Ontology Engineering for Active Catalog

Jihie Kim, S. Ringo Ling, and Peter Will
Information Sciences Institute and Computer Science Department
University of Southern California
4676 Admiralty Way
Marina del Rey, CA 90292, U.S.A.
{jihie, ling, will}@isi.edu

Abstract

The exponential growth of the Internet and increasing communication and computational power have created many opportunities for advancing engineering, manufacturing and business activities. Although rich in content, existing catalog systems are poor in both search quality and the usage of the retrieved information. The Active Catalog brings a conceptually new idea to electronic commerce by providing new, computationally usable, catalog information environment about products and their applications. It is designed for finding and using information in building applications in design, logistics, and electronic commerce in the context of heterogeneous, Internet-based distributed computing environments. This paper focuses on increasing the quality of search results and providing users with easy access to catalog information. We provide various ways of understanding user requirements and powerful reasoning capabilities, so that the catalog service can correctly capture the user goals and improve hit ratio and coverage of the results.

Introduction

A catalog is a basic resource to a number of people and disciplines ranging from shoppers to companies wanting to sell products. Catalog information has traditionally been in paper form and are due for rapid revolution in content and presentation because of the existence of the Internet and the World Wide Web. PartNet and IndustryNet are examples.

Most of the benefits of catalogs and libraries are based on a major prerequisite: that the information is easily and conveniently accessible to the customer (i.e., it is reachable and usable). This assumption does not naturally hold in general. Although rich in content, today's libraries are poor in access and their contents are designed to be consumable by a human reading a document or looking at an image.

Figure 1 shows the search results from various catalog services (in February 1997). The table shows the number of retrieved information items (manufacturers or online pointers) given a set of keywords (DC motor, Servo Motor, and DC Servo Motor). The existing catalog systems, either in paper form or on the Web, cannot fully exploit the product information space. The retrieved information is often ineffectual and hard to manage. All the information services, except for InfoSeek (on company) and EDN, retrieve a large number of items (up to 81,620) in some searches,

Services	keywords	Sources	On-line Info	Comments
■ Thomas Register	DC Motor	260	29	On-line info provided by TR
	Servo Motor	210	24	
	DC Servo Motor	0	0	
■ Industry Net	DC Motor	>500	177	On-line info provided by Indus.Net
	Servo Motor	>500	200	One company may have many entries
	DC Servo Motor	0	0	
■ Yahoo	DC Motor	3	3	
	Servo Motor	28	28	
	DC Servo Motor	81620	n/a	Web pages
■ InfoSeek	DC Motor	9	9	On-line info provided by Infoseek
	Servo Motor	6	6	
	DC Servo Motor	1	1	
■ Alta Vista	DC Motor	10000	n/a	Web pages
	Servo Motor	10000	n/a	
	DC Servo Motor	9000	n/a	
■ EDN	DC Motor	10	0	Only company contract info
	Servo Motor	14	0	
	DC Servo Motor	0	0	

Figure 1: Search results from existing catalogs.

where most of the results are irrelevant to the user goal. Given the size of the results, users often become frustrated facing the fact that they have to perform their own search with the given long list of items. In some other searches, the services produce zero result, wasting users' time and effort. In the cases of EDN and InfoSeek, the items are only a small part of all available information, and users lose the chance to examine other products that are not linked to the retrieved items.

In these catalogs, the information is indexed either by categories or product numbers. Especially, the web-based catalogs heavily rely on keywords and product numbers (e.g., national stock number in the case of PartNet) for searches. Given this limited access support, either users have to know the exact product number of the product in advance, or they have to try different (combinations of) keywords. Many times, the results are a large number of unwanted products. In fact, users often do not have the product numbers, and that may be what they want to learn. Also, in the keyword search case, if the service does not support the keywords specified by the user, the catalog cannot provide any information.

In fact, there exists a gap between the high-level, conceptual mental model of a product needed by a customer and the low-level, physical query that retrieves the needed information from a library. Too often, the customer is required mentally to map the description of a desirable component for solving a particular problem onto a set of searchable attributes recognizable by the library/catalog. That is, the user needs essentially to understand the "schema" of the libraries and the searchable technical attributes of the library contents before mapping the need into a set of queries. This mapping is not trivial and imposes a major mental burden on users, deflecting them from focusing directly on meeting their requirements. Using the information from the library/catalog is equally onerous and often involves manually transcribing the information into the local design tool environment.

