A DESIGN FOR A REUSABLE ADA LIBRARY

John D. Litke
Grumman Data Systems Corporation
1000 Woodbury Road
Woodbury, New York 11797

ABSTRACT

A goal of the Ada language standardization effort is to promote reuse of software, implying the existence of substantial software libraries and the storage/retrieval mechanisms to support them. We propose a searching/cataloguing mechanism that permits full or partial distribution of the database, adapts to a variety of searching mechanisms, permits a changing taxonomy with minimal disruption, and minimizes the requirement for specialized cataloguer/indexer skills. The important observation is that key words serve not only as an indexing mechanism, but also as an identification mechanism, especially via concatenation and as support for a searching mechanism. By deliberately separating these multiple uses, we achieve the modifiability and ease of growth that current libraries require.

Extensive reuse of software is a goal that industry has found difficult to reach. Among the many issues to be solved before extensive reuse is a reality is the design and construction of a software part storage facility. The requirements for such a system exceed those normally found in a conventional software management system and thus demand a new design approach. This paper proposes a new design for a critical component of a software library, the searching and cataloguing mechanism.

All libraries have a common set of functions to perform. Some well known examples are storage of information, accession, discard of materials, and searching and retrieval

of information. Most of these functions for a computerized software parts library can use known computer science or library science methods. However, the function of searching and retrieval has proven difficult and it is in this area that we propose a new design.

As in any library, the function of a searching mechanism is to retrieve any information relevant to a query, not just a precisely specified fact. This means that context and meaning are important elements in the query interpretation process. Furthermore, the information in a library is not static, but rapidly changing both in scope and in terminology used to describe the relevant topics.

A useful searching/retrieval mechanism for a library of software would support inquiry from at least three different points of view. First, it should support the more conventional inquiry by keyword matching to select items by language, machine, author, etc. Second, it should allow searching by conventional topical descriptions in a variety of areas, such as aeronautics, electrical engineering, etc. Third, it should allow searching by algorithmic content, rather than by intended function. For example, we should be able to find an algorithm both by the topical "edge enhancement techniques", and the algorithmic, "fast fourier transforms." If there were a universal topical and algorithmic specification nomenclature, we would need an elaborate, but conventional index. However, all engineering fields have different topical indexing styles that do not map one to one to each other. Further, their algorithmic nomenclature is also not congruent to the nomenclature of other fields. Hence, a complete indexing/classification scheme must contain synonyms, see alsos, analogous references, etc.

If the resulting classification system were stable, the problem is complex enough. Furthermore, not only is the conventional taxonomy unclear and dependent on the engineering field of the user, but also the information that we are indexing is unformatted with an uncontrolled vocabulary.

A superficial solution to this searching and retrieval problem is to propose a computer based database system, complete with proven query languages and report writers.

However, the problem is not one of retrieval of information by unique key word, but retrieval of relevant information given a set of partially applicable key words or phrases. The differences are so substantial that a distinct field of information retrieval has evolved to address the issues as distinct from database technology. A review of the current problems in the field will illuminate the nature of the problem and the import of our proposed solution.

Current research in this field of information retrieval can be summarized in four categories (BART85):

Automatic Indexing
Information Structures
Query Formulation
Query Evaluation.

The workers in the area of automatic indexing are trying to find a way to convert unformatted information with an unstable vocabulary into a formatted, stable vocabulary that can be served with conventional indexing systems of the database community (ABBE75). The heart of the problem is to determine the content of a document by analysis of its text. This problem is analogous to the problems faced by the automatic language translation efforts and has proven very difficult.

The information structures area is trying to determine ways to represent the information contained in documents and the ways that documents relate to one another. There are three principal approaches. The first seeks to develop useful and comprehensive classification mechanisms to apply to all documents. This approach is the established, classic approach of libraries. The second seeks to develop thesauri that capture not only meaning, but relationships in a limited vocabulary of terms. The principal difference of a thesaurus and a classification system is the inclusion of relationships into the scheme, such as "part of", or the narrower/broader relationship of classes, and other ordering relations. Finally, a system that emphasizes relationships will take on the appearance of a semantic net or other network structures. At this end of the spectrum, more meaning is in the relationship between terms, and so the problem of synonyms and partially related terms becomes important.

