be activated whenever a term is read. Using the concepts available in the short-term memory, the triggers perform actions on CD structures. Each concept, after recognized, is stored in the short-term memory. If a request is not satisfied, it is placed in a queue to be satisfied later. The following example ilustrates the use of requests in Conceptual Analysis:

- Input sentence: "Jack read a book".
- Term "Jack": indicates a reference to a male human being named Jack. This reference is stored in the short-term memory under the token *jack*;
- Term "read": instance of the concept reading, which, in CD, is represented by the concept $mtrans([\ actor(\_),\ object(\_)])$ and supplies some expectations which will help to fill in the empty slots in the CD frame;

[8] A request is a kind of trigger in Conceptual Analysis [2].

  - requests for the slot *actor*: if there is some animated being before this concept in short-term memory, fill in the slot *actor* with it — this expectation (request) is satisfied and the resultant CD is $mtrans([actor(jack),\ object(\_)])$;
  - requests for the slot *object*: if there is some physical object after this concept, fill in the slot *object* with it — this request is not satisfied in this moment.

- Term "a": indicates that an instance of a concept should come next;
  - requests of the term "a": if a concept is found after this position in short-term memory, it is marked with *ref(indef)* (indefinite reference) — this request is not satisfied in this moment either.

- Term "book": indicates an instance of a concept which is known to be a physical object and a source of information. Now the request belonging to the term "a" is satisfied and the current concept is marked as *book([ref(indef)])*. Furthermore, the request of the concept "read" that was waiting to be satisfied too is indeed satisfied and the slot *object* is filled with *book([ref(indef)])*;

- Output: the CD structure *mtrans([actor(jack), book([ref(indef)])])*.

The success of the CA depends strongly on how the ontology is modeled; the consistency of the conceptual representations of the terms is crucial. A problem faced in this approach is the need to model a great number of domain and general terms — the CA works on each term of the sentence in order to map it into a conceptual representation. Once the ontology has been totally defined, the CA can map sentences of that specific domain into CDs. In this system, if the following sentences were applied to the CA, the resultant mapping would be:

- Sentence: "Hypothesis in which the faltering accord does not suffer from the referred omission."

```
hypothesis([object(not([object(state([object(state([object(accord([])),
                                                    description(faltering)])),
                                       description(omission([]))])])])])
```

- Sentence: "Declaration embargos against decision which does not contain obscurity, contradiction or omission are inconsistent."

```
implication([cause(state([object(embargo([type(declaration([]))]))]),
         action(critic([argument(embargo([type(declaration([]))])),
               theory(not([object(state([object(decision([actor(_),
                                                   object(_)])),
                 description(or([object1(or([object1(omission([])),
                                             object2(obscurity([]))])),
                         object2(contradiction([]))])])])])])])])),
  consequence(transition([before(embargo([type(declaration([]))])),
            after(state([object(embargo([type(declaration([]))])),
                                 description(inconsistent)]))])])])
```

The complexity of the ontology depends on the application domain.

5.2 Case-based Information Retrieval

Classical IR techniques use, as theoretical basis, statisticals and mathematics methods[1]. This paradigm has achieved satisfactory results but has not been even closer to the accuracy obtained by human beings. Human beings do not use statistical methods to understand a sentence in natural language or to retrieve documents; instead, they use knowledge [3].

Some problems may arise in classical IR tools, such as retrieval of documents which contain the terms of the query in different meanings. Some documents which do present the intended meaning are not retrieved because the terms of the query are different from the terms used in the document. These problems are diminished when semantic is properly used.

The search engine of the proposed system uses a CD similarity measure to determine how much the user query is similar to the sentences in documents. This similarity measure is applied recursively — a CD slot could be filled with another CD structure — on the conceptual structures. The similarity measure takes two elements of a CD for comparison:

- *predicate*: has higher weight in the similarity; if two CD's have both the same predicate, then their similarity is already relevant;
- *slots*: has lower weight in the similarity; both *roles* and *fillers* are compared.

Some rules were used to determine the similarity value. The results were satisfactory, but an ideal similarity measure mechanism using cases of comparison — each type of comparison between two CD structures would have a case indicating how the comparison should be made and which values of similarity should be used — might increase the results. This mechanism would itself be as complex as a CBR tool and that was not a goal in this work.

5.3 System Architecture

The system modules are depicted in Figure 3. An "ontology of use" is created by expanding the terms in the "user ontology" into a suitable representation for the Conceptual Analyzer (CA)[9]. Next, the module *XML2CD* is responsible for the conversion of the relevant part of a XML document into a CD representation[10]. A CD document base is then obtained[11]. These two processes are executed apart from each other and the searching process — although the creation of the "ontology of use" be fast, the generation of CD documents may require a large amount of processing time depending on the size of XML base.

