On2broker: Lessons Learned from Applying AI to the Web

Dieter Fensel, Jürgen Angele, Stefan Decker, Michael Erdmann, Hans-Peter Schnurr, Rudi Studer and Andreas Witt

Institute AIFB, University of Karlsruhe, D-76128 Karlsruhe, Germany
dfe@aifb.uni-karlsruhe.de, http://www.aifb.uni-karlsruhe.de/~dfe

Abstract. Ontobroker applies Artificial Intelligence techniques to improve access to heterogeneous, distributed and semistructured information sources as they are presented in the World Wide Web or organization-wide intranets. It relies on the use of *ontologies* to annotate web pages, formulate queries and derive answers. In the paper we will briefly sketch Ontobroker. Then we will discuss its main shortcomings, i.e. we will share the lessons we learned from our exercise. We will also show how On2broker overcomes these limitations. Most important is the separation of the query and inference engines and the integration of new web standards like XML and RDF.

1 Introduction

The World Wide Web (WWW) currently contains around 300 million static objects providing a broad variety of information sources (cf. [Bharat & Broder, 1998]). The early question of whether a certain piece of information is on the web has become the problem of how to find and extract it. The problem will become even more serious if the web continues to grow at the high speed expected by the W3C (the standardization committee of the WWW). Therefore, dealing with the problem of finding and accessing information in the WWW has become a key issue in overcoming the information overload, i.e. in preventing users from wasting hours in going through useless information and trying to find the piece of information they are interested in.

Artificial Intelligence (AI) has a strong tradition of developing methods, tools and languages for structuring knowledge and information. Therefore it is quite natural to apply its techniques to tackle the above problems. In [Fensel et al., 1997] we coined the metaphor of viewing on the web as a large knowledge-based system, however, providing a very limited querying and inference interface at its current state. In the area of knowledge-based systems *ontologies* have been developed for structuring and reusing large bodies of knowledge (cf. CYC [Lenat, 1995], KIF/Ontolingua [KIF], and CommonKADS [Schreiber et al., 1994]). Ontologies are consensual and formal specification of a vocabulary used to describe a specific domain. Frame-based languages enriched by logical axioms are often used to formulate them (cf. LOOM [MacGregor, 1990] and Frame Logic [Kifer et al., 1995]). Roughly, ontologies correspond to generalized database schemata. However, ontologies can be used to describe the semantic structure of much more complex objects than common databases and are therefore well-suited for describing heterogeneous, distributed and semistructured information sources. In the meantime a number of projects rely on such notions (cf. HERMES [Subrahmanian, to appear], Infomaster [Genesareth et al., 1997], and Information Manifold [Levy et al., 1996]) for integrating information sources. SHOE (cf. [Luke et al., 1996], [Luke et al., 1997]) and Ontobroker (cf. [Fensel et al., 1988], [Decker et al., 1999]) use ontologies for information mediation focussing on the integration of HTML sources distributed throughout the WWW.

Ontobroker provides a broker architecture with three core elements: a query interface for formulating queries, an inference engine used to derive answers, and a webcrawler used to collect the required knowledge from the web. It provides a *representation* language for formulating ontologies. A subset of it is used to formulate queries, i.e. to define the *query language*. An *annotation* language is offered to enable knowledge providers to enrich web documents with ontological information. The strength of Ontobroker is the tight coupling of informal, semiformal and formal information and knowledge. This supports their maintenance and provides a service that can be used more generally for the purpose of *knowledge management* and for integrating knowledge-based reasoning and the semiformal representation of documents.

Applying these techniques to the web and to scenarios of realistic size, however, creates a couple of serious problems and brings up some interesting new insights. We will address the most interesting ones in this paper. Some of the above discussed problems and learned lessons could directly be addressed by Ontobroker and some of them required the redesign of the system now called On2broker. The major new design decisions in On2broker are the clear separation of query and inference engines and the integration of new web standards like XML and RDF. Both decisions are answers to two significant complexity problems of Ontobroker: the computational inference effort for a large number of facts and the human annotation effort for adding semantics to HTML documents.

The content of the paper is organized as follows. First, we describe Ontobroker in section 2 and show how it enables integrated access to HTML pages distributed throughout the WWW. Then, we discuss in section 3 the limitations of this approach. These limitations illustrate the difficulties when making a step forward from solving toy examples to a system that is useful for solving real-world problems. Section 4 draws the consequences and introduces On2broker which overcomes most of the serious limitations of its predecessor. We will also show that such a system can successfully handle a much broader scope of tasks than intended at its inception. Section 5 provides the scope of tasks and domains, On2broker can be applied to. Our conclusions, and a discussion of related and future work are provided in section 6.

2 Ontobroker and its Merits

Ontobroker uses ontologies for information mediation focussing on the integration of HTML sources distributed over the WWW. To achieve this goal, Ontobroker provides three interleaved languages and two tools.

It provides an *annotation* language called HTML^A to enable the annotation of HTML documents with machine-processable semantics. For example, the following HTML page states that the text string „Richard Benjamins“ is the name of a researcher where the URL of his homepage is used as his object id.

```
<html><body><a onto="page:Researcher"><h2>Welcome to my homapge</h2>
My name is <a onto="[name=body]">Richard Benjamins</a>.</body></html>
```

An important design decision of HTML^A was (1) to smoothly integrate semantic annotations into HTML and (2) to prevent the duplication of information. The reason for the former decision was to lower the threshold for using our annotation language. People who are able to write HTML can use it straightforwardly as a simple extension. The rationale underlying the second decision is more fundamental in nature. We do not want to add additional data, *instead we want to make explicit the semantics of already available data*. The same piece of data (i.e., „Richards Benjamins“) that is rendered by a browser is given a semantic in saying that this ascii string provides the name of a researcher. We will later show that this is a significant difference between our approach and approaches like SHOE, RDF, and annotations used in information retrieval.

A *representation* language is used to formulate an ontology. This language is based on Frame logic [Kifer et al., 1995]. Basically it provides classes, attributes with domain and range definitions, is-a hierarchies with set inclusion of subclasses and multiple attribute inheritance, and logical axioms that can be used to further characterize relationships between elements of an ontology and its instances. The representation language introduces the terminology that is used by the annotation language to define the factual knowledge provided by HTML pages on the web. A little example is provided in Figure 1. It defines the class *Object* and its subclasses *Person* and *Publication*. There some attributes are defined and some rules expressing relationships between them, for example, if a publication has a person as an author then the author should have it as a publication.

The *query* language is defined as a subset of the representation language. The elementary expression is:

$$x \in c \wedge attribute(x) = v$$

written in Frame logic:

$$x[attribute \text{ -> } v] : c$$

Complex expressions can be built by combing these elementary expressions with the usual logical connectives ($\wedge$, $\vee$, $\neg$). The following query asks for all abstracts of the

```
Object[].
  Person :: Object.
  Publication::Object.
Person[
   firstName =>> STRING;
   lastName =>> STRING;
   eMail =>> STRING;
   ...
   publication =>> Publication].
Publication[
author =>> Person;
title =>> STRING;
year =>> NUMBER;
abstract =>> STRING].
```

```
FORALL Person1, Person2
   Person1:Researcher [cooperatesWith ->> Person2]
  <-
   Person2:Researcher [cooperatesWith ->> Person1].

FORALL Person1, Publication1
   Publication1:Publication [author ->> Person1]
<->
  Person1:Person [publication ->> Publication1].
```

Figure 1 An excerpt of an ontology (taken from [Benjamins et al., 1998])

publications of the researcher „Richard Benjamins“.

x[*name* -> „Richard Benjamins“; *publication* ->> { *y*[*abstract* -> *z*]}] : *Researcher*

The variable substitutions for *z* are the desired abstracts.