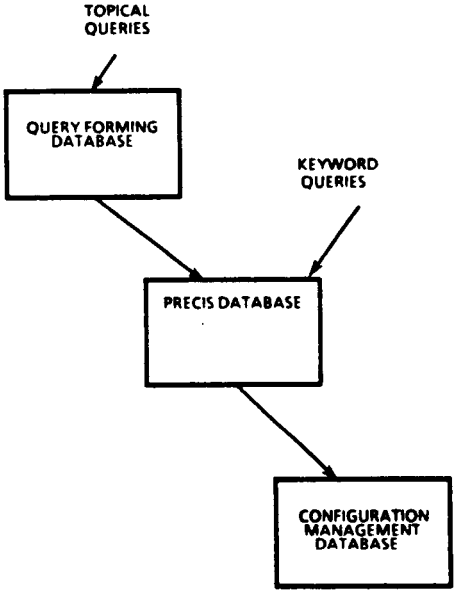
The query formulation area is not independent of the query evaluation area, although it does contain an important component of human interface in addition to the selection logic mechanism. To date there has not been a great deal of success with either boolean logic based systems that allow the formation of "this and that but not the other thing" type of queries, or with combinatoric systems that allow "similar to this or that" type of queries. There are two major difficulties with the boolean logic approach. First, the inquirer must know the allowable vocabulary and indexing scheme so that the query will be well formed. For large, complex systems like a library, this is unlikely for a casual inquirer. Second, boolean logic makes it difficult to specify major/minor selection criteria (BART85), (CROF81), (BOOK85). The combinatoric approach cannot represent boolean constraints easily, often leading to poor selectivity, and difficulty indexing uncertain information.

PROPOSED SOLUTION

The software library searching and retrieval design has several key constraints. First, the automatic classification of entries is beyond a commercially viable solution at present. Since the library will become large, we must minimize the classification effort required until automated solutions are available. Second, the information structures that are desirable are not yet known. Because a library is intended to have a long life, we must minimize the impact of modifications to the information structure or the classification/retrieval structure. Further, it is desirable to distribute widely the classification/retrieval mechanism to the user community to encourage a uniform approach to nomenclature/classification. To make such a distribution practical, the classification mechanism must be separable from the actual documents that it classifies. Finally, a mechanism must be found to classify the subject documents without extensive services from a professional indexer. Since automatic classification systems are still experimental, this implies a reliance on the help of the submitter/author, a heretofore not totally satisfactory source of indexing information. The solution we propose is not a final solution, but rather an architecture that will serve the present and grow gracefully into the future.

Our solution to this problem uses three key ideas supported by three interrelated databases. First, a precis is required from all submitters to minimize the indexing work of the library staff. Second, the classifications required of the author by the precis are guided by a database system supplied by the library to ensure that classifications are reasonable and correct. Furthermore, classification cross references and see alsos are maintained by the library staff, and are not expected of the author. Third, the searching mechanism is separated from the storage/retrieval mechanism to make future modifications to either easier.

The largest of the three interrelated databases is a conventional configuration management system and holds the full text files of all information in the library. (See Figure 1.) Other than indices required by the configuration management function, items are identified by a unique accession number.



Architecture of the Library System

Figure 1.

The second database holds a precis of each logically unique cluster of documents in the library. This precis requires certain information such as author, language, etc. It serves as a brief reference document to a collection of related documents that are logically indexed as one. Such a collection might include the specification, functional design, code, test driver, and test data for a particular module. The precis represents the finest level of granularity that a user can see from the indexing system. Note that the configuration management system is free to control the configuration of such a collection either as a unit, or as separate documents. The design of a precis itself is discussed separately below. The second database is organized conventionally in either a hierarchical or relational model, supporting conventional keyword query mechanisms.

The third database contains the indexing/classification system and serves as a query front end to the precis database. This database supports complex query forms on the classification system, not on the documents that are classified, and enables the user to easily construct complex queries for eventual submission to the precis database. This separation of the querying process from the underlying data containing system will allow change in either mechanism without rewriting the other. It allows users that are satisfied with conventional keyword access to use a conventional system, while allowing independent development of more elaborate searching/retrieval mechanisms.

There are three important properties of this solution. First, separation of the databases allows smaller, indexing databases to be distributed widely without the high cost of distributing large quantities of data in the configuration management database. Second, the version control and dependency relationships are removed from the searching/indexing mechanism to simplify the design and maintenance of the databases. Third, the topical relationships are separated from the precis information. This topical information is stored in a separate, logical tree organization so that variable depth indexing and class/subclass relationships can be maintained with a fixed record and fixed key size indexing scheme. (This was the same problem that led the GRIPHOS database designers to separate the indexing and retrieval mechanism in a similar way.)

This design rests on a plausible, but untested assumptions that the data to be classified is regular enough that:

1. Typical relationships between indexable topical terms are sufficiently generic that the relationships and the data are separable without introducing excessive retrieval error.

2. Classification can be done sufficiently well that the number of false identifications is tolerable. This implies that classification with a fixed vocabulary is possible.

3. The subject matter is sufficiently regular that classifications can be usually identified with a pre-existing taxonomy and that separable relationships are sufficiently generic that significant classes of objects are formed.

A small database is being constructed to test the validity of these assumptions.

