Dieter Fensel

Ontologies: Silver Bullet for Knowledge Management and Electronic Commerce

Abstract. Currently computers are changing from single isolated devices to entry points into a world wide network of information exchange and business transactions called the World Wide Web (WWW). Therefore support in the exchange of data, information, and knowledge exchange is becoming the key issue in current computer technology. Ontologies provide a shared and common understanding of a domain that can be communicated between people and application systems. Therefore, they may play a major role in supporting information exchange processes in various areas. This book discusses the role ontologies will play in knowledge management and in electronic commerce. In addition, I show how arising web standards such as RDF and XML can be used as an underlying representation languages for ontologies.

Preface

...

February 2000 ???

Table of Contents

1 Introduction 1

2 Ontologies 8

3 Application Area Knowledge Management 13

3.1 The pitfalls of current information search 13
3.1.1 How to avoid nonsense and find what you are looking for 14
3.1.2 Information presentation and access is limited 14
3.1.3 How to collect distributed information 15
3.1.4 How to collect implicit information 16

3.2 How Ontobroker overcomes these limitations 16
3.2.1 The Languages 17
3.2.1.1 The Annotation Language 18
3.2.1.2 The Representation Languages 19
3.2.1.3 The Query Languages 21
3.2.2 The Tools 22
3.2.3 Conclusions 25

3.3 The Future beyond Ontobroker 25
3.3.1 On2broker 26
3.3.1.1 The Database Manager: Decoupling Inference and Query Response 26
3.3.1.2 The Info Agent 28
3.3.1.3 Conclusions 29
3.3.2 On-To-Knowledge: Evolving Ontologies for Knowledge Management 30
3.3.3 IBROW: Brokering dynamic reasoning services in the WWW 31

3.4 The service pyramid 32

4 Application Area Electronic Commerce 34

4.1 Application area B2C
Shopbots, Adaptive On-line Stores, and On-line Market places 34
4.1.1 Introduction 34
4.1.2 Shopbots 35
4.1.3 Adaptive On-line Stores 41
4.1.4 On-line Marketplaces 41
4.1.5 Electronic Commerce, Agents and XML 43

4.2 Electronic Trading Systems as B2B Marketplaces 43
4.2.1 Means for representation and translation 45
4.2.2 Means for content descriptions (Ontologies) 46

4.2.3 Chemdex (www.chemdex.com) 47
4.2.4 PaperExchange (www.paperexchange.com) 47
4.2.5 VerticalNet (www.verticalnet.com) 47
4.2.6 Conclusions 47

5 The basic technology: XML, XSL, and XML-QL 49
5.1 Why XML 50
5.2 What is XML 52
5.3 What are DTDs 53
5.4 Linking in XML 55
5.5 Extensible Style Language (XSL) 55
5.6 Query Languages for XML 56
5.7 The Resource Description Framework RDF 58
5.8 Conclusions 59

6 Ontology Languages 61
6.1 Ontology Languages 61
6.1.1 First-order predicate logic languages CycL and KIF 61
6.1.2 Frame-based Approaches: Ontolingua and Frame Logic 65
6.1.3 Description Logics 68
6.2 XML, RDF, and Ontology Languages 71
6.2.1 DTD and Ontologies 71
6.2.2 RDF and Ontologies 73
6.2.3 Comparing RDF and XML 73
6.3 XOL 75

7 Conclusions 78

8 References 80

9 Appendix - Survey on Standards 90
9.1 Survey Papers 90
9.2 Ontology standards 90
9.3 SE standards 92
9.4 WWW standards 93
9.5 Text, Video, and Metadata standards 94
9.6 Electronic Commerce Standards 95
9.7 Electronic Commerce portals 96

1 Introduction

Ontologies are a popular research topic in various communities such as knowledge engineering, natural language processing, cooperative information systems, intelligent information integration, and knowledge management. They provide a *shared and common* understanding of a domain that can be communicated between people and heterogeneous and widely spread application systems. They have been developed in Artificial Intelligence to facilitate *knowledge sharing and reuse*. There, *problem-solving methods* (cf. [Fensel, 2000]) describe the reasoning behaviour and *ontologies* describe the static domain knowledge of a knowledge-based system. Some Examples are KIF [Genesereth, 1991], Ontolingua [Gruber, 1993], CYC [Lenat & Guha, 1990], and KQML [KQML]. Recent articles covering various aspects of ontologies can be found in [Uschold & Grüninger, 1996], [van Heijst et al., 1997], [Studer et al., 1998], [Benjamins et al., 1999], [Gomez Perez & Benjamins, 1999]. An ontology provides an explicit conceptualisation (i.e., *meta-information*) that describe the semantics of the data. They have a similar function as a database schema. The differences are[1]:

- A language for defining ontologies is syntactically and semantically richer than common approaches for databases.
- The information that is described by an ontology consists of semi-structured natural language texts and not tabular information.
- An ontology must be a shared and consensual terminology because it is used for information sharing and exchange.
- An ontology provides a domain theory and not the structure of a data container.

Currently computers are changing from single isolated devices to entry points into a worldwide network of information exchange and business transactions. Therefore support in the exchange of data, information, and knowledge is becoming the key issue in current computer technology. Ontologies provide a shared and common understanding of a domain that can be communicated between people and application systems. Providing shared and common domain structures is becoming essential, and ontologies will therefore become a key asset in information exchange used to describe the structure and semantics of information exchange. Currently, Internet technology and the World Wide Web are the main technological infrastructure for on-line information exchange. It is therefore not surprising to see that a number of initiatives are arising in this area to provide notations for data structures and semantics. For example:[2]

- The *Resource Description Framework (RDF)* (cf. [Miller, 1998], [Lassila & Swick,1999]) provides a standard for describing the semantics of information (via metadata descriptions).
- The *Extendible Markup Language (XML)* ([Connolly, 1997], [XML]) provides a standard for describing the structure of information (and some aspects of its semantics).
- *XML schemes* provide a standard for describing the structure and semantics of data.

---

1. See [Meersman, 1999] for an elaborated comparison of database schemes and ontologies.
2. A comparison of these standards can be found in [van Harmelen & Fensel, submitted].

- The transformation language of XSL *(XSL-T)* [Clark, 1999] provides a standard for describing mappings between different terminologies.
- Various querying languages for XML (XQL, XML-QL [QL, 1998]) provide standards for describing mappings between different terminologies.

These arising standards will allow ontology techniques to enter the market places quickly, supporting various aspects of information exchange. Figure 1 depict the three main application areas of this technology I have identified and which I will discuss during this paper: Knowledge Management, Web Commerce, and Electronic Business. These three business fields also correspond roughly to the three different types of networks[3]:

- *Intranet*: closed user community, company- or organization-wide use (Knowledge Management).
- *Internet*: open access; worldwide user community, for example, on-line shopping Business-to-Consumer (B2C) in Web Commerce.
- *Extranet*: limited access from the outside (Internet) to an Intranet, for example, in Business-to-Business (B2B) in Electronic Business.

Intranet & Internet = Information Search & Knowledge Management (D2D). The competitiveness of companies active in areas with a high market change rate depends heavily on how they maintain and access their knowledge (i.e., their corporate memory). Most information in modern electronic media is textual, visual, and audial and rather weakly structured. This holds for the Internet but also for the large intranets of companies and organizations. Finding and maintaining information is a difficult problem in this weakly structured representation media. An increasing number of companies are realizing that their company's intranets are valuable repositories of corporate information. However, raw information in large quantities does not by itself solve business problems, produce value, or provide competitive advantage. Information

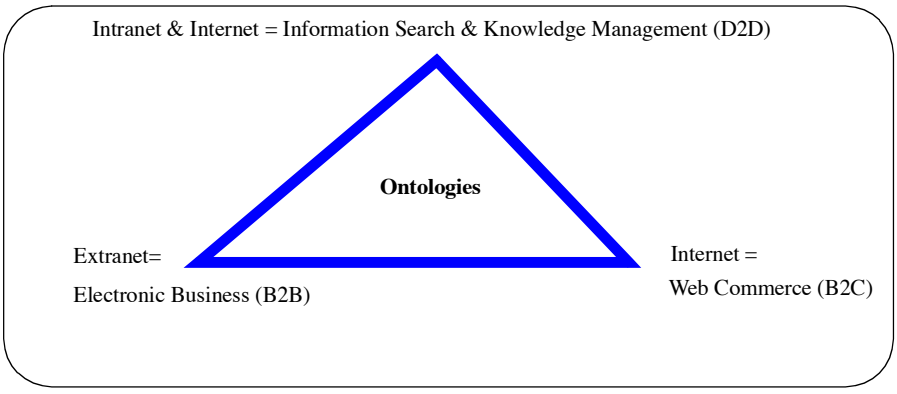


Fig. 1 One technology, three application areas.

3. Except f or Knowledge Management, which also searches for information in the Internet.

is useless without an understanding of how to apply it effectively. But with the volume of information available increasing rapidly, turning information into useful knowledge has become a major problem. *Knowledge Management* is concerned with acquiring, maintaining, and accessing knowledge of an organization. It aims to exploit an organisation's intellectual assets for greater productivity, new value, and increased competitiveness. Due to globalisation and the impact of the Internet, many organizations are increasingly geographically dispersed and organized around virtual teams. Such organizations need knowledge management and organizational memory tools that encourage users to understand each other's changing contextual knowledge and foster collaboration while capturing, representing and interpreting the knowledge resources of their organizations.

With the large number of on-line documents, several document management systems entered the market. However these systems have severe weaknesses:

- *Searching information*: Existing keyword-based search retrieves irrelevant information which uses a certain word in a different context, or it may miss information where different words about the desired content are used.
- *Extracting information*: Human browsing and reading is currently required to extract relevant information from information sources, as automatic agents lack all common sense knowledge required to extract such information from textual representations, and they fail to integrate information spread over different sources.
- *Maintaining* weakly structured text sources is a difficult and time-consuming activity when such sources become large. Keeping such collections consistent, correct, and up-to-date requires a mechanized representation of semantics and constraints that help to detect anomalies.
- *Automatic document generation* [Perkowitz & Etzioni, 1997] discuss the usefulness of adaptive Web sites which enable a dynamic reconfiguration according to user profiles or other relevant aspects. The generation of semi-structured information presentations from semi-structured data requires a machine-accessible representation of the semantics of these information sources.

In the near future, semantic annotations will allow structural and semantic definitions of documents providing completely new possibilities:

- Intelligent search instead of keyword matching.
- Query answering instead of information retrieval.
- Document exchange between departments via XSL translations.
- Definition of views on documents.

In Section 3 I will describe these options in more detail and sketch some existing research prototypes in this area (Ontobroker [Fensel et al., 1998a], On2broker[4] [Fensel et al., 1999a], IBROW[5] [Benjamins et al., 1998], and On-To-Knowledge[6].

Internet = Web Commerce (B2C). Electronic Commerce is becoming an important

---

4. http://www.aifb.uni-karlsruhe.de/www-broker
5. http://www.swi.psy.uva.nl/projects/IBROW3/home.html.
6. http://www.cs.vu.nl/~ontoknowledge

and growing business area. This is happening for two reasons. First, electronic commerce is extending existing business models. It reduces costs and extends existing distribution channels and may even introduce new distribution possibilities. Second, it enables completely new business models or gives them a much greater importance than they had before. What has up to now been a peripheral aspect of a business field may suddenly receive its own important revenue flow. Examples of business field extensions are on-line stores, examples of new business fields are shopping agents, on-line marketplaces and auction houses that make comparison shopping or meditation of shopping processes into a business with its own significant revenue flow.

On-line shops of existing and newly founded enterprises provide new distribution channels with advantages such as: economy in access, overcoming geographical distances, bypassing time limitations in access (like closing hours), anonymity (at least as a psychological fiction), and adaptability to individual customers and customer groups (user profiles and corporative filtering). An example is *Amazon*[7], an on-line bookstore which developed within a few years from a $10.000 investment into a company with stock valued at several billion dollars. Worldwide, more than 100 million people are already using the Internet and a significant percentage of them has used such stores for shopping.

The advantages of on-line stores and the success story of many of them has led to a large number of such shopping pages. The new task for a customer is now to find a shop that sells the product he is looking for, getting it in the desired quality, quantity, and time, and paying as little as possible for it. Achieving these goals via browsing requires significant time and will only cover a small share of the actual offers. Very early, *Bargainfinder* [Krulwich, 1996] and *shopbots* [Etzioni, 1997] (later it became the company *Jango*[8], which has been taken over by *Excite*[9] in the meantime) represented the first approaches for comparison shopping. These shopbots visit several stores, extract product information and present to the customer a instant market overview. Their functionality is provided via *wrappers* that need to be written for each on-line store. Such a wrapper uses a keyword search for finding the product information together with assumptions on regularities in the presentation format of stores and text extraction heuristics. This technology has two severe limitations:

- *Effort*: Writing a wrapper for each on-line store is a time-consuming activity and changes in the outfit of stores cause high maintenance efforts.
- *Quality*: The extracted product information is limited (mostly price information), error prone and incomplete. For example, a wrapper may extract the direct product price but misses indirect costs such as shipping costs etc.

These problems are caused by the fact that most product information is provided in natural language, and automatic text recognition is still a research area with significant unsolved problems. However, the situation will drastically change in the near future when standard representation formalisms for the structure and semantics of data are available. Software agents then can *understand* the product information. Meta-on-line

7. www.amazon.com
8. www.jango.com
9. www.excite.com

stores can be built with little effort and this technique will also enable complete market transparency in the various dimensions of the diverse product properties. The low-level programming of wrappers based on text extraction and format heuristics will be replaced by XSL specifications, which translate different product descriptions in various XML-dialects into each other. An ontology describes the various products and can be used to navigate and search automatically for the required information. Section 4 will go into more details on the role of ontologies in Web Commerce.

Extranet = Electronic Business (B2B). Electronic Commerce in the business to business field (B2B) is not a new phenomena. Initiatives to support electronic data exchange in business processes between different companies existed already in the sixties. In order to exchange business transactions sender and receiver have to agree on a common standard (a protocol for transmitting the content and a language for describing the content) A number of standards arose for this purpose. One of them is the UN initiative *Electronic Data Interchange for Administration, Commerce, and Transport (EDIFACT)* [EDIFACT]. Figure 2 provides an example for a specification of a business transaction in EDIFACT.

In general, the automatization of business transactions has not lived up to the expectations of its propagandists. This can be explained by some serious shortcomings of existing approach like EDIFACT: It is a rather procedural and cumbersome standard, making the programming of business transactions expensive, error prone and hard to maintain. Finally, the exchange of business data via extranets is not integrated with other document exchange processes, i.e., EDIFACT is an isolated standard.

Using the infrastructure of the Internet for business exchange will significantly improve this situation. Standard browsers can be used to render business transactions and these transactions are transparently integrated into other document exchange processes in intranet and Internet environments. The first portals for electronic commerce using Internet facilities are *harbinger.net*, *mysap.com* and *VerticalNet.com*. However, this is currently hampered by the fact that HTML do not provide a means for presenting rich syntax and semantics of data. XML, which is designed to close this gap in current Internet technology, will therefore drastically change the situation (cf. [Glushko et al., 1999]). B2B communication and data exchange can then be modeled with the same means that are available for the other data exchange processes, transaction specifications can easily be rendered by standard browsers, maintenance will be cheap (cf. WebEDI [Westarp et al., 1999] and XML/EDI [Peat & Webber, 1997][10]).

XML will provide a standard serialized syntax for defining the structure and semantics of data. Still, it does not provide standard data structures and terminologies to describe business processes and exchanged products. Therefore, ontologies will have to play two important roles in XML-based electronic commerce:

- *Standard ontologies* have to be developed covering the various business areas. In addition to official standards, on-line marketplaces (Internet portals) may generate de facto standards. If they can attract significant shares of the on-line transactions in a business field they will factually create a standard ontology for this area.

---

10. http://www.xmledi.com/

EDIFACT S93A Sample Document

PURCHASE ORDER

```
UNB+UNOB:1+003897733:01:MFGB-PO+PARTNER ID:ZZ+000101:1050
+00000000000916++ORDERS'
UNH+1+ORDERS:S:93A:UN'
BGM+221+P1M24987E+9'
DTM+4:20000101:102'
FTX+PUR+3++PURCHASE ORDER BEFORE LINE ITEM INSTRUCTIONS'
RFF+CT:123-456'
RFF+CR:1'
NAD+SE+10025392::92++SUPPLIER NAME'
CTA+SR+:STEVE'
NAD+BT+B2::92++COMPAQ COMPUTER CORPORATION+P O BOX 692000
+HOUSTON+TX+77692000+US'
NAD+BY+MFUS::92++COMPAQ COMPUTER CORPORATION'
CTA+PD+:CLARETTA STRICKLAND-FULTON'
NAD+ST+CM6::92++COMPAQ COMPUTER CORPORATION+CCM6 RECEIVING
DOCK:20555 SH 249+HOUSTON+TX+77070+US'
TAX+9+++++++3-00105-5135-3'
CUX+2:USD:9'
PAT+1++1:1:D:45'
PAT+22++1:1:D:30'
PCD+12:2'
TDT+20++++:::AIRBORNE'
LOC+16+COMPAQ DOCK'
TOD+2+NS+:::ORIGIN COLLECT'
LIN+000001++107315-001:BP'
PIA+1+AA:EC+123456:VP'
IMD+F+8+:::PART DESCRIPTION INFORMATION
QTY+21:10000000:PCE'
DTM+2:20000301:102'
FTX+PUR+3++LINE ITEM COMMENTS
PRI+CON:50'
TAX+7++++:::100'
MOA+124:100'
UNS+S'
UNT+29+1'
UNZ+1+000000000000916'
```

Fig. 2 A purchase order in EDIFACT.

- *XSL-based translation services* between different data structures in areas where standard ontologies do not exist or where a particular client wants to use his own terminology and needs translation service from his terminology into the standard. This translation service must cover structural and semantical as well as language differences.

Then, ontology-based trading will significantly extend the degree to which data exchange is automated and will create complete new business models in the participating market segments (cf. [McGuinness, 1999]). Comparing Internet-based electronic commerce in the B2C and B2B one has to admit that B2C is more mature. However, the B2B area will be perspectively more interesting as around 80% of the transaction volume will be in the B2B area.[11]

This book is organized as follows. In chapter 2, I provide a general introduction in Ontologies. In chapter 3, I discuss the knowledge management area where ontologies can be used to further information access in intranets, significantly improving information exchange between departments (D2D). Chapter 4 is devoted to the use of ontologies in Electronic Commerce. Here I deal with the Business-to-Consumer area, i.e., the area of Web Commerce and will show how ontology-based Internet portals will radically change business processes in the Business-to-Business area. In chapters 5 and 6 I take a more technical view. I discuss arising new web standards allowing structural and semantical descriptions of data (RDF, XML, XSL-T, and XQL). Then I discuss possibilities for Ontology Languages and show how standards like XML and RDF can be used to work with ontologies. Finally, I bring my conclusionsin Section 7 . An appendix briefly enumerates the approaches discussed in the book.

11. For example, 55 billion dollar in 2001 in Europe (Forrester Research).

2 Ontologies

Ontologies were developed in Artificial Intelligence to facilitate *knowledge sharing and reuse*. Since the beginning of the nineties ontologies have become a popular research topic investigated by several Artificial Intelligence research communities, including Knowledge Engineering, natural-language processing and knowledge representation. More recently, the notion of ontology is also becoming widespread in fields such as intelligent information integration, cooperative information systems, information retrieval, electronic commerce, and knowledge management. The reason ontologies are becoming so popular is in large part due to what they promise: a shared and common understanding of some domain that can be communicated between people and application systems.

Ontologies are developed to provide a machine-processable semantics of information sources that can be communicated between different agents (software and humans). Many definitions of ontologies have been given in the last decade, but one that, in our opinion, best characterizes the essence of an ontology is based on the related definitions in [Gruber, 1993]: *An ontology is a formal, explicit specification of a shared conceptualisation*. A 'conceptualisation' refers to an abstract model of some phenomenon in the world which identifies the relevant concepts of that phenomenon. 'Explicit' means that the type of concepts used and the constraints on their use are explicitly defined. 'Formal' refers to the fact that the ontology should be machine readable. Hereby different degrees of formality are possible. Large ontologies like WordNet provide a thesaurus for over 100,000 natural language terms explained in natural language (see also [Meersman, 1999] for a discussion of this issue). On the other end of the spectrum is CYC, that provides formal axiomating theories for many aspect of common sense knowledge. 'Shared' reflects the notion that an ontology captures consensual knowledge, that is, it is not restricted to some individual, but accepted by a group. Basically, the role of ontologies in the knowledge engineering process is to facilitate the construction of a domain model. An ontology provides a vocabulary of terms and relations with which to model the domain.

Because ontologies aim at consensual domain knowledge their development is often a cooperative process involving different people, possibly at different locations. People who agree to accept an ontology are said to *commit* themselves to that ontology.

Depending on their generality level, different types of ontologies may be identified that fulfil different roles in the process of building a KBS ([Guarino, 1998], [van Heijst et al., 1997]). Among others, we can distinguish the following ontology types:

- *Domain ontologies* capture the knowledge valid for a particular type of domain (e.g. electronic, medical, mechanic, digital domain).
- *Metadata ontologies* like Dublin Core [Weibel et al., 1995] provide a vocabulary for describing the content of on-line information sources.
- *Generic* or *common sense ontologies* aim at capturing general knowledge about the world, providing basic notions and concepts for things like time, space, state, event etc. ([Fridman-Noy & Hafner, 1997], [Pirlein & Studer, in press]). As a consequence, they are valid across several domains. For example, an ontology about mereology (part-of relations) is applicable in many technical domains [Borst

& Akkermans, 1997].

- *Representational ontologies* do not commit themselves to any particular domain. Such ontologies provide representational entities without stating what should be represented. A well-known representational ontology is the *Frame Ontology* [Gruber, 1993], which defines concepts such as frames, slots, and slot constraints allowing the expression of knowledge in an object-oriented or frame-based way.
- Other types of ontology are so-called *method* and *task ontologies* ([Fensel & Groenboom, 1997], [Studer et al., 1996]). Task ontologies provide terms specific for particular tasks (e.g. 'hypothesis' belongs to the diagnosis task ontology), and method ontologies provide terms specific to particular PSMs (e.g. 'correct state' belongs to the Propose-and-Revise method ontology). Task and method ontologies provide a reasoning point of view on domain knowledge.

Part of the research on ontologies is concerned with envisioning and building enabling technology for the large-scale reuse of ontologies at a world-wide level. In order to enable as much reuse as possible, ontologies should be small modules with a high internal coherence and a limited amount of interaction between the modules. This requirement and others are expressed in design principles for ontologies ([Gruber, 1995], [Guarino, 1995], [Uschold & Grüninger, 1996]).

Assuming that the world is full of well-designed modular ontologies, constructing a new ontology is a matter of assembling existing ones. I [Farquhar et al., 1997] describe the Ontolingua server, which provides different kinds of operations for combining ontologies: inclusion, restriction, and polymorphic refinement. E.g. inclusion of one ontology in another has the effect that the composed ontology consists of the union of the two ontologies (their classes, relations, axioms). The SENSUS system [Swartout et al., 1996] provides a means for constructing a domain specific ontology from given common sense ontologies. The basic idea is to use so-called seed elements which represent the most important domain concepts for identifying the relevant parts of a top-level ontology. The selected parts are then used as starting points for extending the ontology with further domain specific concepts. The SKC project (Scalable Knowledge Composition) [Jannink et al., 1998] aims at developing an algebra for systematically composing ontologies from already existing ones. It will offer union, intersection, and difference as basic operations for such an algebra.

Various kind of formal languages are used for representing ontologies, among others description logics (see e.g. LOOM [MacGregor, 1991] or CYCL [Lenat & Guha, 1990]), Frame Logic [Kifer et al., 1995], and Ontolingua [Gruber, 1993], which is based on KIF (Knowledge Interchange Format) [Genesereth & Fikes, 1992]. It is basically a first-order predicate logic extended with meta-capabilities to reason *about* relations. Languages for expressing ontologies and their relationships to arising web standards will be discussed in a later chapters. In the following, I will provide some illustrations: WordNet, CYC, TOVE, and $(KA)^2$.

WordNet[1] (cf. [Fellbaum, 1999]) is an on-line lexical reference system whose design is inspired by current psycholinguistic theories of human lexical memory. English nouns,

---

1. http://www.cogsci.princeton.edu/~wn/.

verbs, adjectives and adverbs are organized into synonym sets, each representing one underlying lexical concept. Different relations link the synonym sets. It was developed by the Cognitive Science Laboratory at Princeton University. WordNet contains around 100.000 word *meanings* organized in a taxonomy. WordNet groups words into five categories: *noun, verb, adjective, adverb,* and *function word*. Within each category it organizes the words by concepts (i.e., word meanings) and via semantical relationship between words. Examples of these relationships are:

- *Synonymy*: Similarity in meaning of words, which is used to build concepts represented by a set of words.
- *Antonymy*: Dichotomy in meaning of words - mainly used for organizing adjectives and adverbs.
- *Hyponymy*: Is-a relationship between concepts. This is-a hierarchy ensures the inheritance of properties from superconcepts to subconcepts.
- *Meronymy*: Part-of relationship between concepts.
- *Morphological* relations which are used to reduce word forms.

The success of WordNet is based on the fact that it is available on-line, free of charge, and that it is a dictionary based on concepts, i.e. it provides much more than just an alphabetic list of words. A multilingual European version EuroWordNet[2] also exists in the meantime. Specific features of WordNet are its large size (i.e., number of concepts), its domain-independence, and its low level of formalization. With the latter I refer to the fact that WordNet does not provide any definitions of semantics in a formal language. The semantics of concepts is defined with natural language terms. This leaves definitions vague and limits the possibility for automatic reasoning support. WordNet is mainly linguistically motivated. In this respect, WordNet can be seen as one extreme point in a spectrum where CYC defines the other extreme.

CYC[3] [Lenat & Guha, 1990] was initiated in the course of Artificial Intelligence, making common-sense knowledge accessible and processable for computer programs. The lack of common sense knowledge and reasoning was encountered in many if not all application areas of Artificial Intelligence as the main barrier for allowing Intelligence. Take machine learning as an example: on the one hand, learning is a prerequisite of intelligence; on the other hand, intelligence is a prerequisite for meaningful learning. Humans decide based on their common sense knowledge what to learn and what not to learn from their observations. CYC started as an approach to formalize this knowledge of the world and provide it with a formal and executable semantics. Hundreds of thousands of concepts have been formalized in the meantime with millions of logical axioms, rules, and other assertions which specify constraints on the individual objects and classes. Some of them are publicly available at the web page. The upper-level ontology of CYC with 3000 concepts has been made publicly available. These are the most generic concepts which are situated at a high level in the taxonomy of concepts. Most of the more specific concepts are kept secret as property of Cycorp which is the company that commercializes CYC.

---

2. http://www.let.uva.nl/~ewn
3. http://www.cyc.com/.

CYC groups concepts into microtheories to structure the overall ontology. Micro theories are a means to express context dependency of knowledge (i.e., what is right in one context may be wrong in another one, cf. [Lenat, submitted]). They are a means to structure the whole knowledge base, which would be otherwise inconsistent and unmaintainable. Each microtheory is a logical theory introducing terms and defining their semantics with logical axioms. CycL, a variant of predicate logic, is used as language for expressing these theories.

CYC is motivated by ideas from Artificial Intelligence. Like WordNet it is rather large and domain-independent. In difference to WordNet it provides a formal and operational definition of its concepts.

TOVE[4] ([Fox et al., 1993], [Fox & Gruninger, 1997]) is an example of a task and domain-specific ontology. The ontology supports enterprise integration, providing a shareable representation of knowledge. The goal of the TOVE (TOronto Virtual Enterprise) project is to create a generic, reusable data model that has the following characteristics:

- it provides a shared terminology for the enterprise that each agent can jointly understand and use,
- it defines the meaning of each term in precise and unambiguous manner as possible,
- it implements the semantics in a set of axioms that will enable TOVE to automatically deduce the answer to many "common sense" questions about the enterprise, and
- it defines a symbology for depicting a term or the concept constructed thereof in a graphical context.

In consequence, TOVE provides a reusable representation (i.e., ontology) of industrial concepts. Using ontologies for information exchange and business transactions is also investigated in [Uschold et al., 1996].

The *Knowledge Annotation Initiative of Knowledge Acquisition Community (KA)*^{2} (cf. [Benjamins et al., 1999]) was a case study on:

- the process of developing an ontology for a heterogeneous and world-wide (research) community, and
- the use of the ontology for providing semantic access to on-line information sources of this community.[5]

(KA)^2 comprises three main subtasks: (1) ontological engineering to build an ontology of the subject matter, (2) characterizing the knowledge in terms of the ontology, and (3) providing intelligent access to the knowledge. In (KA)^2, an ontology of the Knowledge Acquisition community (cf. an "enterprise knowledge map") was built. Since an ontology should capture consensual knowledge, several researchers cooperated together – at different locations – to construct the ontology in (KA). In this way, it was ensured that the ontology will be accepted by a majority of KA researchers. The design criteria

4. http://www.eil.utoronto.ca/tove/toveont.html.
5. http://www.aifb.uni-karlsruhe.de/WBS/broker/KA2.html.