Ontobroker relies on two tools that give it „life“: a *webcrawler* and an *inference engine*. The *webcrawler* collects web pages from the web, extracts their annotations, and parses them into the internal format of Ontobroker. The *inference engine* takes these facts together with the terminology and axioms of the ontology, and then derives the answers to user queries. To achieve this it has to do a rather complex job. First it translates Frame logic into Predicate logic and second it translates Predicate logic into Horn logic via Lloyd-Topor transformations [Lloyd & Topor, 1984]. The translation process is summarized in Figure 2

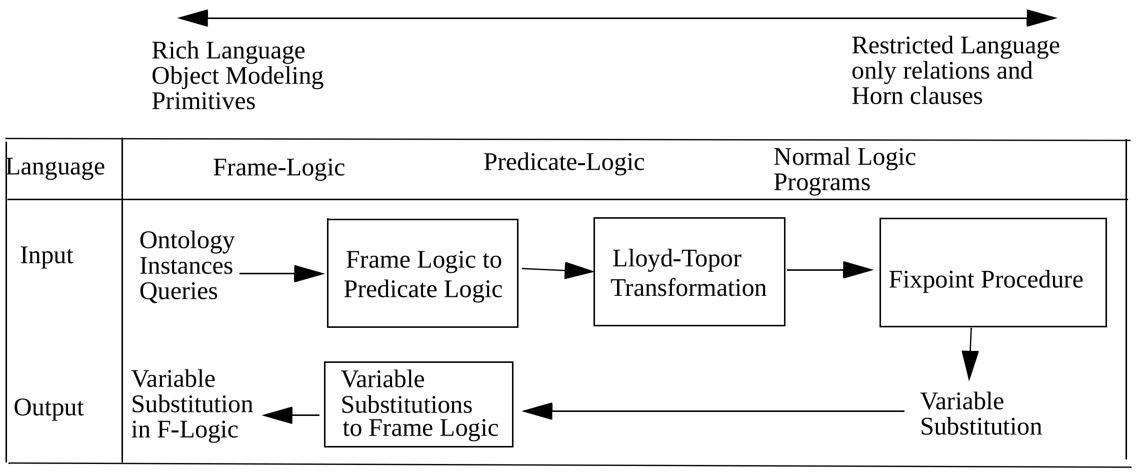


Figure 2 Stages and Languages used in the Inference Engine

As a result we obtain a normal logic program. Standard techniques from deductive databases are applicable to implement the last stage: the bottom-up fixpoint evaluation procedure. Because we allow negation in the clause body we have to carefully select an appropriate semantics and evaluation procedure. If the resulting program is stratified, we use simple stratified semantics and evaluate it with a technique called dynamic filtering (cf. [Kifer & Lozinskii, 1986], [Angele, 1993]) which focuses the inference engine to the relevant parts of a minimal model required to answer the query. But the translation of Frame Logic usually results in a logic program with only a limited number of predicates, so the resulting program is often not stratified. In order to deal with non stratified negation we have adopted the well-founded model semantics [Van Gelder et al., 1991] and compute this semantics with an extension of dynamic filtering [Van Gelder, 1993].

At the start of Ontobroker we encountered a number of interesting aspects of the web environment which we applied the inference engine to: First, we used the type system of the inference engine for abductive inference. Second, we used long text fragments as constants of a logical program. Third, we developed a hyperbolic presentation of the ontology and a tabular interface to improve accessibility. This is essential when the user of the system is not someone who is familiar with logic programming. Fourth we decided to use URLs as object ids. These issues will be discussed in the following.

Using the type system of the inference engine for abductive inference. The representation language of Ontobroker provides domain and range definitions of attributes. Therefore, checking for well-typedness can be carried out. However, given the incomplete nature of the knowledge on the web this nearly always results in typing errors. Assume the following example where researcher Benjamins states that he cooperates with another researcher.

```
<html><body><a onto="page:Researcher"><h2>Welcome to my homepage</h2>
... I cooperate with
<a href="http://www.kmi.uk/~motta" onto="page[cooperatesWith=href]">Enrico Motta
</a>. ...
```

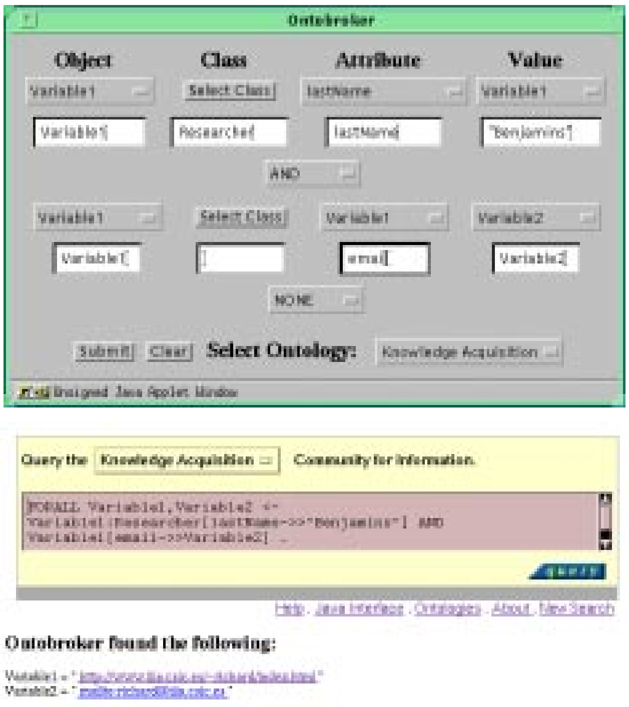
The ontology may provide a definition for *cooperate-with* that assumes if it is applied to a researcher then the range should also be a researcher, i.e.

Researcher[*cooperatesWith* =>>{*Researcher*}]

A typing error then arises if Enrico Motta (denoted by the object-id http://www.kmi.uk/~motta) has not yet been annotated as being a researcher. Given the fact, that most knowledge and especially our annotations will only cover incomplete fragments of the web the type checking facility turned out to be useless at first glance. However, it could be used nicely to abductively derive new knowledge starting with annotated fragments. In fact, we use the type information to deduce that Enrico Motta must also be an researcher. Therefore, we can derive additional knowledge that is not yet directly annotated on the web. The inference rule in Frame logic that realizes this reasoning step is:

$V : T \text{ <- } C\ [A \text{ =>> } T] \quad \wedge \quad O : C\ [A \text{ ->> } V].$

Using long text fragments as constants of a logical program. In the example, we



Variable1 | Researcher | lastName | "Benjamins"

Variable1 | | email | Variable2

Query the Knowledge Acquisition Community for information.

```
FORALL Variable1,Variable2 <-
Variable1:Researcher[lastName->>"Benjamins"] AND
Variable1[email->>Variable2] .
```

Ontobroker found the following:

Variable1 = "[illegible]"
Variable2 = "[illegible]"

Figure 3 The query interface.

described above we asked for the abstracts of the publications of a researcher. An example of an annotation looks like:

```
<H4><A HREF="ftp://ftp.aifb.uni-karlsruhe.de/pub/mike/dfe/paper/www.ps"
name="kt98"><IMG SRC="foto/ps-file.gif"></A>
Dieter Fensel: Search Information in the World Wide Web, Karlsruher Transfer, 12(20),
1998.</H4>
<a onto="#kt98[abstract=body]">The World Wide Web ... discussion.</a></P>
```

Therefore logical variables are substituted for complete abstracts. Even complete papers can be defined as the value of a variable. This causes significant problems for the inference engine. An assumption of typical inference engines is that a logical program uses short constant names and unification can be performed quickly and with small storage demand. In our case, the inference engine tried to use long text fragments as constant names and unification becomes an expensive text processing activity. In consequence, a program has been developed that translates external text fragments into symbolic internal representations of constants to keep unification feasible.

