Textbases

Integrating Access to Text and Structured Data

Text Is Real Data

Text Is Much More Prevalent Than Structured Data

Relational database management systems (RDBMSs) have received more than their share of attention over the past few years when you consider the fact that most business data aren't stored in *any* formal database, let alone a relational one. What we call "structured data"—the rows and columns of simple character strings, numeric data, and dates in an RDBMS—make up a small part of the information within any organization, probably less than 10 percent.

In fact, text, in the form of documents of one sort or another, is probably the most common data resource out there today. Think of all of the letters, meeting notes, reports, E-mail messages, news feeds, procedure manuals, and marketing materials, we deal with every day. There is a wealth of information in these documents that is every bit as valuable as the structured data in relational and other databases that business users access on a daily basis. Integrating access to text and structured data is a critical step in helping companies better leverage a wider range of internal data to improve decision-making and provide a competitive advantage.

The Nature of Text Data

Documents, or textual information, consist of simple character-string data in the ASCII sense, but they are much more complex than that in reality. First, characters in documents are aggregated into words, phrases, and sentences. All text also has a series of other attributes, including, but not limited to, format (font and size), structure (paragraphs and other styles such as title, heading, bullet, etc.), and meaning. Users obviously need ways to search text content beyond the simple character-string functions provided by DBMS products.

Text-Search Products Do Not Solve the Larger Information Access Problem

In the past, this need has been met by full-text-search products. Examples include Fulcrum's SearchServer, Open Text's TextSearch, and Verity's Topic. Designed to index the contents of a set of documents, these products allow the user to create queries on text and efficiently retrieve documents that most closely match the search criteria. As these products have matured, we have seen increasing levels of sophistication in indexing and retrieval techniques as well as performance improvements.

However, two major developments are forcing users to view text as part of a larger whole. First, the sheer volume of text, particularly text already available in electronic form (as on the World Wide Web, or WWW), is exploding. As the size of text that must be searched grows, scalability and performance become issues. Second, users need to be able to reuse text information and to display it with other forms of data. A good example here is a mail-order catalog, where text is combined with images of each product and with structured data, such as item numbers and

prices. Here, text is not an afterthought handled by a separate, special purpose system. Text is real data that need a real database manager with security, concurrency control, audit trails, backup and recovery, and other mainstream DBMS features.

DEFINITION OF TEXTBASE. This combination of DBMS and text-retrieval features and functions create what we might call a *textbase*. A textbase would go beyond text-search, which has the ability to define only a few structured data items. The issue of *managing* text data in a consistent, reliable manner and integrating it with structured relational data in a single storage medium would be the key differences between full-text search and textbase systems.

Here is a scenario that describes how a textbase might work. A Microsoft Word for Windows user selects the "Open" option on the File menu and highlights a document in a local directory. However, the word processor, instead of reading from a local hard drive, transparently issues a "select" statement against a table on the LAN to retrieve the document. The document is actually a column in a row, a set of rows in a table in the database, or a file accessible through a pointer in the database. The database thus becomes a complete repository of information for the organization, and the user can issue queries that access any data in the database. This illustrates the dramatic difference between individuals processing data locally and group collaboration and sharing. One thing a database does well is to enable and *manage* the sharing of information. Security, concurrency control, and transaction management all contribute to this.

Next Step: Extending the Solution to Other Data Types

These customer requirements also apply to other types of complex binary data (e.g., images), isochronous data (e.g., stream data such as video frames or audio that must be coordinated and delivered within a specific time period), and arrays (e.g., time series data). Our premise is that users need and want tools and services that integrate access to text, structured data, and other types of data. This requires the following:

- The ability to store all types of data in a single, logical "database"—structured data, text, images, video, audio, time series, and user-defined data types. This allows easier assessment of potential relationships among data.

- A single language, such as extended SQL, that can access all types of data. The goal is the ability to issue a single query that can express and combine search criteria across all data types and return a single integrated results set. Many current solutions to text-search require the user to issue two queries—one to retrieve text from a text source and another to retrieve structured data and other data types (such as images) if they are stored in a separate database. The client software is then responsible for combining the results.

- Full solutions that include server functionality, easy-to-use client tools (for query, reporting, and application development), and management tools.

The Need to Accommodate Future Requirements

It's unclear whether text-centric products can fulfill these requirements in the long run, especially the ability to accommodate new data types in the future. Two years ago, no one would have anticipated the need for a database to store Web pages in Hypertext Markup Language (HTML) format as a data type. Yet all of the RDBMS vendors plan to integrate their database servers with the Internet/WWW. Who knows what new data types the next wave of computing will bring? Selecting a specialized text solution or a generalized database solution obviously depends on whether they can meet these overall requirements.

In this article, we take a look at the current solutions for handling text as a data type and how each is evolving, the technology required to handle text as data, and the issues that future products will have to address to meet customer requirements.

Current Solutions for Managing Text as Structured Data

Many types of products address text and documents as data that must be accessed and managed. These products are: (1) Full-text search products, (2) extended RDBMSs, and (3) object-relational DBMSs. There are also specialized applications, such as document management systems and Lotus Notes, that integrate and build on text-search and database management technology. (See Illustration 1.) It's unclear which approach will provide the basis for a true textbase solution. However, it appears that relational database vendors are moving the quickest—and have the most money to invest—to integrate text and structured data in a uniformly accessible storage repository.

Full-Text Search Products Provide Limited Access to Non-Text Data

FULL-TEXT SEARCH PRODUCTS. Products from vendors such as Excalibur (which bought ConQuest), Fulcrum, Open Text, Personal Library Software (PLS), and Verity are all focused on full-text search capabilities. They handle text only, with limited ability to index and retrieve text based on structured data elements. They generally do a very good job of searching through text. Most are read-only, require the use of a proprietary application interface, and have limited distributed capabilities. Fulcrum has embraced extended SQL for generating queries. Some, such as Open Text, can search text based on its attributes and structure. Verity is also working in this area.

Many of these vendors are now leveraging their text expertise by turning their attention to the Internet and Web. Online users are faced with such an overwhelming barrage of information that many service providers are embedding text indexing and search capabilities in their Web servers to enable users to find what they want more efficiently. Verity, Open Text, and PLS are taking advantage of this opportunity to market their text-search engines as embeddable products. PLS, for example, provides the search engine for America OnLine.

Another approach is that of using agents and profiles for filtering text according to a particular user's requirements. Verity already ships agent support with its Topic Agent Server. A few vendors, such as Excalibur, have extended their products to handle data types other than text; Excalibur also has image capabilities. Some

 Reproduction in whole or in part is prohibited. For reprints, call (617) 742-5200.

vendors are pursuing partnerships with RDBMS vendors in an effort to become an integral technology here. Excalibur is working with Informix on both text and image. PLS is partnering with Illustra Information Technologies to develop a PLS Text DataBlade for the Illustra database server.