Class: research-topic
Attributes:
Name: <string>
Description: <text>
Approaches: <set-of keyword>
Research-groups: <set-of research-group>
Researchers: <set-of researcher>
Related-topics: <set-of research-topic>
Subtopics: <set-of research-topic>
Events: <set-of events>
Journals: <set-of journal>
Projects: <set-of project>
Application-areas: <text>
Products: <set-of product>
Bibliographies: <set-of HTML-link>
Mailing-lists: <set-of mailing-list>
Webpages: <set-of HTML-link>
International-funding-agencies: <funding-agency>
National-funding-agencies: <funding-agency>
Author-of-ontology: <set-of researcher>
Date-of-last-modification: <date>

Fig. 3 The meta ontology for specifying research topics in $(KA)^2$.

used to build the (KA) ontology were: modularity, to allow more flexibility and a variety of uses, specialization of general concept into more specific concepts, classification of concepts the similar features to guarantee according to inheritance of such features, and standardized name conventions. The ontology for the KA community consists of seven related ontologies: an organization ontology, a project ontology, a person ontology, a research-topic ontology, a publication ontology, an event ontology, and a research-product ontology. The first six ontologies are rather generic, whereas the latter (i.e., the research-topic ontology) is specific for the investigated domain (see Fig. 3). Actually, a meta-ontology (i.e., a template) for describing research topics was first defined. Then this template was instantiated for the research topics. The topics that were identified in a number of international meetings are: reuse, problem-solving methods, ontologies, validation vnd verification, specification languages, knowledge acquisition methodologies, agent-oriented approaches, knowledge acquisition from natural language, knowledge management, knowledge acquisition through machine mearning, knowledge acquisition through Conceptual Graphs, foundations of knowledge acquisition, evaluation of knowledge acquisition techniques and methodologies, and knowledge elicitation. Each of these topics was given to a small group of experts who completed the scheme in Fig. 3 The research topics ontology can be viewed at the (KA) homepage. More details on this initiative and the technology to formalize and access the knowledge are described in chapter 3.

Finally, I would like to mention *Prot*égé (cf. [Grosso et al., 1999]). Protégé-2000 (cf.

[Puerta et al., 1992], [Errikson et al., 1999]) is the latest version of a series of tools developed in the Knowledge Modeling Group at Stanford Medical Informatics to assist developers in the construction of large electronic knowledge bases. Protégé allows developers to create, browse and edit domain ontologies in a frame-based representation, which is compliant with the OKBC knowledge model [Chaudhri et al., 1998]. Starting with an ontology, Protégé automatically constructs a graphical knowledge-acquisition tool that allows application specialists to enter the detailed content knowledge required to define specific applications. Protégé allows developers to customize this knowledge-acquisition tool directly by arranging and configuring the graphical entities in forms, that are attached to each class in the ontology for the acquisition of instances. This allows application specialists to enter domain information by filling in the blanks of intuitive forms and by drawing diagrams composed of selectable icons and connectors. Protégé-2000 allows knowledge bases to be stored in several formats, among others a CLIPS-based syntax and RDF.

3 Application Area Knowledge Management

In the meantime, large Companies have intranets with several million pages. Finding, creating and maintaining information is a rather difficult problem in this weakly structured representation media. On the other hand, the competitiveness of companies active in areas with a high change rate depends heavily on how they maintain and access their knowledge. Therefore, Knowledge Management deals with *acquiring*, *maintaining*, and *accessing* knowledge of an organization. In the following, I will discuss some of the pitfalls of current techniques and will then show some existing approaches that make use of ontologies for providing much stronger support.[1]

3.1 The pitfalls of current information search

Working with the Web is currently done at a very low level: Clicking on links and using key word search for links is the main (if not only) navigation technique. It is like programming with assembler and go-to instructions. The low-level interface may significantly hamper the expected growth of the Web in the future. Currently, the World Wide Web (WWW) contains around 300 million static objects providing a broad variety of information sources [Bharat & Broder, 1998] and other studies estimate that these static pages provide only around 20% of the actual information accessible via the WWW (cf. [Lawrence & Giles, 1998]). Clearly, simple browsing insufficient as a search technique for this amount of information. Therefore, hundreds of keyword-based search engines have sprung up to support search processes for information.[2] Popular examples are: *Alta Vista*, *Google*, *Yahoo!*, and *MetaCrawler*.[3]

Alta Vista was one of the first keyword-based search engines. Like most other search engines it consists of three components:

- A *webcrawler* downloads documents from the Web.
- An *indexer* extracts key terms from these documents. They are used to represent the retrieved document. A term vector represents the frequency with which a term appears in a document.
- A *query interface* receives query terms which are compared with the database of term vectors.[4] Then, documents which have a similarity factor greater than some threshold are presented successively to the client.

Yahoo! extends this automatic approach through human intervention. A taxonomy of search terms is built up and web documents are classified in this taxonomy by human lectors. This human intervention limits the scope of documents (some authors speak about 200 million indexed documents in the case of *Alta Vista* versus 1 Million

---

1. Actually, I only discuss a small number of prototypical approaches. There are many valuable and interesting ontology-based approaches in this area which I do not discuss: (ONTO)^{2}Agent [Arpírez et al., 1998], ONTOSEEK [Guarino et al., 1999], PlanetOnto [Domingue & Motta, in press], and TREVI [Meersman, 1999].
2. One of the earliest ones is described in [Bowman et al., 1994].
3. http://www.altavista.com, http://www.google.com, http://www.yahoo.com, http://www.metacrawler.com. See also http://searchenginewatch.com/.
4. In the simplest case, it is the scalar product of the term vector representing the document and the term vector representing the query.

classified documents in the case of *Yahoo!*). Therefore, whenever a search term does not correspond to a category in *Yahoo!* it automatically passes the query onto *Alta Vista*.

Google differs from Alta Vista in the determination of the relevance of documents [Brin & Page, 1998]. Like Alta Vista it selects documents based on the similarity of terms that appear in query and documents. However, the order in which documents are represented is then determined by their quotation index. That is, documents that are quoted often by other documents are viewed as being more important than documents that are quoted seldomly. Quotation is counted in terms of hyperlinks that point to a certain document. Given the fact, that an average query may retrieve thousands of documents, presenting relevant documents first is a much more important aspect than representing all documents that may be relevant (i.e., *completeness* is often of theoretical importance only).

Finally *MetaCrawler* [Selberg & Etzioni, 1997] is not a search engine on its own but rather an interface that integrates several search engines. This meta-search engine transfers a user query to several search engines, integrating and cleaning up their results. In consequence, the user is freed from interacting with several search engines in cases where one search engine alone does not provide the desired results. However, there are severe problems, which will be discussed in the following.

3.1.1 How to avoid nonsense and find what you are looking for

Imagine that you want to find out about the research subjects of a researcher named *Smith* or *Feather*.[5] Consulting a search engine will result with a huge set of pages containing the key words *Feather*. Preciseness and recall are limited.

- Precision: how many retrieved documents are really relevant?
- Recall: have I found all relevant information?

All pages containing the string *Feather* are returned and many of these pages are completely irrelevant (see Figure 4). The important page may be missing. In the actual example, I can find the homepage with the search expression *feather+research* (see Figure 5). However, imagine that he has a headline like "*Topics of interest*" at the page that is imported by a framed homepage. Such a page does not contain any of the assumed keywords. Even if the person's pages are identified, a significant human search effort is required to investigate these pages until the page that contains the required information has been found. Even search engines specialized in retrieving homepages of persons cannot make use of the information that he is a researcher and are specialized in retrieving address information and not in making sophisticated queries. For example about what a person does.[6]

3.1.2 Information presentation and access is limited

The format of query response is a list of hyperlinks and textual and graphical information that is denoted by them. It requires human browsing and reading to extract

---

5. Not to mention the case where his name is *Cook*.

6. This limitations become rather comical as in the meantime AltaVista retrieved a description of Ontobroker (see the following subsection) as result of the query *feather+research* because this query was used in on-line available publications dealing with Ontobroker (see Figure 6).

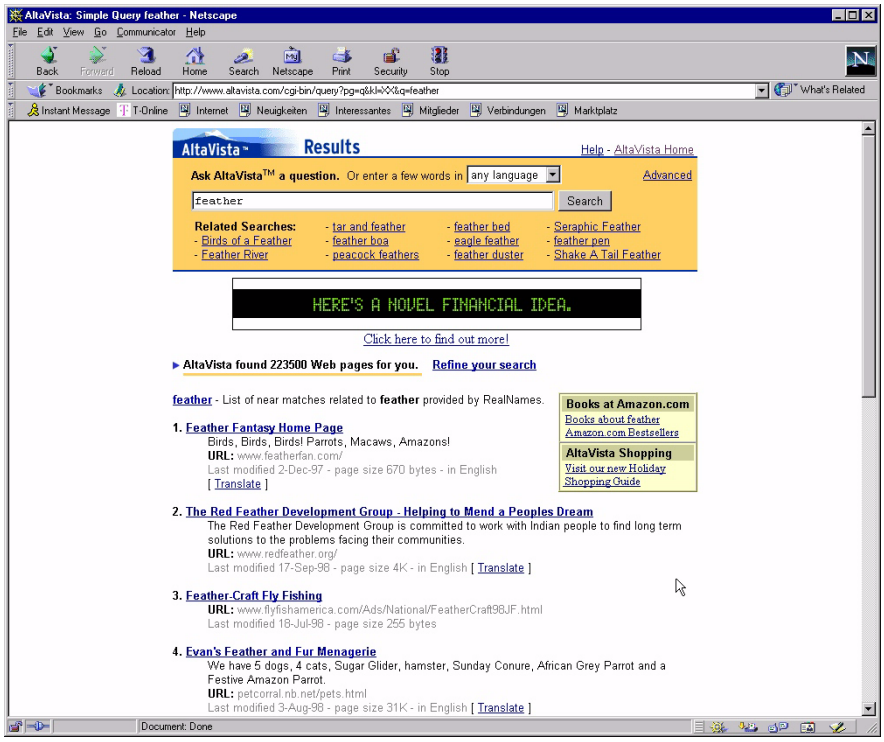


Fig. 4 Searching with *Alta Vista* search for *feather*.

the relevant information from these information sources. Remember, we were looking for the research subjects of Mr. *Feather*. We would like to get a list of research topics like: "*World Wide Web, Ontologies, Knowledge Acquisition, Software Engineering*". However, it requires further human extraction to retrieve this information. This burdens web users with an additional loss of time[7] and seriously limits information retrieval by automatic agents that miss all common sense knowledge required to extract such information from textual representations. A further consequence is that the outcome of a web query cannot directly be processed by another software tool, because a human has to extract and represent it in a way that fits some standard representation.

3.1.3 How to collect distributed information

Still, the above mentioned problems are rather trivial compared to queries that refer to the content of several pages. Imagine that you want to find the research subjects of a research group. You have to determine whether this is on a central page or whether each researcher enumerates them on his pages. Then you have to determine all members of

7. Compare this with databases and SQL, where a user can make precise queries leading to precise answers.

Fig. 5 Searching with *Alta Vista* search for *feather* and *research*.

this research group and go through all their pages. The required search effort and lack of recall make such queries impractical for a large, widely spread ,and heterogeneous group of people (i.e., Web sources). Imagine that you want to extract the research topics of all researchers who also work on ontologies. This shows fairly clearly that the current information access to the WWW cannot handle information that is scattered at several locations and pages.

3.1.4 How to collect implicit information

Finally, each current retrieval service can only retrieve information that is represented by the WWW. This sounds trivial, but it significantly limits query answering capability. Imagine that Feather writes on his homepage that he cooperates with another researcher E. Motta on investigating formal specifications of problem-solving methods. However, you will not find 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. However, an answering mechanism that can make use of the implicit symmetry of cooperation could provide you with this answer. Similarly, because Smith is a researcher and he cooperates on research issues with E. Motta it can be derived that E. Motta is also a researcher and may want to receive this information even if it is not explicitly stated on one of E. Mottas´ pages. Here we would make use of a type information of a relationship.

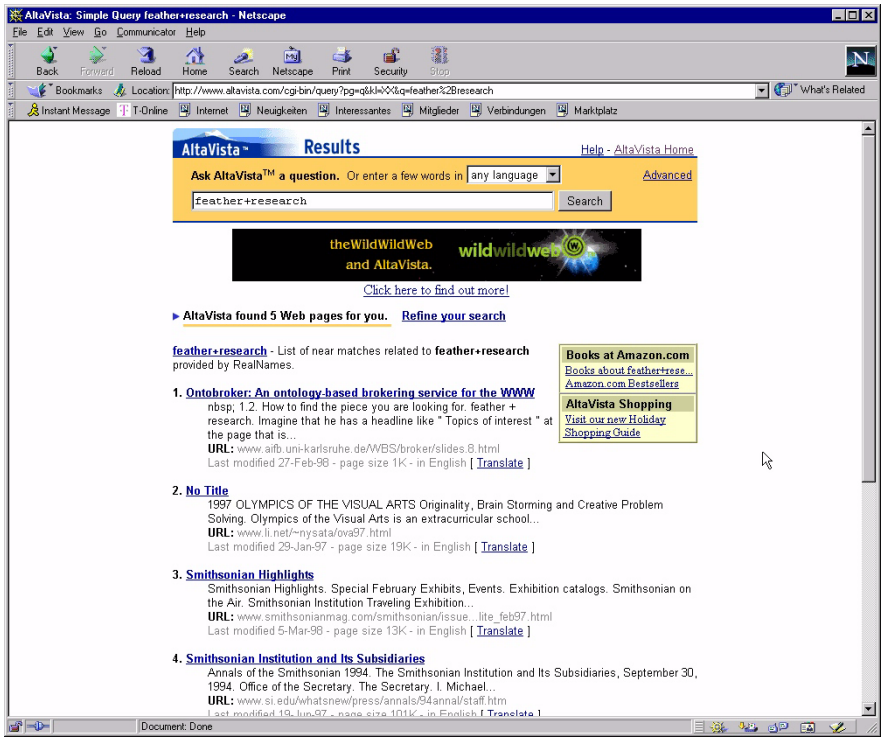


Fig. 6 ... and try it recently.

3.2 How Ontobroker overcomes these limitations

Ontobroker (cf. [Fensel et al., 1998a], [Decker et al., 1999]) applies Artificial Intelligence techniques to improve access to heterogeneous, scattered and semi-structured 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. The gist of the matter is: to define an ontology and use it to annotate/structure/wrap your web documents, and somebody else can make use of Ontobroker's advanced query and inference services to consult your knowledge. To achieve this goal, Ontobroker provides three interleaved languages and two tools. It 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 close coupling of informal, semiformal, and formal information and knowledge. This supports their maintenance and provides a service that can be used more generally for

integrating knowledge-based reasoning with semi-formal represented documents.

3.2.1 The Languages

Ontobroker provides three interleaved languages: An *annotation* language is offered to enable knowledge providers to enrich web documents with ontological information. It provides a *representation* language for formulating ontologies. A subset of it is used to formulate queries, i.e. to define the *query language*.

3.2.1.1 The Annotation Language

Ontobroker 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.</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.

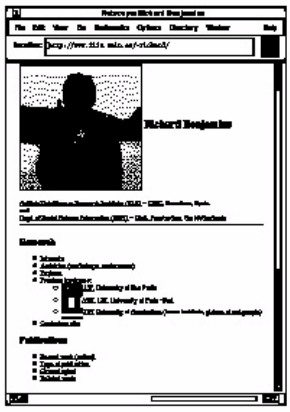
The reason for the former decision was to lower the threshold for using our annotation language. People who are able to write HTML can use it straightforwardly as a simple extension. The pages remain readable by standard browsers like Netscape Navigator or MS Explorer, and information providers can still rely on standard web techniques. 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 semantics saying that this ascii string provides the name of a researcher. This is a significant difference between our approach and approaches like SHOE[8] [Luke et al. 1997], RDF [Lassila & Swick,1999] and annotations used in information retrieval.

In Ontobroker, a frame-based approach has been chosen for the annotation language corresponding to the kind of language used for representing the ontology. Three primitives are provided to annotate web documents :

- An *object* can be defined as an instance of a certain class.

---

8. 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 Ontobroker. 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 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. Ontobroker uses 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.



```
<html>
<head><TITLE> Richard Benjamins </TITLE>
<a onto="page:Researcher"> </a>
</head>

<H1 > <A HREF="pictures/id-rich.gif">
<IMG  align=middle SRC="pictures/richard.gif"></A>
<a onto="page[photo=href]"
HREF="http://www.iiia.csic.es/~richard/pictures/richard.gif"
></a>

<a onto="page[firstName=body]">Richard</a>
<a onto="page[lastName=body]">Benjamins </a>
</h1 > <p>

<A onto="page[affiliation=body]"HREF="#card">
Artificial Intelligence Research Institute (IIIA)</A> -
<a href="http://www.csic.es/">CSIC</a>, Barcelona, Spain
<br>
and <br>
<A onto="page[affiliation=body]"HREF="http://
www.swi.psy.uva.nl/">
...
```

Fig. 7 An example for an annotated web page.

- The *value* of an object´s attribute can be set.
- A *relationship* between two or more objects may be established.

All three primitives are expressed by using an extended version of a frequent HTML tag, i.e. the anchor tag: <a ...> ... . The anchor tag is usually used to define named locations in a web page and links to other locations. Thus, it contains the attributes *name* and *href* to fulfill these purposes. For ontologically annotating a web page Ontobroker provides another attribute to the syntax of the anchor tag, namely the ***onto*** attribute. Typically, a provider of information first defines an object as an element of a certain class. To express this in its HTML extension he would use the following line on a home page:

<a *onto*=´"http://www.iiia.csic.es/~richard" : Researcher´>

URLs are used as object-ids. Each class could possibly be associated with a set of attributes. Each instance of a class can define values for these attributes. For example, the ontology contains an attribute email for each object of class Researcher. If Richard Benjamins would like to provide his email address, he would use this line on his home page:

<a *onto*=' "http://www.iiia.csic.es/~richard"
[email=?mailto:richard@iiia.csic.es?]'>

The object denoted by "http://www.iiia.csic.es/~richard" has the value "mailto:richard@iiia.csic.es" for the attribute email. An example for an annotated web page is given in Figure 7.

In terms of a knowledge-based system, the annotation language provides the means to express factual knowledge (ground literals). Further knowledge is provided by the

Class Hierarchy	Attribute Definitions	Rules

Object[].
Person :: Object.
Employee :: Person.
AcademicStaff :: Employee.
Researcher :: AcademicStaff.
Publication :: Object.
	Person[
	firstName =>> STRING;
	lastName =>> STRING;
	eMail =>> STRING;
	...
	publication =>> Publication].
	Employee[
	affiliation =>> Organization;
	...].
	Researcher[
	researchInterest =>>
	ResearchTopic;
	...].
		FORALL P_1, P_2
		P_1[cooperatesWith ->> P_2]
		<-
		P_2[cooperatesWith->> P_1].
		
		FORALL P, Pub
		Pub:Publication
		[author ->> P]
		<->
		P:Person
		[publication ->> Pub].

c_1 :: c_2 means that c_1 is a subclass of c_2.
$c[a$ ==> $r]$ means that an attribute a is of domain c and range r.
o : $c[a$->> $v]$ means that o is element of c hand has the value v for a.
<- means logical implication and <-> logical equivalence.

Fig. 8 An excerpt of an ontology (taken from [Benjamins et al., 1999]).

ontology. The ontology defines the terminology (i.e., signature) and may introduce further rules (i.e., axioms) that allow the derivation of additional facts that are not stated as extensions.

3.2.1.2 The Representation Languages

A *representation* language is used to formulate an ontology. This language is based on *Frame logic* [Kifer et al., 1995]. F-Logic is a language for specifying object-oriented databases, Frame systems, and logical programs. Its main achievement is to integrate conceptual modeling constructs (classes, attributes, domain and range restrictions, inheritance, axioms) into a coherent logical framework. 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 in the Web. An example is provided in Figure 8. It defines the class *Object* and its subclasses *Person* and *Publication*. Some attributes and some rules expressing relationships between them are defined, for example, if a publication has a person as an author then the author should have it as a publication. Semantically, the language for defining rules is the

fragment of first-order logic that can be transformed via Lloyd-Topor transformations [Lloyd & Topor, 1984] into Horn logic. Syntactically it is different as it incorporates object-oriented modeling primitives. Ontobroker uses a subset of F-logic for defining the Ontologies.

- Class definition:
  $c[]$
  defines a class with name c.
- Attribute definition:
  $c[a{=>>}\{c_1,...,c_n\}]$
  implies that the attribute a can applied to the elements of c (it is also possible to define attributes applied to classes) and an attribute value must be member of all classes $c_1,...,c_n$.
- Is-a relationship:
  $c_1 :: c_2$
  defines c_1 as a subclass of c_2 which implie that:
  - all elements of c_1 are also elements of c_2,
  - all attributes and their value restrictions defined for c_2 are also defined for c_1, and
  - multiple attribute inheritance exists, i.e.
    $c :: c_1[a =>> \{c_3\}]$ and $c :: c_2[a =>> \{c_4\}]$ implies
    $c[a =>> \{c_3,c_4\}]$
- Is-element-of relationship:
  $e : c$
  defines e as an element of the class c.
- Rules like
  - FORALL $x,y\ x[a \text{->>} y] \text{<-} y[a \text{->>} x]$.
  - FORALL $x,y\ x{:}c_1[a_1 \text{->>} y] \text{<->} y{:}c_2[a_2 \text{->>} x]$.

3.2.1.3 The Query Languages

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:

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

In the head of F-Logic rules, variables are all quantified. In the body, variables may be either all or existentially quantified. All quantified variables must additionally be bound by a positive atom in the body. Lloyd-Topor transformation handles these quantifications as follows. Existential quantifiers in the body may be dropped, because every variable in the body of a rule is implicitly existentially quantified. An all-quantification, *forall* $y\ p(y)$, in the body is transformed to a $\neg$ *exists* $y \neg p(y)$. Then Lloyd-Topor transformation produces a set of rules out of this. Queries are handled as rules without a head. Thus the above mentioned conditions for quantifications hold here too.

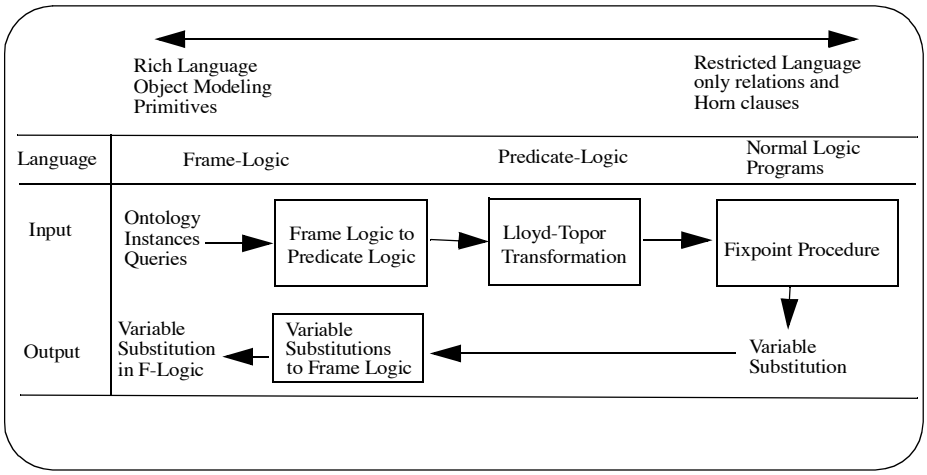


Fig. 9 Stages and Languages used in the Inference Engine.

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 publications of the researcher „Richard Benjamins“.

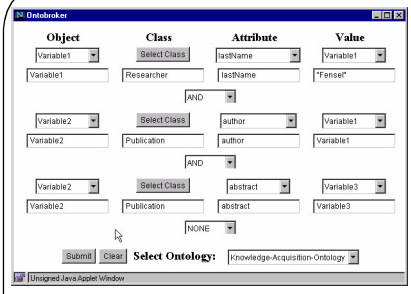
x[*name* -> „Richard Benjamins“; *publication* ->> { *y*[*abstract* -> *z*]}] : *Researcher*

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

3.2.2 The Tools

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 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 Fig. 9.

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 negation in the clause body is allowed, we have to carefully select an appropriate semantics and evaluation procedure. If the resulting program is stratified, Ontobroker uses simple stratified semantics and evaluates it with a technique called dynamic filtering (cf. [Kifer & Lozinskii, 1986], [Fensel et al., 1998b]), which focuses the inference engine on the relevant parts of a minimal model required to answer the query. Dynamic filtering combines bottom-up and top-down evaluation techniques. The top-down part restricts the set of facts which has to be computed to a subset of the minimal model. *Thus infinite minimal models are also possible, because only this subset has to be finite*.[9] 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, Ontobroker uses the *well-founded model*



Researcher* with *name* "*Fensel*" & *Publications* of this *author* & their *abstracts

Ontobroker found the following:

V1 = "http://www.aifb.uni-karlsruhe.de/WBS/dfe/index.html"

V2 = "http://www.aifb.uni-karlsruhe.de/WBS/dfe/publications97.html#EEF+97"

V3 = "Building knowledge-based systems from reusable elements is a key factor in their economic development. However, one has to ensure that the assumptions and functionality of the reused building block fit to each other and the specific circumstances of the actual problem and knowledge. We use the Karlsruhe Interactive Verifier (KIV) for this purpose. We show how the verification of conceptual and formal specifications of knowledge-based systems can be done with it. KIV was originally developed for the verification of procedural programs but it fits well for verifying knowledge-based systems. Its specification language is based on algebraic specification means for the functional specification of components and dynamic logic for the algorithmic specification. It provides an interactive theorem prover integrated into a sophisticated tool environment supporting aspects like the automatic generation of proof obligations, generation of counter examples, proof management, proof reuse etc. Such a support is essential in making verification of complex specifications feasible. We provide some examples on how to specify and verify tasks, problem-solving methods, and their relationships."

V1 = "http://www.aifb.uni-karlsruhe.de/WBS/dfe/index.html"

V2 = "http://www.aifb.uni-karlsruhe.de/WBS/dfe/publications97.html#FS97"

V3 = "During the last years, a number of formal specification languages for knowledge-based systems have been developed. Characteristic for knowledge-based systems are a complex knowledge base and an inference engine which uses this knowledge to solve a given problem. Specification languages for knowledge-based systems have to cover both aspects: they have to provide means to specify a complex and large amount of knowledge and they have to provide means to specify the dynamic reasoning behaviour of a knowledge-based system. This paper will focus on the second aspect, which is an issue considered to be unsolved. For this purpose, we have surveyed existing approaches in related areas of research. We have taken approaches for the specification of information systems (i.e., Language for Conceptual Modelling and Troll), approaches for the specification of database updates and the dynamics of logic programs (Transaction Logic and Dynamic Database Logic), and the approach of Evolving Algebras. This paper, which is a short version of a longer report, concentrates on the methodology of our comparison and on the conclusions we have drawn. The actual comparison between the languages has

Fig. 10 The tabular query interface.

semantics [Van Gelder et al., 1991] and computes this semantics with an extension of dynamic filtering.

A hyperbolic presentation of the ontology and a tabular interface improve the accessibility of Ontobroker. Expecting a normal web user to type queries in a logical language and to browse large formal definitions of ontologies is not very realistic. Therefore, the structure of the query language is exploited to provide a tabular query interface as shown in Figure 10. We also need support for selecting classes and

9. Syntactical rules that ensure that the subset of minimal model that has to be computed remains finite are described in [Fensel et al., 1998b].

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 concepts. With regard 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 and classes at the border of the surrounding circle are only marked with a small circle (see Figure 11). 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.

3.2.3 Conclusions

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 are used to describe the desired content.
- 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
- Keyword-based document retrieval fails to integrate information spread over different sources.
- Finally, each current retrieval service can only retrieve information that is represented by the WWW. No further inference service is provided for deriving implicit information.

Ontobroker[10] is available on the Web and has been applied in a few applications in the meantime. One is the $(KA)^2$ initiative[11] that is developing an ontology for annotating web documents of the knowledge acquisition community [Benjamins et al., 1999].

3.3 The Future beyond Ontobroker

In this subsection I will discuss some new research prototypes that extends Ontobrokers facilities in several dimensions. On2broker (cf. [Fensel et al., 1999a], [Fensel et al., to appear]) is the successor system to Ontobroker. The major new design decisions in

---

10. http://www.aifb.uni-karlsruhe.de/www-broker.
11. The Knowledge Annotation Initiative of the Knowledge Acquisition Community $(KA)^2$.

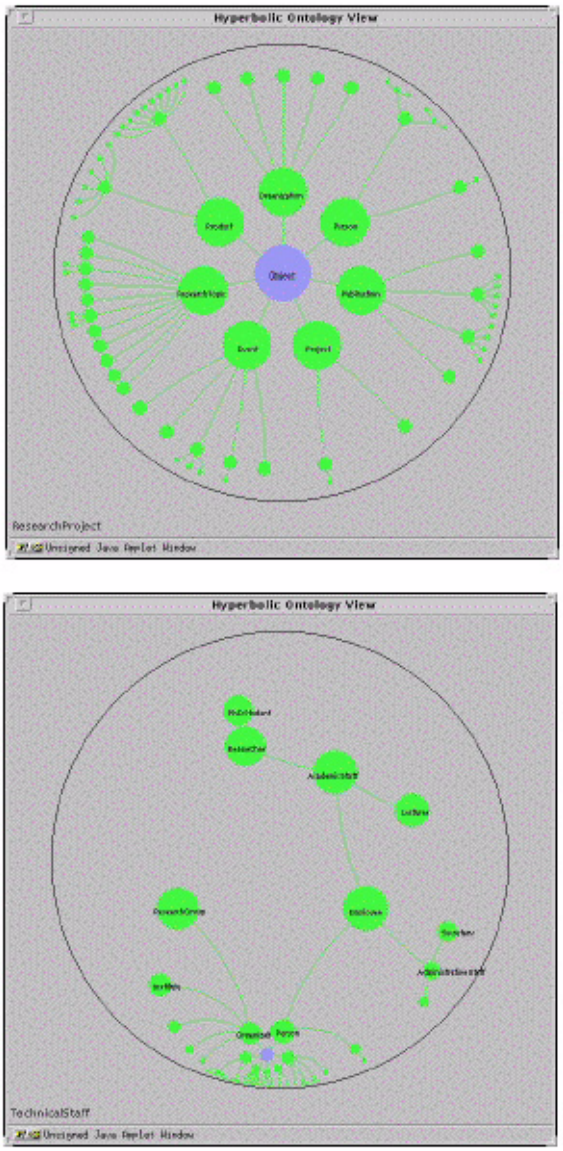


Fig. 11 The hyperbolic ontology interface.

On2broker are the clear separation of the 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 required for a large number of facts and the human annotation effort necessary for adding semantics to HTML documents.