Developing a hyperbolic presentation of the ontology and tabular interface to improve accessibility. Expecting a normal web user to type queries in a logical language and to browse large formal definitions of ontologies is not very practical. Therefore, we exploited the structure of the query language to provide a tabular query interface as shown in Figure 3. There we ask for the researchers whose last name is *Benjamins* and their email addresses. We also need support for selecting classes and attributes from the ontology. To allow the selection of classes, the ontology has to be presented in an appropriate manner. Usually an ontology can be represented as a large hierarchy of

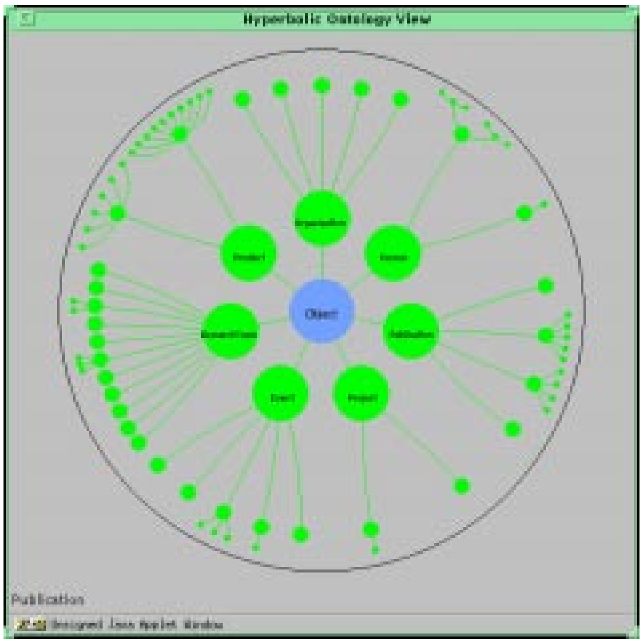


Figure 4 The hyperbolic ontology view.

concepts. With regards to the handling of this hierarchy a user has at least two requirements: first he wants to scan the vicinity of a certain class looking for classes better suitable to formulate a certain query. Second a user needs an overview of the entire hierarchy to allow for a quick and easy navigation from one class in the hierarchy to another class. These requirements are met by a presentation scheme based on Hyperbolic Geometry [Lamping et al., 1995] where classes in the center are depicted with a large circle, whereas classes at the border of the surrounding circle are only marked with a small circle (see Figure 4). The visualization technique allows a quick navigation to classes far away from the center as well as a closer examination of classes and their vicinity. When a user selects a class from the hyperbolic ontology view, the class name appears in the class field of the tabular interface and the user can select one of the attributes from the attribute choice menu as the pre-selected class determines the possible attributes. Based on these interfaces Ontobroker automatically derives the query in textual form and presents the result of the query.

Using URLs as object ids. At first glance, this is very appealing because each object on the web already has an unique identifier which can also be used to denote this object in our frame-based approach. However, such an identifier is not always unique. For example, the following three URLs denote the same web object:

(1) http://www.aifb.uni-karlsruhe/WBS/dfe
(2) http://www.aifb.uni-karlsruhe/WBS/../WBS/dfe
(3) http://www.aifb.uni-karlsruhe/~dfe

One could apply some normalization to eliminate (2) but (3) cannot be eliminated because the way in which the web server interprets the link is not externally visible. Currently we ignore this problem. That is, the same object may have different denotations and we are not aware that these different denotations actually refer to the same object.

A much deeper problem with it stems from the fact that objects in Ontobroker correspond to objects in the real world, and not to objects in the Web-world. E.g. „Richard Benjamins“ is a real-world object. Information about him may be stated in his home-page, but also in many other pages (since facts are not limited to the object corresponding to the page with in which they are stated, i.e. facts about Richard Benjamins can be provided on many places and different locations. Thus in the Ontobroker view of the world, objects are not pages, objects are not even parts of pages, but objects are „outside the Web“, and statements about these objects can be spread over many different pages. Given this, the choice of URLs as object-id's is not such an obvious one anymore. For example, in WebMaster [van Harmelen, 1999] the situation is different. They have deliberately decided that the models you build in WebMaster are about parts of the Web, and not about the world which is described in the Web. Thus, in WebMaster, you would not (could not) say anything about „Richard Benjamins“, but only about „Richard Benjamins homepage“. This is a fundamental difference in the two approaches, WebMaster reasons about the Web (for maintaining it), Ontobroker provides information about real world objects by accessing information sources on the Web.

Ontobroker[1] is available on the web and has been applied in a few applications in the meantime. The most prominent one is the $(KA)^2$ initiative that develops an ontology for annotating web documents of the knowledge acquisition community [Benjamins et al., 1998]. A similar initiative has now been started for the software agent community in AgentLink.

3 Problems in Scaling Up Ontobroker

Ontobroker produces nice and convincing results for small-sized applications. However, there are serious bottlenecks when trying to apply it to larger case studies. Three main problems significantly reduce its usability. First, there is a high human annotation effort. It is hard to convince customers that this effort really pays back through the improved information access. Second, the inference engine becomes slow when the number of facts increases to realistic sizes. Then queries are no longer get answered or the answers arrive too late. Third, neither the query interface nor the way answers are presented fulfil the needs of information retrieval tools. We will discuss each of these problems in more detail in this section.

3.1 The Annotation Effort

Manually adding annotations to web sources requires human effort causing costs in terms of time and money. In an experiment we estimated that the average number of pages a person can annotate is around five per hour. Obviously this number significantly differs

1. http://www.aifb.uni-karlsruhe.de/www-broker.

with the size of the pages, the quality, and the grainsize of annotations. Meanwhile we also developed a click-and browse editor that lowers this effort. Still, annotating the entire web would require an unrealistic effort. This annotation effort may become less problematic by spreading it over the entire web community. Currently the Resource Description Framework (RDF) [Miller, 1998] arises as a standard for annotating web sources with machine-processable metadata. Relating our approach to this standard significantly broadens the range of existing annotations it can be applied to. Another interesting possibility is the increased use of the eXtensible Markup language XML. In many cases, the tags defined by a DTD may carry semantics that can be used for information retrieval. For example, assume a DTD that defines a person tag and within it a name and phone number tag.

```
<PERSON>
        <NAME>Richard Benjamins</NAME>
        <PHONE>+3120525-6263</PHONE>
</PERSON>
```

Then the information is directly accessible with its semantics and can be processed later by Ontobroker for query answering. Expressed in Frame logic, we get:

```
url[NAME --> „Richard Benjamins“; PHONE -->+3120525-6263] : PERSON
```

Annotation is a declarative way to specify the semantics of information sources. A procedural method is to write a program (called *wrapper*) that extracts factual knowledge from web sources. [Ashish & Knoblock, 1997] distinguish three types of information sources on the web: multiple-instance sources, single-instance sources, and loosely-structured sources. The former two types have a stable format that can be used by a wrapper to extract information. Writing wrappers for stable information sources enable us to apply Ontobroker to structured information sources that do not make use of our annotation language. In fact, we applied Ontobroker to the CIA World Fact book (cf. Figure 5)[2]. The Fact book contains a page for each country in the world which presents some general information using a standardized layout. The wrapper program we developed (a one page phyton program) extracts around 40.000 facts providing around 4 MB of factual knowledge about these countries. The strategy of the wrapper is to use a key word based search combined with assumptions on the delimiters of information entries. This strategy was necessary because the pages are hand made and slightly differ in structure for different countries. Also the authors of these pages used HTML as a layout and not as a logical language.[3]