The Convergence of Text and Database Management

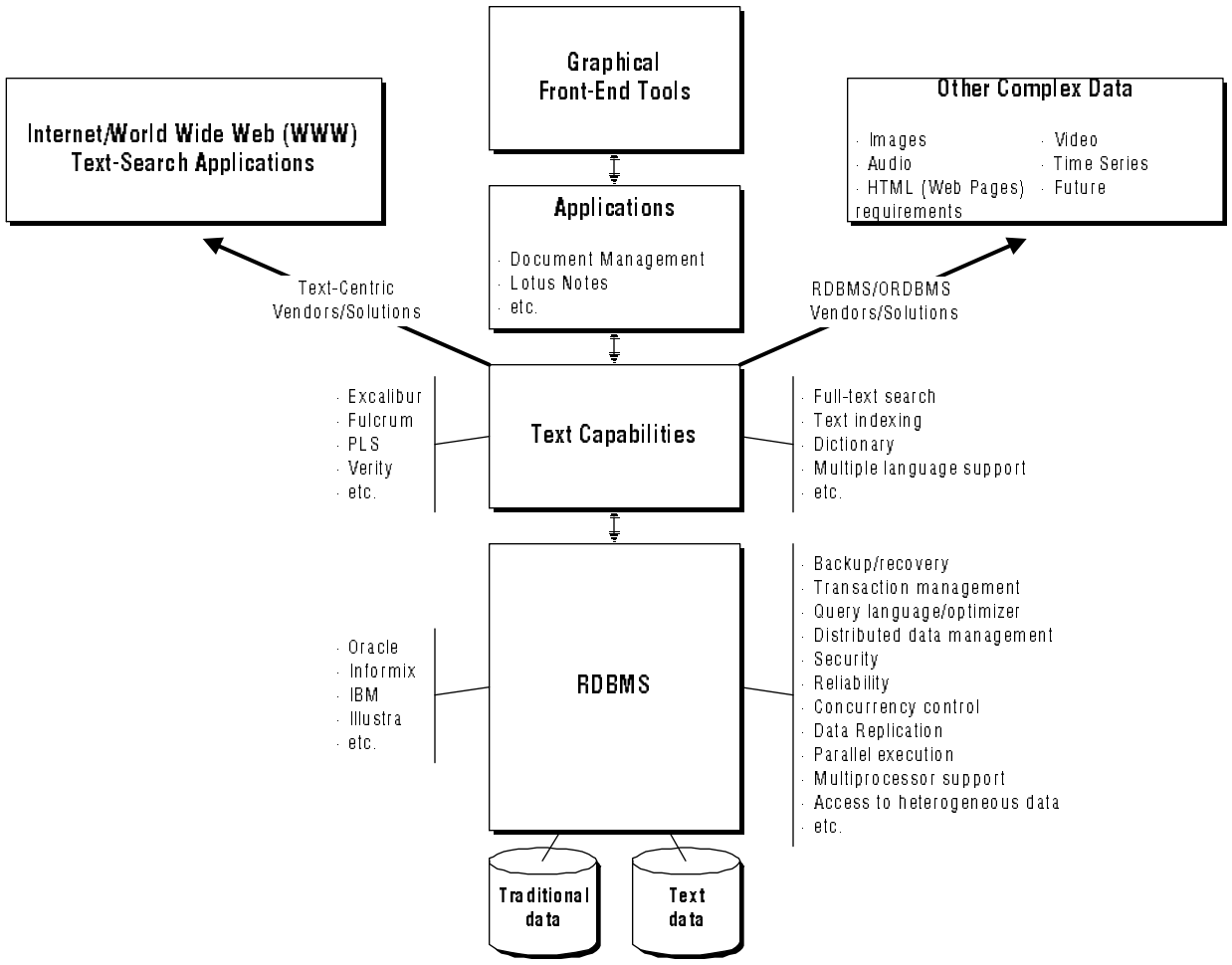

Illustration 1. Combining text-search with database management is the current state-of-the-art in textbases. However, text is only one of many types of complex data that must be addressed in the context of overall information access requirements. The key is an architecture that can be extended to gracefully accommodate unanticipated future data types. The difference between the full-text (text-centric) approach and the database management approach is the difference between specialized and generalized solutions to information access.

InMagic has taken the approach of combining some database-oriented features with text retrieval in its Windows-based DB/TextWorks product. DB/TextWorks adds

features such as a report-writer and the ability to store, display, zoom, rotate, and print images in up to 30 formats.

Document Management Usually Includes Full-Text Search

DOCUMENT MANAGEMENT SYSTEMS. Document management systems are applications that manage and track documents. The user typically assigns database field values (e.g., author, date created, key words) to a document. These fields represent a set of document properties that are then saved in a database, either a desktop database or an RDBMS. Document management usually doesn't turn the document itself into a data element, but creates a database record associated with the document. When retrieving a record, the system can also pull up the document in the form of a binary large object (BLOB, if stored in an RDBMS) or the associated file (if stored in a file system).

Full-text search is an option in most document management systems, with the ability to search on document properties or full text or both. Since full-text search adds overhead to the system, users may not want to include all documents in a full-text index. Some systems allow the user to specify certain properties that are used to determine which documents to index in full-text. Most embed a text-search engine from Verity, Fulcrum, Excalibur, or another third-party vendor.

In addition to retrieval by properties and full-text search, document management systems also provide document management and control features. These can include document check-in/check-out for concurrency control, version control, security, an organizational/storage metaphor (foldering, cabinets, etc.), document assembly, and library services.

Workflow, Agents, and the Web

Workflow is a separate technology that many document management systems are beginning to integrate, including the three top PC offerings: Saros, PC DOCS, and Soft Solutions. Many workflow products also use an RDBMS such as Oracle, Sybase, or Informix under the covers. Open Text, for example, has merged with Odesta, which is integrating Open Text's text-search products with its Livelink document management and workflow capabilities.

Other directions for document management systems are agent technology and the ability to manage documents on the World Wide Web.

Lotus Notes Includes Verity's Text-Search Engine

LOTUS NOTES. Lotus Notes provides group collaboration software for sharing information in the form of documents. Although Notes is neither a document nor a database management system, many organizations are building sophisticated text-centric applications on the Notes platform. Notes stores document information in the form of a set of fields. Each field can be structured (date, name, title, etc.) or unstructured. Unstructured fields are further defined as *text* (with minimal formatting) or *rich text* (e.g., colors, fonts, embedded tables, embedded audio or video).

Notes also contains Verity's full-text search engine. In Notes, each database must be explicitly designated for full-text indexing. This is comparable to indexing at the

 Reproduction in whole or in part is prohibited. For reprints, call (617) 742-5200.

table level in an RDBMS. The full-text index is stored and queries are run where the database resides, which can be either on the client or on the server. (The index is not replicated with the database. It must be indexed locally, or the user must connect to network and run the query against the server where the database is.) Release 4 of Notes adds the ability to full-text search any text in OLE 2 objects embedded in a Notes rich-text document, including labels on illustrations.