On-To-Knowledge[12] is a running projects under the 5th European Framework program. It will provide improved information access in digital networks. On-To-Knowledge develops a three-layered architecture for information access. At the lowest level (the *information level*), weakly-structured information sources are processed to extract machine-processable meta-information from them. The intermediate level (the *representation level*) uses this meta-information to provide automatic access, creation, and maintenance of these information sources. The highest level (called the *access level*) uses agent-based techniques as well as state-of the art querying and visualization techniques that fully employ formal annotations to guide user access of information. At all levels, *ontologies* are the key asset in achieving the described functionality.

IBROW[13] (cf. [Benjamins et al., 1998], [Fensel & Benjamins, 1998], [Fensel et al., 1999b]) is another running project under the 5th European Framework program. It will provide customizable reasoning service in addition to information access. IBROW develops an Internet-based broker for the access of *dynamic reasoning services* in the WWW. This broker can handle web requests of customers for classes of knowledge systems by accessing libraries of reusable problem-solving methods on the Web, and by *selecting*, *adapting*, *configuring,* and *executing* these methods in accordance with the customer's problem and domain. In consequence a user is not only supports in finding information but also in executing the task for which he or she requires such information.

3.3.1 On2broker

The overall architecture of On2broker, which includes four basic engines representing different aspects, is provided in Fig. 12.

- 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 **info agent** is responsible for collecting factual knowledge from the Web using various types of meta annotations, direct annotations like XML and in future also text mining techniques.
- The **inference engine** 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.
- 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.

3.3.1.1 The Database Manager: Decoupling Inference and Query Response[14]

In the worst case, a query may lead to the evaluation of the entire minimal model of a set of facts and rules. This is a computational hard problem (cf. [Brewka & Dix, 1999]). In

12. http://www.cs.vu.nl/~dieter/ontoknowledge.
13. http://www.swi.psy.uva.nl/projects/IBROW3/home.html.

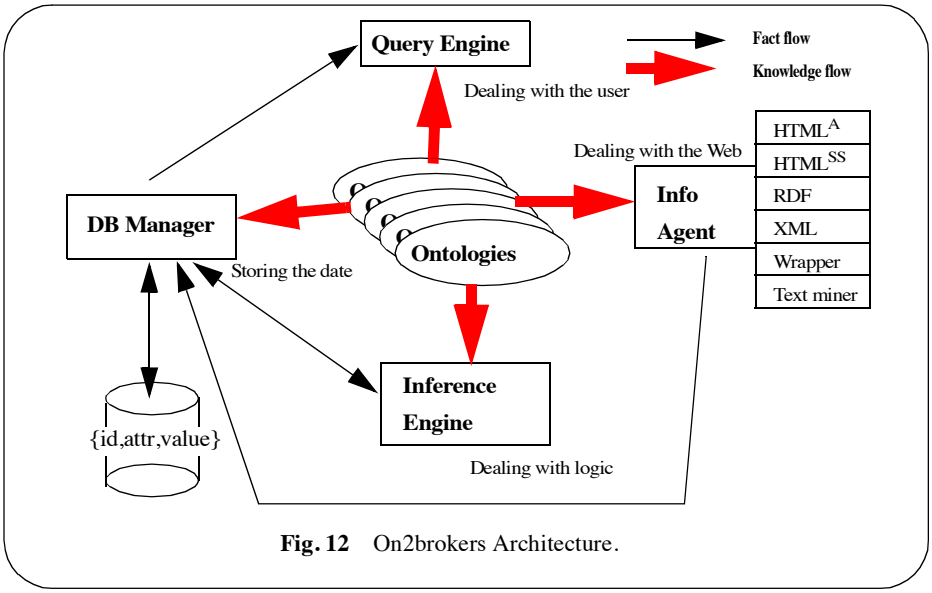


Fig. 12 On2brokers Architecture.

other cases, predicate symbols and constants are used to divide the set of facts into subsets in order to omit those subsets which do not contribute to the answer. This normally reduces the evaluation effort considerably. Ontobroker allows very flexible queries such as "what attributes does a class have". As a consequence, 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*)). This reification strategy implies that the set of facts is only divided into a few subsets. Using few predicates has the consequence that nearly every rule set is not stratified (cf. [Ullman, 1988]) if negation in rules is allowed. Therefore Ontobroker has to make use of the Wellfounded Semantics (cf. [Van Gelder et al., 1991]) because wellfounded Model Semantics also allows us to evaluate non stratified rule sets.

Both points, the small number of predicates and the Wellfounded Model Semantics produce severe efficiency problems. It can only be applied to knowledge bases with less than 100,000 facts. However, it is clear that such an approach should be applicable to millions of facts in order to be of practical relevance. This pointed out a serious shortcoming of the overall system architecture of Ontobroker. In Ontobroker, the query engine and the inference engine are actually *one* engine. The inference engine receives a query and derives the answer. However, an important decision was already made in the design of Ontobroker when the web crawler and the inference engine were seperated. The web crawler periodically collects information from the Web and caches it. The inference engine uses this cache when answering queries. The decoupling of inferences and fact collection is done for efficiency reasons. The same strategy is used by search

14. In terms of the database community On2broker is a kind of data warehouse for data in the Web. Queries are not run on the sources to which On2broker provides access, but on a database into which the source content has been extracted. In addition to the facts that can be found explicitly in the sources, the system also applies rules to derive additional information.

engines in the Web. A query is answered with the help of their indexed cache and not by starting to extract pages from the Web. On2broker refines the architecture of Ontobroker 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. The query engine does not directly interact with the inference engine. Instead it takes facts from the database:

- Whenever inference is a time critical activity, it can be performed in the background independent of the time required to answer the query.
- 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 wild cards, term similarity, and ranking in the query answering mechanism. They can now be directly integrated into the SQL query interface (i.e., in part they are already provided by SQL) and do not require any changes for the much more complex inference engine.

The strict separation of query and inference engines can be weakened for 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 (cf. [Fensel et al., 1998b]). You can make use of this strategy, when deciding which facts are to be put into the database. Either you limit the queries that can be processed by the system or you replace real entries in the database with a virtual entry representing a query to the inference engine. The latter may necessitate a long delay in answering, which, however, may be acceptable for user agents which collect information of the WWW in a background mode. Finally, you can cache the results of such queries to speed up the process in cases where it is asked again. In many application contexts the full flexibility of the query interface is not necessary ,but rather information answering a set of predefined queries. This also holds for the automatic generation of documents. Here, the document results from a query that is executed when the document is retrieved by a user. Therefore, such a document corresponds to a predefined query.

3.3.1.2 The Info Agent

The info agent extracts factual knowledge from web sources. I will discuss the four possibilities I provide in On2broker.

First, On2broker uses Ontobrokers' minor extension of HTML called HTML^{A} to integrate semantic annotations in HTML documents. On2broker uses a *webcrawler* to collect pages from the Web, extracts their annotations, and parses them into the internal format of On2broker.

Second, you can make use of wrappers for automatically extracting knowledge from web sources. 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. Writing wrappers for stable information sources enable the application of On2broker to structured information sources that do not make

use of an annotation language to make explicit the semantics of the information sources.

Third, On2broker can make use of RDF Annotations (cf. [Lassila & Swick,1999]). Manually adding annotations to web sources requires human effort and causes costs in terms of time and money. However, this annotation effort may become less problematic by spreading it over the entire web community. Currently the Resource Description Framework (RDF) is arising as a standard for annotating web sources with machine-processable metadata. *RDF* provides a means for adding semantics to a document without making any assumptions about the internal structure of this document. The info engine of Onto2broker extracts RDF descriptions, and the inference engine of On2broker specialized for RDF is called *SiLRI (Simple Logic-based RDF Interpreter)* [Decker et al., 1998].[15]

Fourth, another interesting possibility is the increased use of the eXtensible Markup language XML. In many cases, the tags defined by a Document Type Definition (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

3.3.1.3 Conclusions

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 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 semi-structured information sources. On2broker is also used to create and maintain such semi-structured information sources, i.e. it is a tool for web site construction and restructuring.

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 semi-structured information presentations *from* other semi-structured ones. The results of a query to On2broker may be inserted as Java-Script data structures into the HTML-stream of a web page. Thus using Java-Script, the query results may be presented in

15. http://www.w3.org/RDF, RDF Software and Products.

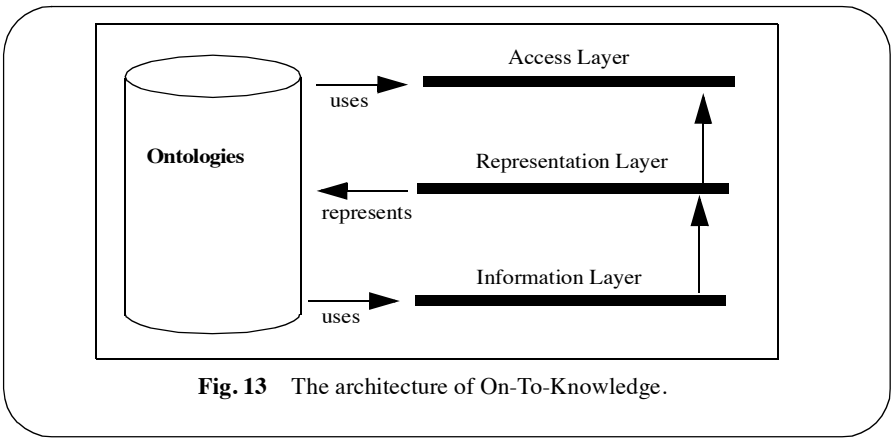


Fig. 13 The architecture of On-To-Knowledge.

every desired form within this page. This allows the insertion of content into a web page which is dynamically generated by On2broker.

Maintenance of weakly structured text sources helps to detect inconsistencies among documents and between documents and external sources, i.e., to detect incorrectness. Maintaining intranets of large organizations and companies is becoming a serious effort, because such networks already provide several million documents. WebMaster ([van Harmelen & van der Meer, 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). Here the Ontology is not used to derive additional facts, but rather to ensure that the provided knowledge is consistent and correct.

3.3.2 On-To-Knowledge: Evolving Ontologies for Knowledge Management

The goal of the On-To-Knowledge project[16] is to support efficient and effective knowledge management. It focuses on acquiring, maintaining, and accessing weakly-structured on-line information sources:

- *Acquiring*: Text mining and extraction techniques are applied to extract semantic information from textual information (i.e., to acquire information).
- *Maintaining*: RDF and XML are used for describing syntax and semantics of semi-structured information sources. Tool support enables automatic maintenance and view definitions of this knowledge.
- *Accessing*: Push services and agent technology support users in accessing this knowledge.

For all tasks, *ontologies* are the key asset in achieving the described functionality. Ontologies are used to annotate unstructured information with structural and semantic information. Ontologies are used to integrate information from various sources and to formulate constraints over their content. Finally, ontologies help to improve user access to this information. Users can define their own personalized view, their user profile, and

16. http://www.cs.vu.nl/~dieter/ontoknowledge.

their information agents in terms of an ontology. On-To-Knowledge focuses especially on working with large, scattered, and heterogeneous ontologies.

This tool environment is embedded in a *methodology* that provides guidelines for introducing knowledge management concepts and tools into enterprises, helping knowledge providers to present their knowledge efficiently and effectively. The methodology will include the identification of goals that are to be achieved by knowledge management tools and will be based on an analysis of business processes and the different roles knowledge workers play in organizations.

Current application cases of On-To-Knowledge are Organizational Memories of large organizations, help desks in call centers, and virtual enterprises.

3.3.3 IBROW: Brokering dynamic reasoning services in the WWW

On2broker and On-To-Knowledge provide query access to static information sources. IBROW[17] (cf. [Benjamins et al., 1998], [Fensel & Benjamins, 1998], [Fensel et al., 1999b]) is a project that has the goal of developing a broker for the access of *dynamic reasoning services* in the WWW. The objective of IBROW3 is to develop intelligent brokers that are able to configure reusable components in knowledge systems through the World-Wide Web. The WWW is changing the nature of software development to a distributive plug & play process which requires a new kind of managing software: *intelligent software brokers*. On successful completion of the project, an intelligent web-broker is provided that can handle Web requests for inference services. The broker is able to handle both the customer and the supplier side of the request. It will access libraries in the Internet, search for appropriate inference services, verify their requirements, request additional information from the customer if needed, adapt the inference services to the particular domain knowledge, plug them together, and execute them via CORBA. Therefore, the user no longer buys, downloads and installs software. Instead he uses it as a computational service provided via the network (cf. [Flammia & McCandless, 1997]).

The overall picture of IBROW3 is illustrated in Figure 14. The intelligent broker will be able to handle requests for reasoners from various customers. Based on these requests it will access different libraries available in the Web and will search them for candidate inference services, which will be adapted and configured into a knowledge system for the customer. Library providers will have to make sure that their libraries comply with the description language UPML [Fensel et al., 1999b] and the interoperability protocol.

IBROW opens the way for a new form of electronic commerce in which the services are intelligent reasoning services. Different business models can be envisioned. In the business-to-consumer (B2C) area we can imagine end users who want to solve a concrete problem such as the classification of plants, filtering of web pages, or selection of suitable algorithms for different kinds of data. Based on stated user requirements, IBROW technology configures a suitable reasoner from generic knowledge components and executes it to provide the consumer with an answer. Depending on the popularity of the consumer request, you could decide to store the configured service for later reuse or to throw it away. Commercial exploitation of such services would require consumers to

---

17. http://www.swi.psy.uva.nl/projects/IBROW3/home.html.

pay, either per use or through subscription.

In a business-to-business (B2B) context, IBROW technology can be used to construct half products, which then need further processing by industries before delivering end products to consumers. For example, a car manufacturer could be interested in a service that helps him to develop and/or adapt a new car design. In another scenario, the IBROW broker provides a service to configure the bare bones of a knowledge system, which then needs to be refined for end consumers based on their particular needs. Yet another model would use IBROW technology to provide an underlying infrastructure to support knowledge engineers in selecting, testing, adapting, refining, and combining generic components into concrete systems.

Ibrow moves work on inference services in the direction of multi-agent systems. Important topics in this area are matchmaking between user requests on the one side and competence descriptions of available agents at the other, as well as the delegation of tasks to heterogeneous agent societies (see for example RETSINA, [Decker et al., 1997], [Sycara et al., 1999]). Linking both areas more closely will be done in the near future.

3.4 The service pyramid

Figure 15 sketches the three layers of services I discussed in this Section. Technology

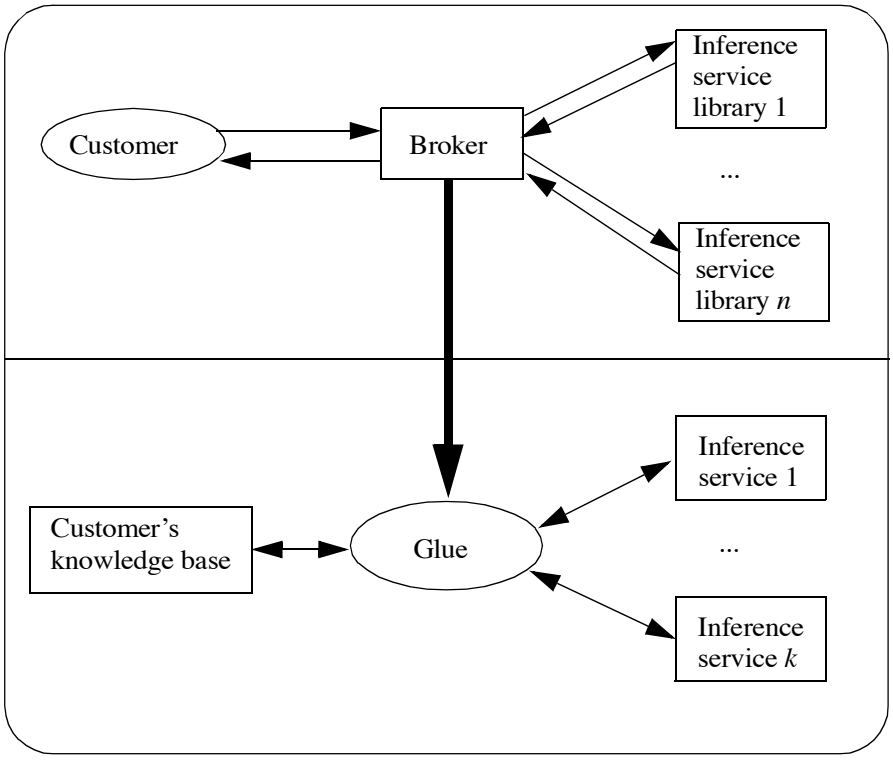


Fig. 14 IBROW: Brokering dynamic reasoning services in the WWW

like search engines in the WWW currently provides support automatic information retrieval which helps in finding information sources. The remaining tasks of extracting the information and using the information to solve a given task remains for the human user. Projects like Ontobroker, On2broker, and On-To-Knowledge add an additional level of service on top by providing automated information extraction support, helping the user in information access and interpretation. Finally, projects like IBROW also provide reasoning service that supports users in task fulfillment. Lets take a travel planning task as an example. Current techniques provide a large number of web pages where information can be found. Intermediate services provide answers to precise questions for travelling connections, specifying locations, dates, and maximal prices. Services like IBROW support in the overall configuration of a trip, where several constraints on the combination of different means of travel and domiciles have to be fulfilled.

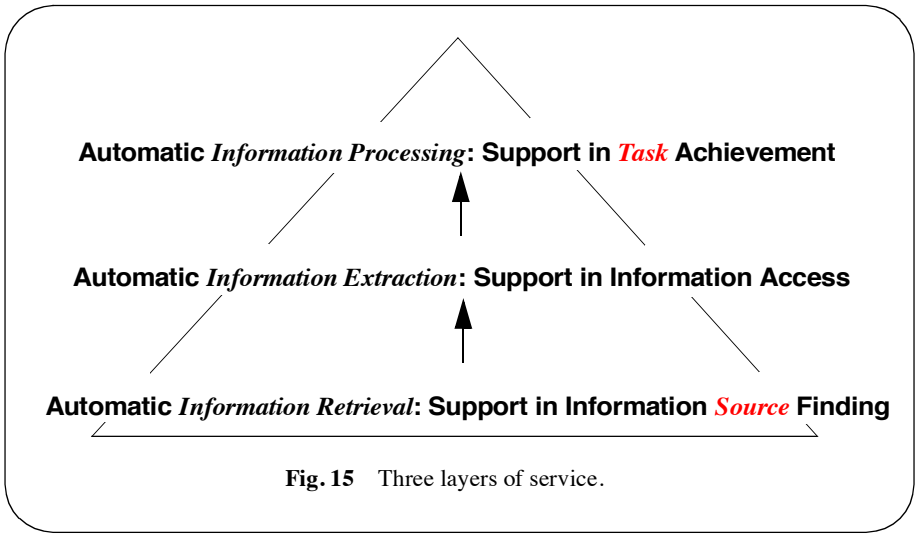


Fig. 15 Three layers of service.

4 Application Area Electronic Commerce

This section will discuss electronic commerce in B2B (Web Commerce) and B2B (Electronic Business). We will examine the need for extended formats and standards for data and information exchange. Therefore, ontologies will play a central role in these areas (cf. [McGuinness, 1999]).

4.1 Application area B2C
Shopbots, Adaptive On-line Stores, and On-line Market places

This subsection will examine the usefulness of ontologies in the consumer market of electronic commerce, i.e., we will discuss the business-to-consumer field (B2C). After a short introduction, we will discuss the field from the consumer perspective (i.e., shopbots), from the seller perspective (adaptive on-line stores), and from the more global and integrating perspective of on-line market places. Finally, we will show how arising technologies will improve the current situation.

4.1.1 Introduction

The area we will be talking about is still new and growing. Therefore, established taxonomies do not exist. The common goal of the approaches we will discuss is to improve the usability of the WWW for electronic commerce by enhancing its accessibility. We will distinguish three types of approaches for better interfaces:

- Intelligent information search agents (i.e., shopping agents) that help customers to find products.
- Intelligent information providers (i.e., on-line stores) that help vendors to present their goods in appropriate manner.
- Intelligent information brokers (i.e., on-line market places) that mediate between buyers and vendors.

Why is there a need for change? Working with the Web is currently done at a very low level. Clicking on links and using key word search for links is the main (if not only) navigation technique. It is like programming with assembler and go-to instructions. This low-level interface significantly hampers the expected growth of the Web and electronic commerce. In particular, it blocks several of the potential superiorities of on-line shopping.

- **Individual product search**. Per definition, on-line stores make product information available on-line. Therefore, physical and time barriers to the access of this information are eliminated. In principle, it requires only some mouse clicks to find and access the desired information. However, finding the right on-line store that sells the desired product at a reasonable price may be very time consuming, too.
- **Corporative product search.** User profiles that support users in searching for products they are likely to fit to their needs could be built up automatically. Corporative strategies try to find similar users and use their product choices as recommendations. However, such possibilities are rarely used currently.
- **Market transparency.** For traditional marketplaces, *complete* product and marketplace information is illusory, i.e., the costs of achieving complete

information are much greater than the savings they provide. With on-line shopping, complete market transparency could be achieved. All information is available on-line and could in principle be easily accessed and compared. However, manually visiting numerous on-line stores and extracting and comparing product information is also not practicable.

- **Easy access.** Buying a product is freed from physical and time barriers and the whole process could nearly be automized. In the extreme case, a software agent could search and buy a product in place of the human client.
- **Negotiation**. Fixed prices turned up at the beginning of the 20th century to lower transaction costs. However, negotiations and auctions help to allocate resources more optimally. Still, the negotiation effort may ouweigh the advantages and lead to unreasonably high demands on time (and transaction costs). Automated negotiation agents and auction houses dispell the argument of high transaction costs and allow optimized resource allocation.

Comparing the current situation with the above sketched potential shows that on-line commerce is far from realizing its future promise. Approaches that take steps towards realizing more of its potential merits will be discussed in the following.

4.1.2 Shopbots

> "Softbots (software robots) are intelligent agents that use software tools and services on a person´s behalf." [Etzioni, 1997]

Shopbots[1] are special-purpose information search, filter, and integration agents[2] providing much better recall and precision than general-purpose search engines, and they usually add some extra service. Their general architecture is shown in Figure 16. This architecture is based on the concept of providing integrated access to heterogeneous and distributed information sources developed in [Wiederhold, 1992], [Wiederhold et al., 1997], and [Wiederhold & Genesereth et al., 1997]. *Wrappers* abstract from syntactical variants in which information is provided by the various sources. The *mediator* accesses these sources (via their wrappers) to answer queries. It has to decide which information source it accesses, it may decompose into subqueries to several information sources and it has to integrate the answers. In the case of on-line shopping, the client interacts with the mediator via a web browser.

In general, three kinds of services types of shopping agents can be distinguished: Passive shopbots that search product information based on explicit user input, active shopbots that try to anticipate user desires and provide proposals, and finally corporative filtering agents that also try to anticipate user desires, however, not only by watching him, but also by watching other users.

An early example of passive shopbots is *BargainFinder* (cf. [Krulwich, 1996]): It returns the prices for a CD in different on-line shops in the Web (see Figure 17 and

---

1. Shopbots try to maximize the utility of buyers, pricebots try to maximize utility of vendors (cf. [Greenwald & Kephart, 1999]).
2. For an introduction to agent technology see [Nwana, 1996] and on intelligent information agents see [Klusch, 1999].

Figure 18). BargainFinder allows users to compare prices among eight compact disc sources offered in the Internet. It is a special search engine (in terms of its purpose and sources). It is specialized for a small set of information sources (i.e., on-line CD stores) and it returns price information only. Basically it is program that automatically queries the cgi-scripts of product provider pages (wrapper-aspect) and integrates the results (mediator-aspect). It was developed by Anderson Consulting as a tool for the end consumer market segment of electronic commerce. Various *softbots* have been developed at the University of Washington (cf. [Doorenbos et al., 1997], [Etzioni, 1997]). Shopbot is a general framework for customizing special-purpose shopping agents. Machine Learning techniques are applied to vendor sites to semi-mechanize the wrapper generation process (see also [Kushmerick, 1997], [Muslea et al., 1998]). The application of automated procedures for wrapper constructions is possible because many on-line store use similar navigation and layout rules for accessing their sortiment. This approach has been commercialized by the company *Jango*[3] which has been purchased by *Excite*[4] in the meantime.

A large number of similar companies exist in the meantime and we are close to the situation of needing meta-shopping agents that select from the various shopping agents: *Bookblvd* (http://www.bookblvd.com/), *Bottom Dollar* (http://www.bottomdollar.com/), *Buyer's Index* (http://www.buyersindex.com/), *CompareNet* (http://www.compare.net/), *Dealpilot* (http://www.dealpilot.com/), *Jango* (http://www.jango.com/), *Junglee* (http://www.junglee.com/)[5], *MyShop* (http://www.myshop.de), *Shopfind* (http://

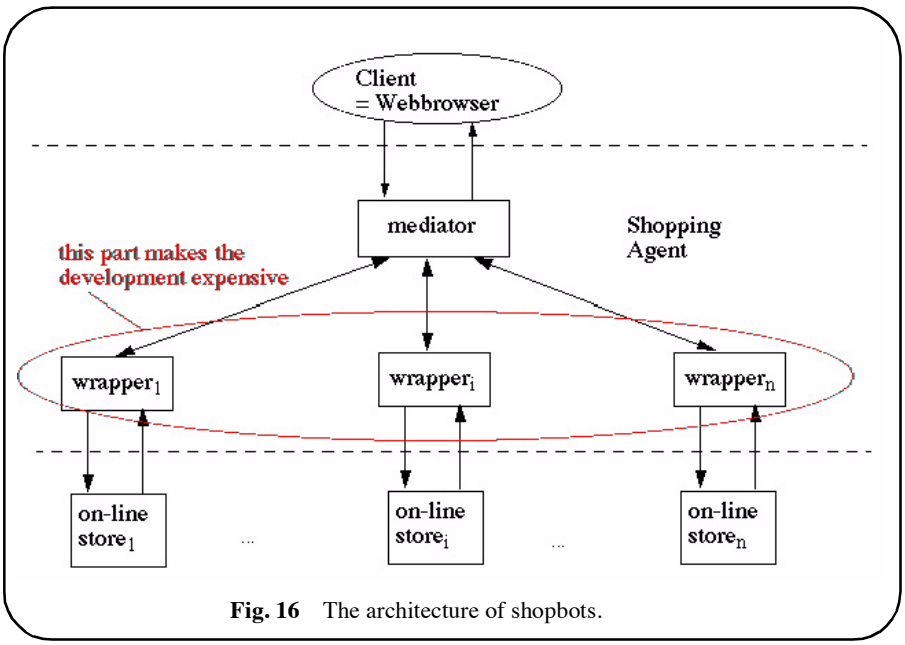


Fig. 16 The architecture of shopbots.

3. www.jango.com
4. www.excite.com
5. Bought by Amazon.

www.shopfind.com/), and *Shopper* (http://www.shopper.com).

Originally these shopbots had problems in finding a business model . Web users do not want to pay because they are used to free service. Product providers do not want to fund the agent because of its ability to always find the cheapest source. Actually BargainFinder was blocked and Shopbot disguised itself as an applet for a while. Product providers would fund the agent if it manipulated the search results. This would eliminate objectivity however which is a requirement for high acceptance. Financing by banners requires a very high traffic, which is difficult for a shopping agent to achieve. In the end, most of them were bought by Internet portals, which could provide an additional feature, or investors tried to build an Internet portal with them. It is quite natural to view them as a specific feature of search engines, like jango which was sold to Excite.

Current, shopbot technology is mostly passive. The user has to play the active part and the shopbots help in collecting the required information. Active shopbots would search for product information which may be interesting for its user. An example for such type of agents in the area of information search is *Letzina* (cf. [Lieberman, 1998a], [Lieberman, 1998b]). Letzina is an information reconnaissance agent. As you are looking at a page, Letzina does an incremental breadth-first search from the current

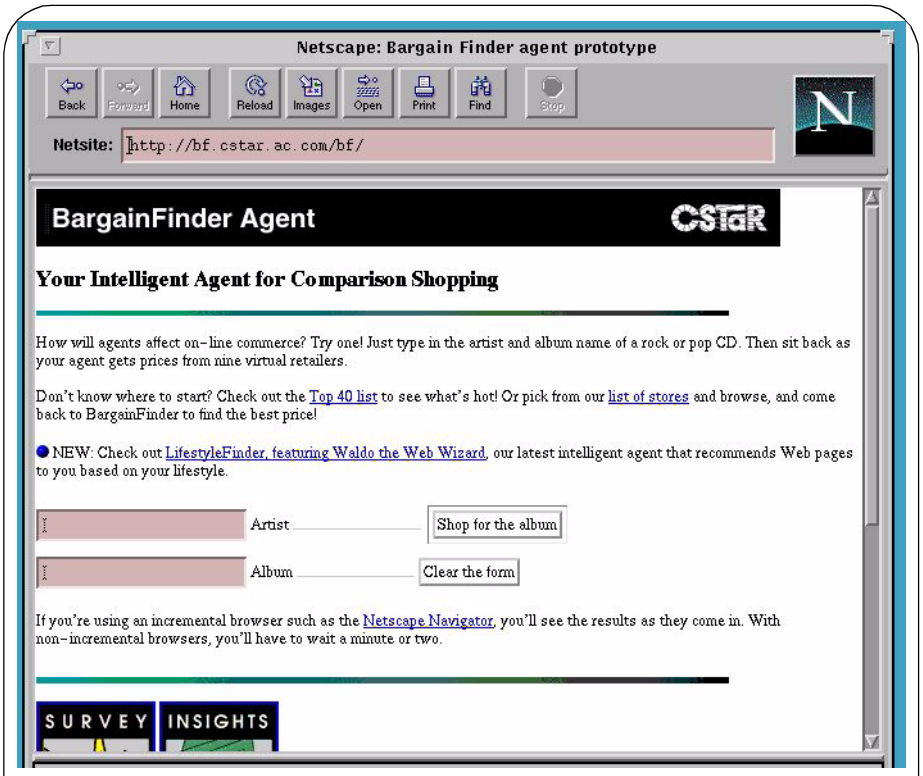


Fig. 17 Bargainfinder (http://bf.cstar.ac.com/bf).