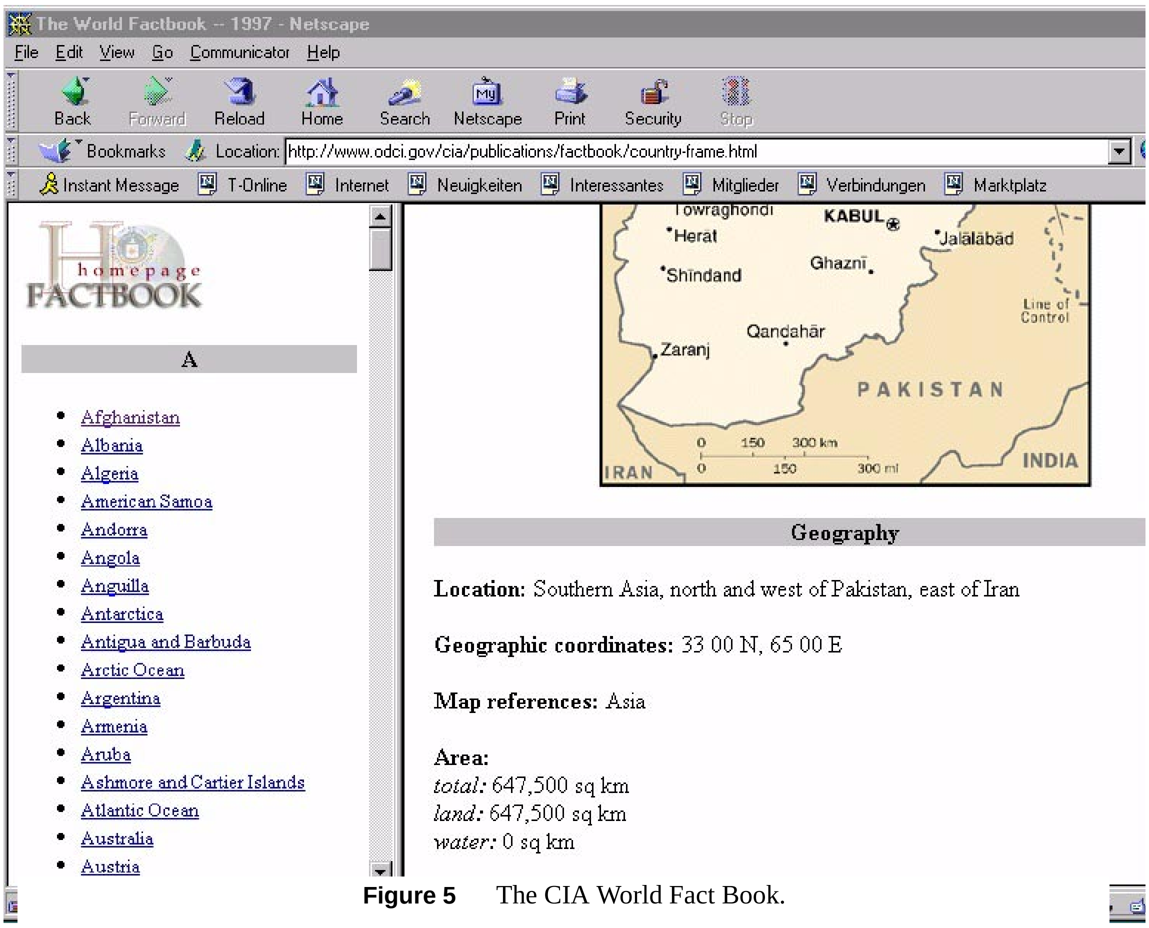
This experiment proved that it is already possible to exploit structure and regularity in current web sources (i.e., HTML documents) to extract semantic knowledge from it without any additional annotation effort. Successfully overcoming the annotation effort for this information source made us aware of a second bottleneck in Ontobroker. The inference engine required around a minute response time when confronted with the amount of factual knowledge provided by the wrapper. We will discuss this problem in the next section.

2. http://www.aifb.uni-karlsruhe.de/www-broker.

3. For example, they indicate headings by <bold> and not by <Heading$_i$>-tags.

3.2 The Inference Effort

In the worst case a query may lead to the evaluation of the entire minimal model. This is a computational hard problem (cf. [Brewka & Dix, 1999]). In other cases predicate symbols and constants are used to divide the set of facts into subsets in order to omit those subsets which can not contribute to the answer. This normally reduces the evaluation effort considerably. Ontobroker allows very flexible queries such as „which attributes has a class“. The consequence of this is, that the entire knowledge is represented by only a few predicates such as the predicate *value* which relates a class *c* to its attribute *att* and the corresponding attribute value *v* (*value*(*c*,*att*,*v*)). Thus the set of facts is divided into a few subsets only. The last version of our inference engine used the minimal model as semantics. If the set of rules is stratified [Ullman, 1988] an answer to a query may be evaluated efficiently using minimal model semantics. Using only few predicates has the consequence that nearly every rule set is not stratified if we allow negation in rules. This was the reason that we had to use the Wellfounded Semantics in our approach now. Wellfounded Semantics coincides with Minimal Model Semantics in cases where the rule set is stratified. Beyond that the Wellfounded Model Semantics also allows us to evaluate non stratified rule sets.


The World Factbook -- 1997 - Netscape
File Edit View Go Communicator Help
Back Forward Reload Home Search Netscape Print Security Stop
Bookmarks Location: http://www.odci.gov/cia/publications/factbook/country-frame.html
Instant Message T-Online Internet Neuigkeiten Interessantes Mitglieder Verbindungen Marktplatz
homepage FACTBOOK
A
Afghanistan
Albania
Algeria
American Samoa
Andorra
Angola
Anguilla
Antarctica
Antigua and Barbuda
Arctic Ocean
Argentina
Armenia
Aruba
Ashmore and Cartier Islands
Atlantic Ocean
Australia
Austria
Towraghondi
KABUL
Herat
Jalalabad
Shindand
Ghazni
Line of Control
Qandahar
Zaranj
PAKISTAN
IRAN
INDIA
0 150 300 km
0 150 300 mi

Geography

Location: Southern Asia, north and west of Pakistan, east of Iran

Geographic coordinates: 33 00 N, 65 00 E

Map references: Asia

Area:
total: 647,500 sq km
land: 647,500 sq km
water: 0 sq km

Figure 5 The CIA World Fact Book.

Both points, the small number of predicates and the Wellfounded Model Semantics raised severe efficiency problems. Up to now we applied our inference engine only to knowledge bases with less than 100,000 facts. It is clear that our approach must be applicable to millions of facts in order to be of practical relevance. Additionally the inference engine loads the entire knowledge base into RAM to evaluate a query. In many cases only a small subset of the knowledge base may be sufficient to answer the query. This strongly restricts the possible size of our knowledge base and causes in many cases an additional runtime overhead. The problems in applying the inference engine to the full-sized applications pointed out a serious shortcoming of the overall system architecture of Ontobroker. Currently, the query engine and the inference engine are actually one engine. The inference engine receives a query and derives the answer. However, in the case of the CIA World Fact book there are no axioms present in the ontology (cf. Figure 6). And the query interface of Ontobroker is restricted to simple SQL-like queries (i.e., it can be used to ask for ground objects ids and ground attributes values fulfilling certain properties). That is, we applied a powerful inference mechanism to a problem that could be solved by much simpler means. The need to separate the query and inference engines was the clear lesson we learned from this exercise. The next section will provide further arguments for this need.

3.3 The Limited Query Interface

The query interface of Ontobroker lacks the standard capabilities of professional keyword-based web search engines and information retrieval systems. Terms must be typed precisely as they are stored (as facts or ontological expressions) in the system. Unification in logic programming assumes a perfect match of terms, i.e. Benjamins does not match with Benjamin. Similarly, Ontobroker does not allow the truncation of expressions (i.e., Benjam*) and the ranking of the answers found is determined by the

```
Country :: Object.
Country[ has_name =>> string;
        has_geography =>> geography;
        has_people =>> people;
        has_government =>> government;
        has_economy =>> economy;
        has_communication =>> communications;
        has_transportation =>> transportation].
```

```
Geography :: Object.
geography[
        map =>> string;
        flag =>> string;
        location =>> string;
        geographic_coordinates =>> string;
        has_area =>> area;
        has_boundaries =>> boundaries;
        coastline =>> string;
        has_maritime_claims =>> maritime_claims;
        climate =>> string;
        terrain =>> string;
        has_elevation_extremes =>> elevation_extremes;
        natural_resources =>> string;
        has_land_use =>> land_use;
        irrigated_land =>> string;
        natural_hazards =>> string].
```

Figure 6 A snapshot of the ontology developed for describing the knowledge of the CIA World Fact Book.

internal order of the inference process and does not make any sense to the user. In general, two strategies exist to deal with these shortcomings.