Full-text searches can be run on a scheduled or command-driven basis, and they can be combined with a full-text agent for automatic processing after the documents are retrieved. The user can specify a wide range of actions, including the ability to run external programs. This aspect makes Notes' full-text search very powerful when used as part of a process. It can be used to filter documents or to take a set of documents and do something with them (e.g., list, forward, mark urgent, or print). One example is running a regular full-text search across customer call reports to get information on a competitor. Documents retrieved could then be put in a special database.

On the downside, the way Lotus has implemented full-text searching in Notes makes it harder to use than it needs to be. (And a good Notes application shouldn't have to be full-text-searched anyway. It should present the user with the right documents within a view.) For example, a Boolean operator can't be specified by a point-and-click; it must be entered manually. And the user must designate what databases are to be searched. If you don't know about a database, you can't search it. When searching across multiple databases, Notes returns the results out of context, sacrificing the Notes structure.

In summary, the Notes implementation of full-text search capability strongly supports business processes (with agents and workflow) and both local and remote use through replication. Its major limitation is its weakness as an ad hoc query platform. The user must know if a database exists and where it is in order to index or search it. People simply don't use the full-text search in Lotus very much. Its primary use is to index personal E-mail for searching.

Extending the RDBMS: A Generalized Solution

RDBMS EXTENSIONS. In the past, the RDBMS vendors have largely ignored all types of complex data. The best that most RDBMSs can offer today is a BLOB data type, usually up to 2GB in size. But the RDBMS is only educated to deal natively with alpha/numeric data types—fixed- and variable-length character, numeric, date and time data. It has little or no understanding of the internal structure of a BLOB and cannot interpret the contents. The only alternatives are to tag BLOB data with keywords stored in other fields in the database (the same approach usually taken when storing such data in a file system) or leave processing of BLOB data, other than storage and retrieval, to the front-end application.

Lately, however, we find interesting developments coming from the RDBMS vendors. Many are extending the relational database server to better handle a broad range of complex data types, not just text. These extensions are not trivial, and they generally require a rearchitecting of major components within the RDBMS.

For example, IBM is beta-testing a series of DB2 Relational Extenders for Text, Image, Video, Audio, and Fingerprints. Informix Software is extending its Informix-OnLine data server using what it calls an Extensible Framework. Initial implementations that address text and image are already in controlled release, with general availability scheduled for mid-1996. Oracle's TextServer will become an integrated component of the Oracle7 RDBMS in Release 7.3, due out in Q2 1996. Sybase is one major player without a stated strategy in this area. The company brought in Fulcrum's full-text product as a partner but offers no common development or administrative tools.

RDBMS directions include extensibility in the database server itself for new data types, new functions, new access methods, and new search techniques, with SQL extensions to take advantage of all this. We are early in the development cycle here. Version 2 of IBM's DB2 common server, for example, has user-defined data types, but they are limited to data types that extend base data types. Full user-defined abstract data types will come in a future release. DB2 does not have user-defined access methods yet, either. In the case of Informix and Oracle, the user will be initially restricted to extended facilities provided by the database vendor.

Extended RDBMSs from established vendors have the opportunity to become a key foundation for integrated information access across large organizations. And the database products that are open to multiple third-party technologies will provide the highest level of flexibility for the user.

It should be noted that Microsoft is taking a very different approach to support for nontraditional data than the other RDBMS vendors. One reason is the fact that Microsoft also owns the operating system on which its database products run. Microsoft plans to embed the interfaces to all data types within its future object-oriented Cairo file system. Using OS facilities rather than building a separate architecture within the DBMS eliminates the need to "reinvent the wheel." It makes the interfaces more broadly available to applications and services in addition to database management systems. We expect access to extended data (file) types to surface sometime in 1996 in the form of OLE DB, a new set of OLE interfaces for data access. The questions with OLE DB are whether Microsoft will deliver the technology on schedule and whether OLE DB will perform adequately to meet user requirements.

Object-Relational Is Built from the Ground up for Complex Data

OBJECT-RELATIONAL DATABASE MANAGERS. Object-relational database management systems (ORDBMSs)—as opposed to pure object databases—are a relatively new approach to the problem of effectively handling complex data in a database. ORDBMSs such as Illustra from Illustra Information Technologies purport to offer the best of both the relational and object-oriented worlds. These products build on the relational foundation of good security, integrity, reliability, and SQL support while adding object-oriented extensibility to handle complex data. This evolutionary approach leverages existing skills and knowledge gained in the relational world while supporting the development of new kinds of applications. (For more details on

 Reproduction in whole or in part is prohibited. For reprints, call (617) 742-5200.

Current Solutions for Managing Text as Structured Data

ORDBMSs, refer to the article on "Object-Relational Database Managers" in *Distributed Computing Monitor*, Vol. 10, No. 2, February 1995.)

The ORDBMS products have a window of opportunity to provide an alternative to the RDBMS for handling complex data while the RDBMS vendors struggle to catch up in this area. The challenges for ORDBMSs are that they are sold by small companies, they need to coexist with relational databases given its installed base, and they need to implement robust DBMS functionality such as multiprocessor support and parallel processing, distributed data management, and sophisticated management and monitoring tools.

Text Has Specific Technology Requirements

Any good solution to handling text data must include the requisite text-search capabilities. Here is a brief overview of major features by which to evaluate products. (See Illustration 2.)