When the user enters a query in natural language, that sentence is mapped into a CD structure — as seen in Section 5. Next, the search engine retrieves each CD document and compares each of them with CD format of the query. In the module *similarity*, the similarity between the document and the user query is returned[12]. Finally, the resultant documents are returned ranked by similarity.

[9]The "user ontology" utilizes a simple notation to facilitate its creation, however that is not the format used by the CA.

[10]The legal documents were originally in XML format.

[11]A document may generate various CD representations: the one with higher similarity is chosen to represent the whole document.

[12]The similarity is normalized from 0 to 1.

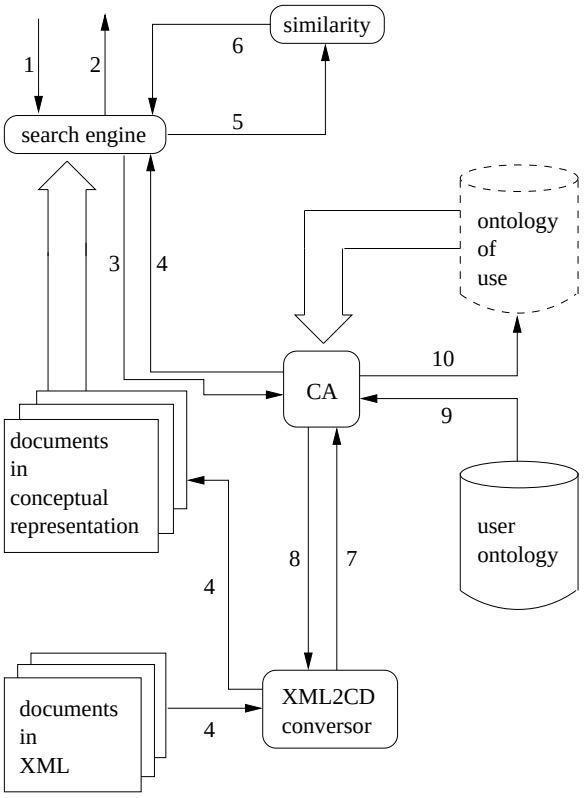

Figure 3: System Architecture.

6 Conclusions and Future Work

Memory modeling is the most important issue in semantic systems. Classical techniques used to understand the world as a set of objects. This view is not suitable for systems where interaction and events over time must be dealt with. Besides, building a memory model which properly represents the domain facilitates and clarifies the development of applications.

The effectiveness of the CA is directly determined by the ontology and, thus, by the memory. If the ontology isss well defined, the results will be satisfactory. In order to obtain a well defined ontology, the modeling used needs to be accurate and representative. The use of conceptual knowledge instead of statistic and mathematical methods in the search engine makes the results more precise.

Classical techniques of memory modeling based on objects have not been useful to provide meaning for interactive tasks and domains where the environment changes over time. The problem is centered in the conotation given to the interpretation of the world when models are built. Memory modeling should be oriented to take into account sequences of events and to

understand the world as an environment where agents collect perceptions (pictures) and might be required to act upon them.

References

[1] R. Baeza-Yates and B. Ribeiro-Neto. *Modern Information Retrieval*. ACM Press, New York, 1999.

[2] L. Birnbaum and M. Selfridge. Conceptual Analysis of Natural Language. In Schank and Riesbeck [9], chapter 13, pages 318–353.

[3] C. K. Riesbeck and R. C. Schank, editors. *Inside Case-based Reasoning*. Lawrence Erlbaum Associates, Hillsdale, NJ, 1989.

[4] S. W. Russel. Semantic categories of nominals for conceptual dependency analysis of natural language. Technical Report STAN-CS-72-299, Computer Science Department, Stanford University, Stanford, Jul 1972.

[5] R. C. Schank. Intention, memory and computer understanding. Technical Report STAN-CS-71-193, Computer Science Department, Stanford University, Stanford, Jan 1971.

[6] R. C. Schank. Conceptual Dependency: A Theory of Natural Language Understanding. *Cognitive Psychology*, 3:552–631, 1972.

[7] R. C. Schank. The fourteen primitive actions and their inferences. Technical Report STAN-CS-73-344, Computer Science Department, Stanford University, Stanford, Mar 1973.

[8] R. C. Schank. *Dynamic Memory Revisited*. Cambridge University Press, Cambridge, UK, 1999.

[9] R. C. Schank and C. K. Riesbeck, editors. *Inside Computer Understanding*. Lawrence Erlbaum Associates, Hillsdale, NJ, 1981.

[10] A. Valente, editor. *Legal Knowledge Engineering*. IOS Press, Amsterdam, Netherlands, 1995.

[11] R. W. van Kralingen, editor. *Legal Knowledge Engineering*. Kluwer Law International, The Hague, Netherlands, 1995.