- First, the following features can be integrated into the inference process: (a) we could allow equality and express equality axioms for terms;[4] (b) we could include specific built-in predicates that externally define same term equalities (cf. [Cohen, 1998]); and (c) we could directly change the unification mechanism.
- Second, we could use the classical inference engine but provide additional post-processing service by the query interface. The query engine can decide how to deal with the results of the inference engine in a way that meets the user’s needs. In On2broker we generally follow this second approach.

The choices we made in changing and extending Ontobroker to On2broker will be discussed in the following section.

4 On2broker: Lessons Learned and Problems Solved

On2broker takes two main lessons from the experiences we collected with Ontobroker. It overcomes the inference bottleneck and it broadens the scope of web sources it can be applied to. Both aspects will be discussed in this section. Finally we show how On2broker can communicate with other information mediators and softbots [Etzioni, 1997].

4.1 Decoupling Inference and Query Response

In the design of Ontobroker we already made an important decision when we separated the web crawler and the inference engine. The web crawler periodically collects information from the web and caches it. The inference engine uses this cache when answering queries. The decoupling of inferencing and fact collection is done for efficiency reasons. The same strategy is used by search engines on the web. A query is answered with help of their indexed cache and not by starting to extract pages from the web. On2broker refines this architecture by introducing a second separation: *separating the query and inference engines*. The inference engine works as a demon in the background. It takes facts from a database, infers new facts and returns these results back into the database (cf. Figure 7). The query engine does not directly interact with the inference engine. Instead it takes facts from the database. It is an SQL frontend to this database and the tabular and hyperbolic query interface of Figure 3 and Figure 4 can still be used for it. Separating query and inference engine has some clear advantages:

- Whenever inference is a time critical activity it can be performed in the background independently of the time required to answer the query.

4. For reasons of computational complexity, Ontobroker does not provide equality reasoning.

- Using database techniques for the query interface and its underlying facts provides robust tools that can handle mass data.
- It is relatively simple to include things like truncation, term similarity and ranking in the query answering mechanism. They can now directly be integrated into the SQL query interface (i.e., in part they are already provided by SQL) and does not require any changes to the much more complex inference engine.

For example, the ontology of the CIA World Fact Book does not contain any rules. It is therefore large overhead to use an inference engine for query answering. With Ontobroker, it takes a minute to read and translates all 40.000 facts before the inference engine even starts with its first inference. A simple database update in On2broker performs much faster.

In the simplest case the format in which the data are stored in the database is one large table with three columns: object-id, attribute (i.e., property), and value, and a row for each fact. This may cause new efficiency problems when millions of facts have been extracted. However, we can make use of several refinements to deal with this problem.

First, we can define a database per ontology. Second, we can use one ontology to structure its database. Simplified, each concept in an ontology may corresponds to a table and each attribute defines a column of this table.[5] Both strategies do not reduce the number of facts that need to be stored but the number of facts that need to be checked for answering a query.

More general, the strict separation of query and inference engines can be weakened for

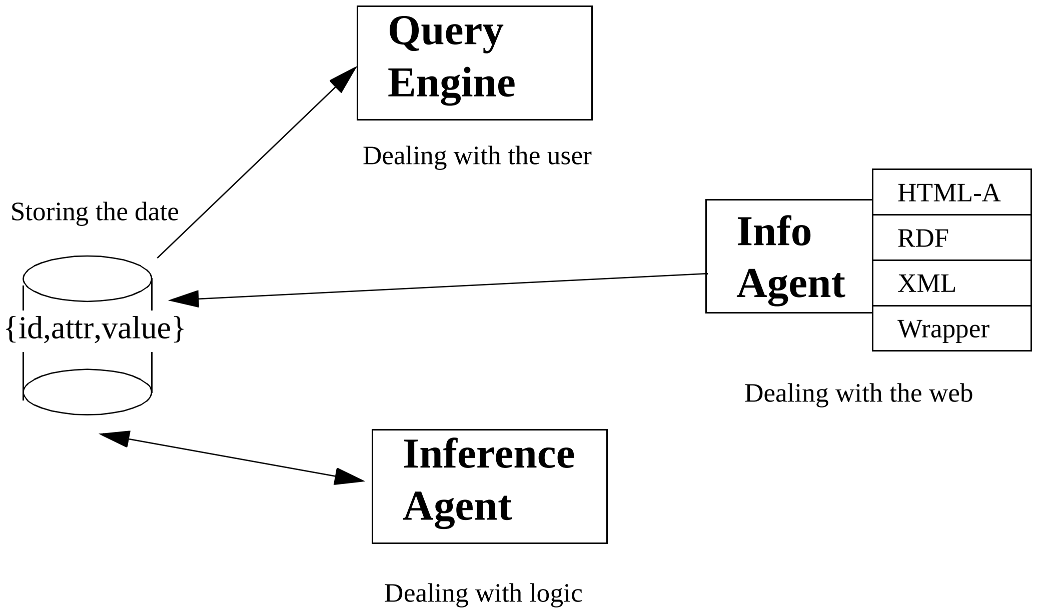


Figure 7 The gist of On2broker.

5. The actual solution is more complex reflecting the is-a relationship between concepts including subset relationships of sub concepts and attribute inheritance.

cases where this separation would cause disadvantages. In many cases it may not be necessary to enter the entire minimal model in a database. Many facts are of intermediate or no interest when answering a query. The inference engine of On2broker incorporates this in its dynamic filtering strategy which uses the query to focus the inference process. We can make use of this strategy when deciding which facts are to be put into the database. Either one limits the queries that can be processed by the system or one replaces real entries in the database with a virtual entry representing a query to the inference engine. The latter may require large delay in answering which, however, may be acceptable for user agents that collects information of the WWW in a background mode. Finally, we can cache the results of such queries to speed up the process in cases where it is asked again. These strategies bring query and inference engines closer together again. They are, however, no longer directly coupled but mediated by the database manager. Developing such an intelligent integration of database and inference techniques is an important line of the future work in On2broker. In general, in the literature on dataware housing provides two alternatives:

- Materializing views (i.e., queries)
- Realizing views by run time queries

Both solutions are also present in the web (e.g., by the on-line store providers junglee and jango[6]) and both have their merits and shortcomings (cf. [Harinarayan et al., 1996]). By making the architecture of On2broker more flexible as it was allows domain and task specific customization. Elaborating these issues are an important line of future work on On2broker.

4.2 How to extend HTML^A

HTML^A is appealing because (1) it is a simple and straightforward extension of an existing technology and because (2) it prevents any duplication of information. However it is not a widely used standard. The actual annotations which we will find in the web will therefore be rather small. An alternative is to write wrappers for non-annotated sources. However, this burdens us with some programming effort. Therefore, we extended the webcrawler (called Info agent in On2broker) of Ontobroker to include two new web standards RDF and XML that both provide meta information in a complementary manner.

RDF[7] [Miller, 1998] provides means for adding semantics to a document without making any assumptions about the internal structure of this document. It is an XML application (i.e., its syntax is defined in XML) customized for adding meta information to Web documents. It is currently under development as a W3C standard for content descriptions of web sources and will be used by other standards like PICS-2, P3P, and DigSig.