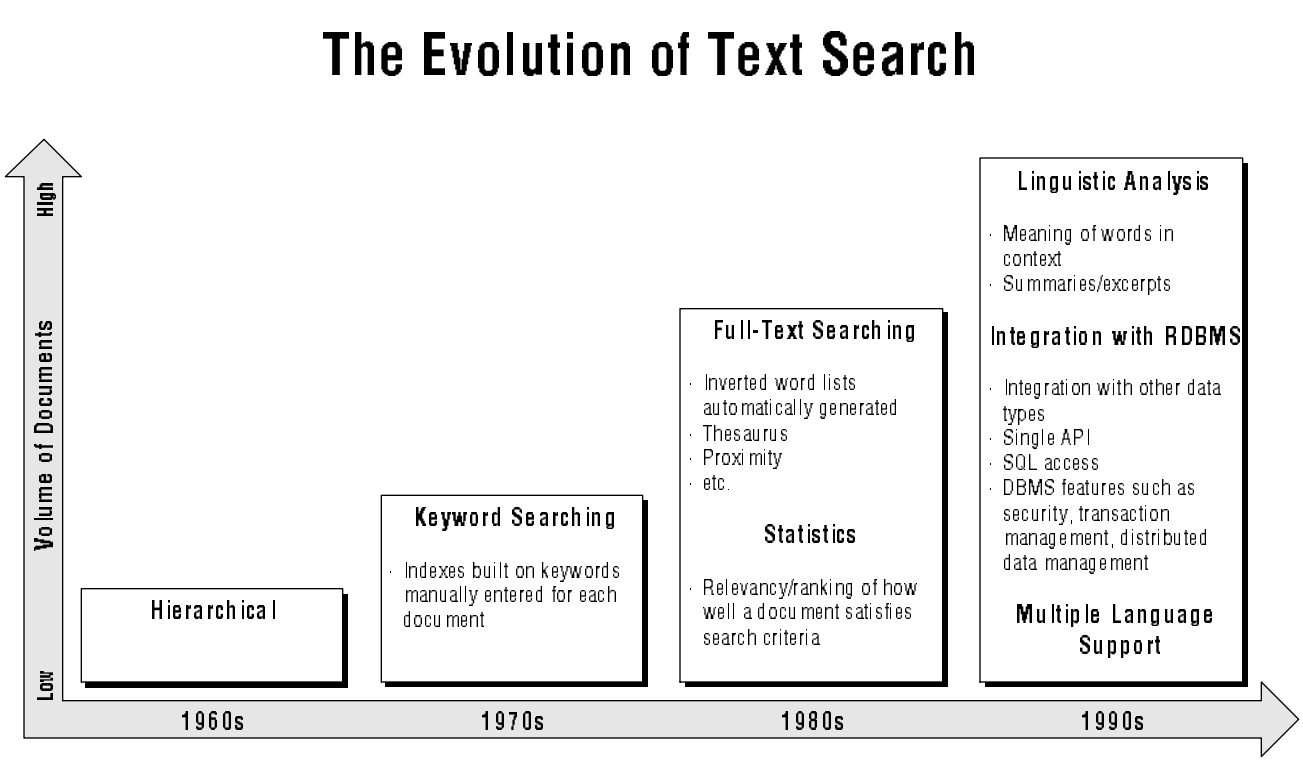

Illustration 2. Text-search capabilities have evolved significantly over the past three decades.

Full-Text Search Is Core Technology

FULL-TEXT SEARCH. The primary focus of efforts to address text retrieval requirements have been full-text search engines. These retrieve documents based on words, phrases, or concepts contained in the documents, and they have been around

for a long time. Indexing techniques and the ability to determine the relevancy of text according to a user's query have evolved dramatically. The key objective is to help the user find *all* of the relevant documents and *only* the relevant documents. The user doesn't want to miss important documents nor retrieve irrelevant ones.

Word Indexing Techniques

In full-text searching, the user specifies one or more words that must be contained in a document for it to be included in the results set, or "hit list." Most full-text search engines use the same indexing technology, generating what is called an inverted word list for indexing purposes. Two concepts are important here. The first is a *stop list*, which identifies all words that are not indexed or searchable by the user. Examples of stop words are common, insignificant words such as *a*, *an*, *the*, and *to*. Stop words are not indexed and cannot be included in a search. The second concept is a *go list*, which includes all of the searchable words. A shortcoming for using stop words is the search engine's inability to find, for example, the phrases "association of" or "return on investment" in a document. A good text-search product should give the user the choice of whether or not to specify stop words.

The full set of searchable (indexed) words is collapsed into a set of unique words, and each of these is reduced to its root or stem word with location pointers to the appropriate documents. Thus, each word needs to be stored only once in the index along with the number of its occurrences and locations in each document. A million words might be reduced to 20,000 unique words, for example. A major issue is flexibility in maintaining indexes as documents are added, updated, or deleted.

Some products use other indexing techniques. Open Text stores words in the index as strings of bits rather than as individual words, an approach that facilitates support for foreign languages, especially double-byte character sets. IBM also uses a statistics-based approach to indexing. Oracle's TextServer uses bitmapped indexes for word locations, and its ConText technology provides linguistic analysis in addition to standard inverted-word indexing.

Search Techniques to Find and Rank All Relevant Text

Other important features of full-text search are aimed at finding all of the documents that might match the search criteria, beyond the individual words specified by the user.

- **Thesaurus.** The search engine uses an internal thesaurus to automatically invoke synonyms for search terms on behalf of the user. Thus, the user doesn't have to know or remember to specify all relevant related words in a query. For example, for a query including the word "car," the search engine might also look for "automobile" and "vehicle." A customizable thesaurus is valuable.

- **Wildcards.** Wildcards cover both truncated words (e.g., heli* would find helicopter, heliport, helium, etc.) and masks for specific characters (b?t finds bat, bet, and bit).

- **Proximity.** The user can search for words or phrases in proximity to each other. The given words must be adjacent or must occur within a specified number of words. One issue is whether proximity searches are uni- or bi-directional.

 Reproduction in whole or in part is prohibited. For reprints, call (617) 742-5200.

Text Has Specific Technology Requirements

- **Multiple Arguments.** The ability to specify multiple search arguments using Boolean *and*, *or*, and *not* operators and to nest these operators is important.

- **Phrase Search.** The user can specify a phrase for search purposes, such as "sale ends today." Use of stop words can limit the ability to search by phrase.

- **Fuzzy Search.** Fuzzy search techniques can be used to take into account misspellings or OCR errors in documents. For example, a fuzzy search on the word "relational" should find "relatonal" as well. Note that fuzzy searching tends to expand the number of retrieved documents rather than reduce it.

- **Search on Structure.** The user can search for words or phrases in a particular structural component of a document, such as a search for "Washington" in a headline or "Fowles" in the bibliography. Open Text's TextSearch, for example, understands the structure of SGML and HTML documents.

- **Query-by-Example.** Some text-search products, such as Fulcrum's SearchServer, allow the user to take a retrieved document and use it as an example for further refining a search (e.g., "find more documents like this one").

- **Placing Search Results in Context.** Full-text-search products usually rank and display retrieved documents in descending order according to their degree of relevancy to the query criteria. Each product uses its own algorithms for assessing relevancy based on word counts, frequency, and other criteria. Another aspect of document context is the ability to group, or cluster, documents by subject. Verity recently introduced support for clustering.

Flexibility of Deployment

Text-search products have to be flexible in order to appeal to a wide variety of user organizations:

- **Multiple Language Support.** Some products, such as Fulcrum's SearchServer, Open Text's TextSearch, and Verity's Topic, support full-text search of documents in multiple languages. IBM's Text Extender for DB2 will support full-text search in at least 17 languages.

- **Multiple Text Formats.** Most text-search products support multiple document formats for both indexing and viewing. A key feature is to include text in all formats in a single, unified index.

- **Distributed Capabilities.** Because documents are often created locally on the desktop, there are many issues regarding sharing documents and making them searchable across the network. Does a copy of the document have to be moved to a separate location, or can the text-search product index and access the document where it resides? Can documents be indexed in native format? Can searches span multiple document databases? Open Text's Latitude product, for example, already supports distributed indexes; Verity will introduce this in early 1996.

Text Has Specific Technology Requirements

- **Support for Other Data Types.** Most products support structured fields, and some, such as DB/TextWorks, also provide access to multimedia data associated with a document (but not content-based retrieval of this multimedia data).

Understanding the Meaning of Text

LINGUISTIC ANALYSIS. Linguistic analysis, or natural language processing, of text involves actually parsing sentences to determine what text is about—its meaning—and what concepts are covered, not just what words are used. The same word can have a different meaning and a different level of importance depending on how and where it is used in a sentence or a document. Benefits of linguistic analysis include:

- **Natural Language Queries.** Natural-language queries allow users to generate more free-form queries and the text engine to interpret what the user wants. This provides flexibility and ease of use.