Our catalog system, called *Active Catalog*, is an attempt to improve this situation a) by increasing the quality of the information search environment to give users better results and easier access to the library contents and b) by improving the means of consumption of the information by computers after it is retrieved.

In this article, we focus on improving information search environment. Our goal is to (1) deliver better quality results in the sense that all the retrieved items meet user requirements rather than yielding a lot of irrelevant results, and (2) retrieve all of the available information in the library that are closely related with the requirements. That is, we want to improve the correctness (hit ratio) and the completeness (coverage) of the search.

We approach these goals, done here in the context of electro-mechanical engineering parts/products, by addressing several fundamental HCI, cognitive issues. As noted, there can be a gap between the user's mental model of a part and the set of queries needed to access it. We humans cannot perform high quality reasoning and provide correct judgment if there exists no direct mapping between a real world object and our mental model of that object; and we are much less productive when we are diverted by too many distracting tasks; we cannot keep track of many threads or recall all what we know. Last, but not least, we cannot access material if we are not aware of its existence. We aim at providing technologies that will enable users to dedicate their mental resources to their own tasks at hand and are using a knowledge-based approach to do it.

The next section describes Active Catalog system as a whole. A rich set of models of different types in the retrieved product information allow a user to evaluate dynamic and behavioral aspects of the products in a system via interactive simulation built from or using down-loaded models, viewers or even applets of simulation code. The subsequent section describes our information search environment for improving correctness and completeness of the search results. The process will be discussed in terms of the servo-control domain. The key contribution of the work is to provide various ways of understanding user requirements and to exploit domain knowledge with powerful reasoning capabilities to extract all the products that are implied by the user input. Finally, related work and discussion are presented.

Background

Active Catalog is designed for both finding and using information in building applications in design, logistics, and electronic commerce in the context of heterogeneous,

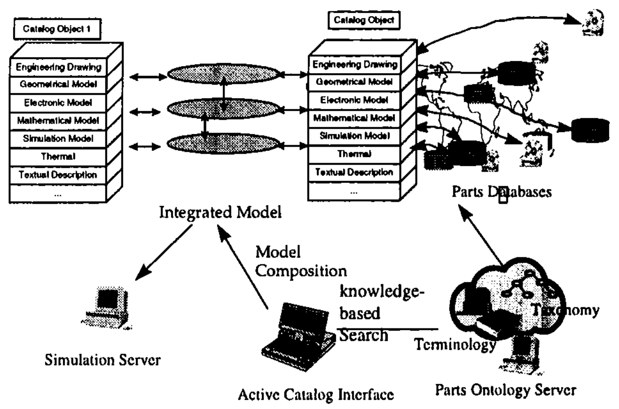


Figure 2: Active catalog architecture.

internet-based distributed computing environments. Its design requirements are driven by its vision i.e., finding and using information in building applications, and the nature of its computing environments in which the vision is to be achieved. We examine the design requirements of Active Catalog in the following paragraphs.

Information is active and dynamic: Information in Active Catalog is used not only for human understanding, but also for computer consumption. In particular, information is used by computers as building blocks in application developments. Active Catalog provides not only data, but active information such as programs, which can be directly executed on user machines. The information is also dynamic, as the program can process the data in user environments in response to users individual needs and current situations. A Java applet is one example of this active and dynamic information. However, Active Catalog goes further by providing information that can be used as building blocks in application developments.

Information is composable and can be integrated: Java applets are directly executable on user machines, but they are essentially restricted to run as stand-alone applications due to security concerns. Active Catalog is based on the assumption that information can be shared and used in a cooperative environment. Its objective is to support designers in their simulation analysis. Information is assumed to be down-loaded from a few but trustable sites, and designers can incorporate the down-loaded modules into their applications. Active Catalog builds on the existing strength of software reuse and object-oriented programming and uses ontologies to support semantic composition and integration.

Information is multi-dimensional and share-able in multiple domains: Active Catalog information is not just simply software components. It is software that represents physical parts and objects which have multiple views and interact with objects in other domains. For example, a motor is usually considered as a electro-mechanical device with torque and RPM (revolutions per minute). Torque and RPM are coupled and delivered to other devices, such as pumps and mechanisms. A motor has structural information, such as its components of stators and rotors. Its shaft diameter is a constraint on its connection to other devices. Finally, a motor has its geometric dimensions and weight, which can influence its physical placement. Active Catalog is to support reasoning about this multi-disciplinary behaviors, using ontologies and by mapping functions across ontologies.