The data model of RDF provides three object types: resources, property types, and statements (cf. [Lassila & Swick, 1998]):

6. http://www.junglee.com and http://www.jango.com.

7. See http://www.w3c.org/RDF and http://www.cs.ukc.ac.uk/people/staff/djb1/research/metadata/rdf.shtml.

- A *resource* (*subject*) is an entity that can be referred to by an address at the WWW (i.e., by an URI). Resources are the elements that are described by RDF statements.
- A *property type* (*predicate*) defines a binary relation between resources and / or atomic values provided by primitive datatype definitions in XML.
- A *statement* (*object*) specifies for a resource a value for a property. That is, statements provide the actual characterizations of the web documents.

A simple example is

Creator(http://www.w3.org/Home/Lassila) = *Ora Lassila*[8]

stating that the creator of the web document http://www.w3.org/Home/Lassila is Ora Lassila. Values can also be structured entities

Creator(http://www.w3.org/Home/Lassila) = $X \wedge$
Name(X) = *Ora Lassila* $\wedge$ *Email*(X) = lassila@w3.org

where X either denotes an actual (i.e., the homepage of Ora Lassila) or virtual URI. In addition, RDF provides bags, sequences, and alternatives to express collections of web sources. Finally, RDF can be used to make statements about RDF-statements, i.e. it provides meta-level facilities.

Claim(*Ralph Swick*) = (*Creator*(http://www.w3.org/Home/Lassila) = *Ora Lassila*)

states that Ralph Swick claims that Ora Lassila is the creator of the resource http://www.w3.org/Home/Lassila. The info engine of Onto2broker can deal with RDF descriptions. We make use of the RDF Parser SiRPAC[9] that translates RDF descriptions into triples that can directly be put into our database.

RDF still requires the annotation effort for creating metadata but this effort is now shared by the entire web community. XML provides the chance to get metadata „for free“, i.e. as side product of defining the document structure. XML allows the definition of new tags with the help of a DTD and provides semantic information as a by-product of defining the structure of the document. A DTD defines a tree structure to describe documents and the different leaves of the tree have tags that provides semantics of the elementary information units presented by them. That is, the structure and semantics of a documents are interleaved. In particular, a document must be written using XML and the specific tagging provided by its DTD. On2broker is able to read such DTD, to translate it into an ontology, and to translate XML documents into its internal triple representation.

Further extensions that are currently under development are the use of style sheets for meta-annotations (currently style sheets containing meta-information concerning the rendering style of a tag) and the combination with keyword-based text mining and management tools like Jaspar [Davis et. al., 1995].

8. We skip the awark syntax of RDF because a tool can easily present it like shown above.
9. http://www.w3.org/RDF/Implementations/SiRPAC/

4.3 Enabling Agents to Access On2broker

In the effort to create efficient search mechanisms in the WWW information *mediators*, the like of Metacrawler[10], and softbots that access other search engines will become increasingly important. That is why in On2broker the decision was taken to implement the query interface as an Java^{TM} *Remote Method Invocation (RMI) Server.* This allows us to make the Java^{TM} interface publicly available and thus give meta search engines more efficient access to the database. While the current method, used by the query interface applet, returns the results in a HTML format as does the CGI skript in Ontobroker, the interface can easily be extended to have methods return the result in any other format convienent to the invoking application. With the current release of Sun's JDK 1.2 an implementation of the query server with a CORBA interface will also be available to make accessibility even easier for applications not written in Java^{TM}.

5 Accessing, Creating, and Maintaining Semistructured Documents in Intranets and the Internet

Ontobroker was presented as a means to improve access to information provided in intranets and in the internet (cf. [Fensel et al., 1997]). Its main advantages compared to keyword-based search engines are:

- Keyword-based search retrieves irrelevant information that use a certain word in a different meaning or it may miss information where different words about the desired content are used.
- The query responses require human browsing and reading to extract the relevant information from these information sources. This burdens web users with an additional loss of time and seriously limits information retrieval by automatic agents that miss all common sense knowledge required to extract such information from textual representations
- Key word based document retrieval fail to integrate information spread over different sources.
- Finally, each current retrieval service can only retrieve information that is represented by the WWW.[11]

Ontobroker uses semantic information for guiding the query answering process. It provides the answers with a well-defined syntax and semantics that can be directly understood and further processed by automatic agents or other software tools. It enables a

10. http://www.metacrawler.com.

11. This sounds trivially true, but it significantly limits query answering capability. Imagine that Feather writes on his homepage that he cooperates with another researcher E. Motta. You will completely miss this information for E. Motta if he does not repeat the information (with the reverse direction) on his homepage and you are only consulting his page. An answering mechanism that can make use of the implicit symmetry of cooperation could provide you with this answer.

homogeneous access to information that is physically distributed and heterogeneously represented in the WWW and it provides information that is not directly represented as facts in the WWW but which can be derived from other facts and some background knowledge. Still, the range of problems it can be applied to is much broader than information access and identification in semistructured information sources. On2broker is also used to create and maintain such semistructured information sources, i.e. it is a tool for web site construction and restruction.

Automatic document generation extracts information from weakly structured text sources and creates new textual sources. Assume distributed publication lists of members of a research group. The publication list for the whole group can automatically be generated by a query to On2broker. A background agent periodically consults On2broker and updates this page. The gist of this application is that it generates semistructured information presentations *from* other semistructured ones. At the moment, answer of On2broker have to manually transformed into web sources. Providing flexible layout wrappers is essential for lowering this effort.

Maintenance of weakly structured text sources helps to detect inconsistencies among documents and to detect inconsistencies between documents and external sources, i.e., to detect incorrectness. WebMaster [van Harmelen, 1999] developed a constraint language for formulating integrity constraints for XML documents (for example, a publication on a page of a member of the group must also be included in the publication list of the entire group). Again such a service can be provided by On2broker. We can either incorporate the inference of WebMaster in it or use the existing inference engine in a different way. We mentioned that we currently use the type system for abductively inferring new facts paying tribute to the sloppiness and openness of the WWW. Using the type system for checking integrity constraints may be the right way to make use of it for homogeneous and well designed intranets of companies and organizations. Maintaining intranets of large organizations and companies become a serious effort because such networks already provide several million documents. Therefore it is no surprise that first serious application projects of On2broker actually refer rather to intranet sides and not directly to the internet.

6 Conclusions

The overall picture of On2broker is provided in Figure 8 which includes four basic engines representing different aspects.

- The **info agent** is responsible for collecting factual knowledge from the web using various style of meta annotations, direct annotations like XML and in future also text mining techniques.
- The **inference agent** uses facts and ontologies to derive additional factual knowledge that is only provided implicitly. It frees knowledge providers from the burden of specifying each fact explicitly.