- **Document Abstracts.** Oracle's TextServer with ConText can automatically generate an abstract, or summary, of document content and has a "speedread" feature for quickly scanning reduced amounts of text. (See the section "Oracle's TextServer with ConText," page 19.) Verity recently added abstracting capability to Topic. Another example is Xsoft's Visual Recall 2.0. Visual Recall uses a number of criteria, including position of text in the document, weighting of word frequency, and elimination of short sentences, to identify the key points of a document. As in Oracle's ConText, key points can be presented both in context (i.e., highlighted in the document itself) or as bulleted items in a "key excerpt" list. Abstracts allow the user to decide whether or not to retrieve the entire document, saving network bandwidth.

Simplifying Application Development

APPLICATION INTERFACE. Most text-search products, including those of Oracle, Verity, and Excalibur, have a proprietary interface and query language for text retrieval. In contrast, Fulcrum, Informix, and IBM are all taking an extended SQL approach to accessing text. Oracle will move to a SQL interface in its next TextServer release.

What Can the User Do with Retrieved Documents?

POST-RETRIEVAL PROCESSING. An area that many text-search products have not explored adequately is the ability to *do* something with the documents (besides read them) once they have been retrieved. Examples are to update the contents of all the retrieved documents, to generate a report, or to fax or E-mail the documents. InMagic's DB/TextWorks, for example, includes a report writer. The addition of agents in text-search products such as Verity's Topic addresses this need as well. Agents provide real-time filtering of text based on user-defined profiles. Agents can also be used to automatically search text and execute user-defined processing on the retrieved documents.

How Accessible Is the Technology?

A key to the success of any technology is how easily it can be used and accessed. This means the availability of graphical client tools (ad hoc query and reporting, query-by-example, etc.), of management tools (Oracle, for example, has a set of graphical administrative utilities for its TextServer product), and of a diverse

 Reproduction in whole or in part is prohibited. For reprints, call (617) 742-5200.

portfolio of applications built on the text-search technology. Applications include document management, Web servers, and the like.

Integrating Text into an RDBMS

Combining text with structured data has its own set of architectural requirements. Extending any DBMS to deal effectively with complex objects is itself complex, involving new indexing and optimization methods, content-based retrieval, and functions to directly manipulate content. Just as the RDBMS vendors have had to implement ways to access data stored in heterogeneous data managers, they now must develop ways to access what we might call "heterogeneous" data types. These are all extensions of the database management *service* to meet customer demands for integrated access to data that can be turned into useful information.

The "Complex Data Package"

The basics of building access to text and other complex data into a relational database management platform include extensibility in four critical areas. The key is an extensible architecture that permits the creation and manipulation of new (and often unanticipated) data types. Object-relational DBMSs such as Illustra are based on this model and already provide such extensibility. The four areas are:

- **Data Types.** RDBMSs have traditionally focused on relatively simple, scalar data types—characters, numbers, and dates. All major products also offer BLOB support up to 2GB but without much in the way of intelligent handling of BLOB contents. This "fixed" data-type model limits the user's ability to map business objects to database structures. The addition of data types that better represent both complex data (text, images, video, etc.) and collections of objects (e.g., arrays) plus the ability to create user-defined data types (UDTs) increase flexibility to meet new application requirements.

- **Functions.** The RDBMS must also include the functions, or methods, associated with new data types. A time-series data type might have several calendar functions associated with it, for example. These functions can be provided by the database vendor, third-party vendors, or the user (user-defined functions, or UDFs).

- **Indexing or Access Methods.** All the RDBMS products use B-trees for indexing traditional data. Some are adding other indexing techniques, such as bitmapped indexes, to speed access to data for certain types of queries. But none of the RDBMS vendors yet allow the incorporation of user-defined access methods within the database server to capture specialized search techniques. Illustra, on the other hand, supports indexing of multiple complex data types, and the company is working with partner PLS to integrate the PLS text indexing and search engine within Illustra as part of the PLS Text DataBlade.

- **Data Access Language.** Another level of integration is the ability to issue a single query that can include reference to any and all data types and functions. For the RDBMS and object-relational products, this is an extension of SQL, not

a new, separate interface aimed at a subset of data types. Tools for users that mask the need to write SQL statements are particularly important here.

Content-Based Retrieval

QUERY BY CONTENT. The RDBMS also has to understand and manipulate the contents of text and other complex data types. Relational generally provides a simple, one-dimensional, character-oriented way to index and search for data (e.g., =, <, <=, etc.). But the world of applications is changing, and users want the same text-search capabilities in the DBMS that they find in specialized text products. Users also want to do queries like "find all the pictures of houses that look like this" in a library of photographs or "find sound effects that sound like this" in a file of audio clips. We are beginning to see the RDBMS and ORDBMS vendors integrate these types of query-by-content facilities in text-search engines and other complex data facilities.

The Optimizer Is Key to Performance

OPTIMIZER. The importance of the optimizer in database performance cannot be overstated. With the addition of new data types, functions, and access methods, the optimizer itself must be extended to understand the relationships among them, the cost of executing UDFs, and how to take advantage of new access methods. This is coming in gradual steps. The first step is to allow the optimizer to continue using its existing, "regular" optimization techniques on queries that include UDFs. The second step is to provide special optimization intelligence. One example is the cost of a UDF. In IBM's DB2 Version 2.1, the user can specify several parameters for a UDF to tell the optimizer how much it costs to execute the UDF in both I/Os and CPU resources. Illustra also supports identifying UDF costs. With this information, the optimizer can make more intelligent decisions about how to evaluate a SQL statement. It may decide to evaluate other parts of the query first if a UDF is particularly expensive.

Integrating new access methods is more difficult. Illustra has gone the farthest here with its DataBlade architecture. Currently, IBM's DB2 server itself doesn't understand an index built by the Text Extender. But the DB2 optimizer does figure out when to invoke the Text Extender, which then uses its own techniques to find the requested text data efficiently.

Flexibility in Storage Options

DATA STORAGE. Today, text is likely to be created using a word processor and to be stored in a file system, not a database. The RDBMS must be flexible enough to allow text to be stored either in the database or in separate files with pointers to those files. Verity and Open Text can index documents at their source location; the documents do not need to be physically moved into a central database for access. In the case of Illustra and DB2, documents can be stored either in the database or in an external file. Informix will allow documents to be stored externally in the second half of 1996. Another feature important to storage of complex data is data compression.

There are also issues relevant to flexible memory management and transaction logging when large data objects are dealt with. Informix and IBM have done the best job so far in addressing these issues.

 Reproduction in whole or in part is prohibited. For reprints, call (617) 742-5200.