Information is accessible syntactically and semantically: Currently, many search mechanisms, such as Yahoo and Lycos, exist to located web and HTML documents. Also many catalog web-sites allow users to perform relational database queries through their CGI interfaces. While these search mechanisms are useful, they rely on keyword search coupled with statistical correlation techniques to identify potential matches. Active Catalog makes use of these mechanisms. However, Active Catalog also provides semantic search capability by making use of ontologies and cross mappings between products and applications.

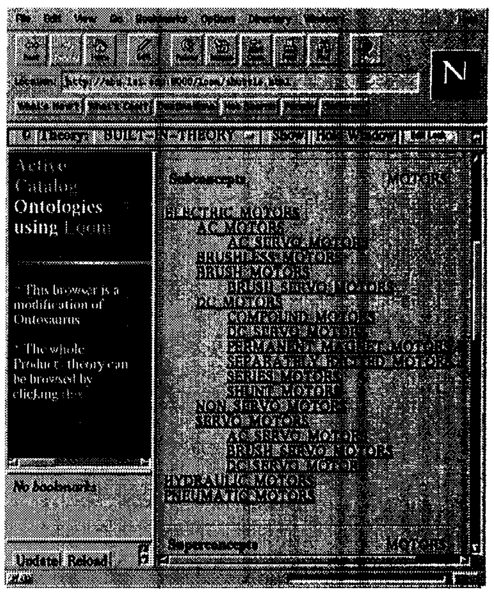
Information is interoperable and executable in distributed environments: Active Catalog information is used in heterogeneous and network-based distributed environments. Interoperation among software modules in such environments has been a major issue. Unix Sockets, RPC (Remote Procedural calls), DDE (Dynamic Data Exchange) are earlier work and OLE, CORBA and Java are current technologies to address this issue. Active Catalog uses a number of approach to handle the interoperation issue: by representing information in a platform independent fashion, and by using interface and wrapping agents for interoperating software modules. After an application is built, it can be also be executed on different machines.

The semantic network of Active Catalog is modeled in Loom (MacGregor 1990; 1994), a knowledge representation system language and environment developed at ISI for constructing intelligent applications. The Loom system provides deductive support for the declarative portion of the Loom language. Declarative knowledge in Loom consists of definitions, rules, facts, and default rules. The semantic model of servo control system consists of a set of definitions, rules and facts, matching Loom's power. To support building domain ontology and incremental development of a shared ontology, Active Catalog is using Ontosaurus(Swartout *et al.* Nov 1996), a web-based ontology editor for Loom.

Searching for Catalog Information

Increasing hit ratio

The *hit ratio* of retrieved information can be defined as the proportion of the useful items among the retrieved items. To increase the hit ratio of the retrieved information, i.e. to make most of the retrieved items meet the user goal, it is important to fully understand the user requirements. Active Catalog builds a set of ontologies and semantic network of products to support various ways of understanding user requirements, and searches are performed based on the understanding. Depending on the user preferences and the available input formats of the requirements, the user can pursue different ways (or combination of ways) of finding the products. Customers can retrieve product information



```
Subconcepts                         MOTORS
ELECTRIC MOTORS
    AC MOTORS
        AC SERVO MOTORS
    BRUSHLESS MOTORS
    BRUSH MOTORS
        BRUSH SERVO MOTORS
    DC MOTORS
        COMPOUND MOTORS
        DC SERVO MOTORS
        PERMANENT MAGNET MOTORS
        SEPARATELY EXCITED MOTORS
        SERIES MOTORS
        SHUNT MOTORS
    NON SERVO MOTORS
    SERVO MOTORS
        AC SERVO MOTORS
        BRUSH SERVO MOTORS
        DC SERVO MOTORS
HYDRAULIC MOTORS
PNEUMATIC MOTORS
Superconcepts                       MOTORS
```

* This browser is a modification of Ontosaurus

* The whole Product-theory can be browsed by clicking the

Figure 3: Taxonomy of motors.

by describing the applications and their contexts, or the functional descriptions with a set of attribute values, as well as product categories and product numbers. For example, the user can ask for *a motor that can be used for a servo control system with lowest price (in a search space) and weighs less than 1 pound.* Given the user requirement model, it is mapped to the real product instances using a powerful reasoning mechanism, based on the semantics of the domain.