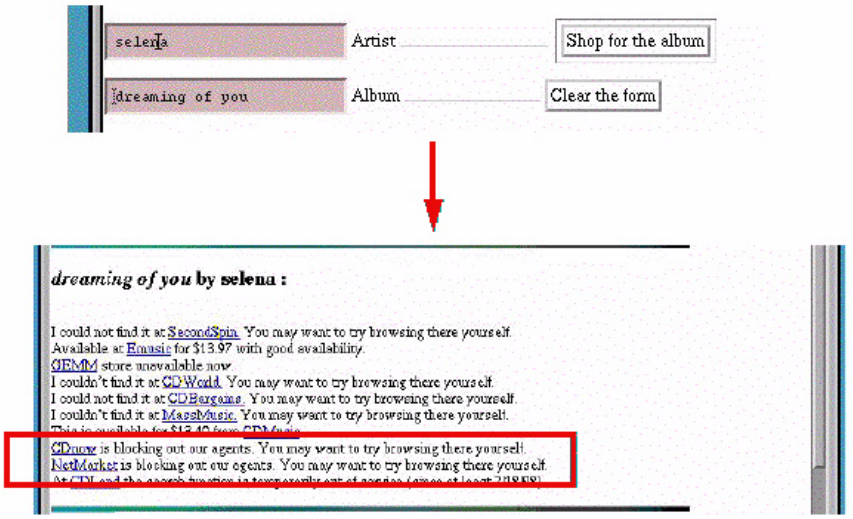


Fig. 18 Response of Bargainfinder.

page, previewing the links. It uses an automatically generated user profile to recommend new information sources. Central for all active information agents is that they require knowledge about the preferences of their users. Letzina directly watches the user's behavior and builds a profile from it. However, this can be achieved in several ways:

- *Lifestyle Finder* (cf. [Krulwich, 1997]): querying the user for his preferences. *Lifestyle Finder* recommends documents matching your interests based on your answers to a set of questions.
- *Alexa*: watches all its users and anticipates their preferences. Alexa recommends new additional web sources "similar" to those currently visited by its user by watching what its other users have selected as their next page.
- *Firefly* (cf. [Shardanand & Maes, 1995]) is a commercialized corporate filtering agent. Firefly asks for ratings of specific musical artists, correlates each user with others who share their tastes, and recommends songs or albums which their cohorts have rated highly.

Lifestyle Finder[6] is a prototype developed by Anderson Consulting. The idea is to take data about a user´s likes and dislikes and generate a general profile of the user. These profiles can be used to retrieve documents matching user interests; recommended music, movies, or other similar products; or carry out other tasks in a specialized fashion. A user profile is generated through a dialogue with a user (see Figure 20) where the answers are used to classify the user into one out of 62 demographic clusters that were derived from surveys.

6. http://bf.cstar.ac.com/lifestyle

Fig. 19 Jango (http://www.jango.com)

Alexa[7] can be best understood as an intelligent proxy. A normal proxy caches parts of the Web visited by a user. It reduces overload of the net (here it helps the net provider) and makes off-line browsing possible (here it helps the user). An "intelligent" proxy caches the pages a user *will* visit. An intelligent proxy is a user agent that collects pages a user is or may be interested in. It need a user profile and knowledge about web sources. Alexa is a plug-in for web browsers that archives and stores all web sources that are visited by its users. It is a proxy of all web sources visited by its user community. In that sense it will become a web archive. It also stores and analyses the web browsing of its users to recommend interesting web sources to them. It enables commenting of web sources and it recommends to its user a set of pages that may be related with the current page a user is looking at.

Firefly is a spin-off of MIT bought by Microsoft in the meantime. Its technology is

7. http://www.alexa.com

based on Ringo (cf. [Shardanand & Maes, 1995]) which uses social filtering techniques to guide users through information sources and product catalogues. It consists mainly of three steps:

- The system maintains a user profile, a record of the user's interests in specific items.
- It compares this profile to the profiles of other users and weights each profile for its degree of similarity with the user's profile.
- Finally, it considers a set of the most similar profiles and uses information contained in them to recommend items to the user.

[Hagel III & Singer, 1999] argue that such (social) shopping agents may be an early form of what they call *infomediaries*. These infomediaries are aware of customer desires via extensive profiling and help them to find the products they need via the extensive market survey and analysis as well as social filtering techniques they can apply based on their large number of customers. They also help to protect these profiles as property of

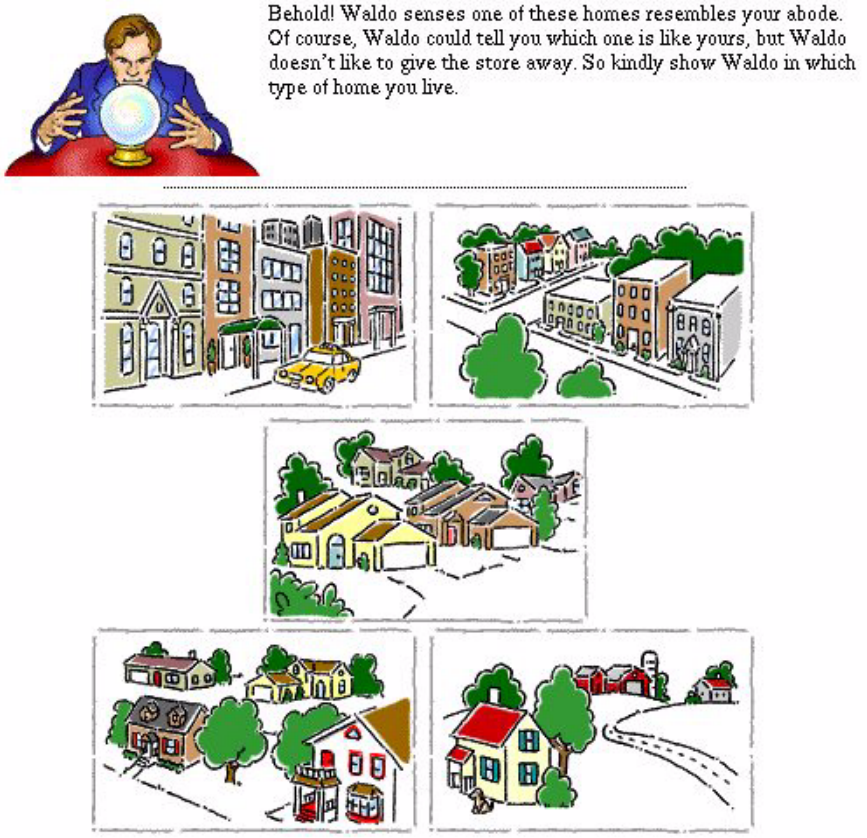


Fig. 20 Lifestyle Finder.

their users. Because they mediate between buyers and vendors they can provide a twofold service: They can protect buyers from spam and, in cases where buyers allow it, they can provide guided access to potential clients for vendors. They may reshape the business model in many economic branches because they will become the portals that mediate customer-vendor relationships. They may help to actually realize for clients the total market transparency that is possible with electronically available commercial offers. Also they enable one-to-one marketing for the broad public. As a general perspective [Hagel III & Singer, 1999] expect that power will shift from vendors to consumers via these infomediaries. Currently, consumers are atomized and generally lack a means for cooperative actions. Informediators may become powerful representatives of their clients. A step into this directions are web portals like Accompany.[8] Here customers come together to form selling groups which enable them to ask for large discounts.

4.1.3 Adaptive On-line Stores

Establishing on-line stores is a routine activity in the meantime.[9] However, most on-line store do not employ the full power of the new medium: *Adaptivity* and *intelligence*. Shops must be adaptable to user preferences and his specific current context. Then they can fully employ the superiority of the on-line media. Physically rearranging the presentation of the entire offer of a supermarket for each client and each point in time he visits it is not affordable. In case of on-line stores, precisely this can be achieved (cf. [Perkowitz & Etzioni, 1997], [Perkowitz & Etzioni, 1999]).

Lets look at an example from information presentation: *WebWatcher*[10] (cf. [Joachims et al., 1997]) is a tour guide agent for the World Wide Web developed at CMU (Carnegie Mellon). For example, it is applied for the CMU School of Computer Science Front Door as a tour guide for visitors. First, you have to tell it which information you seek. Then it

- accompanies you from page to page as you browse the Web,
- highlights hyperlinks that may be of interest for you, and
- you can provide feedback to it.

WebWatcher recommendations were learned from feedbacks from earlier users. Basically it learns a function:

UserChoice : Page x Goal x Link -> [0,1]

where page is the current URL the user is visiting, goal is the information need he or she stated atin the beginning, and link represents the next URLs recommended by the system if their value for the function *UserChoice* is over some threshold. Clearly we could expect similar concepts in on-line stores adapting to the specific needs of their individual customers.

4.1.4 On-line Market Places

Until now, we discussed intelligent support in finding information (i.e., products) and

---

8. http://www.accompany.com
9. See for example http://www.heitlinger-wein.de/ that was built within a seminar with students.
10. http://www.cs.cmu.edu/People/webwatcher

Fig. 21 WebWatcher

intelligent support in presenting information (i.e., products). That is, we discussed direct support for somebody who is searching for a house and direct support for somebody who is trying to sell a house. And what is missing ? The guy who makes most of the money, the mediator. Take Priceline[11] as an example. It is an auction house for airline tickets in the B2C segment. Customers can set a bid and airline companies can accept it and sell a ticket to the price the customer is offering. In its first year, Priceline sold tickets for 35 million dollars making a profit of more than 100 million dollars. After its first year Priceline was introduced to the stock market and its value on the first day was nearly 10 billion dollars. *Negotiation* helps to fairly allocate limited resources. Fixed prices are a phenomena which is only around 100 years old . However, there are impediments to using negotiation.

- In the physical world, certain types of auctions require all parties to be geographically together in, say, an auction house.
- Negotiating may also be too complicated or frustrating for the average consumer.
- Moreover, some negotiation protocols perform over an extended period of time that does not suit impatient or time-constrained consumers

Therefore, real-world negotiations accrue transaction costs. Most of these impediments

11. http://www.priceline.com

of negotiation disappear in the digital world. Instead of human buyers and sellers agents will meet in artificial houses to negotiate prices. Examples of auction houses[12] are: *Adauction*[13], *Alando*[14], *Amazon*[15], *Andsold*[16], *Artnet*[17], *Auction Bot*[18], *eBay´s Auction house*[19], *FreeMarkets*[20], *Kasbah*[21], *Mondus*[22], *National Transport Exchange*[23], *Nextag*[24], *OnSale*[25], *Ricardo*[26], and *Ron Angels*[27]. With Kasbah [Chavez & Maes, 1996] a user can create an agent, give it some strategic direction, and send it off into a centralized marketplace. Kasbah agents proactively seek out potential buyers or sellers and negotiate with them on behalf of their owner.

4.1.5 Electronic Commerce, Agents, and XML

We have described a number of approaches that provide automatic support in electronic commerce ([O´Leary, 1997b]). Currently, the mixture of natural language, gifs, and layout information of HTML is the major barrier for the *automatization* of electronic commerce, because the semantics of the information is only understood by human users. Therefore, no real automatic processing of this information can be provided. This significantly hampers the realization of the advantages of electronic commerce. The information service provided by shopping agents is limited: they heuristically extract some information, but they cannot fully understand natural language and the effort for developing and maintaining shopping agents is costly.

The new standard XML will significantly improve the situation. HTML is a layout language for presenting textual documents whereas XML is a language for defining the structure and semantics of information. Therefore, it enables directed information search and the exchange of structured data (for example, between databases). In consequence, the automated processing of information will be possible and electronic commerce can be executed by software agents. Still XML only provides a standardized syntax for exchanging data. Defining the structure and semantics (i.e., the vocabulary and its meaning), is required additionally. This is precisely what can be achieved with ontologies (cf. [Glushko et al., 1999], [Maes et al., 1999]). We will show some of these interesting perspectives in the context of B2B.

---

12. They also include *reverse* auction houses like Priceline and Nextag, where customers submit a request together with a maximum price and suppliers can decide whether to accept this request.
13. http://www.adauction.com
14. http://www.alando.de (meanwhile bought by ebay).
15. www.amazon.com/auctions
16. www.andsold.de
17. http://www.artnet.com
18. http://auction.eecs.umich.edu
19. http://www.ebay.com
20. http://www.Freemarkets.com
21. http://kasbah.media.mit.edu
22. http://www.mondus.com
23. http://www.nte.com
24. http://www.nextag.com
25. http://www.onsale.com
26. http://www.ricardo.de
27. http://www.ronangels.com

4.2 Electronic Trading Systems as B2B Market Places

Electronic Commerce (EC) in the business to consumer area (B2C) is already a well-established business field (e.g., the bookstore Amazon and the auction house Priceline are companies with billions of dollars in stocks). EC is currently still being developed in the B2B area. In the long run, however, it will be the more interesting area, as around 80% of the transaction volume will be in the B2B area. In general, there are three business cases for electronic commerce in the B2B area (see Figure 16):

- **1:1**. Two companies exchange business transactions electronically. They need to negotiate a joint system and data exchange standard (often EDIFACT[28] & converter). With new Internet technology it can be done via TCP/IP and XML. These Internet technologies provide a better integration with other data exchange processes and tool environments but do not essentially change the business models.
- **1:*N***. One company exchanges business transactions electronically with a number of other (smaller) companies. Usually it is a large vendor or a large buyer that creates this network. It dictates a joint system and data exchange standard (often EDIFACT & converter). With Internet technology it can be done via TCP/IP and XML. Again, a better integration in other data exchange processes and tool environments and a lower threshold for acceptance is achieved without changing the business model. However, a new interesting aspect is the on-line availability of product catalogues.
- ***N*:*M***. *M* companies exchange business transactions electronically with *N* companies in a *fragmented* market. An Internet-based marketplace can help significantly to bring both sides together. It provides instant market overview and offers comparison shopping. *This marketplace will significantly change the business model of this market segment. From the business point of view, these marketplaces are the most interesting kind of EC*. Basically, it will replace or at

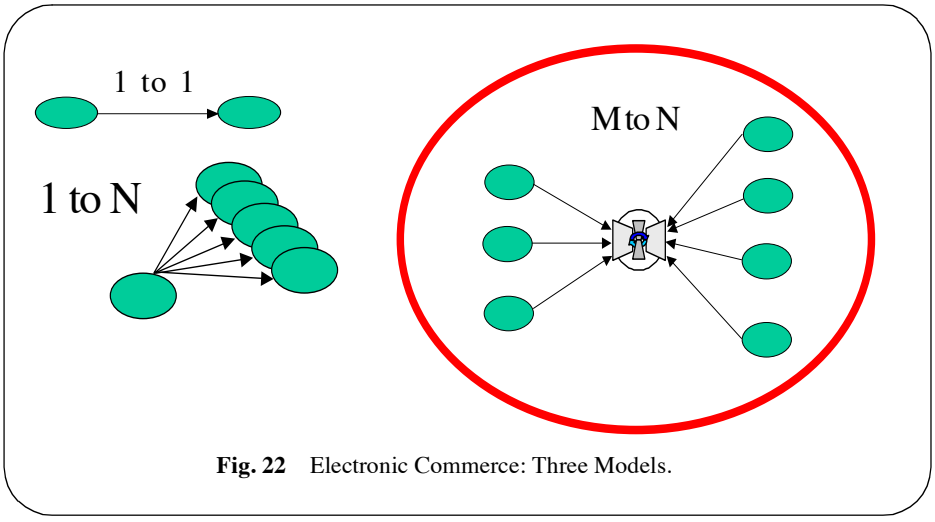


Fig. 22 Electronic Commerce: Three Models.

28. [EDIFACT].

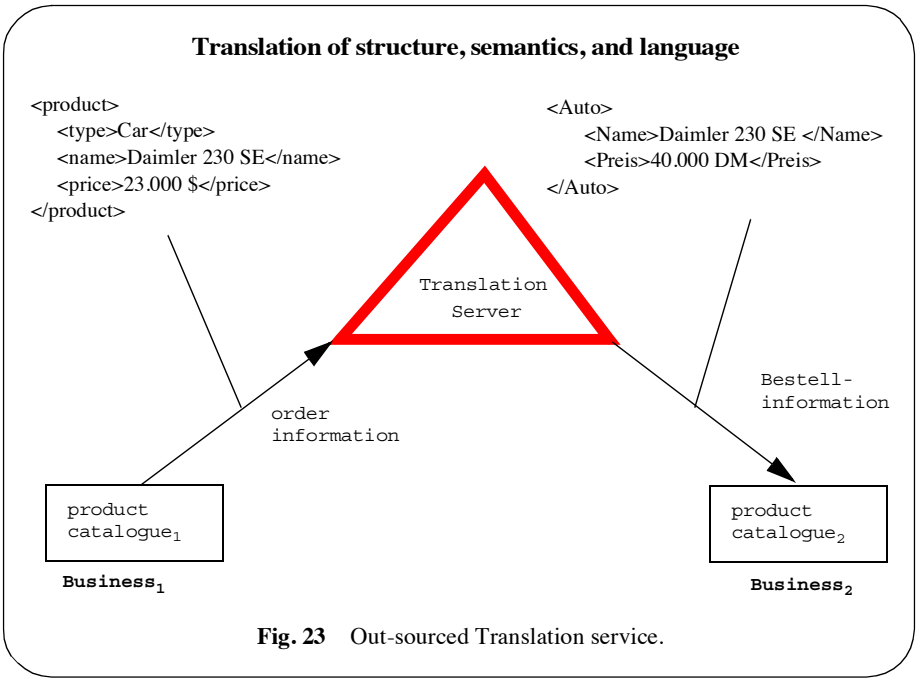


Fig. 23 Out-sourced Translation service.

least compete with traditional mediation agents, like wholesale traders.

Where are the bottlenecks that must be passed in order to set up such marketplaces? Currently, electronic commerce is seriously hampered by the lack of standards:

- *Lack of a means for representation and translation:* The Internet standard HTML not provides syntax and semantics for information, and existing standards for EC, like EDIFACT, are isolated, cumbersome, and costly.
- *Lack of a means for content descriptions (ontologies):* There are no standard product descriptions (catalogues) in the various subsegments.

In consequence, there is a clear need and a large commercial potential for new standards for data exchange and domain modeling.

4.2.1 Means for representation and translation[29]

The Resource Description Framework (RDF) ([Miller, 1998], [Lassila & Swick,1999]) provides a standard for describing the semantics of information (via meta-data descriptions). The Extendible Markup Language (XML) provides a standard for describing structure of information (and some aspects of its semantics). XML schemes will provide a standard for describing the semantics of data. The Extendible Stylesheet Language (XSL) provides a standard for describing mappings between different terminologies. Very likely, XML/EDI (cf. [Bryan, 1998]) will replace cumbersome and isolated EDIFACT. However, none of these standards provide a standardized

29. See Section 5 and 6 for more details on the representation formalisms.

vocabulary, i.e., different business agents may use different document type definitions (DTDs) even if both agree on the use of XML. Therefore, there is still a need for translation service.

Business Model out-sourced translation service: A company can serve as an out-sourced translation service (see Figure 23) that enables communication between different business units using different terminologies.

4.2.2 Means for content descriptions (Ontologies)

Ontologies provide a shared and common understanding of a domain that can be communicated between people and application systems. Providing shared and common domain structures becomes essential, and their providers own a key asset in information exchange (comparable to portals in the B2C area). In the B2B sector, ontologies correspond to standardized product catalogues. There have been various attempts to achieve standardized content descriptions (cf. [de Carvalho Moura et al., 1998], [Li, to appear]): *Common Business Library (CBL)*[30] of Commerce Net, *Commerce XML (cXML)*[31], *Dublin core*[32], <indecs>[33], *ICE*[34], *IOTP*[35], *OBI*[36], *OFX*[37], *Ontology Org*[38], *RossettaNet*[39]. Other approaches like *Chemdex*[40], a web-based marketplace for life science products, may lead from the bottom-up to a joined ontology for an industrial area where it can attract large transaction volume. All these standardization efforts would enable direct communication between different agents.

Business Model: "Standard terminologies". Building a B2B market place for an industrial segment with a rich content model (see Figure 24). A standard ontology can be used by various agent to buy and sell products.

These branch portals may use *mysap.com* and *harbinger.net* types of infrastructure as their underlying technology enriching it with *rich and area-specific domain models*. In the following, we will discuss two examples of this type of branch portals: *Chemdex* and *PaperExchange*; and an approach for customizing branch portals called *VerticalNet.com*.[41]

4.2.3 Chemdex (www.chemdex.com)

Chemdex enables life science enterprises, researchers, and suppliers to effectively buy

---

30. http://www.commerce.net
31. http://www.oasis-open.org
32. [Weibel et al., 1995]. See http://purl.oclc.org/dc/.
33. http://www.indecs.org.
34. http://www.w3.org/TR/NOTE-ice.
35. http://www.ietf.org/html.charters/trade-center.html
36. http://openbuy.org.
37. http://www.ofx.net/.
38. http://www.ontology.org/
39. http://www.rosettanet.org/ (see also http://www.extricity.com/).
40. http://www.chemedex.com
41. Other examples for such portals are http://www.commerceone.com, http://www.mondus.com, and http://www.inference.com/.

and sell research products through a marketplace, a secure, Internet-based procurement solution. The worldwide market for life science research supplies is more than $10 billion annually, incl. $4 billion for reagents. Chemdex provides a database of 240,000 research products, the average order size is between $200 and $400. Over 100 suppliers are included. Chemdex acts as a principal in purchasing products from its suppliers and reselling them to its own customers. CHEMDEX receives a percentage fee on product sales. However, few customers have adopted the CHEMDEX procurement solution up to now. Genentech is the most important customer, accounting for 82% of revenues in the last quarter.

4.2.4 PaperExchange (www.paperexchange.com)

PaperExchange enables buyers and suppliers of the paper industry to trade in an on-line spot market (classifieds). It started offering products on its marketplace in 1998/99. Since the start, around 2000 items have been offered on the market. It is owned by a publisher. The average order size is between $5,000 and $50,000. PaperExchange is only a spot market, i.e. no product catalogues are integrated. It is designed to serve the international market, but currently limited to the US market. The registration is mandatory and PaperExchange allows members to post anonymously until a transaction is agreed upon. Until “clearing” is implemented, members agree to handle all payments directly with each other without intervention by PaperExchange. The seller agrees to pay PaperExchange a fee equal to 3% of the total notional value of any completed transaction.

4.2.5 VerticalNet (www.verticalnet.com)

VerticalNet[42] set up web sides (more than 30 meanwhile) as B2B solutions for specific industrial segments. Each web side forms a community of vendors and buyers in a

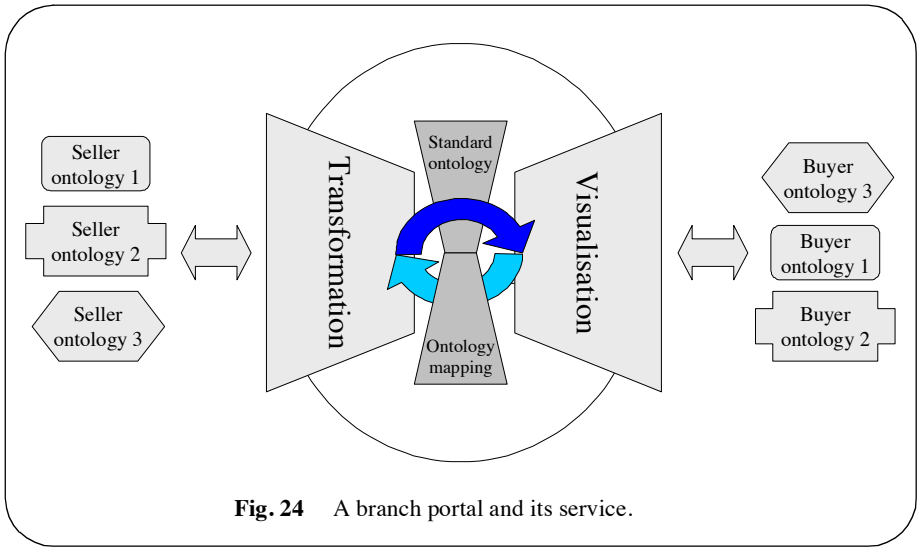


Fig. 24 A branch portal and its service.

42. http://www.VerticalNet.com

specific area (see Figure 25). Vertical trade communities are introduced in segments with a substantial number of buyers and suppliers, a high degree of fragmentation on both the supply and demand sides, and significant on-line access.

4.2.6 Conclusions

In a nutshell, such portals for industrial segments must provide a rich domain model (product catalogue) of the domain, the integration of various other non-standard product catalogues and a translation service that makes these differences transparent to the users, as well as an integration with infrastructure that processes the business transactions. The business model for such portals only makes up a small share of alltransactions (much lower than what is usually wanted by the wholesalers). In consequence such portals will be able to mediate large shares of the entire turnover of an industrial segment. Advantages of such portals are (cf. [Wildemann]): worldwide access to the market, reduction of transaction costs, shorter selling and buying processes, and transparent decisions and transparent market surveys.

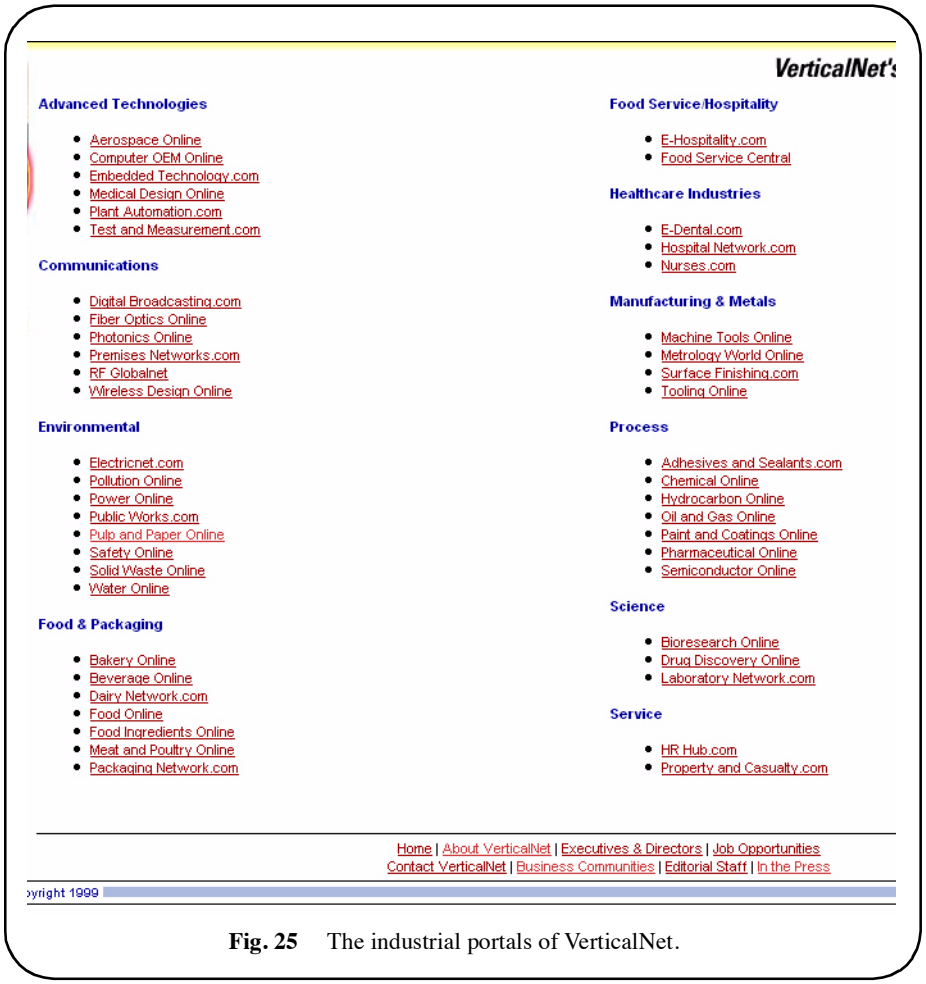


Fig. 25 The industrial portals of VerticalNet.

5 The basic technology: XML, XSL, and XML-QL

The last two sections discussed promising application areas of ontologies: knowledge management and electronic commerce in the B2C and B2B areas. This section is devoted to the technological infrastructure that will enable ontologies to be put into practise. I already mentioned the fact that computers currently are shifting from a single isolated devices to entry points into a worldwide network of information exchange and business transactions. Therefore, support in data, information, and knowledge exchange is becoming the key issue in current computer technology. In consequence, there are serious efforts in the direction of a new standard for defining and exchanging data structures. The *eXtendible Markup Language XML* is a new web standard that provides such facilities. In this section we will therefore investigate XML in detail before we show in Section 6 how it relates to the use of ontologies for information exchange.

The content of this section is organized as follows. First I will explain the need for XML in Section 5.1. In Section 5.2 I will explain its core concepts. I will explain the use of DTDs for defining constraints over valid XML documents in Section 5.3 and its linking concept in Section 5.4. XML is a language for defining static information sources. Languages that add dynamics to it are discussed in Section 5.5. and 5.6., i.e. XSL and query languages for XML such as XQL, XML-QL, or LOREL.[1] They enable the transformation of documents (i.e., view definition) and allow us to query documents. Section 5.7. introduces RDF (cf. [Miller, 1998], [Lassila & Swick,1999]), an application of XML for the purpose of describing metadata, i.e., semantics of information sources.