Manipulating Text

INSERTS AND UPDATES. It is also interesting to consider the possibilities offered by a DBMS for inserting and updating text. Think about the need to change a product name throughout a large set of documents describing the products offered by your company. Wouldn't it be convenient to issue a single update statement that replaces the old name with the new one everywhere in every document, respecting the *context* in which the text currently appears? To accomplish this, the language with which one describes these documents will have to evolve from many specific formats (Microsoft Word, Lotus AmiPro, etc.) into a standard way of representing text data. Standard Generalized Markup Language (SGML) is probably the most promising effort here. This is based on IBM's GML and is the origin of the Hypertext Markup Language (HTML) used on the Web. A standard format facilitates much more intelligent querying and updating of text. On the down side, SGML is difficult to use because it imposes on authors the burden of following what may be a complex structure and usually forces them to use an SGML editor instead of a familiar word processor.

Leveraging Parallel Processing

PERFORMANCE AND SCALABILITY. As users gather more and more data into data warehouses and online production systems, the issues of performance and scalability become more important.

One of the promises of RDBMS is vastly improved performance through parallel execution of database operations. The time to process a single query, for example, is reduced by having multiple threads and processors work on it in parallel on a symmetric multiprocessing (SMP) or massively parallel processing (MPP) platform. Currently, most database vendors have imposed restrictions on what can be parallelized because of the complexity inherent in parallel operations. Adding new layers of complexity with extended database objects and access methods will only exacerbate this situation. For example, will new search engines for text have internal parallelism? Will a text-search engine have to run on every node in an MPP environment? Informix, for example, supports parallel queries on text data on SMP platforms in the first release of its text-search extension to OnLine. Parallel text-search will be supported on MPP as well with OnLine 8.2 in late 1996.

Adding Navigation via Object IDs

It is also reasonable to expect that navigation through the database needs to go beyond the table joins that are so basic to the relational data model. Object-oriented and object-relational database management systems use what are called *object IDs* (OIDs) to facilitate navigational access in addition to value-based access to data. Each object has a unique OID, and navigation among objects is done via these OIDs. This offers a new way to index data and to efficiently access denormalized data. RDBMSs try to minimize duplication of data through normalization of tables, but there is still redundancy because primary keys themselves are repeated many times.

Informix will introduce OIDs for navigation among database objects in a future release. It will then be possible to have direct pointers to rows within other rows, eliminating the need to execute joins using indexes. In cases where OIDs are more efficient, the RDBMS won't have to do a join to find related data.

Full-Text Search Products

A Sampling of Products

Here are thumbnail sketches of the two text-search industry leaders—Verity and Fulcrum Technologies—plus an overview of Open Text, an interesting competitor. (For more detailed information on Verity and Fulcrum, see "Beyond Full-Text Search" in the *Workgroup Computing Report*, December, 1995, Vol. 18, No. 12.)

Verity: Put Topic Everywhere through OEM Partnerships

VERITY. As a long-time vendor of text-search technology, Verity's Topic is already embedded in many popular applications, including Lotus Notes, PC DOCS, and Netscape, with more coming all the time. Verity is concentrating on its OEM partnerships. Its goals are to make the Topic family of information retrieval and filtering tools ubiquitous in the marketplace and to have information in any text-based format indexed by the Topic engine and accessible by a single client interface. Verity's new product line adds servers for indexing Internet and CD-ROM text as groundwork for interoperability among different text collections in the organization. Another new feature is the ability to deploy agents that can watch information services and notify users when relevant information arrives. Topic can access data in an RDBMS with native drivers for Sybase and Oracle, with ODBC support coming in the next release. However, the user cannot yet access Topic-indexed text and RDBMS-based structured data in a single query unless all of the data are indexed by Topic. We also expect to see Verity develop a DataBlade for Illustra in the future.

Fulcrum: Standards, Integration with RDBMS, Intuitive Search Methods

FULCRUM. Fulcrum Technologies is focusing its information-retrieval efforts and its SearchServer product line in several areas. One is the need to bridge the worlds of structured and unstructured data. Fulcrum is doing this with support for industry standards such as SQL and ODBC and with tighter integration with the RDBMS, such as the ability to use SearchServer to index text in the RDBMS. Another thrust is improved natural-language methods of searching text. SearchServer already supports Intuitive Searching, the ability to base a query on interesting words, phrases, or paragraphs that the user highlights in a document already retrieved. The next version of SearchServer, coming in Q1 1996, will add other advanced query capabilities. Fulcrum recently added the requisite products to search text on the Internet and WWW as well.

Open Text: Focus on Performance

OPEN TEXT. Open Text provides some interesting features in its text database product line in addition to its string-bit indexing of all words for full-text search. The Open Text TextSearch engine supports multiple languages and includes a native SGML parser, enabling it to understand and search on structure in SGML-formatted documents (and HTML documents by extension). A document is indexed and searched in its native format; it is not translated. The company is focusing heavily on performance. TextSearch includes parallel execution capabilities and the ability to search across multiple distributed document-databases as if they were a single logical database. The company also claims to provide the same level of performance regardless of the length of the search criteria or the size of the results set.

Open Text also provides a free Web index service—the goal is to index the entire Web—and its technology is embedded in other products, such as Yahoo!'s

 Reproduction in whole or in part is prohibited. For reprints, call (617) 742-5200.

electronic guide to the Web and IBM's InfoMarket Search. (See Illustration 3.) Open Text uses the catchy phrase "client/surfer" for its Web software.

Open Text's Web Index

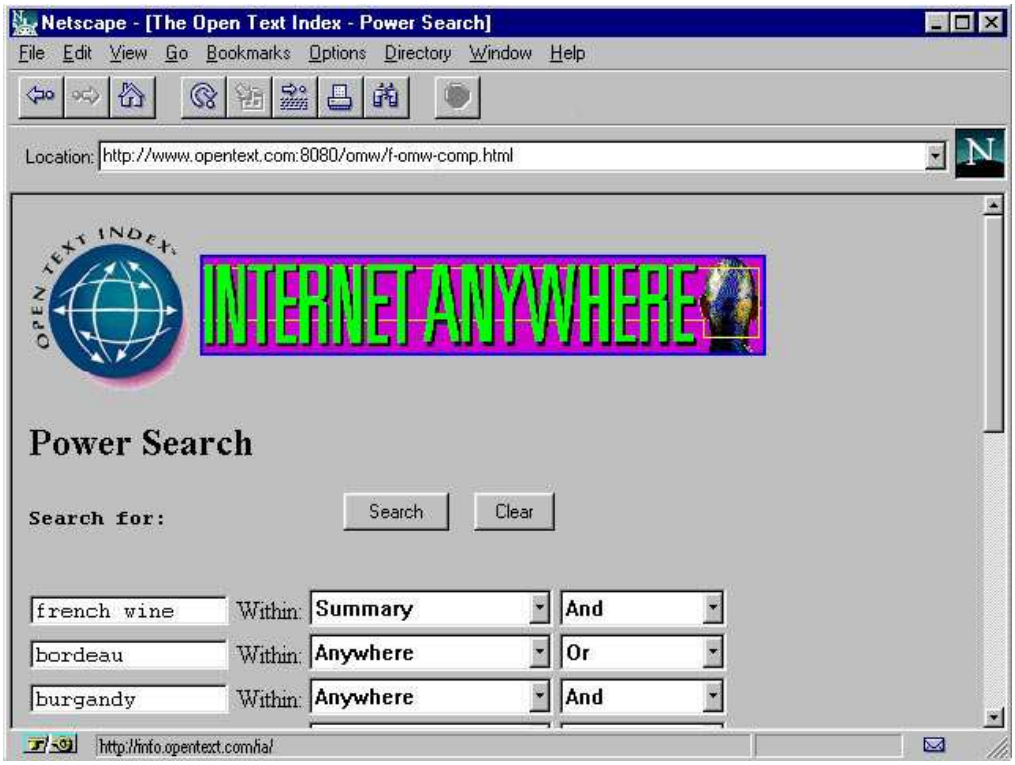

Illustration 3. Open Text's interface for searching the World Wide Web illustrates the ability to search in both full-text and structural elements of documents. Here, the user is looking for "french wine" in the document summary and for the words "bordeau" or "burgundy" anywhere in the document.