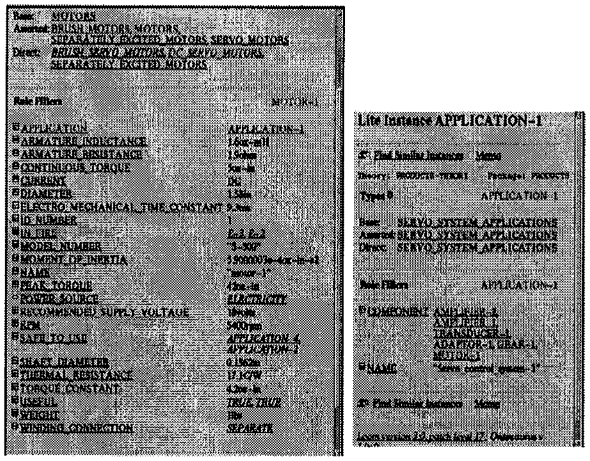
The tools for specifying the requirements should incorporate the mental model of the product users, so that the description can be correctly and easily captured in the requirement understanding phase. The user should not feel that describing the requirement using the tools imposes an extra effort. Also, the search mechanism should guide the user to describe the requirement properly.

The process of understanding the user goals should also capture the proper context or the application that the user is interested in. By filtering out irrelevant part in the information space and focusing on the context, the process can exploit the constraints implied by the context. For example, if an aircraft engineer needs to fix a motor, the system should rely on aircraft domain semantics. However, if he/she wants to fix a motor in his/her own washing machine, the system excludes parts for aircraft.

The key component in this framework is to structure knowledge in such a way that given product description, only essential search paths are explored, restricting unnecessary search effort as much as possible. Currently, we support five ways of specifying the user requirements to meet the goal.

1. *Browsing ontologies:*

   User can follow the taxonomies of product classes, attributes, and values to reach the products they want. For example, user can explore the taxonomies of motors, fol-



Base: MOTORS
Asserted: BRUSH_MOTORS, MOTORS, SEPARATELY_EXCITED_MOTORS, SERVO_MOTORS
Direct: BRUSH_SERVO_MOTORS, DC_SERVO_MOTORS, SEPARATELY_EXCITED_MOTORS

Role Fillers	MOTOR-1
APPLICATION	APPLICATION-1
ARMATURE_INDUCTANCE	1.6oz-mH
ARMATURE_RESISTANCE	1.9ohm
CONTINUOUS_TORQUE	5oz-in
CURRENT	DC
DIAMETER	1.5in
ELECTRO_MECHANICAL_TIME_CONSTANT	9.3ms
ID_NUMBER	1
IN_USE	E-3, E-2
MODEL_NUMBER	"S-300"
MOMENT_OF_INERTIA	5.9000007e-4oz-in-s2
NAME	"motor-1"
PEAK_TORQUE	43oz-in
POWER_SOURCE	ELECTRICITY
RECOMMENDED_SUPPLY_VOLTAGE	[illegible]
RPM	5400rpm
SAFE_TO_USE	APPLICATION-4, APPLICATION-1
SHAFT_DIAMETER	0.1562in
THERMAL_RESISTANCE	17.1C/W
TORQUE_CONSTANT	4.2oz-in
USEFUL	TRUE, TRUE
WEIGHT	1lb
WINDING_CONNECTION	SEPARATE

(a) Part to applications

Lite Instance APPLICATION-1

Find Similar Instances Menu

Theory: PRODUCTS-THEORY Package: PRODUCTS

Types APPLICATION-1

Base: SERVO_SYSTEM_APPLICATIONS
Asserted: SERVO_SYSTEM_APPLICATIONS
Direct: SERVO_SYSTEM_APPLICATIONS

Role Fillers	APPLICATION-1
COMPONENT	AMPLIFIER-2, AMPLIFIER-1, TRANSDUCER-1, ADAPTOR-1, GEAR-1, MOTOR-1
NAME	"Servo control system-1"

Find Similar Instances Menu

Loom version 2.0 patch level 17. Ontosaurus [illegible]

(b) Example application

Figure 4: A motor instance and its application.