In the meantime there are large quantities of information material about XML. The official web pages about XML are hosted by the W3C[2], the standardization committee of the World Wide Web:

- XML (Recommendation 1.0)
  general: http://www.w3.org/XML/
  specification: http://www.w3.org/TR/1998/REC-xml-19980210
- XML Linking (Working Drafts)
  XPointer: http://www.w3.org/TR/1998/WD-xptr-19980303
  XLink: http://www.w3.org/TR/1998/WD-xlink-19980303
- XML Stylesheet (Working Drafts)
  general: http://www.w3.org/Style/XSL/
  specification: http://www.w3.org/TR/WD-xsl
- Document Object Model (DOM): http://www.w3.org/DOM/
- Query languages for XML: http://www.w3.org/TandS/QL/QL98/

In addition, there is an excellent FAQ list at http://www.ucc.ie/xml/ and have been numerous books dealing with XML in the meantime (e.g., [Connolly, 1997]). Articles and tutorials on XML can be found at:

- http://metalab.unc.edu/pub/sun-info/standards/xml/why/xmlapps.html
- http://www.inrialpes.fr/inria/seminaires/XML1-10.12.98/sld00000.htm

---

1. [QL, 1998]
2. http://www.w3c.org

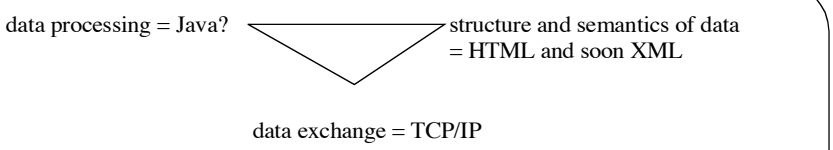


Fig. 26 The three aspects for standardization.

- http://www.inrialpes.fr/inria/seminaires/XML2-10.12.98/sld00000.htm
- http://www.gca.org/conf/paris98/bosak/sld00000.htm
- http://www.heise.de/ix/raven/Web/xml/

Finally, Robin Covers´ side at OASIS is one of the richest information collections which exists on these topics in the web: http://www.oasis-open.org/cover/xml.html.

5.1 Why XML

In general, three aspects of programming can be standardized in order to support information and knowledge exchange between various agents (see Figure 26):

- defining a language for describing the structure and semantics of data,
- defining a language for processing data, and
- defining a protocol for exchanging data.

The success of the Internet is based on its simple TCP/IP protocol that enabled document exchange between various networks using different protocols. Therefore, different protocols used by different nodes become transparent when accessible via an Internet gateway. The Internet standardizes the protocol for data exchange.

The World Wide Web introduced an additional standard for defining the data structure of the documents exchanged. The success of the World Wide Web was brought about by the simple HTML language for describing and rendering documents in the WWW. Browsers that enable their representation were soon developed (Mosaic, Netscape) . The advantages of HTML are:

- HTML is so simple that anybody from the street can write a home page.
- HTML is so simple that is was quickly possible to create browsers that can render home pages.
- The fact that the Web has 100 million users in the meantime would clearly be impossible without this simplicity.

However, HTML is *"too simple"* for the next Web generation with one billion users. The basic principles of HTML is that it is a mark-up language with predefined tags. A HTML document has the general structure of <begin-tag>...<end-tag> where "..." is arbitrary information. The tags are predefined and are translated by browsers into predefined formats. HTML was intended to be a language to describe the *structure* of documents, however, it is mainly used as an *layout language*. An example of a simple home page is given in Figure 27.

```
<html>
  <head>
    <title>A beautiful homepage</title>
  </head>
  <body>
    <h2>This is Fensels' Homepage!</h2>
    <img src="dfe.jpg" width="69" height="95">
    <p>Hello! My Name is <em>Dieter Fensel
    </em> and my email is:<br>
    dfe@aifb.uni-karlsruhe.de and my phone
    number is:<br> 6084751</p>
  </body>
</html>
```

Fig. 27 A simple HTML example.

The weaknesses of HTML ... First, HTML does not provide much semantics for the data, for example, "<h2>..</h2>" has a very vague semantics. It only let us know that *"This is Fensels' Homepage!"* is a level 2 heading. This is a strong limitation for automatic information search and processing ,and HTML is therefore a serious limitation for meaningful information search in the WWW. Second, HTML does not allow us to define structured data and, in consequence, no data exchange is possible with HTML. All data must be expressed in terms of a document structure (i.e., headings, tables, paragraphs etc.). And it is not possible to adapt tags to specific applications, i.e., only simple documents can be represented with HTML. Therefore, HTML is a serious

```
<?XML version="1.0"?>
<homepage>
    <heading>This is Fensels' Homepage!</heading>
    <paragraph>
        Hello! My Name is
        <name>Dieter Fensel</name>
        and my email is:<br/>
        <email>dfe@aifb.uni-karlsruhe.de</email>
        and my phone number is:<br/>
        <phone type="office">6084751</phone>
        <phone type="private">9862155</phone>
    </paragraph>
</homepage>
```

Fig. 28 A simple XML example.

limitation for electronic commerce because it cannot be used for electronic data interchange.

... are the strengths of XML. First, in XML tags can define the semantics of the data, for example,

<Name>*Dieter Fensel*</Name>

Second, XML provides arbitrary trees (graphs) as data structures, for example,

<Person> <Name>*Dieter Fensel*</Name> <Phone>*6084751*</Phone> </Person>

Third, XML allows the definition of application-specific tags and can therefore be customized for the exchange of specific data structures.

5.2 What is XML

XML is a tag-based language for describing tree structures with a linear syntax. It is a successor of SGML, which was developed long ago for describing document structures. However, whereas HTML is too simple to serve our purpose SGML was viewed to be too complex to become a widely used standard. Therefore XML simplifies some aspects of SGML that were not viewed as being essential. An example for a simple XML document is provided in Figure 28.

XML provides seven different means for presenting information:

1 Elements.
Elements are the typical element of a markup: <*tag*> contents </*tag*>

2 Attributes
Attributes are name-value pairs defined within a tag
<tag *attribute-name*="*attribute-value*"> ... </tag>

3 References

References can be used to write symbols in a text that would otherwise be interpreted as commands, for example, "<" can be written as *<* to use it as text in XML. References can also be used to define macros. Often-used texts or links can be defined as macros and have to be written and maintained only at one place. Entity references always start with "&" and end with ";".

4 Comments

Comments begin with <!-- and end with -->. XML processors could ignore comments.

5 Process Instructions

Processing Instructions (PI) are the procedural element in an otherwise declarative approach. Processing Instructions have the form:

<?name pidata?>

A XML processor could ignore Processing Instructions like comments, however, must pass them through to the application. The application executes all Processing Instructions it knows. An example for a PI is:
<?xml:stylesheet type="text/css2" href="style.css" ?>

6 CDATA

CDATA represents arbitrary strings in XML Documents which are not interpreted by an XML parser.

<![CDATA[
XML uses <begin-tag> and <end-tag> to structure documents.
]]>

7 Prolog

The XML declaration: <?XML version="1.0"?> is obligatory. In addition a prolog may contain further elements. An XML document may use a document type declaration either by containing its definition or by pointing to it. Such a document type declaration defines a grammar for XML documents and is called a *Document Type Definition DTD*. An external definition which is pointed to by a reference:

<!DOCTYPE Name SYSTEM "name.dtd">

an intern definition looks like

<!DOCTYPE Name *[*
<!ELEMENT Name (#PCDATA)>
]>

5.3 What are DTDs

In this Section I will explain the usefulness of DTDs and then show how DTDs can be defined. An XML document is *wellformed*, if

- the document starts with an XML-declaration;
- all tags with contents have begin and end tags; tags without contents have an end tag or end with "/>".
- there must be a root (XML documents are trees);

An XML document is *valid*, if it is *wellformed*, and if the document uses an DTD it

```
<?XML version="1.0"?>
    <!DOCTYPE name [
        <!ELEMENT name (title*, first name | initial, middle name?, last name +)>]>
    <!DOCTYPE first name [
        <!ELEMENT first name #PCDATA[1]>]>

    <name>
        <title>Privatdozent</title>
        <title>Dr.</title>
        <first name>Dieter</first name>
        <last name>Fensel</last name>
    </name>
```

1. parseable character data

Fig. 29 A simple element declaration.

respects this DTD. Therefore, DTDs are not necessary for XML documents, however, they provide the possibility to define stronger constraints for documents.

A DTD consists of three elements: *Element* declaration that define composed tags and value ranges for elementary tags, *attribute* declaration that define attributes of tags, and finally *entity* declaration.

An example for element declarations and a valid XML document is given in Figure 29. In this the following hold true:

- ? = zero or one appearance of an element
- * = zero to n-appearances of an element
- + = one to n-appearances of an element
- a | b = a or b appearances of an element

Attribute declarations regulate the following aspects: the elements that may have an attribute; the attributes they have; the values an attribute may have; and the default value of an attribute. Its general form is:

<!ATTLIST element-name
    attribute-name_1 attribute-type_1 default-value_1
    ...
    attribute-name_n attribute-type_n default-value_n
>

There are six attribute types: CDATA = string, ID = Unique key, IDREF and IDREFS = Reference for one or several IDs in the document, ENTITY or ENTITIES = name of one or several entities, NMTOKEN or NMTOKENS = value is one or several words, and a list of names (enumeration type).

Finally, four types of default values can be distinguished:

- #REQUIRED; The attribute must have a value.
- #IMPLIED; The attribute must have a value and no default value is defined.
- “value”; This value is the value of the attribute if nothing else is defined explicitly.
- #FIXED “value”; If the attribute is used it must have this default value.

Entities enable the definition of symbolic values. This may provide shortcuts for long expressions, for example *dfe* for *Privatdozent Dr. Dieter Andreas Fensel*.

<*!ENTITY* dfe “Privatdozent Dr. Dieter Andreas Fensel”>

Even more important, it significantly improves the maintainability of XML documents. Elements that appear in several places within a document need only to be changed once based on their central description.

5.4 Linking in XML

HTML is the shortcut for HyperText Markup Language. Therefore, HTML provides two main aspects. Representation of textual information and hyperlinks between various documents and subsections of them. XML incorporates a similar but generalized linking mechanism. It is not yet fixed, i.e. there is still no defined standard. However there are drafts that show us what the standard will look like.[3] In general, three kinds of links will be provided: simple links, extended links, and extended pointers.

Simple links resemble HTML links, for example,

<*LINK XML-LINK=”SIMPLE” HREF=”*locator”>text</*LINK*>

However, the locator may be an URL (as in HTML), a query (see XML-QL), or an extended pointer (see below).

Extended links can express relations with more than two addressees:

<*ELINK XML-LINK=”EXTENDED”* ROLE=”ANNOTATIONS”>
<*LOCATOR XML-LINK=LOCATOR”* HREF=”text.loc>text</*LOCATOR*>
<*LOCATOR XML-LINK=LOCATOR”* HREF=”Annot$_1$.loc>Ann$_1$</*LOCATOR*>
<*LOCATOR XML-LINK=LOCATOR”* HREF=”Annot$_2$.loc>Ann$_2$</*LOCATOR*>
</*ELINK*>

Extended Pointers (XPointer). In HTML an URL can point to a specific part of a document, for example, “http://www.a.b/c#name” where “*name*” is the name of an anchor tag. In XML you can jump to an arbitrary location within a document, for example, in a list of employees you can jump to the row of employees with the name *Miller* and in a list of employees you can jump to the 10th row or to the row of employee with the ID “007”.Therefore, XPointers are similar to general queries.

5.5 Extensible Style Language (XSL)

A browser can render a HTML document because it knows all HTML tags. Therefore it can use predefined style information. However, in XML tags can be defined by the information provider. How does a browser render XML documents? It requires additional style sheet information for this purpose. *Cascading Stylesheets (CSSs)* define how a browser should render XML documents. It has already been developed for HTML to allow more flexibility in layout, helping to bringing HTML back to its

---

3. see http://www.w3.org/TR/1998/WD-xptr-19980303 for XPointer and http://www.w3.org/TR/1998/WD-xlink-19980303 for XLink.

original purpose of being a language for describing the structure of documents instead of their layout. A more expressive possibility in the case of XML is XSL. It is the coming standard for expressing format information for XML documents, however, it can do much more than this. CSS defines for each element of a document how it should be rendered. XSL allows us to define views that manipulate the structure and elements of an document before they are rendered. Therefore, XSL even enables the translation of one XML document into another using a different DTD. This is important in cases where different users may wish to have different *views* of the information captured in a XML document. In electronic commerce applications, it enables different product presentations for different clients and different product presentations for different user groups (for example, clients who build and maintain product catalogues versus users who access these catalogues). Therefore, XSL has more expressive power than CSS. It is comparable to the expressiveness of DSSSL which is used for presenting SGML documents. However, in contrast to the Lisp syntax of DSSSL, XSL has a XML Syntax.

At the moment, XSL can be used to translate XML documents into HTML by the server. HTML is just another XML dialect and this translation by the server is required because most browsers currently do not support XML and XSL for rendering. In general, the dynamic manipulation of XML documents can be used to create different sides from the same data sources, and to realize dynamically changing sides according to user preferences or contexts. XML is a standard language for defining tagged languages. However, XML does not provide standard DTDs, i.e., each user can/may/must define his own DTD. For exchanging data between different users relying on different DTDs, you have to map different DTDs onto each other. You can use XSL to translate XML documents using DTD_1 into XML documents using DTD_2 providing the translation service required for electronic commerce mentioned earlier. *Exactly here is the importance of XSL* in our context.

How does XSL achieve this? XSL is a language for expressing stylesheets. Each stylesheet describes rules for presenting a class of XML source documents. There are two parts to the presentation process: First, the result tree is constructed from the source tree. Second, the result tree is interpreted to produce formatted output on a display, on paper, in speech or on other media.

The first part is achieved by *associating patterns with templates*.

- A *pattern* is matched against elements in the source tree.
- A *template* is instantiated to create part of the result tree.
- The result tree is separate from the source tree.

In consequence, the structure of the result tree can be completely different from the structure of the source tree. In constructing the result tree, the source tree can be filtered and reordered, and arbitrary structure can be added.

The second part, formatting, is achieved by using the formatting vocabulary specified in this document to construct the result tree.

- Formally, this vocabulary is an XML name space. Each element type in the vocabulary corresponds to a formatting object class.
- A formatting object class represents a particular kind of formatting behavior.

The following is an example of a simple XSL stylesheet that constructs a result tree for a sequence of paraelements. The result-ns="fo" attribute indicates that a tree using the formatting object vocabulary is being constructed. The rule for the root node specifies the use of a page sequence formatted with any font with serifs. The paraelements become block formatting objects which are set in 10 point type with a 12 point space before each block.

```
<xsl:stylesheet>
 xmlns:xsl="http://www.w3.org/TR/WD-xsl"
 xmlns:fo="http://www.w3.org/TR/WD-xsl/FO"
 result-ns="fo">
 <xsl:template match="/">
  <fo:basic-page-sequence font-family="serif">
    <xsl:apply-templates/>
  </fo:basic-page-sequence>
 </xsl:template>

 <xsl:template match="para">
  <fo:block font-size="10pt" space-before="12pt">
    <xsl:apply-templates/>
  </fo:block>
 </xsl:template>
</xsl:stylesheet>
```

Fig. 30 A simple XSL document.

- Each attribute in the vocabulary corresponds to a formatting property.
- A formatting object can have content, and its formatting behavior is applied to its content.

An example for an XSL file is provided in Figure 30.

5.6 Query Languages for XML

The need for a query language for XML becomes obvious when comparing the WWW with a database. With a database you could ask goal-oriented direct queries. In the Web, only vague search queries can be formulated (asking for sides that contain some key terms) and the human reader must extract and integrate the information from the sides. This makes information extraction expensive and automatic information extraction nearly impossible. Query languages for XML will provide query answering service like it is provided by databases. Currently, there exists still a number of proposals (XQL, XML-QL, LOREL). Most of them can be found in [QL, 1998]. The availability of large amounts of data in the Web raises several issues that the XML standard does not address. In particular, what techniques and tools should exist:

- for extracting data from large XML documents,

- for translating XML data between different ontologies (DTD's),
- for integrating XML data from multiple XML sources,
- and for transporting large amounts of XML data to clients or for sending queries to XML sources.

Therefore, much effort is currently being spent in developing a query language for XML that allows SQL-like queries over XML documents. Interestingly, there seems to be close connections between a query language and XSL (cf. [Schach et al., 1998]). Actually, the current redesign of XSL clearly separates document layout aspects from document transformation aspects. The sublanguage for the latter aspect, called XSL-T [Clark, 1999], is in fact nothing more than a query language for XML.

5.7 The Resource Description Framework RDF

XML provides semantic information as a by-product of defining the structure of the document. It prescribes a tree structure for documents and the different leaves of the tree have well-defined tags and contexts with which the information can be understood. That is, the structure and semantics of document are interwoven. The *Resource Description Framework (RDF)*[4] (cf. [Miller, 1998], [Lassila & Swick,1999]) provides a means for adding semantics to a document without making any assumptions about its structure. RDF is an infrastructure that enables the encoding, exchange and reuse of structured metadata. Search engines, intelligent agents, information broker, browsers and human users can make use of *semantic* information. RDF 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 description of Web sources and will be used by other standards such as PICS-2, P3P, and DigSig (see Figure 31).

The data model of RDF provides three object types: *subjects*, *predicates*, and *objects*

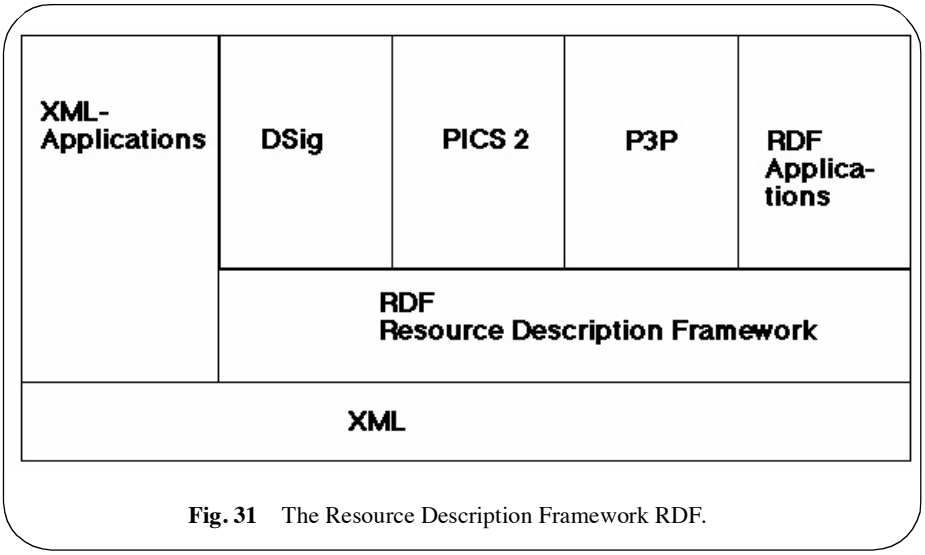


Fig. 31 The Resource Description Framework RDF.

4. http://www.w3c.org/Metadata/

(cf. the schema definition of RDF [Brickley et al., 1998]).

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

A simple example is[5]

Author(http://www.cs.vu.nl/ frankh) = Frank

This states that the author of the named Web document is Frank. Values can also be structured entities:

Author(http://www.cs.vu.nl/ frankh) = *X*
Name(*X*) = Frank
Email(*X*) = frankh@cs.vu.nl

where *X* denotes an actual (i.e., the homepage of Frank) or a virtual URI. In addition, RDF provides bags, sequence, 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(Dieter)=(Author(http://www.cs.vu.nl/ frankh) = Frank)

states that *Dieter* claims that *Frank* is the author of the named resource.

RDF schemes (RDFS) [Brickley et al., 1998] provide a basic type schema for RDF based on core classes, core property types and core constraints.

Three core classes are provided by the RDF Schema machinery

- *Resource* (i.e., the class of all subjects),
- *Property Type* (i.e., the class of all predicates), and
- *Class* (i.e., the class of all values of predicates).

Core property types of RDFS are:

- *instanceOf* and *subClassOf*: *instanceOf* defines a relationship between a *resource* and an element of *Class,* and *subClassOf* defines a relationship between two elements of *Class*. *subClassOf* is assumed to be transitive.
- *Constraint* is a subclass of *PropertyType*. It has the two core instances *range* and *domain* applicable to property types having a class as value. *Range* and *domain* define the range and domain of property types respectively.

5.8 Conclusions

XML, XSL, and RDF are complementary technological means that will enable ontological support in knowledge management and electronic commerce. XML provides a standard serial syntax for exchanging data. In consequence, ontology-based

---

5. I will skip the awkward syntax of RDF, because simple tooling can easily present it in a more common format such as shown here.

data and information exchange can abstract from these aspects. A DTD allows us to define the structure and elementary tags of an XML document. We will see in the next chapter how such a DTD can be generated from an Ontology and vice versa how an Ontology can be derived from a DTD. XSL allows us to translate between different XML documents, i.e., documents relying on different DTDs. Finally, the Resource Description Framework RDF provides a standard for describing machine-processable semantics of data. Therefore, they can immediately be used for representing ontologies (i.e., as a syntax for ontology specification). The relationships between new and arising Web standards on the one hand and ontology languages on the other hand will be discussed in the next chapter.

6 Ontology Languages

The previous chapter described recent web standards that enable the definition of the structure and semantics of data and information in the Web. This chapter will discuss languages for describing ontologies. Ontologies are formal theories about a certain domain of discourse and therefore require a formal logical language to express them. In the second halfe of this chapter I will relate both types of languages, i.e. I will investigate how recent web standards such as XML and RDF can be used as languages that express Ontologies or at least some of their aspects. Finally, I will discuss XOL, a proposal for an XML-based standard for expressing ontologies.

6.1 Ontology Languages

I will discuss some Ontology languages that are well know in the community and that are prototypical for a specific language paradigm. These are:

- CycL and KIF [Genesereth, 1991] as representatives for enriched first-order predicate logic languages.
- Ontolingua [Farquhar et al., 1997] and Frame Logic [Kifer et al., 1995] as representatives for frame-based approaches. Both incorporate frame-based modeling primitives in a first-order logical framework, however they apply very different strategies for this.
- Description Logics that describe knowledge in terms of concepts and role restrictions used to automatically derive classification taxonomies.

6.1.1 First-order predicate logic languages CycL and KIF

CycL was developed in the Cyc project [Lenat & Guha, 1990] for the purpose of specifying the large common-sense ontology that should provide Artificial Intelligence to computers. Far from having attained this goal, Cyc still provides the worldwide largest formalized ontology. CycL is a formal language whose syntax is derived from first-order predicate calculus. However, CycL extends first-order logic through the use of *second* order concepts. Predicates are also treated as constants in expressions. The vocabulary of CycL consists of terms: semantic constants, non-atomic terms, variables, numbers, strings, etc. Terms are combined in CycL expressions, ultimately forming closed CycL sentences (with no free variables). A set of CycL sentences forms a knowledge base. In the following, we will discuss the main concepts of CycL. More details can be found at its homepage[1].

Constants are the vocabulary of the CycL language; more precisely, they are the "words" that are used in writing the axioms (i.e., the closed formulas) that comprise the content of any CycL knowledge base. Constants may denote (1) individuals, (2) collections of other concepts (i.e., sets which correspond to unary predicates), (3) arbitrary predicates that enable the expression of relationships among other constants, and functions. Constants must have unique names.

Predicates express relationships between terms. The *type* of each argument of each predicate must be specified; that is the appropriate formulas must be asserted to be true,

---

1. http://www.cyc.com/cycl.html/

i.e. p (A_0) with A_0 of type T and $p(c)$ implies that c is element of T.

Variables stand for terms (e.g., constants) or formulas whose identities are not specified. A variable may appear anywhere that a term or formula can appear.

Formulas combine terms into meaningful expressions. Each formula has the structure of a parenthesized list. That is, it starts with a left parenthesis, then there follow a series of objects which are commonly designated A_0, A_1, A_2, etc., and at the end there is a corresponding right parenthesis. The object in the A_0 position may be a predicate, a logical connective, or a quantifier. The remaining arguments may be terms (e.g., constants, non-atomic terms, variables, numbers, strings delimited by double quotes ("...")), or other formulas. Note the recursion (i.e., the second-order syntax) here; A_i in one formula might itself be an entire CycL formula. Each atomic formula must begin with a predicate or a variable in order to be well-formed. The simplest kind of formula is a formula in which the A_0 position is occupied by a predicate and all the other argument positions are filled with terms (or variables):

(likesAsFriend DougLenat KeithGoolsbey)
(colorOfObject ?CAR ?COLOR)

The first formula above is called a ground atomic formula, since none of the terms filling the argument positions are variables. The second formula is not a ground atomic formula; it refers to the variables ?CAR and ?COLOR.

Logical connectives are used to build more complex formulas from atomic formulas (and/or other complex formulas). The most important logical connectives are: *and*, *or*, and *not*. New connectives can be introduced simply by inserting a formula to that effect into the knowledge base; thus

(*isa new-connector Connective*).

Complex Formulas. We can compose the above connectives, of course, and have complex expressions such as

(*and*... (*or* ... (*xor A* (*and* ...))...)...)

Quantification: universal quantification and existential quantification. Universal quantification corresponds to expressions like *every, all, always, everyone,* and *anything*, while existential quantification corresponds to expressions like someone, something, and somewhere. CycL contains **one universal quantifier**, *forAll*, and **four existential quantifiers**, *thereExists*, *thereExistAtLeast*, *thereExistAtMost*, and *thereExistExactly*. Additional quantifiers can be introduced by making the appropriate assertions -- declaring the new quantifier to be an instance of Quantifier, and giving a definition of it, probably in terms of existing quantifiers, predicates, and collections. To be considered a closed sentence -- a well-formed formula -- all the variables in an expression need to be *bound* by a quantifier before they are used.

Second-order Quantification: Quantification is also allowed over predicates, functions, arguments, and formulas.

Functions. Like most predicates, most functions have a fixed arity. For each function assertions that specify the type of each argument must be entered into the CycL knowledge base.

Microtheories ([Lenat & Guha, 1990], [Guha, 1993]). A microtheory, or context ([McCarthy 1993], [Lenat, submitted]), is a set of formulas in the knowledge base. Each

formula must be asserted in at least one microtheory. Microtheories are fully reified objects, and thus they cannot only contain CycL formulas, they can also participate in CycL formulas.

Each formula has an associated **truth value** (in each microtheory). CycL contains five possible non-numeric truth values, of which the most common are default true and monotonically true. The other truth values are default false, monotonically false, and unknown. In addition, CycL accommodates Bayesian probabilities and dependencies, and (separately) fuzzy truth values, attributes, and sets. All CycL-compliant system must support at least one "true" and one "false".

- **Monotonically true** means: true with no exceptions. Assertions which are monotonically true are held to be true in every case, that is, for every possible set of bindings -- not just currently known bindings -- to the universally quantified variables (if any) in the assertion and cannot be overridden.
- Assertions which are **default true**, in contrast to monotonically true, can have exceptions. They are held to be true only in most cases (usually meaning most of the relevant cases likely to be encountered in the current context) and can be overridden without needing to alert the user

In a nutshell, CycL uses predicate logic extended by *typing,* i.e. functions and predicates are typed, *reification,* i.e. predicates and formulas are treated as terms and can be used as expressions within other formulas, and microtheories that define a *context* for the truth of formulas.

The *Knowledge Interchange Format KIF* [Genesereth & Fikes, 1992] is a language designed for use in the exchange of knowledge between disparate computer systems (created by different programmers at different times, in different languages, etc.). Different computer systems can interact with their users in whatever forms are most appropriate to their applications (for example Prolog, conceptual graphs, natural language, etc.). Being a language for knowledge interchange, KIF can also be used as a language for expressing and exchanging Ontologies.[2] The following categorical features are essential to the design of KIF.

- The language has *declarative semantics*.
- The language is *logically comprehensive* -- at its most general it provides for the expression of arbitrary logical sentences. In this way, it differs from relational database languages (like SQL) and logic programming languages (like Prolog).
- The language provides a *means for the representation of knowledge about knowledge*. This allows the user to make knowledge representation decisions explicit and to introduce new knowledge representation constructs without changing the language.

Semantically, there are four categories of **constants** in KIF -- object constants, function constants, relation constants, and logical constants. Object constants are used to denote individual objects. Function constants denote functions on those objects. Relation constants denote relations. Logical constants express conditions about the world and are

---

2. Actually, KIF was not presented in this way because its origins are older than the current O-hip.

either true or false. KIF is unusual among logical languages in that there is no syntactic distinction between these four types of constants; any constant can be used where any other constant can be used. The differences between these categories of constants are entirely semantic. This feature *reifies* second-order features in KIF. It is possible to make statements about statements.

There are three disjoint types of **expressions** in the language: *terms*, *sentences*, and *definitions*. Terms are used to denote objects in the world being described, sentences are used to express facts about the world, and definitions are used to define constants. A *knowledge base* is a finite set of sentences and definitions.

There are six types of **sentences**.

sentence ::= constant | equation | inequality | relsent | logsent | quantsent

We have already mentioned constants. An equation consists of the = operator and two terms. An inequality consist of the /= operator and two terms. An implicit relational sentence consists of a constant and an arbitrary number of argument terms terminated by an optional sequence variable.

The syntax of **logical sentences** depends on the logical operator involved. A sentence involving the *not* operator is called a negation. A sentence involving the *and* operator is called a conjunction and the arguments are called conjuncts. A sentence involving the *or* operator is called a disjunction and the arguments are called disjuncts. A sentence involving the => operator is called an implication; all of its arguments but the last are called antecedents and the last argument is called the consequent. A sentence involving the <= operator is called a reverse implication; its first argument is called the consequent and the remaining arguments are called the antecedents. A sentence involving the <=> operator is called an equivalence.

There are two types of **quantified sentences** -- a universally quantified sentence is signaled by the use of the forall operator, and an existentially quantified sentence is signaled by the use of the exists operator. The first argument in each case is a list of variable specifications. Note that according to these rules it is permissible to write sentences with *free* variables[3], i.e. variables that do not occur within the scope of any enclosing quantifiers.

Finally, there are three types of **definitions** -- unrestricted, complete, and partial. Within each type there are four cases, one for each category of constant. For more details see the KIF homepage.[4]

KIF and CycL have features in common. Both languages are oriented on predicate logics. Also, both provide an important extension of first-order logic. They allow the reification of formulas as terms used in other formulas. Therefore, KIF and CycL allow meta-level statements. In addition to this, CycL provides richer modeling primitives than KIF (e.g., various quantifiers and microtheories). This stems from the fact that CycL is a modeling language for ontologies whereas KIF was designed as an exchange format for ontologies. As I will discuss later, both languages are close in spirit to RDF. Second-order elements (i.e., formulas as terms in meta-level formulas) and global scope

---

3. Very different from CycL where free variables are forbidden.
4. http://logic.stanford.edu/kif/kif.html

of properties (i.e., predicates) are common features.