Informix Software: Parallel Processing of Complex Data

Building on the OnLine Parallel Database Architecture

Informix's goal is to extend the Informix-OnLine parallel database architecture to handle all types of data. The company is adding an extensible framework around the OnLine RDBMS server on which to build new functionality. The extensible framework provides support for complex objects (text, image, audio, video, multidimensional), high-volume performance via new indexing technologies and the ability to leverage parallel processing capabilities, support for OODBMS capabilities (e.g., object IDs), and support for industry standards (OLE, SQL3 for object-relational extensions). It essentially moves Informix toward an object-relational architecture and enables Informix, ISVs, VARs, and users to extend the database by building class libraries. (See Illustration 4.)

The programmer/developer will interact with the extensible framework through SQL language extensions, abstract and collection data types, extended stored procedures (methods), and object IDs for navigation. OnLine's internal architecture will include new parallel indexing and search methods, extended query optimization, and APIs for access to low-level functionality.

Informix Extensible Framework Architecture

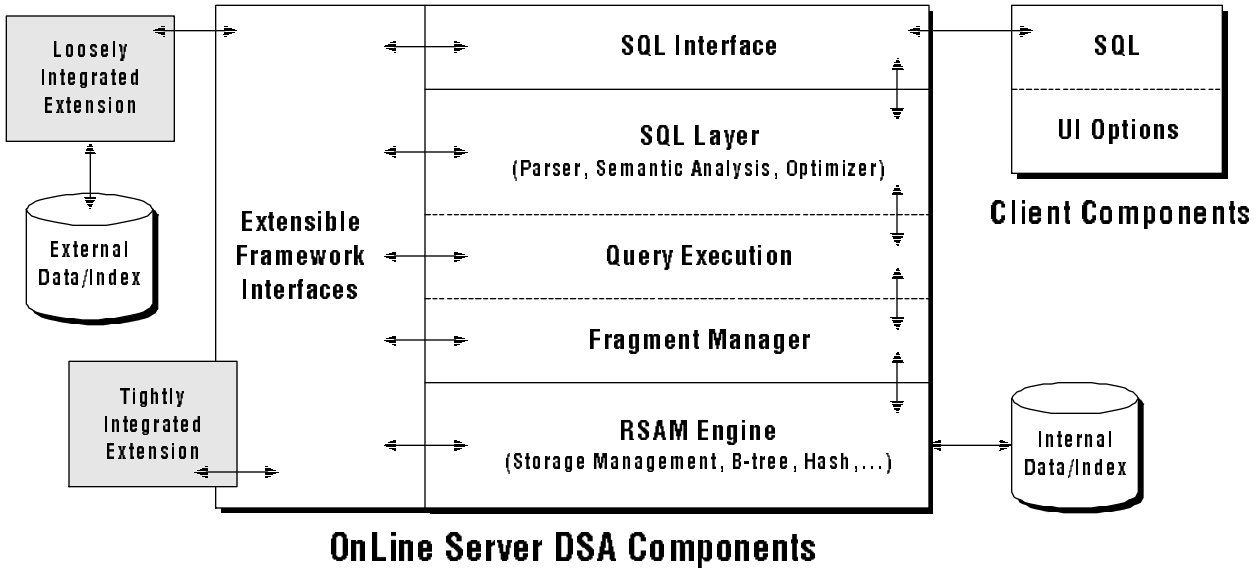

Illustration 4. Informix is extending its parallel Informix-OnLine Dynamic Scalable Architecture (DSA) to incorporate support for complex data types such as text and image.

Informix will apply its parallel query architecture to text data with the ability to partition both the text index and the text data. This will all be layered on Informix's existing BLOB storage facilities, new indexing methods based on Excalibur's Adaptive Pattern Recognition Processing (APRP) bitmap technology within the Informix engine. It will also provide support for unstructured text of any length, keyword/proximity/phrase-based indexing and search algorithms, and filters to separate text data from formatting information for faster searching. Excalibur technology will also be used for image extensions to the OnLine server. Informix states that it will provide for internal storage of text in the first release of the text search product (first half of 1996) followed by image search and options for external storage by the end of 1996.

 Reproduction in whole or in part is prohibited. For reprints, call (617) 742-5200.

IBM's Relational Extenders

Relational Extenders Are Built on DB2 2.1 Features

IBM is currently beta-testing a set of Relational Extenders for its DB2 database server. There are Relational Extenders for Text, Image, Video, Audio, and Fingerprints. IBM has built these using the object-relational facilities introduced in Version 2.1 of the DB2 common server—user-defined data types, user-defined functions, large objects (LOBs), triggers, and check constraints. IBM plans to bundle the Text Extender with the next release (2.1.1) of the DB2 common server for AIX, HP-UX, Solaris, and Windows NT. This will begin shipping in Q1 1996. IBM is working on integrating the other Extenders with DB2.

Text Extender

IBM's DB2 Text Extender incorporates the search technology found in the Search Manager/2 text-search engine that worked against Version 1 of DB2 on OS/2. It has been adapted to work within the IBM Extender architecture and to take advantage of DB2's object-relational support. Documents/text can be stored within the DBMS (up to 2GB each, the limit of BLOBs) or outside the DBMS via a pointer. There is no limit on size if the text is stored outside the DBMS.

Text Extender uses a number of different search techniques including synonyms, search by phrases, proximity, multiple arguments (using Boolean logic), and wildcards. Text Extender also has extensive multilanguage full-text search capability. Linguistic dictionaries are supplied for 17 languages in addition to English, including double-byte character sets.

Document Ranking, Free-Form Query, Agents and Profiles Coming

FUTURES. For future releases of its Text Extender, IBM is considering support for the following:

- **More Free-Form Query.** Instead of using a specific word or phrase with proximity language, users would be able to specify search criteria in a free-form fashion. For example, instead of defining a query as "[company XYZ] and [desktop] adjacent to [publishing]," the user could request documents that contain references to "company XYZ products used for desktop publishing." The Text Extender would come up with a ranked list of search results, with documents that most closely match the search criteria at the top of the list. The user would determine which document is closest to the desired search criteria and tell the search engine to use that as a model to come up with a better list. Natural-language searches provide a greater linguistic understanding of what the user wants. IBM's research division has been working on linguistic analysis techniques for many years and considers linguistic analysis very important in handling text.