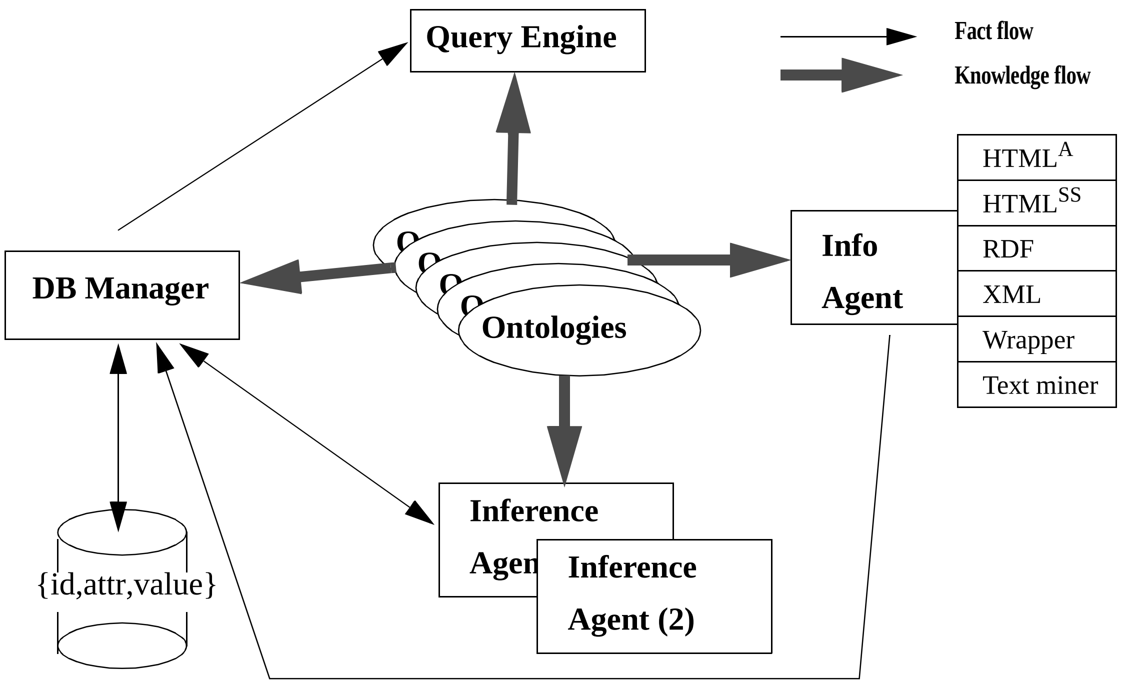


Figure 8 On2brokers Architecture.

- The **query engine** receives queries and answers them by checking the content of the databases that were filled by the info and inference agents.
- The **database manager** is the backbone of the entire system. It receives facts from the Info agent, exchanges facts as input and output with the inference agent, and provides facts to the query engine.

Ontologies are the overall structuring principle. The info agent uses them to extract facts, the inference agent to infer facts, the database manager to structure the database and the query engine to provide help in formulating queries. On2broker introduces this architecture to overcome some serious shortcoming we encountered when applying AI techniques to the web environment with our prototypical system Ontobroker. In the following we will compare On2broker with related work and outline directions of future work.

Current web standards like HTML are very limited in the access to information sources they provide. Therefore, many extensions of HTML are proposed in the literature. [Perkowitz & Etzioni, 1997b] propose A-HTML that extends HTML by the use of meta-information enabling adaptive web sites. This extension is close in spirit to our extensions of HTML^A. The main difference concerns the content and purpose of these annotations. A-HTML enables dynamic reconfiguration whereas HTML^A supports query access to web pages. However, generating web pages via queries to On2broker brings both approaches together. Still, the HTML generator of On2broker is rather straightforward and requires further improvement. At the current state it provides a list of query substitutions and much more sophisticated output specifications are required to use it to produce useful HTML pages.

SHOE (cf. [Luke et al., 1996], [Luke et al., 1997]) introduced the idea of using ontologies for annotating web sources. There are two main differences to our approach. First, the annotation language is not used to annotate existing information in web pages but to add additional information and annotate them. That is, in SHOE the same information must be repeated and this redundancy may cause significant maintenance problems. For example, an affiliation must once be provided as a text string rendered by the browser and a second time as annotated meta information. In this respect, SHOE is close to meta tags in HTML. On2broker use the annotations to directly add semantics to textual information that is also rendered by a browser. A second difference is the use of inference techniques and axioms to infer additional knowledge. SHOE relies only on database techniques. Therefore, no further inference service is provided. Ontobroker uses an inference engine to answer queries. Therefore, it can make use of rules that provide additional information. However, this decision also limited the size of problems it can successfully be applied to. On2broker takes an intermediate position. It uses an inference engine to derive additional facts. Its query interface is, however, coupled to a database easily scaling up to large datasets.

An excellent survey of database techniques applied to the WWW is provided by [Florescu et al., 1998]. In their outline, they characterize a web site management system as consisting of wrappers, mediators, declarative web site specification, and an HTML generator. The Info agent of On2broker provides such a wrapper, and the ontology together with the inference agent and database manager provide the mediator. The query engine of On2broker allows the declarative specification of web site (relying also on additional axioms provided by the ontology). This is, however, hampered by the already mentioned weak HTML generator of On2broker.

FLORID [Ludäscher et al., 1998] uses Frame logic for defining access to web sources as does On2broker. However, FLORID directly relies on the syntactical structure of web sources and does not use any metadata approach. Simplified, a HTML page is an object and each tag (including links) is an attribute of this object. Therefore, it is possible to ask for headings of a page or for all pages reachable from it. On2broker lifts from this syntactical level and provides semantic access to facts spread over different web sources. This orientation on semantics rather than on syntax and the use of a logical background theory (i.e., an ontology) is also the main difference between On2broker and approaches like STRUDEL [Fernandez, 1998] and WebQQL [Aroneca, 1997].

The use of *one* ontology for annotating web documents will never scale up for the entire web. Neither will an ontology be suitable for all subjects and domains nor will ever such a large and heterogeneous community as the web community agree on a complex ontology for describing all their issues. For example, the Dublin Core community[12] has been working for years to establish a simple core ontology for adding some meta information to on-line documents. [Fensel et al., 1997] sketch out the idea of an *ontogroup*. Like a news groups, it is based on a group of people who are joined by a

12. http://purl.org/metadata/dublin_core.

common interest and some agreement as to how to look at their topic. An ontology can be used by such a group to express this common ground and to annotate their information documents. A broker can make use of these annotations to provide intelligent information access. The ontology describes the competence of the broker, i.e. the area in which it can provide meaningful query response. In consequence, several brokers will arise, each covering different areas or different points of views on related areas. Facilitators and softbots [Etzioni, 1997] guide a user through this knowledgeable network superimposed on the current internet (cf. [Dao & Perry, 1996], [Sakata et al., 1997]). Therefore, work on relating and integrating various ontologies (cf. [Jannink et al., 1998]) will become an interesting and necessary research enterprise which will also be addressed in the future course of the On2broker project helping to evolve „the Web from a Document Repository to a Knowledge Base.“ [Guha et al., 1998]

Acknowledgment. We thank Igor Becker, Abdullah Dagasan, and Christian Hoffmann, for their contributions to the On2broker project, Jeff Butler for correcting the English, and Frank van Harmelen for his very critical and constructive comments.

References

[Angele, 1993] J. Angele: *Operationalisierung des Models der Expertise mit KARL*, Infix, St. Augustin, 1993.

[Aroneca, 1997] G. Aroneca: WebQQL: Exploiting document Structure in Web Queries. Masther´s thesis, University of Toronto, 1997.

[Ashish & Knoblock, 1997] N. Ashish and C. Knoblock: Semi-automatic Wrapper Generation for Internet Information Sources. In *Proceedings of the IFCIS Conference on Cooperative Information Systems (CoopIS),* Charlston, South Carolina, 1997.

[Benjamins et al., 1998] R. Benjamins, D. Fensel, and A. Gomez Perez: Knowledge Management Through Ontologies. In *Proceedings of the Second International Conference on Practical Aspects of Knowledge Management (PAKM'98),* Basel, Schweiz, 29.-30. Oktober 1998.

[Bharat & Broder, 1998] K. Bharat and A. Broder: March´98 Measurement of Search Engines. Update of the paper Estimating the Relative Size and Overlap of Public Search Engines, *Proceedings of the 7th International World Wide Web Conference (WWW7)*, Brisbane, Australia, *Computer Networks and ISDN Systems*, 30(1-7), April 14-18, 1998.

[Brewka & Dix, 1999] G. Brewka and J. Dix: Knowledge Representation with Logic Programs. In D. Gabbay et al. (eds.), *Handbook of Philosophical Logic* (2nd ed.), vol 6, Methodologies, Reidel Publ., 1999.

[Cohen, 1998] W. W. Cohen: The WHIRL Approach to Integration: An Overview. In *Proceedings of the AAAI´98 Workshop on AI and Information Integration*, Madison, WI, July 26-27, 1998.