6.1.2 Frame-based Approaches: Ontolingua and Frame Logic

The central modeling primitive of predicate logic are predicates. *Frame-based* and *object-oriented* approaches take a different point of view. Their central modeling primitive are classes (i.e., frames) with certain properties called attributes. These attributes do not have a global scope but are only applicable to the classes which they are defined for (they are typed), and the "same" attribute (i.e., the same attribute name) may be associated with different range and value restrictions when defined for different classes. In the following I will discuss two frame-oriented approaches: Ontolingua (cf. [Gruber, 1993], [Farquhar et al., 1997]) and Frame logic [Kifer et al., 1995].

Ontolingua[5] was designed to support the design and specification of ontologies with a clear logical semantics based on KIF. Ontolingua extend KIF using additional syntax to include the intuitive bundling of axioms into definitional forms with ontological significance and a Frame Ontology to define object-oriented and frame-language terms.[6] The set of KIF expressions that Ontolingua allows is defined in an ontology called the Frame Ontology. The Frame Ontology specifies the representation primitives that are often supported by special-purpose syntax and code in object-centered representation systems (e.g., classes, instances, slot constraints, etc.). Ontolingua definitions are Lisp-style forms that associate a symbol with an argument list, a documentation string, and a set of KIF sentences labeled by keywords. An Ontolingua ontology is made up of definitions of classes, relations, functions, distinguished objects, and axioms that relate these terms.

A **relation** is defined with a form like the following:

(define-relation *name* (?A_1 ?A_2)
:def (KIF formula)

The arguments ?A_1 and ?A_2 are universally quantified variables ranging over the items in the tuples of the relation. This example is a binary relation, so each tuple in the relation has two items. Relations of greater arity can also be defined. The sentence after the *:def* keyword is a KIF sentence stating logical constraints over the arguments. Constraints on the value of the first argument of a binary relation are domain restrictions, and those on the second argument of a binary relation are range restrictions. There may also be complex expressions stating relationships among the arguments of relation. The *:def* constraints are necessary conditions, which must hold if the relation holds over some arguments. It is also possible to state sufficient conditions or any combination.

A **class** is defined by a similar form with exactly one argument called the instance variable. In Ontolingua, classes are treated as unary relations to help unify object- and relation-centered representation styles.

A **function** is defined like a relation. A slight variation in syntax moves the final argument outside of the argument list. As in definitions of relations, the arguments to a

---

5. http://ontolingua.stanford.edu/

6. The Ontolingua Server as described in [Farquhar et al., 1997] has extended the original language by providing explicit support for building ontological modules that can be assembled, extended, and refined in a new ontology.

```
class relation (?relation)
class function (?function)
class class (?class)
relation instance-of (?individual ?class)
function all-instances (?class) :->
    ?set-of-instances
function one-of (@instances) :-> ?class
relation subclass-of (?child-class ?parent-class)
relation superclass-of (?parent-class ?child-class)
relation subrelation-of
    (?child-relation ?parent-relation)
relation direct-instance-of (?individual ?class)
relation direct-subclass-of
    (?child-class ?parent-class)
function arity (?relation) :-> ?n
function exact-domain (?relation) :->
    ?domain-relation
function exact-range (?relation) :-> ?class
relation total-on (?relation ?domain-relation)
relation onto (?relation ?range-class)
class n-ary-relation (?relation)
class unary-relation (?relation)
class binary-relation (?relation)
class unary-function (?function)
relation single-valued (?binary-relation)
function inverse (?binary-relation) :-> ?relation
function projection (?relation ?column) :-> ?class
function composition
    (?relation-1 ?relation-2) :-> ?binary-relation
relation composition-of
    (?binary-relation ?list-of-relations)
function compose
    @binary-relations) :-> ?binary-relation
relation alias (?relation-1 ?relation-2)
relation domain (?relation ?class)
relation domain-of
    (?domain-class ?binary-relation)
relation range (?relation ?class)
relation range-of (?class ?relation)
relation nth-domain
    (?relation ?integer ?domain-class)
relation has-value
    (?domain-instance ?binary-relation ?value)
function all-values (?domain-instance
    ?binary-relation) :-> ?set-of-values
relation value-type
    (?domain-instance ?binary-relation ?class)
function value-cardinality
    (?domain-instance ?binary-relation) :-> ?n
relation same-values
    (?domain-instance ?relation-1 ?relation-2)
relation inherited-slot-value
    (?domain-class ?binary-relation ?value)
function all-inherited-slot-values
    (?domain-class ?binary-relation) :-> ?set-of-
values
relation slot-value-type (?domain-class
    ?binary-relation ?range-class)
function slot-cardinality
    (?domain-class ?binary-relation) :-> ?n
relation minimum-slot-cardinality
    (?domain-class ?binary-relation ?n)
relation maximum-slot-cardinality
    (?domain-class ?binary-relation ?n)
relation single-valued-slot
    (?domain-class ?binary-relation)
relation same-slot-values
    (?domain-class ?relation-1 ?relation-2)
class class-partition (?set-of-classes)
relation subclass-partition (?c ?class-partition)
relation exhaustive-subclass-partition
    (?c ?class-partition)
relation asymmetric-relation (?binary-relation)
relation antisymmetric-relation
    (?binary-relation)
relation antireflexive-relation (?binary-relation)
relation irreflexive-relation (?binary-relation)
relation reflexive-relation (?binary-relation)
relation symmetric-relation (?binary-relation)
relation transitive-relation (?binary-relation)
relation weak-transitive-relation
    (?binary-relation)
relation one-to-one-relation (?binary-relation)
relation many-to-one-relation (?binary-relation)
relation one-to-many-relation (?binary-relation)
relation many-to-many-relation
    (?binary-relation)
relation equivalence-relation (?binary-relation)
relation partial-order-relation (?binary-relation)
relation total-order-relation (?binary-relation)
relation documentation (?object ?string)
```

Fig. 32 The Frame Ontology of Ontolingua (see [Gruber, 1993]).

function are constrained with necessary conditions following the *:def* keyword.

Finally, it is possible to define **individuals** in an ontology.

The frame ontology is expressed as second-order axioms in Ontolingua. It contains a complete axiomatization of classes and instances, slots and slot constraints, class and relation specialization, relation inverses, relation composition, and class partitions. Each second-order term is defined with KIF axioms. A list of the Frame Ontology vocabulary is given in Figure 32.

Frame logic [Kifer et al., 1995] was already sketched out in chapter 3. It is a language for specifying object-oriented databases, Frame systems, and logical programs. Its main achievement is to integrate conceptual modeling constructs (classes, attributes, domain and range restrictions, inheritance, axioms) into a coherent logical framework. 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 the relationships between elements of an ontology and its instances.

The alphabet of an F-logic language consists of a set of function symbols and a set of variables. A term is a normal first-order term composed of function symbols and variables, as in predicate calculus. A language in F-logic consists of a set of formulas constructed of the alphabet symbols. As in many other logics, formulas are built from simpler formulas by using the usual connectives *not*, *and*, and, *or* and the quantifiers *forall* and *exists*. The simplest kind of formulas are called molecular F-formulas. A molecule in F-logic is one of the following statements:

- Assertion of the form $C :: D$ or of the form $O : C$, where C, D, and O are id-terms. The first expression models subclass relationship and the second statement models is-element-of relationship.
- An object molecule of the form O[*a* ';'-separated list of method expressions]. A method expression can be either a data expression or a signature expression. O is a term denoting an object (which may refer to an instance or a class). "*a*" further specifies properties of this object.
  - Data expressions take one of the following two forms:
    A *scalar* expression *ScalarMethod* $@Q_1,...,Q_k \rightarrow T$
    A *set-valued* expression *SetMethod* $@ R_1,...,R_l \rightarrow\!\!> \{S_1,...,S_m\}$
    Data expressions specify that the method m applied to the object O and the parameter $Q_1,...,Q_k$ deliver the value T. They can be either single-valued or may return a set.
  - Signature expressions also take two forms:
    A *scalar* signature expression *ScalarMethod* $@V_1,...,V_n \Rightarrow (A_1,...,A_r)$
    A *set-valued* signature expression *SetMethod* $@ W_1,...,W_s \Rightarrow\!\!> (B_1,...,B_t)$
    Signature expressions define types for applying methods (i.e., attributes) to object. A method m applied to the object O and the parameter $V_1,..,V_m$ must deliver a value that is element/subclass of $A_1,...,A_r$.

F-formulae are built of simpler F-formulae in the usual manner by means of logical connectives and quantifiers.

Ontolingua and Frame logic integrate frames (i.e., classes) into a logical framework. The main difference between Ontolingua and Frame logic is the manner in which they realize frame-based modeling primitives in a logical language. Ontolingua characterizes the frame-based modeling primitives via axioms in the language. Frame logic defines their semantics externally via an explicit definition of their semantics. Simplified: Ontolingua applies standard semantics of predicate logic and uses axioms in this logic to exclude models that do not fit to the semantics of its modeling primitives. Frame logic provides a more complex semantics compared to predicate logic. The modeling primitives are explicitly defined in the semantics of Frame logic. A second difference

between Frame logic and Ontolingua arises from the fact that Ontolingua inherits the powerful reification mechanism from KIF which allows the use of formulas as terms of (meta-level) formulas. In Frame logic, predicate names can be bound to variables but not entire formulas.

6.1.3 Description Logics

The main effort of research in knowledge representation is directed at providing theories and systems for expressing structured knowledge and for accessing and reasoning with it in a principled way. *Description Logics (DL)* (cf. [Brachman & Schmolze, 1985], [Baader et al., 1991]), also known as *terminological logics*, form an important powerful class of logic-based knowledge representation languages.[7] They result from early work in semantic networks and define a formal and operational semantics for them. Description Logics try to find a fragment of first-order logic with high expressive power which has still a decidable and efficient inference procedure (cf. [Nebel, 1996]). Implemented systems are, for example, BACK, CLASSIC, CRACK, FLEX, K-REP, KL-ONE, KRIS, LOOM, and YAK.[8]

A distinguishing feature of Description logics is that classes (usually called concepts) can be defined intensionally in terms of descriptions that specify the properties that objects must satisfy in order to belong to the concept. These descriptions are expressed using a language that allows the construction of composite descriptions, including restrictions on the binary relationships (usually called roles) connecting objects.

Figure 33 provides the syntax definition of the core language of CLASSIC. Its main modeling primitives are concept expressions and individual expressions (cf. [Borgida et al., 1989]). A CLASSIC database is for the most part a repository of information about individual objects. Objects have an intrinsic identity and are related to each other through binary relationships; these are called *roles* (elsewhere known as attributes or properties). Individuals will be grouped into collections indirectly by means of descriptions that apply to all members of a collection | We will call these descriptions concepts or classes. The data definition language allows the definition of concepts either by grouping individuals together extensionally, or grouping individuals implicitly through the use of intensional descriptions in regard to their structure. Complex CLASSIC concepts are formed by composing expressions using a small set of constructors.

The simplest kind of description you can form in CLASSIC is a **primitive concept**. Primitive concepts are simple but not necessarily atomic; each primitive concept except for the topmost concept (which we call THING), is expected to have at least one parent (more general) concept. The simplest kind of primitive is one whose only parent is essentially vacuous, namely THING. For example, the concept of a CAR might be defined in this way:

(PRIMITIVE THING car)[9]

This expression means that whatever it designates is simply a type of THING with some

---

7. Links to most papers, project, and research events in this area can be found at http://dl.kr.org/.
8. http://www.research.att.com/sw/tools/classic/imp-systems.html
9. The example is taken from [Borgida et al., 1989].

<concept-expr> ::=
    THING | CLASSIC-THING | HOST-THING |
    [these three are built-in primitives]
    <concept-name> |
    (AND <concept-expr> +)[1] |
    (ALL <role-expr> <concept-expr>) |
    (AT-LEAST <positive-integer> <role-expr>) |
    (AT-LEAST <positive-integer> <role-expr>) |
    (AT-MOST <non-negative-integer><role-expr>) |
    (SAME-AS (<role-expr> +) (<role-expr>+)) |
    (TEST <fn> <realm>) |
    (ONE-OF <individual-name>+) |
    (PRIMITIVE <concept-expr> <index>) |
    (DISJOINT-PRIMITIVE
        <concept-expr> <partition-index> <index>)
<individual-expr> ::=
    <concept-expr> |
    (FILLS <role-expr> <individual-name>) |
    (CLOSE <role-expr>) |
    (AND <individual-expr>+)
<realm> ::= host | classic
<concept-name> ::= <symbol>
<individual-name> ::= <symbol> | <host-lang-expr>
<role-expr> ::= <symbol>
<index> ::= <number> | <symbol>
<partition-index> ::= <number> | <symbol>
<fn> ::= a unary function with boolean return type that can be evaluated in the host language.

---

1. + means one or more values separated by blanks.

Fig. 33 The grammar of the CLASSIC language (taken from [Borgida et al., 1989]).

unspecified difference from THING in general. This is quite the opposite of the case with other (non-primitive) concepts, as we shall see in a moment.

Primitives can also have non-trivial parents. Thus, SPORTS-CAR might be defined as a subconcept of both CAR and another concept, EXPENSIVE-THING:

(PRIMITIVE (AND CAR EXPENSIVE-THING)
sports-car).

In fact, the parent of a primitive concept can be any CLASSIC concept, including another primitive. Primitives thus specify *necessary* conditions: if $Corvette_1$ is an instance of SPORTS-CAR, then it is both a CAR and an EXPENSIVE-THING. But note that there is no sufficiency condition specified for primitive concepts.

The CLASSIC language of concepts allows us to go substantially beyond the simple IS-

A hierarchies of more traditional semantic data models. It offers three special ways of describing objects in terms of their structure. As we shall see, these constructors allow some class membership relations be determined by inference. CLASSIC's three complex constructors are role *value restrictions*, *cardinality bounds*, and *co-reference constraints*. Role value restrictions are type constraints that hold for the fillers for some single role.

Value restriction. For example, the concept expression

(ALL thing-driven CAR)

describes any object that is related by the thing-driven role solely to individuals describable by the concept CAR.

Bounds restrict the number of fillers for roles. For example,

(AT-MOST 4 thing-driven)

describes any object that is related to at most 4 distinct individuals through the thing-driven role.

(AT-LEAST 3 wheel)

describes any object that is related to at least 3 distinct individuals through the wheel role.

Co-reference constraints specify simple equalities between single-valued roles or, more generally, chains of such roles. For example, the expression

(SAME-AS (driver) (insurance payer))

describes all those individuals whose filler for the driver role is the same as the payer of their insurance role.

Each of the constructors acts as part of both necessary and sufficient conditions for concepts in which they appear (as long as they are not used in a primitive concept, in which case there are no sufficient conditions).

It is important to note that the meaning of concepts in CLASSIC is determined by their structure. This implies that certain relationships exist between concepts by virtue of their definition. For example, it is quite possible for several different concept expressions to denote the same class:

(AND (ALL thing-driven CAR)
(ALL thing-driven EXPENSIVE-THING))

is the same concept as

(ALL thing-driven
(AND CAR EXPENSIVE-THING)),

Various studies have examined extensions of the expressive power of such a language and the trade-off in computational complexity for deriving is-a relationships between concepts and individuals in such a logic. Efficient implementations for core sets of primitives in these languages have been developed in the meantime (cf. [Borgida & Patel-Schneider, 1994], [MacGregor, 1994], and [Horrocks & Patel-Schneider, 1999]), see for example DLP[10] and the FaCT system[11].

---

10. http://www.bell-labs.com/user/pfps/
11. http://www.cs.man.ac.uk/~horrocks/software.html.

6.2 XML, RDF, and Ontology Languages

In the following I will examine how XML and RDF can be used to express ontologies.

6.2.1 DTD and Ontologies

The closest thing that XML offers for ontological modeling is the Document Type Definition (DTD) which defines the legal nestings of tags and introduces attributes for them. Defining tags, their nesting, and attributes for tags may be seen as defining an ontology. However, there are significant differences between an ontology and DTD.

- First, a DTD specifies the legal *lexical* nesting in a document, which may or may not coincide with an *ontological* hierarchy (subclass relationship). That is, there is nothing in a DTD that corresponds to the is-a relationship of classes that is usually central in an ontology.
- Second, and in consequence, DTDs lack any notion of inheritance. In an ontology, subclasses inherit attributes defined for their superclasses and superclasses inherit instances defined for their subclasses. Both inheritance mechanisms do not exist for DTDs.
- Third, DTDs provide a rather poor means for defining the semantics of elementary tags. Basically, a tag can be defined as being composed of other tags or being a string. Usually, ontologies provide a much richer typing concept for describing elementary types.
- Fourth, DTDs define the order in which tags appear in a document. For ontologies, in contrast, the ordering of attribute descriptions does not matter.

We will use an example to clarify these differences (see Figure 34).

- Concept c_1 has two attributes, a_1 and a_2.
  This implies that the domains of a_1 and a_2 are the elements of c_1. The range of a_1 is

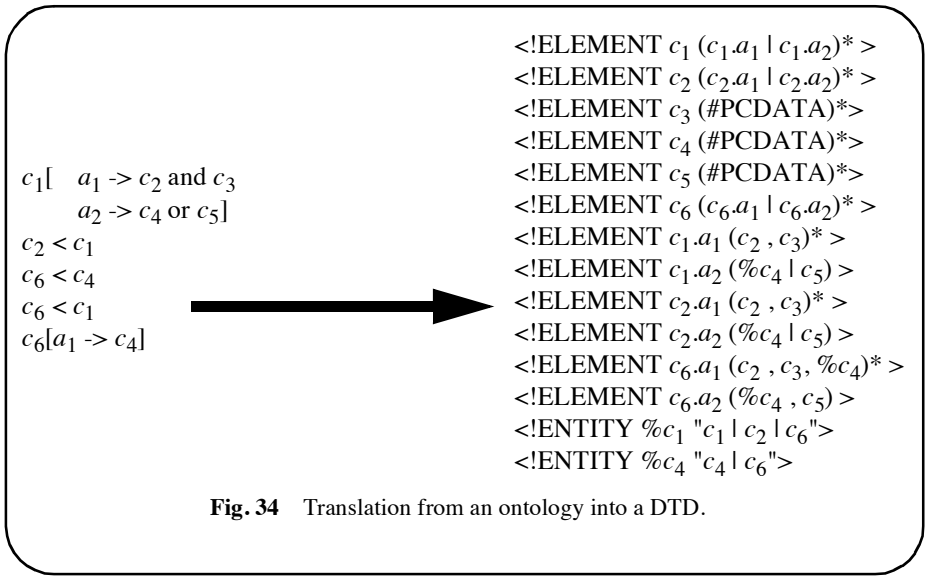


Fig. 34 Translation from an ontology into a DTD.

the intersection of c_2 and c_3 and the range of a_2 is the union of c_4 and c_5.

- c_2 is defined as a subclass of c_1.
  This implies that all attributes defined for c_1 are also applicable for c_2. In addition, each element of c_2 is also an element of c_1.
- Finally c_6 is a subclass of c_4 and c_1.
  Therefore it inherits the attributes a_1 and a_2 from c_1.
  In addition, it refines the range restriction of attribute a_1 to c_2 and c_3 and c_4. That is, the value of a_1 applied to an element of c_6 must also be an element of c_4. This is not necessary for an element of c_1 that is not also an element of c_6.

When translating this ontology into a DTD we first define c_1 as an element having two subtags $c_1.a_1$ and $c_1.a_2$, i.e.,

<c_1>
  <$c_1.a_1$>...</$c_1.a_1$>
  <$c_1.a_2$>...</$c_1.a_2$>
</c_1>

would be a valid document. Therefore, we reify the attribute names with the concept names to distinguish different appearances of attributes in various concepts. A number of problems arise in this translation process:

- The sequence of attribute values of an object does not matter in an ontology, i.e.
  $o[a_1=5, a_2=3]$ and $o[a_2=3, a_1=5]$ are equivalent.
  We express this by $(a_1 \mid a_2)^*$. However, this implies that an object may have several values for the same attribute (which is allowed for set-valued but not for single-valued attributes).
- The attribute a_1 has for c_1 as range the intersection of c_2 and c_3. That is, a value of the attribute is an object for which the attributes of c_2 and c_3 can be applied. We express this via $(c_2 , c_3)^*$ which again imply that an object may have several values for the same attribute.
- The only primitive data type is PCDATA, i.e. arbitrary strings.

We can also see the two aspects of inheritance in the translation process

- First, we have to add all inherited attributes and their inherited range restrictions explicitly. For example:

  <!**ELEMENT** $c_6.a_1$ (c_2*, c_3*, %c_4*)* >
- Second, the value of an attribute may also be the element of a subclass of its value type (i.e., a super class inherits all elements of its subclasses). Therefore, whenever a class is used as a range restriction we have to add all its subclasses. For this we use the entity mechanism of DTDs. For example,

  <!**ENTITY** %c_1 "c_1 | c_2 | c_6">

More details and further aspects of ontology to DTD translations can be found in [Erdmann & Studer, 1999], [Rabarijoana et al., 1999], and [van Harmelen & Fensel, submitted]. In a nutshell, DTDs are rather weak in regard to what can be expressed with them.

Work on *XML-schemes* (cf. [Malhotra & Maloney, 1999]) may well contribute to

bridging the gap between DTD's and ontologies. Schemes will be introducing mechanisms for constraining document structure and content, mechanisms to enable inheritance for element, attribute, and data type definitions, mechanisms for application-specific constraints and descriptions, mechanisms to enable the integration of structural schemes with primitive data types, primitive data typing, including byte, date, integer, sequence, etc., and they shall allow the creation of user-defined data types.

Up to now I have discussed the mapping from ontologies to DTDs. [Welty & Ide, 1999] discuss a mapping from DTDs to an ontological representation. [Welty & Ide, 1999] want to provide the reasoning service of description logic to query and manipulate XML documents. DTDs are therefore translated automatically into a representation of an ontology in description logic. This ontology simply consists of each element in the DTD. The taxonomy can be derived by the classifier of the description logic CLASSIC based on the use of entities and the type attributes.

6.2.2 RDF and Ontologies

RDFS can be used directly to describe an ontology. *Objects*, *Classes*, and *Properties* can be described. Predefined properties can be used to model *instance of* and *subclass of* relationships as well as *domain restrictions* and *range restrictions* of attributes. A speciality of RDFS is that properties are defined globally and are not encapsulated as attributes in class definitions. Therefore, a frame or object-oriented ontology can only be expressed in RDFS by reifying the property names with class name suffixes (as we already saw for XML). In regard to ontologies, RDF provides two important contributions:

- a standardized syntax for writing ontologies,
- a standard set of modeling primitives like *instance of* and *subclass of* relationships.

On the one hand, RDFS provides rather limited expressive power. A serious weakness of RDF is that it lacks a standard for describing logical axioms. RDFS allows the definition of classes and properties through their types (by providing their names). No intensional definitions or complex relationships via axioms can be defined. On the other hand, RDFS provides a rather strong reification mechanism. RDF expressions can be used as terms in metaexpressions. Here, RDFS provides reified second-order logic like it is used in CycL and KIF. Neither Frame logic nor most Description Logics provide such an expressivity[12]. However, there are good reasons for this restriction in the latter approaches. This feature makes it very difficult to define a clean semantics in the framework of first-order logic and disables sound and complete inference services.[13] In the case of RDFS, such an inference service remains possible because of the otherwise restricted expressive power that does not provide any rule language. That is, RDFS provides syntactical features of second-order logic without requiring actually second-order semantics.

---

12. Exceptions are described in [Calvanese et al., 1995] and [De Giacomo & Lenzerini, 1995], however, without an implemented reasoning service.

13. The problems stems from the fact that, because terms in second-order logic may be arbitrary formulas, already term unification in second-order logic (i.e., one simple substep in deduction) requires full deduction in first-order logic which is undecidable in the general case.

6.2.3 Comparing RDF and XML

RDF is an application of XML for the purpose of representing metadata. For example, the RDF statements

date(http://www.xyz.de/Example/Smith/) = July 1999
subject(http://www.xyz.de/Example/Smith/) = Intelligent Agents
creator(http://www.xyz.de/Example/Smith/) = http://www.xyz.de/~smith/
name(http://www.xyz.de/~smith/) = John Smith
email(http://www.xyz.de/~smith/) = smith@organisation.de

can be represented in linear XML syntax (see Fig. 35). This may raise the question why there is a need for RDF at all, because all metadata represented in RDF can also be presented in XML. However, RDF provides a *standard* form for representing metadata in XML. Directly using XML to represent metadata would result in its being represented in various ways.

The difference becomes even more obvious when considering how to represent an ontology in RDF or XML. Above I showed how an ontology can be used to generate a DTD describing the structure of XML documents. However, I did not discuss how the ontology itself could be represented in XML. For defining a standardized manner in which ontologies can be represented in XML we have to decide about two questions:

- What are the epistemological primitives used to represent an ontology (i.e., things like classes, is-a relationships, element-of relationships, attributes, domain and range restrictions etc.)? Basically these are decisions about the *meta*-ontology used to represent ontologies.
- How can these concepts be represented in the linear syntax of XML.

```
<? xml version=1.0”?>
<RDF
    xmlns=”http://www.w3c.org/1999/02/22-rdf-syntax-ns#”
    xmlns:DC=”http://www.purl.org/DC#/”
    xmlns:y=”http://www.description.org/schema”>

    <Description about=”http://www.xyz.de/Example/Smith/”>
        <DC:date rdf:resource=”July 1999”/>
        <DC:subject rdf:resource=”Intelligent Agents”/>
        <DC:creator rdf:resource=”http://www.xyz.de/~smith/”/>
    </Description>

    <Description about=”http://www.xyz.de/~smith/”>
        <DC:name rdf:resource=”John Smith”/>
        <DC:email rdf:resource=”smith@organisation.de”/>
    </Description>

</RDF>
```

Fig. 35 XML representation of RDF statements.

There are a number of different possibilities, and this makes clear how RDFS comes into the story. RDFS provides a fixed set of modeling primitives for defining an ontology (classes, resources, properties, is-a and element-of relationships, etc.) and a standard way to encode them in XML. Using XML directly for the purpose of representing ontologies would require us to duplicate this standardization effort.[14]

6.3 XOL and OIL

Currently two proposals have been made to unify ontology and web languages: XOL and OIL. We will conclude our discussion by briefly dealing with these new approaches.

6.3.1 XOL

The BioOntology Core Group[15] recommends the use of a frame-based language with an XML syntax for the exchange of ontologies for molecular biology. The proposed language is called XOL[16] (cf. [Karp et al., 1999], [McEntire et al., 1999]). The ontology definitions that XOL is designed to encode include both schema information (meta-data), such as class definitions from object databases – as well as non-schema information (ground facts), such as object definitions from object databases.

The syntax of XOL is based on XML. The modeling primitives and semantics of XOL are based on OKBC-Lite, which is a simplified form of the knowledge model for the *Open Knowledge Base Connectivity (OKBC)*[17] ([Chaudhri et al., 1997], [Chaudhri et al., 1998]). OKBC is an application program interface for accessing frame knowledge representation systems. Its knowledge model supports features most commonly found in knowledge representation systems, object databases, and relational databases. OKBC-Lite extracts most of the essential features of OKBC, but omits some of its more complex aspects. XOL was inspired by Ontolingua. XOL differs from Ontolingua, however, as it has an XML-based syntax rather than a Lisp-based syntax. Still, the semantics of OKBC-Lite which underly XOL are extremely similar to the semantics of Ontolingua.

The design of XOL deliberately uses a generic approach to define ontologies, meaning that the single set of XML tags defined for XOL (defined by a single XML DTD) can describe any and every ontology. This approach contrasts with the approaches taken by other XML schema languages, in which a generic set of tags is typically used to define the schema portion of the ontology and the schema itself is used to generate a second set of application-specific tags (and an application-specific DTD), which in turn are used to encode a separate XML file that contains the data portion of the ontology. Compare the XOL definitions in Figure 36. All of the XML elements of this specification (meaning all the words inside brackets), such as *class*, *individual*, and *name* are generic, i.e., they pertain to all ontologies. All of the ontology-specific information is in the text portion of

---

14. However, it would also allow us to part with some of the unusual design decisions of RDF.
15. http://smi-web.stanford.edu/projects/bio-ontology/
16. http://www.ontologos.org/Ontology/XOL.htm.
17. http://www.ai.sri.com/~okbc/. OKBC has also been chosen by FIPA as its exchange standard for ontologies, see http://www.fipa.org, FIPA 98 Specification, Part 12 Ontology Service (cf. [FIPA 98, part 12]).

```
<class>
    <name>person</name>
</class>
<slot>
    <name>age</name>
    <domain>person</domain>
    <value-type>integer</value-type>
    <numeric-max>150</numeric-max>
</slot>
<individual>
    <name>fred</name>
    <type>person</type>
    <slot-values>
    <name>age</name>
    <value>35</value>
    </slot-values>
</individual>
```

Fig. 36 An example in XOL (taken from [Karp et al., 1999]).

the XML file, that is between the pairs of elements. In contrast, approaches discussed earlier in this book might use this type of XML markup to define the individual Fred as shown in Figure 37.

What are the advantages of the generic approach taken by XOL relative to the non-generic approach? The primary advantage of the XOL approach is simplicity. Only one XML DTD need be defined to describe any and every ontology. Using the non-generic approach, every ontology must define a second, ontology-specific, DTD for describing the data elements of the ontology. Furthermore, rules would have to be defined that describe exactly how that second DTD is derived from the schema portion of the ontology, and most likely, programs would have to be written to generate such DTDs from schema specifications. The XML language provides no formal machinery to define those rules. The entire DTD defining *valid* XOL documents is given in Figure 38. XOL appears rather promising because it provides ontological modeling primitives expressed in one of the most important information exchange standards: XML.

6.3.2 OIL

OIL[18] (cf. [Fensel et al., 2000], [Horrocks et al., to appear]) unifies three important

```
<person>
    <name>fred</name>
    <age>35</age>
</person>
```

Fig. 37 Nonreusable Ontology Specification.