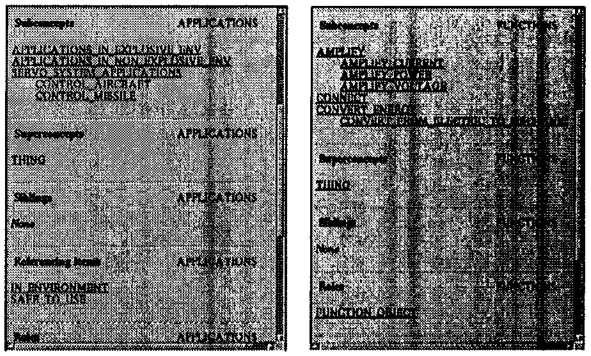
lowing the **isa** links and attribute links (such as size) in the ontology. The taxonomy is organized as a graph rather than a tree structure by taking advantage of Loom's multiple inheritance. At the top levels of the taxonomy are a few abstract type-definitions for motors, such as *Hydraulic Motors, Pneumatic_motors*, and *Electric_motors*, with *Brush_motors, Brushless_motors, AC_motors*, and *DC_motors* being subclasses of *Electric_motors*. In the middle levels are a rich set of abstract classes defining varieties of motor classes. At the bottom levels, a large number of motor classes are defined by combining types from top and middle levels. For example, *AC current, Brushless, Servo motors* are defined as a sub-class of *AC_motors, Brushless_motors*, and *Servo_motors*. Figure 3 is a small sample of the present motor taxonomy, displayed using Ontosaurus.

There are several benefits to making a taxonomy with a graphical structure. Since most bottom-level, specific motor types have multiple parents, the users of the taxonomy can reach the same target types via different paths. For example, a *AC current, Brushless, Servo motors* are reachable via any of (or combination of) the following motor types: *AC_motors, Brushless_motors, Servo_motors*, and all the motor types that are the parents (in the type hierarchy) of these three types.

Attributes describe properties of the objects in the semantic network. Also, a value constraint specifies the range of all possible values for an attribute. A unique feature of attributes and values is that, like motor types, they are organized hierarchically in the semantic network in order to facilitate search. For example, the value constraint for the attribute, *Power Source* of *Motors*, is *Power type*, which could be *Electricity, Water*, or *Gas*.

2. *Products to applications:*

   Given a product, a number of popular applications of the product in the past can be accessed. In Figure 4-(a), the motor instance has a pointer to an application, called



Subconcepts APPLICATIONS

APPLICATIONS_IN_EXPLOSIVE_ENV
APPLICATIONS_IN_NON_EXPLOSIVE_ENV
SERVO_SYSTEM_APPLICATIONS
    CONTROL_AIRCRAFT
    CONTROL_MISSILE

Superconcepts APPLICATIONS

THING

Siblings APPLICATIONS

None

Referencing Items APPLICATIONS

IN_ENVIRONMENT
SAFE_TO_USE

Roles APPLICATIONS

(a) Ontology of applications

Subconcepts FUNCTIONS

AMPLIFY
    AMPLIFY_CURRENT
    AMPLIFY_POWER
    AMPLIFY_VOLTAGE
CONNECT
CONVERT_ENERGY
    CONVERT_FROM_ELECTRIC_TO_[illegible]

Superconcepts FUNCTIONS

THING

Siblings FUNCTIONS

None

Roles FUNCTIONS

FUNCTION_OBJECT

(b) Ontology of functions

Figure 5: Example ontologies for applications and functions.

   application-1. The applications of the product describes a set of domain contexts in which the part can be used. They include details of the application environment, how the part fits into the application, and a set of pointers to other components participated in the applications. Figure 4-(b) shows the components of *application-1* and the pointers to the instances. Also, an application can describe various characteristics of the composition of the components and their usage. For example, a servo motor can be used for building a servo control system in a specific environment, and it needs a compatible adaptor, a transducer, etc. An application can point to other similar applications as references.

3. *Applications (or functions) to products:*

   We provide a semantic network of application, intertwined with the product taxonomy. A set of applications can be reached from products as explained above. Also, given the product requirements, user can follow the thread of application context and its related specifications to find parts for building a system and its sub-systems. For example, as shown in Figure 5-(a), an application ontology can have a set of applications including *servo system applications* which has *control aircraft* for gimbal control applications and *control missile* for missile servo control as its sub-classes. The semantic network is built based on the engineer's typical requests and search patterns.