[Dao & Perry, 1996] S. Dao and H. Perry: Information Mediation in Cyperspace: Scalable Methods for Declarative Information Networks, *Journal of Intelligent Information Systems (JIIS)*, 6:151—170, 1996.

[Davis et. al., 1995] J. Davis, R. Weeks, and M. Revett: Jaspar: Communication Information Agents for the WWW. In *Proceedings of the 4th International World Wide Web Conference*

(WWW4), Boston, Massachusetts, USA, December 11-14, 1995.

[Decker et al., 1999] S. Decker, M. Erdmann, D. Fensel und R. Studer: Ontobroker: Ontology based Access to Distributed and Semi-Structured Information. In R. Meersman et al. (eds.), *Semantic Issues in Multimedia Systems*, Kluwer Academic Publisher, Boston, to appear 1999.

[Etzioni, 1997] O. Etzioni: Moving Up the Information Food Chain, *AI Magazine*, 18(2), 1997.

[Farquhar et al., 1997] A. Farquhar, R. Fikes, and J. Rice: The Ontolingua Server: a Tool for Collaborative Ontology Construction, *International Journal of Human-Computer Studies (IJHCS)*, 46(6):707—728, 1997.

[Fensel et al., 1997] D. Fensel, M. Erdmann, and R. Studer: Ontology Groups: Semantically Enriched Subnets of the WWW. In *Proceedings of the 1st International Workshop Intelligent Information Integration during the 21st German Annual Conference on Artificial Intelligence*, Freiburg, Germany, September 9-12, 1997.

[Fensel et al., 1988] D. Fensel, S. Decker, M. Erdmann und R. Studer: Ontobroker: The Very High Idea. In *Proceedings of the 11th International Flairs Conference (FLAIRS-98)*, Sanibal Island, Florida, USA, 131-135, Mai 1998.

[Fernandez, 1998] M. Fernandez, D. Florescu, J. Kang, A. Levy, and D. Suciu: Catching the boat with Strudel: Experience with a web-site management system. In *Proceedings of the ACM SIGMOD Conference on Management of Data*, Seattle, WA, 1998.

[Florescu et al., 1998] D. Florescu, A. Levy, and A. Mendelzon: Database Techniques for the World-Wide Web: A Survey, *Sigmod Records*, 27(3):59—74, 1998.

[Van Gelder, 1993] A. Van Gelder: The Alternating Fixpoint of Logic Programs with Negation, *Journal of Computer and System Sciences*, 47(1):185—221, 1993.

[Van Gelder et al., 1991] A. Van Gelder, K. Ross, J. S. Schlipf: The Well-Founded Semantics for General Logic Programs, *Journal of the ACM*, 38(3): 620—650, 1991.

[Genesareth et al., 1997] M. R. Genesareth, A. M. Keller, and O. Duschka: Infomaster: An Information Integration System. In *Proceedings of 1997 ACM SIGMOD Conference*, May 1997.

[Guha et al., 1998]R. V. Guha, O. Lassila, E. Miller and D. Brickley: Enabling Inferencing. In *Proceedings of the W3C Query Language Workshop (QL-98)*, Boston, MA, December 3-4, 1998.

[Harinarayan et al., 1996]V. Harinarayan, A. Rajaraman, and J. D. Ullmann: Implementing Data Cubes Efficiently. In *Prodeedings of the 1996 ACM SIGMOD International Conference on Management of Data*, 1996.

[van Harmelen, 1999] F. van Harmelen: WebMaster: Knowledge-based Verification of Web-pages, submitted to *IEA/AIE'99* (to be published, Springer Verlag, 1999).

[Jannink et al., 1998] J. Jannink, S. Pichai, D. Verheijen, and G. Wiederhold: Encapsulation and Composition of Ontologies. In Proceedings of the AAAI´98 Workshop on AI and Information Integration, Madison, WI, July 26-27, 1998.

[KIF] Knowledge Interchange Format. Draft proposed American National Standard (dpANS) NCITS.T2/98-004. http://logic.stanford.edu/kif/dpans.html.

[Kifer et al., 1995] M. Kifer, G. Lausen, and J. Wu: Logical Foundations of Object-Oriented and Frame-Based Languages, *Journal of the ACM*, 42, 1995.

[Kifer & Lozinskii, 1986] M. Kifer, E. Lozinskii: A Framework for an Efficient Implementation of Deductive Databases. In *Proceedings of the 6th Advanced Database Symposium,* Tokyo, 1986.

[Lamping et al., 1995] L. Lamping, R. Rao, and Peter Pirolli.: A Focus+Context Technique Based on Hyperbolic Geometry for Visualizing Large Hierarchies. In *Proceedings of the ACM SIGCHI Conference on Human Factors in Computing Systems*, 1995.

[Lassila & Swick, 1998] O. Lassila and R. R. Swick (eds.): Resource Description Framework (RDF) Model and Syntax Specification, *W3C Working Draft*, August 19, 1998. http://www.w3c.org/TR/WD-rdf-syntax.

[Lenat, 1995] D. B. Lenat: CYC: A Large-Scale Investment in Knowledge Infrastructure, *Communications of the ACM* 38(11), 1995.

[Levy et al., 1996] A. Y. Levy, A. Rajaraman, and J. J. Ordille: Querying Heterogeneous Information Sources Using Source Descriptions. In *Proceedings of the 22nd International Conference on Very Large Databases, VLDB-96,* Bombay, India, September, 1996.

[Lloyd & Topor, 1984] J. W. Lloyd and R. W: Topor: Making Prolog more Expressive, *Journal of Logic Programming,* 3:225—240, 1984.

[Ludäscher et al., 1998] B. Ludäscher, R. Himmeröder, G. Lausen, W. May, and C. Schlepphorst: Managing semistructured data with FLORID: A deductive object-oriented perspective, *Information System*, 23(8), 1998.

[Luke et al., 1996] S. Luke, L. Spector, and D. Rager: Ontology-Based Knowledge Discovery on the World-Wide Web. In *Proceedings of the Workshop on Internet-based Information Systems* at the *AAAI-96*, Portland, Oregon, August 4-8, 1996.

[Luke et al., 1997] S. Luke, L. Spector, D. Rager, and J. Hendler: Ontology-based Web Agents. In P*roceedings of First International Conference on Autonomous Agents*, 1997.

[MacGregor, 1990] MacGregor: *LOOM Users Manual*, ISI/WP-22, USC/Information Sciences Institute, 1990.

[Miller, 1998] E. Miller: An Introduction to the Resource Description Framework, *D-Lib Magazine*, May 1998.

[Perkowitz & Etzioni, 1997b] M. Perkowitz and O. Etzioni: Adaptive Web Sites: an AI Challenge. In *Proceedings of the 15th International Joint Conference on AI (IJCAI-97),* Nagoya, Japan, August 23-29, 1997.

[Sakata et al., 1997] T. Sakata, H. Tada, T. Ohtake: Metadata Mediation: Representation and Protocol. In *Proceedings of the 6th International World Wide Web Conference (WWW6),* Santa Clara, California, USA, April 7-11, 1997.

[Schreiber et al., 1994] G. Schreiber, B. Wielinga, J. M. Akkermans, W. Van De Velde, and R. de Hoog: CommonKADS. A Comprehensive Methodology for KBS Development, *IEEE Expert*, 9(6):28—37, 1994.

[Subrahmanian, to appear] V.S. Subrahmanian, S. Adali, A. Brink, R. Emery, J. J. Lu, A. Rajput, T. J. Rogers, R. Ross, C. Ward: HERMES: Heterogeneous Reasoning and Mediator System, submitted.

[Ullman, 1988] J. D. Ullman: *Principles of Database and Knowledge-Base Systems,* vol I, Computer Sciences Press, Rockville, Maryland, 1988.