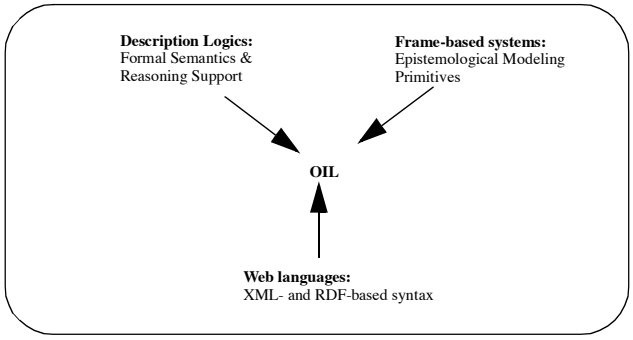


Fig. 39 The three roots of OIL.

aspects provided by different communities (see Figure 39): Formal semantics and efficient reasoning support as provided by Description Logics, epistemologically rich modeling primitives as provided by the Frame community, and a standard proposal for syntactical exchange notations as provided by the Web community.

Description Logics (DL). DLs describe knowledge in terms of concepts and role restrictions that are used to automatically derive classification taxonomies. The main effort of research in knowledge representation is in providing theories and systems for expressing structured knowledge and for accessing and reasoning with it in a principled way. In spite of the discouraging theoretical complexity of their results, there are now efficient implementations for DL languages, see for example DLP[19] and the FaCT system.[20] OIL inherits from Description Logic its *formal semantics* and the *efficient reasoning support* developed for these languages. In OIL, *subsumption* is decidable and with FaCT we can provide an efficient reasoner for this.

Frame-based systems. The central modeling primitives of predicate logic are predicates. Frame-based and object-oriented approaches take a different point of view. Their central modeling primitives are classes (i.e., frames) with certain properties called attributes. These attributes do not have a global scope but are only applicable to the classes they are defined for (they are typed) and the ”same” attribute (i.e., the same attribute name) may be associated with different range and value restrictions when defined for different classes. A frame provide a certain context for modeling one aspect of a domain. Many other additional refinements of these modeling constructs have been developed and have led to the incredible success of this modeling paradigm. Many frame-based systems and languages have been developed and, re-named as object-orientation they have conquered the software engineering community. Therefore, OIL incorporates the *essential modeling primitives* of frame-based systems into its language. OIL is based on the notion of a concept and the definition of its superclasses and attributes. Relations can also be defined not as an attribute of a class but as an independent entity having a certain domain and range. Like classes, relations can be

18. http://www.ontoknowledge.org/oil.
19. http://www.bell-labs.com/user/pfps/
20. http://www.cs.man.ac.uk/~horrocks/software.html. Actually OIL uses FaCT as its inference engine.

```
<!ELEMENT
    (module | ontology | kb | database | dataset)
    ( name, (kb-type | db-type)?, package?, version?, documentation?, class*,
    slot*, individual*)>
<!ELEMENT name (#PCDATA)>
<!ELEMENT kb-type (#PCDATA)>
<!ELEMENT documentation (#PCDATA)>
<!ELEMENT class
    ( name, documentation?,
    (subclass-of | instance-of | slot-values)* )>
<!ELEMENT slot
    ( name, documentation?,
    (domain | slot-value-type | slot-inverse | slot-cardinality | slot-maximum-cardinality |
    slot-minimum-cardinality | slot-numeric-minimum | slot-numeric-maximum
    | slot-collection-type | slot-values)* )>
<!ATTLIST slot
    type ( template | own ) "own">
<!ELEMENT individual
    ( name, documentation?, (type | slot-values)* )>
<!ELEMENT slot-values
    ( name, value*,
    (facet-values | value-type | inverse
    | cardinality | maximum-cardinality | minimum-cardinality
    | numeric-minimum | numeric-maximum | some-values
    | collection-type | documentation-in-frame)* )>
<!ELEMENT facet-values
    ( name, value* )
<!ELEMENT subclass-of     (#PCDATA)>
<!ELEMENT instance-of     (#PCDATA)>
<!ELEMENT domain          (#PCDATA)>
<!ELEMENT slot-value-type (#PCDATA)>
<!ELEMENT slot-inverse            (#PCDATA)>
<!ELEMENT slot-cardinality        (#PCDATA)>
<!ELEMENT slot-maximum-cardinality (#PCDATA)>
<!ELEMENT slot-minimum-cardinality (#PCDATA)>
<!ELEMENT slot-numeric-minimum     (#PCDATA)>
<!ELEMENT slot-numeric-maximum     (#PCDATA)>
<!ELEMENT slot-collection-type     (#PCDATA)>
<!ELEMENT value-type              (#PCDATA)>
<!ELEMENT inverse                 (#PCDATA)>
<!ELEMENT cardinality             (#PCDATA)>
<!ELEMENT maximum-cardinality     (#PCDATA)>
<!ELEMENT minimum-cardinality     (#PCDATA)>
<!ELEMENT numeric-minimum         (#PCDATA)>
<!ELEMENT numeric-maximum         (#PCDATA)>
<!ELEMENT some-values             (#PCDATA)>
<!ELEMENT collection-type         (#PCDATA)>
<!ELEMENT documentation-in-frame   (#PCDATA)>
```

Fig. 38 The XOL DTD (see http://www.ontologos.org/Ontology/XOL.htm).

arranged in a hierarchy.

Web standards: XML and RDF. Modeling primitives and their semantics are one aspect of an Ontology language. Second, we have to decide about its syntax. Given the current dominance and importance of the WWW, a syntax of an ontology exchange language must be formulated using existing web standards for information representation. OIL is closely related to XOL and can be seen as an extension of XOL. For example, XOL only allows necessary but not sufficient class definitions (i.e., a new class is always a sub-class of, and not exactly equal to, its specification) and only class names, but not class expressions (except for the limited form of expression provided by slots and their facets), can be used in defining classes. The XML syntax of OIL was mainly defined as an extension of XOL. Another candidate for a web-based syntax for OIL is RDF together with RDFS. In regard to ontologies, RDFS provides two important contributions: a standardized syntax for writing ontologies, and a standard set of modeling primitives like instance of and subclass of relationships. Therefore, OIL offers two syntactical variants: one based on XML schema and one based on RDF schema.

7 Conclusions

Currently computers are shifting from single isolated devices to entry points into a world wide network of information exchange and business transactions. Therefore, support in data, information, and knowledge exchange is becoming the key issue in current computer technology. Ontologies provide a shared and common understanding of a domain that can be communicated between people and heterogeneous application systems. In consequence, they will play a major role in supporting information exchange processes in various areas. Their impact may be as important as the development of programming languages in the seventies and eighties.

The book discussed the role ontologies will play in *knowledge management* and in *electronic commerce*. Both are increasingly important areas that determine the economic success of companies and societies. Knowledge management is concerned with effective and efficient access to internal and external knowledge that enable a company to be informed of its environment. Electronic commerce enables new and additional business relationships to customers and suppliers. Both areas require the integration of heterogeneous and distributed data and information sources. The success of the Internet and the World Wide Web provide the basis for this integration. However, they only provide a necessary not a sufficient condition. Having a telephone connection does not help if both partners do not speak the same language. Ontologies can provide the required translation service. Large industrial groups are working on standards for industrial segments and Internet portals may create de-facto standards. Translation service into these standards enables the various players to communicate. In the book I showed how ontologies, i.e. formal and consensual domain theories can provide the ground for this integration and translation processes. I also showed how new and arising Web standards, such as RDF and XML, can be used as an underlying representation language for ontologies.

The use of *one* ontology for all application contexts will never be possible. Neither will an ontology be suitable for all subjects and domains nor will such a large and heterogeneous community as the Web community ever agree on a complex ontology for describing all their issues. For example, the Dublin Core community (cf. [Weibel et al., 1995], [Weibel, 1999]) 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 group, it is based on a group of people who are joined by a common interest and some agreement as to how to look at their topic.[1] 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 (i.e., middle agents [Decker et al., 1997]) and softbots [Etzioni, 1997] guide a user through this knowledgeable network superimposed on the current Internet

1. VerticalNet powers Internet portals that realizes such ontology groups for various industrial segments.

(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 helping to evolve „the Web from a Document Repository to a Knowledge Base.“ [Guha et al., 1998]

Figure 40 sketches the Web of the future. Currently, access to the Web is at a level comparable to programming in the sixties. HTML provides text fragments and pointers that allow the user to jump to other information pieces like go-to instructions in assembler programs. The Web of the next generation will have various intermediate information access, extraction, and integration services between the raw information sources and the human user. Brokers provide advanced access to information sources and describe their competence via ontologies. User agents will access these brokers and will meet the information needs of their human users. Facilitators and matchmakers guide these user agents and therefore mediate between information sources and information needs. Developing new abstraction layers which enable better service and functionality is one of the key tendencies in computer science. Currently, we are encountering this in the area of information access, extraction, exchange, and integration.

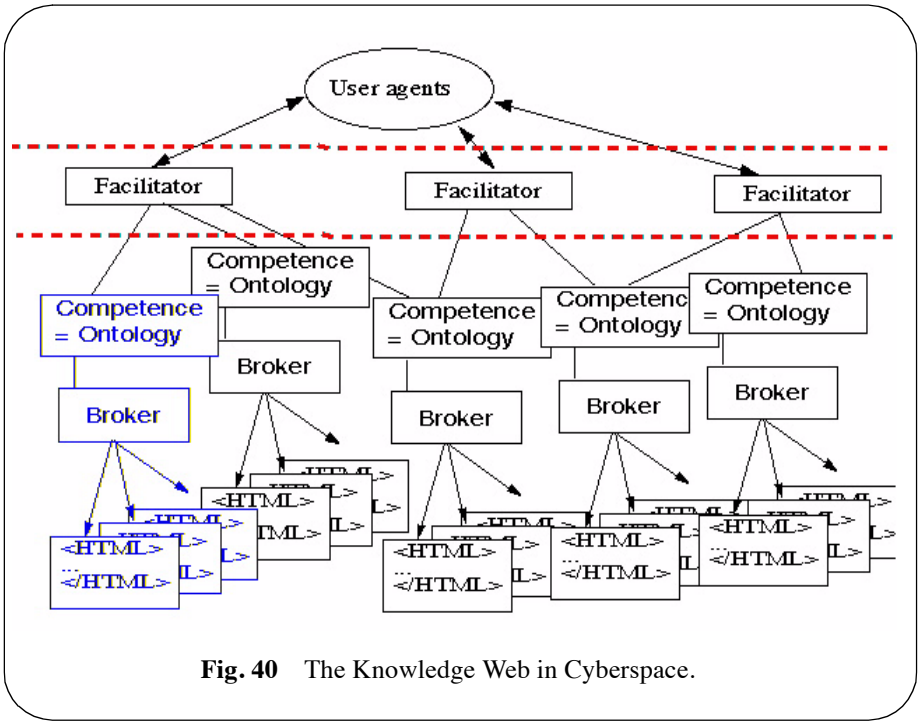


Fig. 40 The Knowledge Web in Cyberspace.

8 References

[Arpírez et al., 1998] J. Arpírez, A. Gómez-Pérez, A. Lozano, S. Pinto: (ONTO)^{2}Agent: An ontology-based WWW broker to select ontologies. In *Proceedings of the ECAI-98 Workshop on Applications of Ontologies and PSMs,* Brighton. England. August 1998. pp. 16-24.

[Baader et al., 1991] F. Baader, H.-J. Bürckert, J. Heinsohn, J. Müller, B. Hollunder, B. Nebel, W. Nutt, and H.-J. Profitlich: Terminological Knowledge Representation: A Proposal for a Terminological Logic. DFKI Technical Memo TM-90-04, Deutsches Forschungszentrum für Künstliche Intelligenz, Kaiserslautern, 1990. Updated version, taking into account the results of a discussion at the ``International Worksop on Terminological Logics," Dagstuhl, May 1991.

[Benjamins et al., 1998] V. R. Benjamins, E. Plaza, E. Motta, D. Fensel, R. Studer, B. Wielinga, G. Schreiber, Z. Zdrahal, and S. Decker: An Intelligent Brokering Service for Knowledge-Component Reuse on the World-WideWeb. In *Proceedings of the 11th Banff Knowledge Acquisition for Knowledge-Based System Workshop (KAW´98)*, Banff, Canada, April 18-23, 1998.

[Benjamins et al., 1999] V. R. Benjamins, D. Fensel, S. Decker and A. Gomez Perez: (KA)^2: Building Ontologies for the Internet: a Mid Term Report, *International Journal of Human-Computer Studies*, 51:687-712, 1999.

[BizTalk] http://www.biztalk.org/.

[Bharat & Broder, 1998] K. Bharat and A. Broder. A technique for measuring the relative size and overlap of public Web search engine. In *Proceedings of the 7th International WWW Conference*, Brisbane, Australia, April 14-18, 1998, pp. 379-388.

[Booch et al., 1999] G. Booch, J. Rumbaugh, and I. Jacobson: *The Unified Modeling Language User Guide*, Addison-Wesley, 1999.

[Borgida et al., 1989] A. Borgida, R. J. Brachman, D. L. McGuinness, and L. A. Resnick: CLASSIC: A Structural Data Model for Objects. In *Proceedings of the 1989 ACM SIGMOD International Conference on Mangement of Data*, pp. 59-67, 1989.

[Borgida & Patel-Schneider, 1994] A. Borgida and P. F. Patel-Schneider: A Semenatics and Complete Algorithm for Subsumption in the CLASSIC Description Logic, *Journal of Artificial Intelligence Reserach*, 1:277-308, 1994.

[Borst & Akkermans, 1997] W. N. Borst and J. M. Akkermans: Engineering Ontologies, *International Journal of Human-Computer Studies*, 46 (2/3):365-406, 1997.

[Bowman et al., 1994] C. M. Bowman, P. B. Danzig, U. Manber, and M. F. Schwartz: Scalable Internet Resource Discovery: Research Problems and Approaches, *Communications of the ACM*, 37(8), pp. 98-107, August 1994.

[Brachman & Schmolze, 1985] R. Brachman and J. Schmolze: An overview of the KL-ONE Knowledge Representation System, *Cognitive Science*, 9(2):171-216, 1985.

[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.

[Brin & Page, 1998] S. Brin and L. Page: The Anatomy of a Large-Scale Hypertextual Web Search Engine. In *Porceedings of the 7th International World Wide Web Conference (WWW-7)*, Brisbone, Australia, April 14-18, 1998.

[Brickley et al., 1998] D. Brickley, R. Guha, and A. Layman (eds.): *Resource description framework (RDF) schema specification*. W3C Working Draft, August 1998. http://www.w3c.org/TR/WD-rdf-schema.

[Bryan, 1998] M. Bryan (ed.): Guidelines for using XML for Electronic Data Interchange, http://www.geocities.com/WallStreet/Floor/5815/guide.htm, January 1998. See also on XML/EDI http://www.xmledi.org and http://www.geocities.com.

[Calvanese et al., 1995] D. Calvanese, G. De Giacomo, and M. Lenzerini: Structured objects: Modeling and resoning. In *Proceedings of the Fourth International Conference on Deductive and Object-Oriented Databases (DOOD-95)*, Lecture Notes in Computer Science (LNCS), Springer, 1995.

[CBL] http://www.commerce.net and http://www.veosystems.com.

[Chaudhri et al., 1997] V. K. Chaudhri, A. Farquhar, R. Fikes, P. D. Karp, and J. P. Rice: *Open Knowledge Base Connectivity 2.0.*, Technical Report KSL-98-06, Knowledge Systems Laboratory, Stanford., July 1997.

[Chaudhri et al., 1998] V. K. Chaudhri, A. Farquhar, R. Fikes, P. D. Karp, and J. P. Rice: OKBC: A Foundation for Knowledge Base Interoperability. In *Proceedings of the National Conference on Artificial Intelligence (AAAI98)*, pp. 600-607, July 1998.

[Chavez & Maes, 1996] A. Chavez and P. Maes: Kasbah: An Agent Marketplace for Buying and Selling Goods. In *Proceedings of the First International Conference on the Practical Application of Intelligent Agents and Multi-Agent Technology*, London, UK, April 1996.

[Clark, 1999] J. Clark: XSL Transformations (XSLT) Version 1.0, W3C Working Draft, August 1999. See http://www.w3.org/TR/WD-xslt.

[cXML] http://www.oasis-open.org and http://www.cxml.org.

[de Carvalho Moura et al., 1998] A. M. de Carvalho Moura, M. L. Machado Campos, and C. M. Barreto: A survey on metadata for describing and retrieving Internet resources, *World Wide Web*, 1:221-240, 1998.

[Connolly, 1997] D. Connolly: *XML: Principles, Tools, and Techniques*, O'Reilly, 1997.

[Dao & Perry, 1996] S. Dao and B. Perry: Information Mediation in Cyberspace: Scalable Methods for Declarative Information Networks, J*ournal of Intelligent Information Systems*, 6(2/3), 1996.

[De Giacomo & Lenzerini, 1995] G. De Giacomo and M. Lenzerini: What's in an Aggregate: Foundations for Description Logics with Tupels and Sets. In

Proceedings of the 14th International Conference on Artificial Intelligence (IJCAI-95), Montreal, Canada, 1995.

[Decker et al., 1997] K. Decker, K. Sycara and M. Williamson: Middle-Agents for the Internet. In *Proceedings of the 15th International Joint Conference on AI (IJCAI97)*, Nagoya, Japan, August 23-29, 1997.

[Decker et al., 1998] S. Decker, D. Brickley, J. Saarela, J. Angele: A Query Service for RDF. In [QL, 1998].

[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, 1999.

[Domingue & Motta, in press] J. Domingue and E. Motta: Planet-Onto: From News Publishing to Integrated Knowledge Management Support, *IEEE Intelligent Systems*, in press.

[Doorenbos et al., 1997] R. B. Doorenbos, O. Etzioni, and D. S. Weld: A Scalable Comparison-Shopping Agent for the World-Wide Web. In *Proceedings of the AGENTS 97 Conference*, 1997. http://www.cs.washington.edu/research/shopbot.

[Dublin Core] http://purl.oclc.org/metadata/dublin_core.

[ECMA, 1997] European Computer Manufactur's Association (ECMA): *Portable Common Tool Environment (PCTE) - Mapping from CASE Data Interchange Format (CDIF) to PCTE*, 1997. http://www.ecma.ch/stand/Ecma-270.htm.

[EDIFACT] United Nation: *UN/EDIFACT-Directory*. http://www.unece.org/trade/untdid, 1999.

[Erdmann & Studer, 1999] M. Erdmann and R. Studer: *Ontologies as Conceptual Models for XML Documents*, research report, Institute AIFB, University of Karlsruhe, 1999.

[Errikson et al., 1994] H. Eriksson, A. R. Puerta, and M. A. Musen: Generation of knowledge-acquisition tools from domain ontologies, *International Journal of Human Computer Studies (IJHCS)*, 41:425-453, 1994.

[Errikson et al., 1999] H. Eriksson, R. W. Fergerson, Y. Shahar, and M. A. Musen: Automated Generation of Ontology Editors. In *Proceedings of the Twelfth Workshop on Knowledge Acquisition, Modeling and Management (KAW99)*, Banff, Alberta, Canada, October 16-21, 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*, 46:707-728, 1997.

[Fellbaum, 1999] Christiane Fellbaum (ed.): *WordNet: An Electronic Lexical Database*, MIT Press, 1999.

[Fensel, 2000] D. Fensel: *Understanding, Developing and Reusing Problem-Solving*

Methods, to appear as Lecture Notes of Artificial Intelligence (LNAI), Springer-Verlag, Berlin, 2000.

[Fensel & Benjamins, 1998] D. Fensel and V. R. Benjamins: Key Issues for Automated Problem-Solving Methods Reuse. In *Proceedings of the 13th European Conference on Artificial Intelligence (ECAI-98)*, Brighton, UK, August 1998, pp. 63-67.

[Fensel & Groenboom, 1997] D. Fensel and R. Groenboom: Specifying Knowledge-based Systems with Reusable Components. In *Proceedings 9th International Conference on Software Engineering and Knowledge Engineering (SEKE '97)*, Madrid 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., 1998a] 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.

[Fensel et al., 1998b] D. Fensel, J. Angele, and R. Studer: The Knowledge Acquisition And Representation Language KARL, *IEEE Transactions on Knowledge and Data Engineering*, 10(4):527-550, 1998.

[Fensel et al., 1999a] D. Fensel, J. Angele, S. Decker, M. Erdmann, H.-P. Schnurr, S. Staab, R. Studer, and A. Witt: On2broker: Semantic-Based Access to Information Sources at the WWW. In *Proceedings of the World Conference on the WWW and Internet (WebNet 99),* Honolulu, Hawaii, USA, October 25-30, 1999.

[Fensel et al., 1999b] Dieter Fensel, V. Richard Benjamins, Enrico Motta, and Bob Wielinga: UPML: A Framework for knowledge system reuse. In *Proceedings of the International Joint Conference on AI (IJCAI-99)*, Stockholm, Sweden, July 31 - August 5, 1999.

[Fensel et al., 2000] D. Fensel, I. Horrocks, F. Van Harmelen, S. Decker, M. Erdmann, and M. Klein: OIL in a nutshell. In *Proceedings of the Workshop on Applications of Ontologies and Problem-solving Methods*, 14th European Conference on Artificial Intelligence ECAI'00, Berlin, Germany August 20-25, 2000.

[Fensel et al., to appear] Dieter Fensel, Stefan Decker, Michael Erdmann, Hans-Peter Schnurr, Rudi Studer, and Andreas Witt: Lessons learnt from Applying AI to the Web. To appear in *Journal of Cooporative Information Systems*.

[Finin et al., 1997] T. Finin, Y. Labrou, and J. Mayfield: KQML as an agent communication language. In Jeff Bradshaw (ed.), *Software Agents*, MIT Press, Cambridge, 1997,

[FIPA 98, part 12] Foundation for Intelligent Physical Agents (FIPA): FIPA 98 Specification, Part 12, Ontology Service, October 1998. http://www.fipa.org.

[Flammia & McCandless, 1997] G. Flammia and M. McCandless: From Software to

Service: the Transformation of Srink-wrapped Software on the Internet, *IEEE Expert*, 12(2), 1997.

[Fox et al., 1993] M. S. Fox, J. F. Chionglo, and F. G. Fadel: A Common Sense Model of the Enterprise. In *Proceedings of the 2nd Industrial Engineering Research Conference*, Norcross GA, USA, 1993.

[Fox & Gruninger, 1997] M. S. Fox and M. Gruninger: On Ontologies and Enterprise Modelling. In *Proceedings of the International Conference on Enterprise Integration Modelling Technology 97*, Springer Verlag, 1997.

[Fridman-Noy & Hafner, 1997] N. Fridman-Noy and C.D. Hafner: The State of the Art in Ontology Design, *AI Magazine*, 18(3):53-74, 1997.

[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.

[Genesereth, 1991] M. R. Genesereth: Knowledge Interchange Format. In *Proceedings of the Second International Conference on the Principles of Knowledge Representation and Reasoning (KR-91)*, J. Allenet al., (eds), Morgan Kaufman Publishers, 1991, pp 238-249. See also http://logic.stanford.edu/kif/kif.html.

[Genesereth & Fikes, 1992] M.R. Genesereth and R.E. Fikes: Knowledge Interchange Format, Version 3.0, Reference Manual. Technical Report, Logic-92-1, Computer Science Dept., Stanford University, 1992. http://www.cs.umbc.edu/kse/.

[Glushko et al., 1999] R. J. Glushko, J. M. Tenenbaum, and B. Meltzer: An XML Framework for Agent-based E-commerce, *Communications of the ACM*, 42(3), 1999.

[Gomez Perez & Benjamins, 1999] A. Gomez Perez and V. R. Benjamins: Applications of Ontologies and Problem-Solving Methods, *AI-Magazine*, 20(1):119-122, 1999.

[Greenwald & Kephart, 1999] A. R. Greenwald and J. O. Kephart: Shopbots and Pricebots. In *Proceedings of the 16th International Joint Conference on AI (IJCAI-99)*, Stockholm, Sweden, July 31 - August 6, 1999.

[Grosso et al., 1999] W. E. Grosso, H. Eriksson, R. W. Fergerson, J. H. Gennari, S. W. Tu, and M. A. Musen: Knowledge Modeling at the Millennium (The Design and Evolution of Protégé-2000). In *Proceedings of the Twelfth Workshop on Knowledge Acquisition, Modeling and Management (KAW99)*, Banff, Alberta, Canada, October 16-21, 1999.

[Gruber, 1993] T. R. Gruber: A Translation Approach to Portable Ontology Specifications, *Knowledge Acquisition*, 5:199—220, 1993.

[Gruber, 1995] T.R. Gruber: Towards Principles for the Design of Ontologies used for Knowledge Sharing, *International Journal of Human-Computer Studies*, 43:907-928, 1995.

[Guarino, 1995] N. Guarino, Formal Ontology: Conceptual Analysis and Knowledge

Representation, *International Journal of Human-Computer Studies*, 43(2/3):625-640, 1995.

[Guarino, 1998] N. Guarino (ed.): *Formal Ontology in Information Systems*, IOS Press, Amsterdam, 1998.

[Guarino et al., 1999] N. Guarino, C. Masolo, and G. Vetere: OntoSeek: Content-Based Access to the Web, IEEE Intelligent Systems, 14(3), 1999.

[Guha, 1993] R. V. Guha: *Context Dependence of Representations in Cyc*, MCC Technical Report, CYC 066-93, 1993.

[Guha et al., 1998] R. V. Guha, O. Lassila, E. Miller and D. Brickley: Enabling Inferencing. In [QL, 1998].

[Hagel III & Singer, 1999] J. Hagel III and M. Singer: *Net Worth. Shapping Markets When Customers Make the Rules*, Havard Business School Press, Boston, Massachusetts, 1999.

[van Harmelen & Fensel, submitted] F. van Harmelen & D. Fensel: Surveying notations for machine-processable semantics of Web sources, submitted.

[van Harmelen & van der Meer, 1999] F. van Harmelen and J. van der Meer: WebMaster: Knowledge-based Verification of Web-pages. In *Proceedings of the Second International Conference on The Practical Applications of Knowledge Management (PAKeM99)*, London, UK, April 1999, pp. 147-166.

[van Heijst et al., 1997] G. van Heijst, A. Schreiber, and B. Wielinga: Using Explicit Ontologies in KBS development, *International Journal of Human Computer Studies*, 46:183-292, 1997.

[Horrocks et al., to appear] I. Horrocks, D. Fensel, J. Broekstra, S. Decker, M. Erdmann, C. Goble, F. van Harmelen, M. Klein, S. Staab, R. Studer, and E. Motta: The Ontology Inference Layer OIL, submitted. http://www.ontoknowledge.org/oil.

[Horrocks & Patel-Schneider, 1999] I. Horrocks and P. F. Patel-Schneider: Optimising Description Logic Subsumption, *Journal of Logic and Computation*, 9(3):267-293, 1999.

[Hunter & Armstrong, 1999] J. Hunter and L. Armstrong: A Comparison of Schemas for Video Metadata Representation. In *Proceedings of the Eighth International World Wide Web Conference (WWW8)*, Toronto, Canada, May 11-14, 1999.

[indecs] http://www.indecs.org.

[IOTP] http://www.ietf.org/html.charters/trade-center.html. http://www.otp.org.

[ISO/IEC, 1990] ISO/IEC 10027 IRDS Framework, 1990.

[ISO/IEC, 1995a] ISO/IEC 10746 Reference Model of Open Distributed Processing, 1995.

[ISO/IEC, 1995b] ISO/IEC 11179 Information Technology - Specification and Standardization of Data Elements, 1995.

[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.

[Joachims et al., 1997] T. Joachims, D. Freitag, and T. Mitchell: WebWatcher. A Tour Guide for the World Wide Web. In *Proceedings of the 15th International Joint Conference on AI (IJCAI-97)*, Nagoya, Japan, August 1997.

[Karp et al., 1999] P. D. Karp, V. K. Chaudhri, and J. Thomere: XOL: An XML-Based Ontology Exchange Language, Version 0.3, July 3, 1999. ftp://smi.stanford.edu/pub/bio-ontology/OntologyExchange.doc.

[Kent, 1999] R. E. Kent: Conceptual Knowledge Markup Language. In *Proceedings of the Twelfth Workshop on Knowledge Acquisition, Modeling and Management (KAW99),* Banff, October 1999. See http://sern.ucalgary.ca/KSI/KAW/KAW99/papers/Kent1/CKML.pdf

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

[Klusch, 1999] M. Klusch (ed.): Intelligent Information Agents: Agent-Based Information Discovery and Management on the Internet, Springer-Verlag Berlin/Heidelberg, 1999.

[KQML] KQML: http://www.cs.umbc.edu/kqml/.

[Krulwich, 1996] B. Krulwich: The BargainFinder agent: Comparison price shopping on the Internet. In *Agents, Bots, and other Internet Beasties, SAMS.NET publishing* (division of Macmillan publishing), pp. 257-263, May, 1996. See also http://bf.cstar.ac.com/bf/.

[Krulwich, 1997] B. Krulwich: Lifestyle Finder. Intelligent User Profiling Using Large-Scale Demographic Data, *AI Magazine*, 18(2), 1997.

[Kushmerick, 1997] N. Kushmerick: *Wrapper Induction for Information Extraction*. Ph.D. Dissertation, Department of Computer Science & Engineering, University of Washington, 1997. Available as Technical Report UW-CSE-97-11-04.

[Labrou & Finin, 1995] Y. Labrou and T. Finin: Comments on the specification for FIPA '97 Agent Communication Language, University of Maryland, Baltimore County, Baltimore Maryland, USA, February 28, 1997. See http://www.cs.umbc.edu/kqml/papers/fipa/comments.shtml

[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,1999] O. Lassila and R. Swick. Resource Description Framework (RDF). W3C proposed Recommendation, January 1999. http://www.w3c.org/TR/WD-rdf-syntax.

[Lawrence & Giles, 1998] S. Lawrence and C. L. Giles: Searching the world wide Web,

Science, 280(4):98-100, 1998.