   Users can follow the network of functional description of the products. The functional ontology classifies the products in terms of their generic functions. As shown in Figure 5-(b), *amplify power, amplify voltage, and amplify current* are subclasses of class *amplify*, and these classes can be used for finding different kinds of amplifiers.

4. *Using query template:*

   Parts (or sub-assemblies of an application) can be retrieved by selecting a set of constraints given by the query template. A template consists of a list of attributes for a class of products. Users can fill the template by specifying value constraints for all or a subset of the attributes.

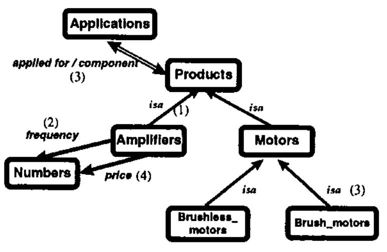


Figure 6: An example of semantic network.

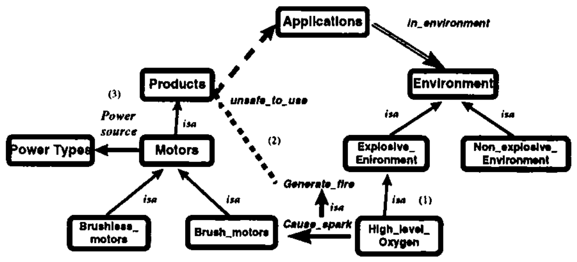


Figure 7: Semantic network with implication rules.

For example, in the context of finding motors, user can search for *AC motors whose peak torque is greater and 50 oz-in, weight is less than 2 pound, recommended supply voltage is 18, and the manufacturer is Company-x.* This is converted into a Loom query to retrieved such motors.

5. *Combination of the descriptions:*

The above set of access paths can be progressively combined for arbitrary access to the system. This relies on Loom's reasoning capabilities. A set of implication rules, taxonomies, and semantic network can all be used for the queries. For instance, the system can find *a high frequency (> 30) amplifier for brush servo motors with the lowest price* using the semantic network shown in Figure 6. The search can be performed in multiple ways. The user can begin the search by following the product taxonomy to reach amplifiers (1), and then specifying the required frequency (greater than 30) (2). Given the amplifiers satisfying the property, their application as a component of a servo system with brush motors can be specified (3). Finally, by computing the minimum of the prices of the intermediate results, the amplifier meeting all the requirements can be retrieved (4). The same result can be produced via different search paths. The query template given to the user at a moment should reflect the fact that the scope of the search is limited to the given class. Thus, the valid queries will be limited to be the ones consistent with the class of products.

Increasing coverage

The *coverage* of retrieved information can be defined as the proportion of the useful items in the retrieved information among all the useful items in the domain. The first requirement for increasing coverage is to populate the catalog system with all the useful products in the domain, so the library itself is a complete repository of products with richness. Second, the search needs to find all the related parts meeting the user specification by extracting all the products that are implied but not necessarily explicitly specified in the user input. To support the first requirement for a given domain, we can build comprehensive ontologies for the domain. For example, our current pump ontology framework has 300 classes and 3000 attributes. We are working on populating the ontology with servo control instances, collaborating with domain engineers. We are also working toward constructing an information architecture that provide tools for the manufacturers and knowledge engineers to edit the catalog information easily and to build Active Catalogs automatically.

To ensure that all the products closely related with user requirements are retrieved, ontologies should be organized and indexed so that all the implied relations can be exploited in the search. That is, the results should include the transitive closure of the product description. This can be achieved by adding a set of rules and constraints for the implied relations, and letting the search engine reason based on these rules and constraints.

Active Catalog uses Loom's representation and inference mechanism, suited for creating explicit semantic relationships and deriving facts that exist implicitly in the knowledge base. Thus, all the instance meeting the requirements (or closely related with the goal) can be retrieved using the deductive engine. For example, when the user needs *electric motors for high oxygen level environment,* it should retrieve all the instances that satisfy that property but not necessarily explicitly meet the specified properties. That is, using the semantic network shown in Figure 7, it retrieves all *brushless motors.* The inference rules/constraints used are: (1) *high level oxygen environment is explosive*; (2) *brush incurs spark in the explosive environment and it is not safe;* and (3) *the power source of brushless motors is electricity.*

Related Work

Ontolingua, the Ontology project at Stanford (Grunber 1992; Farquhar *et al.* 1995) provides a framework for building ontologies, and unifying and disseminating models, as Ontosaurus does. The system is a knowledge-base independent language designed for ease of translation between different knowledge bases. It is not designed to reason about services, a key in dynamic access to information sources.