PRECIS DATABASE

One of the difficulties with conventional keyword approaches to database access is that the keyword vocabulary must be controlled to eliminate misspellings, synonyms, etc. Further, the indexing terms are not readily expanded or changed without structural database changes. For these reasons, the precis database will be indexed with commonly agreed keys that have a finite, known range and represent quantifiable characteristics. For example, we will index on module name, version, author, machine, language, operating system, etc. This information will be obtained from each submitter to the database by requiring the submission of summary information according to a specified style. From this summary information, a trained indexer will prepare a precis with a controlled vocabulary for inclusion in the precis database, and the original summary information will be retained in the configuration managed database.

To form a connection between the precis database and the query forming database, the author must supply up to five classification codes. These codes will be selected by using the query forming database itself to explore the classification space and will be critiqued by a professional indexer for suitability. The classification codes need not be at a uniform depth of classification.

QUERY FORMING DATABASE

The query forming database bears the responsibility to support sophisticated inquiry into the precis database. There are two problems to be solved, while honoring several constraints. First, we must devise an indexing and classification system that will allow flexible searching of the precis database. Second, we must design an effective human interface to the query forming mechanism that does not require extensive knowledge of the underlying classification system.

To be an effective tool, the solution must also honor several constraints. First, it must support multiple views of algorithm classification for a variety of engineering disciplines. (The notions of subset/superset classifications are especially varied among disciplines.) Second, it must allow retrieval at uniform levels of detail so that broader scope documents are not retrieved together with narrower scope documents unintentionally. Third, it must grow gracefully with minimal or no re-classification of existing software. Fourth, it must be easy to provide classification guidance to submitters of software so that they can provide effective aid to the professional indexers.

The solution that we propose establishes a relationship among classification systems in the query forming database and not among items in the precis database. Thus relationships among library modules are represented implicitly via the classification system and not via explicit links in the precis or configuration database. We require the explicit relationships between elements to be maintained by the management database, since such are more static, while the more dynamic classification relationships are maintained in the classification system itself.

To obtain a classification that is familiar and useful to a wide variety of engineers, we propose a hierarchy of overlapping classification systems, whose inter-relationships are determined and maintained by expert indexers. The top level classification system will use the taxonomy of Dissertation Abstracts for field identification. Under each field, the indexing scheme customary to that field will be used. For example, mechanical engineers would use the scheme of the Applied Mechanics Reviews, Computer Scientists would use Computing Reviews, etc. The depth of the indices will vary from field to field. Since some areas of knowledge such as algorithms are found very frequently in software systems, a finer classification resolution such as that provided by the CALGO system will be needed. It will be up to the professional indexing staff to provide cross references and see also relationships among and between the classifications. The only change that will be made to the adopted classification system is to apply a more uniform numbering system to adjacent levels. This allows the database retrieval mechanism to refer to any classification with a unique code.

This design allows multiple classification schemes to exist and be interrelated with no overt cooperation from authors. The scheme can be revised and extended with little or no alteration of existing software. Deleted classification terms are simply translated by automated means into designated alternate categories, while new terms require reclassification only incrementally. (The simultaneous deletion of an old classification and re-assignment to a variety of new ones would require some re-classification.)

The system allows a user to self classify a module in his/her own vocabulary, while automatically supplying the cross references to other vocabularies and fields. Since the classification database is physically separate from the precis database, users can choose to use printed versions of the classification schemes and more laboriously search the precis database by conventional keyword selections. Finally, since a mechanism is provided to search the classification system itself, the classification can be made much more elaborate and precise without making the searching job harder.

Since the whole intent of the query forming database is to enable the user to more readily search the library, the quality of the user interface will be highly dependent on good physical implementation as well as a good match to people's searching strategies. We envision that the user will be able to select categories and sub-categories as she navigates freely through the classification system, able at any time to retrieve some or all of the precis implicitly selected, refine the query by elaboration, deletion, or addition of additional keyword specific qualifiers (such as language) as required. A modern, menu driven system with mouse and optional text entry is a candidate interface paradigm.

Each entry in the classification database will contain a count of the number of entries in the precis database that are indexed by that classification. This will allow a user to sharpen the query procedure without necessarily accessing the large precis database. Items can also be classified to any depth that the author deems appropriate. Exhaustive precision in classification is not required.

REFERENCES

(BART85) Bartschi, Martin "An Overview of Information Retrieval Subjects," Computer, (May 1985) pp 67-84.

(CROF81) Croft, W.B. "Document Representation in Probablistic Models of Information Retrieval," J. ASIS, V 32, 1981, pp 451-457

(ABBE75) Abbey, Scott, "GRIPHOS as a Relational Database System," Dissertation 1975, State University of Stony Brook, Stony Brook, New York.

(BOOK85) Bookstein, Abraham "Implications of Boolean Structure for Probabilistic Retrieval" Proc 8th International ACM SIGIR Conf, Montreal, Canada, June 5-7, 1985 pp 1177.