[Lenat, submitted] D. B. Lenat: The Dimensions of Context Space, submitted. http://casbah.org/resources/cycContextSpace.shtml.

[Lenat & Guha, 1990] D. B. Lenat and R. V. Guha: *Building large knowledge-based systems. Representation and inference in the Cyc project*, Addison-Wesley, Reading, Massachusetts, 1990.

[Li, to appear] H. Li: XML and Industrial Standards for Electronic Commerce, to appear.

[Lieberman, 1998a] H. Lieberman: Integrating User Interface Agents with Conventional Applications, *ACM Conference on Intelligent User Interfaces*, San Fransisco, January 1998.

[Lieberman, 1998b] H. Lieberman: Beyond Information Retrieval: Information Agents at the MIT Media Lab, *Künstliche Intelligenz*, 3/98:17-23.

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

[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 *First International Conference on Autonomous Agents (AA'97),* 1997.

[MacGregor, 1991] R. MacGregor: Inside the LOOM Classifier, *SIGART Bulletin*, 2(3):70-76, June 1991.

[MacGregor, 1994] R. MacGregor: A Description Classifier for the Predicate Calculus. In *Proceedings of the 12th National Conference on AI (AAAI-94)*, pp. 213-220, 1994.

[Maes et al., 1999] P. Maes, R. H. Guttman, and A. G. Moukas: Agent that Buy and Sell, *Communications of the ACM*, 42(3), March 1999.

[Malhotra & Maloney, 1999] A. Malhotra and M. Maloney: XML-Schema Requirements. W3C Note, February 1999. http://www.w3.org/TR/NOTE-xml-schema-req.

[Manola 1998] F. Manola: Towards a Web Object Model, In *Proceedings of the Workshop on Compositional Software Architectures*, Monterey, California January 6-8, 1998. http://www.objs.com/workshops/ws9801/papers/paper022.html

[McCarthy 1993] J. McCarthy: Notes on Formalizing Context. In *Proceedings of the 13th International Conference on Artificial Intelligence (IJCAI-93)*, Chambery, France, 1993.

[McEntire et al., 1999] R. McEntire, P. Karp, N. Abernethy, F. Olken, R. E. Kent, M. DeJongh, P. Tarczy-Hornoch, D. Benton, D. Pathak, G. Helt, S. Lewis, A. Kosky, E. Neumann, D. Hodnett, L. Tolda, and T. Topaloglou: *An Evaluation of Ontology Exchange Languages for Bioinformatics*. August 1999, ftp://smi.stanford.edu/pub/

bio-ontology/OntologyExchange.doc.

[McGuinness, 1999] D. L. McGuinness: Ontologies for Electronic Commerce. In *Proceedings of the AAAI '99 Artificial Intelligence for Electronic Commerce Workshop* , Orlando, Florida, July, 1999.

[Meersman, 1999] R. A. Meersman: The use of lexicons and other computer-linguistic tools in semantics, design and cooperation of database systems. In Y. Zhang (ed.), *CODAS Conference Proceedings,* Springer Verlag, 1999.

[Microsoft (a)] Microsoft: *Open Information Model*. http://msdn.microsoft.com/repository/OIM and http://www.mdcinfo.com/OIM.

[Microsoft (b)] Microsoft: *XML Interchange Format (XIF)*. http://msdn.microsoft.com/repository/.

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

[Muslea et al., 1998] I. Muslea, S. Minton, and C. Knoblock: Wrapper Induction for Semistructured, Web-based Information Sources. In *Proceedings of the Conference on Automatic Learning and Discovery*, Pittsburgh, 1998.

[Nebel, 1996] B. Nebel: Artificial Intelligence: A Computational Perspective. In G. Brewka (ed.), *Principles of Knowledge Representation*, CSLI publications, Studies in Logic, Language and Information, Stanford, 1996.

[Nwana, 1996] H. S. Nwana: Software Agents: An Overview, *Knowledge Engineering Review*, 11(3), 1996.

[OAGIS] Open Applications Group Integration Specification, http://www.openapplications.org.

[OCF] Open Catalog Format, http://www.martsoft.com/ocp.

[O´Leary, 1997b] D. E O´Leary: The Internet, Intranets, and the AI Renaissance, *IEEE Intelligent Systems*, January 1997.

[OMG, 1997] Object Management Group (OMG): Meta Object Facility (MOF) Specification, 1997.

[OMG, 1998] Object Management Group (OMG): Stream-Based Model Interchange, 1998.

[Peat & Webber, 1997] B. Peat and D. Webber: XML/EDI - the E-business framework, August 1997, http://www.geocities.com/WallStreet/Floor/5815/startde.htm.

[Perkowitz & Etzioni, 1997] 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.

[Perkowitz & Etzioni, 1999] M. Perkowitz and O. Etzioni: Adaptive Web Sites: Conceptual Clustering Mining. In *Proceedings of the 16th International Joint Conference on AI (IJCAI-99)*, Stockholm, Sweden, July 31 - August 6, 1999.

[Pirlein & Studer, in press] Th. Pirlein and R. Studer: Integrating the Reuse of Commonsense Ontologies and Problem-Solving Methods, *International Journal*

of Expert Systems: Research and Applications.

[Puerta et al., 1992] A. R. Puerta, „J. W. Egar, S. W. Tu, M.A. and Musen: A Multiple-method Knowledge-Acquisition Shell for the Automatic Generation of Knowledge-acquisition Tools, *Knowledge Acquisition*, 4(2):171—196, 1992.

[QL, 1998] *Proceedings of W3C Query Language Workshop (QL'98) - The Query Languages Workshop*, Boston, Massachussets, December 3-4, 1998. http://www.w3.org/TandS/QL/QL98/.

[Rabarijoana et al., 1999] A. Rabarijoana, R. Dieng, and O. Corby: Exploitation of XML for Corporative Knowledge Management. In D. Fensel and R. Studer (eds.), *Knowledge Acquisition, Modeling, and Management, Proceedings of the European Knowledge Acquisition Workshop (EKAW-99)*, Lecture Notes in Artificial Intelligence, LNAI 1621, Springer-Verlag, 1999.

[RETML] Real Estate Transaction Markup Language (RETML), http://www.rets-wg.org.

[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.

[Swartout et al., 1996] B. Swartout, R. Patil, K. Knight, and T. Russ: Toward Distributed Use of Large-scale Ontologies. In B. R. Gaines and M. A. Musen, (eds.), *Proceedings of the 10th Banff Knowledge Acquisition for Knowledge-Based Systems Workshop*, Banff, Canada, 1996.

[Schach et al., 1998] D. Schach, J. Lapp, and J. Robie: Querying and Transforming XML. In [QL, 1998].

[Schreiber et al., 1999] G. Schreiber, H. Akkermans, A. Anjewierden, R. de Hoog, N. Shadbolt, W. Van de Velde, and B. Wielinga: *Knowledge Engineering and Management. The CommonKADS Methodology*, MIT Press, 1999.

[Selberg & Etzioni, 1997] M. Selberg and O. Etzioni: The Metacrawler architecture for resource aggregation on the Web, *IEEE Expert*, January-February 1997.

[Shardanand & Maes, 1995] U. Shardanand and P. Maes: Social Information Filtering: Algorithms for Automating "Word of Mouth". In *Proceedings of the SIG Computer and Human Interaction, ACM Press*, New York, 1995.

[Staudt et al., 1999] M. Staudt, A. Vaduva, and T. Vetterli: Metadata Management and Data Warehousing, Swiss Life, Information Systems Research, Technical Report, no 21, 1999. http://research.swisslife.ch/Papers/data/smart/meta.ps.

[Studer et al., 1996] R. Studer, H. Eriksson, J. H. Gennari, S. W. Tu, D. Fensel, and M. Musen: Ontologies and the Configuration of Problem-Solving Methods. In B. R. Gaines and M. A. Musen, (eds.), *Proceedings of the 10th Banff Knowledge Acquisition for Knowledge-Based Systems Workshop*, Banff, Canada, 1996.

[Studer et al., 1998] R. Studer, V. R. Benjamins, and D. Fensel: Knowledge Engineering: Principles and Methods, *Data and Knowledge Engineering (DKE)*, 25(1-2):161-197, 1998

[Sycara et al., 1999] K. Sycara, J. Lu, M. Klusch, and S. Widoff: Matchmaking among Heterogeneous Agents on the Internet, In *Proceedings of the AAAI Spring Symposium on Intelligent Agents in Cyberspace*, Stanford, USA, 1999.

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

[Uschold et al., 1996] M. Uschold, M. King, S. Moralee, and Y. Zorgio: The Enterprise Ontology, *Knowledge Engineering Revie*w, 11(2), 1996.

[Uschold & Grüninger, 1996] M. Uschold and M. Grüninger: Ontologies: Principles, Methods and Applications, *Knowledge Engineering Review*, 11(2), 1996.

[Weibel, 1999] S. Weibel: The State of the Dublin Core Metadata Initiative April 1999, *D-Lib Magazine,* 5(4), 1999.

[Weibel et al., 1995] S. Weibel, J. Gridby, and E. Miller: OCLC/NCSA Metadata Workshop Report, Dublin, EUA, 1995. http://www.oclc.org:5046/oclc/research/conferences/metadata/dublin_core_report.html.

[Westarp et al., 1999] F. v. Westarp, T. Weitzel, P. Buxmann, and W. König: The Status Quo and the Future of EDI - Results of an Empirical Study. In *Proceedings of the European Conference on Information Systems (ECIS'99)*, 1999.

[Welty & Ide, 1999] C. Welty and N. Ide: Using the Right Tools: Enhancing retrieval From Marked-Up Documents, *Computers and the Humanities,* 33:1-2, Special Issue on the Tenth Anniversary of the Text Encoding Initiative, 1999. http://www.cs.vassar.edu/faculty/welty/papers/.

[Wiederhold, 1992] G. Wiederhold: Mediators in the Architecture of Future Information Systems, *IEEE Computer*, 25(3):38-49, 1992.

[Wiederhold et al., 1997] G. Wiederhold: Value-added Mediation in Large-Scale Information Systems. In R. Meersman and Leo Mark (eds.): *Database Application Semantics*, Chapman and Hall, 1997, pages 34-56.

[Wiederhold & Genesereth et al., 1997] G. Wiederhold and M. Genesereth: The Conceptual Basis for Mediation Services. *IEEE Expert*, September/October, pp. 38-47, 1997.

[Wildemann] H. Wildemann: Arbeitskreis Virtuelle Maerkte - Einkauf von Commodities, http://www.bwl.wiso.tu-muenchen.de,

[XML] http://www.w3c.org/xml.

9 Appendix - Survey of Standards

The appendix summarizes most of the approaches we discussed in this book. In fact, we enumerate standards in the areas of ontologies, software, WWW languages, text, video and metadata, and electronic commerce. Finally, we mention shopbots, auction houses, and B2B portals.

9.1 Survey Papers

[de Carvalho Moura et al., 1998] reviews existing metadata standards for indexing on-line information sources. [Hunter & Armstrong, 1999] evaluates and compares several schemes for video metadata representation based on current Web technology. [Manola 1998] discusses a couple of standards related to developing an object-oriented model for Web documents. [Li, to appear] discusses XML-based industrial standards arising in the area of electronic commerce. [Staudt et al., 1999] discusses and compares several standards for metadata descriptions developed in Software Engineering, Data Warehousing, and the WWW. Finally [McEntire et al., 1999] evaluates several representation formalism that can be used to define ontology exchange languages. This analysis is part of the development activity of XOL.

9.2 Ontology standards

Ontology.Org[1] is an independent industry and research forum focussed on the application of ontologies in Internet commerce. It is the central goal of Ontology.Org to use ontologies to address the problems that impact the formation and sustainability of large electronic trading groups. The main barrier to electronic commerce lies in the need for applications to share information, not in the reliability or security of the Internet. The problem is particularly acute when a large number of trading partners attempt to agree and define the standards for interoperation. Ontology.Org decided to focus on this area and solve the problems that impact the formation and sustainability of large electronic trading groups - based on the principles of ontology.

CommonKADS[2] is the leading methodology to support structured knowledge engineering. It was gradually developed and has been validated by many companies and universities in the context of the European ESPRIT IT Program. It now is the European de facto standard for knowledge analysis and knowledge-intensive system development, and it has been adopted as a whole or been partly incorporated in existing methods by many major companies in Europe, as well as in the US and Japan. **CML2** (**Conceptual Modeling Language**)[3] is a semi-formal notation for the specification of COMMONKADS knowledge models, including the specification of ontologies.

Cyc[4] [Lenat & Guha, 1990] was initiated in the course of Artificial Intelligence making common-sense knowledge accessible and processable for computer programs. CYC started as an approach to formalize this world knowledge and provide it with a formal and executable semantics. In the meantime hundred of thousands of concepts have been

1. http://www.ontology.org
2. http://www.commonkads.uva.nl/
3. http://www.swi.psy.uva.nl/projects/kads22/cml2doc.html
4. http://www.cyc.com/.

formalized with millions of logical axioms -- rules and other assertions -- which specify constraints on the individual objects and classes. **CycL** was developed in the Cyc project [Lenat & Guha, 1990] for the purpose of specifying the large common sense ontology that should provide Artificial Intelligence to computers. Far from having attained this goal, Cyc still provides the worldwide largest formalized ontology. CycL is a formal language whose syntax is derived from first-order predicate calculus.[5]

The **Foundation for Intelligent Physical Agents (FIPA)**[6] defines standards for promoting agent-based applications, services, and equipments. As part of its standards it defines an *ontology service* [FIPA 98, part 12] and the way it can be accessed (i.e., the communication aspect). In addition, a metaontology that describes the knowledge types that can be accessed via an ontology service is defined. Here, the knowledge model of OKBC is used to provide an object-oriented representation of knowledge.

IBROW[7] [Fensel & Benjamins, 1998] is a project that has the goal of developing a broker for the access of *dynamic reasoning services* in the WWW. The objective of IBROW3 is to develop intelligent brokers that are able to distributively configure reusable components into knowledge systems through the World-Wide Web. Part of IBROW is the development of **UPML**[8] [Fensel et al., 1999b] which is a proposal for standardizing the description of reasoning components including their ontologies.

The **Knowledge Interchange Format (KIF)**[9] [Genesereth, 1991] provides a common means of exchanging knowledge between programs with different knowledge representation techniques. It can express first-order logic sentences with some additional second-order capabilities.

KQML or the **Knowledge Query and Manipulation Language**[10] [Finin et al., 1997] is a language and protocol for exchanging information and knowledge. It is part of a larger effort, the ARPA Knowledge Sharing Effort which is aimed at developing techniques and methodology for building large-scale knowledge bases which are sharable and reusable. KQML is both a message format and a message-handling protocol to support run-time knowledge sharing among agents. KQML can be used as a language for an application program to interact with an intelligent system or for two or more intelligent systems to share knowledge in support of cooperative problem solving. A related approach is the **FIPA '97 Agent Communication Language**[11] (compare [Labrou & Finin, 1995]).

OIL[12] unifies three important aspects provided by different communities: Formal semantics and efficient reasoning support as provided by Description Logics, epistemologically rich modeling primitives as provided by the Frame community, and a standard proposal for syntactical exchange notations as provided by the Web

---

5. http://www.cyc.com/cycl.html/
6. http://www.fipa.org
7. http://www.swi.psy.uva.nl/projects/IBROW3/home.html.
8. http://www.aifb.uni-karlsruhe.de/WBS/ibrow/
9. http://logic.stanford.edu/kif/kif.html
10. http://www.cs.umbc.edu/kqml/
11. http://www.fipa.org
12. http://www.ontoknowledge.org/oil.

community. It is a proposal for a standard ontology language for Web-based applications.

OML[13] and **CKML**[14] [Kent, 1999] are a family of languages for expressing ontologies. Translations into RDF and XML and other languages exist.

The **Open Knowledge Base Connectivity (OKBC)**[15] ([Chaudhri et al., 1997], [Chaudhri et al., 1998]) is an API for accessing and modifying multiple, heterogeneous knowledge bases. It is a successor of the Generic **Frame Protocol (GFP)**. Its knowledge model defines a metaontology for expressing ontologies in an object-oriented frame-based manner.

Ontolingua[16] (cf. [Gruber, 1993], [Farquhar et al., 1997]) has been designed to support the design and specification of ontologies with a clear logical semantics based on KIF. Ontolingua extends KIF with additional syntax to capture intuitive bundling of axioms into definitional forms with ontological significance and a Frame Ontology to define object-oriented and frame-language terms.

TOVE[17] [Fox et al., 1993] is an example of a task and domain-specific ontology. The ontology supports enterprise integration, providing a shareable representation of knowledge.

The **XML-Based Ontology Exchange Language (XOL)** (cf. [McEntire et al., 1999], [Karp et al., 1999]) has been developed in the area of molecular-biology information. The semantics of XOL is based on OKBC-Lite, a simplified version of the knowledge model of OKBC, and its syntax is based on XML.

9.3 SE standards (see [Staudt et al., 1999])

The **Case Data Interchange Format (CDIF)** is a Software Standard for the exchange of metadata between CASE tools. It has been adopted by ISO and defines the structure and content of metadata.

The **Information Resource Dictionary System (IRDS)** [ISO/IEC, 1990] standard, developed by ISO/IEC, addresses the requirements and architecture of dictionary systems. The four-layered architecture is similar to CDIF and MOF.

Meta object Facility (MOF) [OMG, 1997] adopted by the Object Management Group (OMG) is a metastandard. It defines a *meta*-metamodel that can be used to define the metamodel of other modeling approaches. The main purpose of OMG MOF is to provide a set of CORBA interfaces that can be used to define and manipulate a set of interoperable metamodels. The **XML Metadata Interchange (XMI)** [OMG, 1998] provides rules for transforming MOF based meta models into XML DTDs. It therefore provides a general exchange standard between different modeling approaches using XML for this purpose.

---

13. http://www.ontologos.org/OML/OML.htm
14. http://www.ontologos.org/OML/..\..\CKML\CKML.htm
15. http://www.ai.sri.com/~okbc/. OKBC has also been chosen by FIPA as exchange standard for ontologies, see http://www.fipa.org, FIPA 98 Specification, Part 12 Ontology Service (cf. [FIPA 98, part 12]).
16. http://ontolingua.stanford.edu/
17. http://www.eil.utoronto.ca/tove/toveont.html.

Object Data Management Group (ODMG)[18] defines a standard for storing objects. Its purpose is to ensure the portability of applications across different Database management systems. All Database management systems provide a data definition language, or DDL, that enables the user to define the schema. The **ODMG Object Definition (ODL)**[19] is a database schema definition language extension of the OMG Interface Definition Language (IDL) that describes the standard of an object type and its attributes, relationships and operations.

The **Open Information Model (OIM)** [Microsoft (a)] is a Software Engineering Standard developed by Microsoft and passed to the Meta-Data Coalition which is a group of vendors interested in the definition, implementation, and evaluation of metadata interchange formats. The OIM model suite is a core model of the most commonly used metadata types. In particular the Knowledge Management Model is of interest in the context of ontology specification. The **XML Interchange Facility (XIF)** [Microsoft (b)] is based on XML and used to exchange OIM models (like XMI does for MOF).

The **Portable Common Tool Environment (PCTE)** [ECMA, 1997] standard from ECMA and ISO/IEC is a repository standard. It provides an object base and various functions to manipulate these objects.

Reference Model of Open Distributed Processing (RM-ODP) [ISO/IEC, 1995a] is a metastandard for specifying open, widely spread systems. The RM-ODP provides the framework and enables ODP standards to be developed for specifying components that are mutually consistent and can be combined to build infrastructures matching user requirements. It has been adopted by ISO/IEC.

The **Unified Modeling Language (UML)** [Booch et al., 1999] adopted by the OMG is a Software Engineering Standard. It is a language for specifying, visualizing, constructing, and documenting artifacts of software systems. It is highly compatible with RM-ODP.

9.4 WWW standards

The **Document Content Description (DCD) for XML**[20] proposes a structural schema facility for specifying rules covering the structure and content of XML documents. The DCD proposal incorporates a subset of the XML-Data Submission and expresses it in a way which is consistent with the Resource Description Framework.

The **Resource Description Framework (RDF)**[21] (cf. [Miller, 1998], [Lassila & Swick,1999]) provides a means for adding semantics to a document without making any assumptions about its structure. RDF is an infrastructure that enables the encoding, exchange, and reuse of structured metadata.

Schema for Object-Oriented XML (SOX)[22] SOX is a schema language (or metagrammar) for defining the syntactic structure and partial semantics of XML

---

18. http://www.odmg.org/
19. http://www.odmg.org/standard/standardoverview.htm#Object Definition Language (ODL)
20. http://www.w3.org/TR/NOTE-dcd
21. http://www.w3c.org/Metadata/
22. http://www.w3.org/TR/NOTE-SOX/

document types. As such, SOX is an alternative to XML DTDs and can be used to define the same class of document types (with the exception of external parsed entities). However, SOX extends the language of DTDs by supporting an extensive (and extensible) set of datatypes, inheritance among element types, namespaces, polymorphic content, embedded documentation, and features to enable robust distributed schema management.

The **eXtendible Markup Language XML**[23](e.g., [Connolly, 1997]) is a tag-based language for describing tree structures with a linear syntax. It is a successor to SGML, which was developed long ago for describing document structures.

XML - Query Languages[24] [QL, 1998] are notations for addressing and filtering the elements and text of XML documents. A standard for these query languages is currently under development.

The purpose of **XML schema**[25] is to define and describe classes of XML documents by using constructs to constrain and document the meaning, usage and relationships of their constituent parts: datatypes, elements and their contents, as well as attributes and their values. Schema constructs may also provide for the specification of additional information, such as default values.

XSL-T[26] [Clark, 1999] is a language for transforming XML documents into other XML documents. XSLT is designed for use as part of XSL, which is a stylesheet language for XML. In addition to XSLT, XSL includes an XML vocabulary for specifying formatting. XSL specifies the styling of an XML document by using XSLT to describe how the document is transformed into another XML document that uses the formatting vocabulary. XSLT can also be used independently of XSL.

9.5 Text, Video, and Metadata standards [Manola 1998]

Dublin Core [Dublin Core] is a set of metadata attributes for describing documents in electronic networks.

ISO/IEC 11179 [ISO/IEC, 1995b] is a family of standards. It is for the specification and standardization of data element descriptions and semantic content definitions.

The Harvest's **Summary Object Interchange Format (SOIF)** is a syntax for representing and transmitting descriptions of (metadata about) Internet resources as well as other kinds of structured objects.

The **Text Encoding Initiative (TEI)** Guidelines enable the encoding of a wide variety of textual phenomena to any desired level of fine-grainedness and complexity.

The **Warwick Framework** [ISO/IEC, 1995b] defines a container architecture for aggregating distinct packages of metadata.

The **Machine Readable Code Record Format (MARC)** is an international standard and is the format most commonly used for academic catalogues. The Library of Congress was responsible for developing the original MARC format in 1965-66.

---

23. http://www.w3.org/XML/
24. http://www.w3.org/TandS/QL/QL98/
25. http://www.w3.org/TR/xmlschema-1/ and http://www.w3.org/TR/xmlschema-2/
26. http://www.w3.org/TR/xslt

Similar work was in progress in the United Kingdom where the Council of the British National Bibliography set up the BNB MARC Project to examine the use of machine-readable data in producing the printed British National Bibliography (BNB). In 1968 the MARC II project began as an Anglo-American effort to develop a standard communications format. Despite cooperation, two versions emerged, UKMARC and USMARC, which reflected the respective national cataloguing practices and requirements of BNB and the Library of Congress.

The **EAD Document Type Definition (DTD)**[27] is a standard for encoding archival finding aids using the Standard Generalized Markup Language (SGML). The standard is maintained in the Network Development and MARC Standards Office of the Library of Congress (LC) in partnership with the Society of American Archivists.

The **Multimedia Content Description Interface MPEG-7**[28] will specify a standard set of descriptors that can be used to describe various types of multimedia information. MPEG-7 will also standardize ways to define other descriptors as well as structures (Description Schemes) for the descriptors and their relationships to allow fast and efficient searching for material which a user is interested in. MPEG-7 will also standardize a language to specify description schemes, i.e. a Description Definition Language (DDL). that is printed on paper.

9.6 Electronic Commerce Standards [Li, to appear]

The **BizTalk framework** [BizTalk] is an XML framework for application integration and electronic commerce. It provides XML schemes and a set of tags for defining messages between applications. It has mainly been developed by Microsoft.

Common Business Library (CBL) [CBL] is being developed by Veosystems. It is a public collection of DTDs and modules that can be used to develop XML-based electronic commerce applications.

Commerce XML (cXML) [cXML] is being developed by Ariba to provide common business objects and the definition of request/response processes based on XML.

The **Information and Content Exchange protocol (ICE)** [29] is being developed for use by content syndicators and their subscribers. The ICE protocol defines the roles and responsibilities of syndicators and subscribers, defines the format and method of content exchange, and provides support for the management and control of syndication relationships in traditional publishing contexts and in B2B relationships.

The ***in*teroperability of *d*ata in *e-c*ommerce *s*ystems <in*d*ecs>** initiative [indecs] is funded under the European Commission info2000 program embracing multimedia rights clearance. The meta data model promises to provide a semantic and syntactic framework suitable for systems to support workflow, database design, rights management, bibliographic cataloguing, data exchange and e-commerce.

The **Internet Open Trading Protocol (IOTP)** [IOTP] is defined by the Internet Engineering Task Force (IETF) and mainly deals with electronic payment systems.

---

27. http://lcweb.loc.gov/ead/
28. http://drogo.cselt.stet.it/mpeg/standards/mpeg-7/mpeg-7.htm
29. http://www.w3.org/TR/NOTE-ice.

Security, authentication, and digital signatures are its major concerns.

The **Open Applications Group Integration Specification (OAGIS)** [OAGIS] is defined by the Open Application Group (OAG) to integrate business applications. OAGIS defines a vocabulary of business terms and more than ninety different types of Business Object Documents to be exchanged.

The **Open Buying on the Internet (OBI) Consortium**[30] is a non-profit organization dedicated to developing open standards for B2B electronic commerce. It is managed by CommerceNet. With OBI, different business-to-business purchasing systems can interoperate. It supports multi-vendor requirements, customer-specific catalogs, and secure processing on the Web.

The **Open Catalog Format (OCF)** [OCF] is the content language of the Open Catalog Protocol, an XML-based software protocol to support the exchange of complex data between product catalogs.

The **Open Financial Exchange (OFX)**[31] is a unified specification for the electronic exchange of financial data between financial institutions, businesses and consumers via the Internet. Created by CheckFree, Intuit, and Microsoft in early 1997, Open Financial Exchange supports a wide range of financial activities, including consumer and small business banking; consumer and small business bill payment; billing and investments, including stocks, bonds, and mutual funds. Other financial services, including financial planning and insurance, will be added in the future and ncorporated into the specification.

The **Real Estate Transaction Markup Language (RETML)** [RETML] is an open standard for exchanging real estate transaction information. It was created by the National Association of Realtors.

RosettaNet[32] focuses on building a master dictionary to define properties for products, partners, and business transactions in electronic commerce. This master dictionary, coupled with an established implementation framework (exchange protocols), is used to support the electronic commerce dialog known as the Partner Interface Process or PIP. RosettaNet PIPs create new areas of alignment within the overall EC and IT supply-chain's electronic commerce processes, allowing electronic commerce and IT supply-chain partners to scale electronic commerce.

The **UN/SPSC**[33] began as a merger between the United Nation's Common Coding System (UNCCS), itself based on the United Nations Common Procurement Code (CPC), and Dun & Bradstreet's Standard Product and Service Codes (SPSC). The UN/SPSC is a hierarchical classification, with five levels. Each level contains a two-character numerical value and a textual description.

WebEDI [Westarp et al., 1999] and **XML/EDI**[34] [Peat & Webber, 1997] are initiatives to integrate new Web technology and electronic commerce. Electronic data interchange

---

30. http://www.openbuy.org/
31. http://www.ofx.net/.
32. http://www.rosettanet.org/ and http://www.extricity.com
33. http://www.spsc.org/
34. http://www.xmledi.com/

is being shifted from EDIFACT to better-suited Web standards like XML.

9.7 Electronic Commerce Portals

Shopbots (B2C): Bookblvd[35], Bottom Dollar[36], Buyer's Index[37], CompareNet[38], consumerreviews.com[39], Dealpilot[40], Epinions[41], Jango[42], Junglee[43], MyShop[44], Shopfind[45], and Shopper[46].

Auction houses (B2C): Accompany[47], Adauction[48], Alando[49], Amazon[50], Andsold[51], Artnet[52], Auction Box[53], eBay´s Auction house[54], FreeMarkets[55], Kasbah[56] Mondus[57], National Transport Exchange[58], Nextag[59], OnSale[60], Ricardo[61], and Ron Angels[62].

B2B portals: Chemdex[63], CommerceOne[64], harbinger.net[65], Mondus[66], mysap[67], PaperExchange[68], VerticalNet[69].

35. http://www.bookblvd.com/
36. http://www.bottomdollar.com/
37. http://www.buyersindex.com/
38. http://www.compare.net/
39. http://www.consumerreviews.com
40. http://www.dealpilot.com/
41. http://www.epinions.com/
42. http://www.jango.com/
43. http://www.junglee.com/
44. http://www.myshop.de
45. http://www.shopfind.com/
46. http://www.shopper.com
47. http://www.accompany.com
48. http://www.adauction.com
49. http://www.alando.de
50. http://www.amazon.com/auctions
51. http://www.andsold.de
52. http://www.artnet.com
53. http://auction.eecs.umich.edu
54. http://www.ebay.com
55. http://www.Freemarkets.com
56. kasbah.media.mit.edu
57. http://www.mondus.com
58. http://www.nte.com
59. http://www.nextag.com

60. http://www.onsale.com
61. http://www.ricardo.de
62. http://www.ronangels.com
63. http://www.chemdex.com
64. http://www.commerceone.com
65. http://www.harbinger.net
66. http://www.mondus.com
67. http://www. mysap.com
68. http://www.paperexchange.com
69. http://www.verticalnet.com