Stanford "how things work" project is developing a service to search for products by supplying a description of the function users would like the product to achieve.(Iwasaki *et al.* 1996) Our approach is more general than function-based search in that we support various ways of capturing user requirements including the functional description. By analyzing the user goals and mapping the semantics of the queries into the information in the library, and employing powerful reasoning based on domain knowledge, we extract most of the useful product information in the system.

A number of other researchers are exploring knowledge-based design environments. In particular, SHADE system (McGuire *et al.* 1993) uses an ontology based approach to facilitate collaboration between product development and integrated manufacturing. SHADE is based on Ontolingua while Active Catalog will be based on Ontosaurus. The differences between the two ontology servers apply in the same way to the differences between SHADE and Active Catalog. Also, Active Catalog provides more powerful mechanisms for information search and evaluation by providing various information access processes and simulations of many modalities, in addition to viewing texts and drawings.

Mediator (Gaines, Norrie, & Lapsley 1995) is a knowledge support system for managing activities in manufacturing process. Although it provides KL-One like inference services and powerful graphical capabilities, Mediator focuses on accessing remote files and applications rather than on helping users to access, evaluate, and consume information in distributed task environments.

Summary and Future Work

This research focused on improving the information search environment in catalog service by delivering better quality results and easier access to the library. In order to achieve this goal, we provide various ways of understanding user requirements and powerful reasoning capabilities to extract all the products that are implied by the user input. These are supported via a set of ontologies and the semantic model of the user's daily work and a structure that captures the nature of the domain knowledge. By providing a mapping between semantic models and information source models, Active Catalog can translate a semantic query into corresponding queries of physical data repositories.

We are building an information architecture for Active Catalogs, so that manufacturers can easily construct new catalogs using the framework provided by the architecture. The databases that store up-to-date the vendor information in the Internet can be "wrapped" using the architecture services to support semantic queries.

Once the satisfactory products are found by Active Catalog, they can be "bought". Currently we plan to integrate with FAST, an electronic commerce system operating at ISI to implement real purchases.

Acknowledgment

We wish to thank Ping Luo for his earlier work in Active Catalog. We also would like to thank Ramesh Patil and Tom Russ for their Ontosaurus development and support. This work was supported by DARPA under contract J-FBI-95-159. The contents represent the opinions of the authors, and do not reflect official positions of DARPA or any other government agency.

References

Farquhar, A.; Fikes, R.; Pratt, W.; and Rice, J. 1995. Collaborative ontology construction for information integration. Technical Report KSL-95-63, Knowledge System Laboratory, Stanford University.

FAST. FAST. http://info.broker.isi.edu/1/fast.

Gaines, B. R.; Norrie, D. H.; and Lapsley, A. Z. 1995. Mediator: an intelligent information system supporting the virtual manufacturing enterprise. In *Proceedings of 1995 IEEE International Conference on Systems, Man and Cybernetics*, 964–969.

Grunber, T. 1992. Ontolingua: A mechanism to support portable ontologies. Technical Report KSL-91-66, Knowledge System Laboratory, Stanford University.

IndustryNet. Industry Net. http://www.industry.net.

Iwasaki, Y.; Fikes, R.; Farquhar, A.; and Engelmore, R. 1996. Function-based engineering part retrieval.

MacGregor, R. 1990. *LOOM User Manual.* USC/Information Sciences Institute. Working Paper ISI/WP-22.

MacGregor, R. 1994. A description classifier for the predicate calculus. In *Proceedings of the Twelfth National Conference on Artificial Intelligence.*

McGuire, J. G.; Kuokka, D. R.; Weber, J. C.; Tenenbaum, J. M.; Gruber, T. T.; and Olsen, G. R. 1993. SHADE: Technology for knowledge-based collaboration engineering. *Journal of Concurrent Engineering: Applications and Research (CERA)* 1(2).

PartNet. Part Net. http://www.part.net.

Swartout, W. R.; Patil, R.; Knight, K.; ; and Russ, T. Nov. 1996. Toward distributed use of large-scale ontologies. In *Proceedings of the Banff Knowledge Acquisition Workshop.*