- **Relevance Feedback.** In the first release, the user will get only a list of documents. There is no feedback on relevance/ranking relative to the search criteria. This will be added in the future.

- **Text-Mining through Agents and Profiles.** Text-mining is for users who would like to be made aware of developments in a certain area. Users can create

individual interest profiles that represent the subjects about which they want information. Agents then act on the user's behalf to generate queries as documents come in. What the user gets back is a document list.

- **Ability to Combine Different Content-Based Algorithms in one Query.** The goal of combining content-based algorithms is to invoke multiple search engines on behalf of the developer or user. The text-search engine is integrated with DB2 in its first release. Support for image, video, and audio file types, formats, and other attributes will be delivered in the same time frame. IBM already has a Query by Image Content (QBIC) product on the market. The company intends to integrate this technology into its image and video Extenders in Q2 1996 and is continuing research on audio.

Oracle's TextServer with ConText

The Next Generation of SQL*TextRetrieval

Oracle currently offers its TextServer Release 3.1. This is a renamed and enhanced product that replaces SQL*TextRetrieval and adds the ConText linguistic analysis technology acquired in 1992 from Artificial Linguistics. ConText includes language-parsing technology and a set of dictionaries. TextServer represents about $15 million in server licenses and $15 million in consulting revenues for Oracle.

TextServer has a client/server architecture. The client, the Oracle Text Developer's Workbench, is a set of text viewers, graphical administration utilities (Developer/2000 applications), and APIs. The APIs include a Developer/2000 interface in the form of PL/SQL packages, C libraries in the form of DLLs for Windows, and shared subroutine libraries for Unix. The administrative utilities, also available through the C API, are for user maintenance, setting up text databases, managing text partitions, making ad hoc queries, and creating text indexes. The server component is the TextServer3 with ConText kernel. TextServer is part of Oracle's Media Server product line; Oracle uses TextServer for its own corporate repository of data sheets and customer proposals.

Oracle started its text development efforts in 1986 and delivered SQL*TextRetrieval in 1990. This was a separate product from the Oracle database server, although it allowed the user to store formatted or unformatted documents in Oracle tables as well as in external files. SQL*TextRetrieval and, now, TextServer use a fully inverted index method for indexing text with a proprietary bitmapped location index rather than a conventional pointer system. (Oracle will incorporate this bitmapped indexing technique into its core DBMS server in Release 7.3.) The indexes are highly compressed and are cached in memory to reduce disk requirements and increase performance. There are also flexible indexing methods, such as incremental indexes or immediate indexing of new or changed documents.

TextServer can partition text tables across disks. The partitions make up a *virtual text table*. Partitioning allows TextServer to index and search in parallel on SMP, MPP, and clustered systems, providing scalability as the size of the database grows.

 Reproduction in whole or in part is prohibited. For reprints, call (617) 742-5200.

Oracle's TextServer with ConText

ConText Adds Linguistic Analysis

CONTEXT. ConText is a sophisticated natural-language processing technology that can identify themes and content in unstructured text.

Theme Search. ConText adds linguistic analysis, what Oracle calls an "about," or theme search, to other types of text searches. Instead of basing relevancy on the number of times specific words or phrases show up in a document, a theme search invokes linguistic processing that analyzes the actual meaning of words and their relative importance within and among documents. For example, searching for the word "russia" in a set of documents will most likely turn up a different hit list than doing a theme search on "russia." Documents that *contain* the word "russia" are not necessarily *about* Russia, and ConText can make this distinction. Theme searches can be combined with other types of word searches as well.

To do this, ConText uses a dictionary of 600,000 words, each with 1,000 possible attributes (types of words, such as generic gender noun or destination preposition) plus 20,000 grammar and syntax rules. It also has a huge hierarchy of words that represent "families" of words. The usage of words in the document and their relative importance calls up families of words that are tracked across the document. ConText can track up to 500 per document. These represent the themes in the document depending on how long they are used in the document and how well developed they are.

ConText has its own method of relevance ranking based on the relative weighting of words depending on their thematic importance in the document. TextServer also includes the de facto standard Salton method of relevance ranking (a word has the same weighting each time it occurs in a document) and the ability for the developer to create a user-defined method for ranking documents.

Summaries, Speedreading, and Index Generation

Document Summary. With ConText, TextServer can automatically generate a document summary, or what Oracle calls the "gist" of the document. The user can quickly see what a document is about and decide whether or not to read more of the document. Summaries also minimize network traffic in a distributed client/server environment. Another use for summaries is to build separate document abstracts.

Speedreading. SpeedRead is a viewer that allows the user to specify a "reduction percent" for scanning through retrieved documents. The user can start, for example, with a "glance" level (which typically displays only 10 to 20 percent of the document), and then switch to a Read level (50 or 60 percent) to study the document in more detail. SpeedRead can be viewed in two modes: highlight mode, where all of the text is shown with the important points highlighted, and summary mode, where only the important points are displayed.

Back-of-Book Index Generation. Another interesting feature of ConText is the ability to automatically generate an index for a document like the ones found in the back of reference books. An index is much more useful for finding information in a document than is a table of contents. There is considerable flexibility in formatting the index, such as including "see also" references and the ability to combine all

entries in the index below the second level. The user can also highlight an entry in the index and jump to that reference in the document.

Oracle points out that building a document index is very processor-intensive. For example, on a Pentium-based platform, ConText would build an index at roughly laser-printer speed (10 to 12 pages per minute). So users must be careful where and when indexes are built for very large documents.

TextServer Functions Will Be Integrated with Oracle7 Release 7.3

INTEGRATION WITH ORACLE7. In Release 7.3 of Oracle7, due out in production in Q2 1996, Oracle will collapse TextServer functionality into an Oracle7 Text Option. TextServer processing will become Oracle7 processes within the Oracle7 server environment. Text processing will now be able to take advantage of core Oracle7 features, such as parallel index builds, and the administrative tools can be used to configure resources for text. On the front end, the user will be able to issue extended SQL statements.

Illustra Leads the Charge to Object-Relational

Extensibility through DataBlades

The Illustra object-relational database manager from Illustra Information Technologies is an extended RDBMS that incorporates new data types, functions, and data access techniques using the concept of "DataBlade" software modules. A DataBlade is an object library that trains the Illustra server in a new data domain. An organization buys the underlying server engine and framework (the "razor") once and then adds DataBlades as needed to build a model of the business. Currently, DataBlades do not offer application-specific class libraries. They are building blocks to help the developer create specific types of applications faster. Illustra expects complete application objects in third-party DataBlades in the future.

In Illustra, the developer can define arbitrarily complex data types that are then used as templates for defining columns in tables. Data in these columns have structure, unlike the unstructured BLOBs in an RDBMS. Thus Illustra can provide content-based access to complex data such as multimedia objects.

New access methods are used to provide fast indexing for nontraditional data. In addition to relational B-trees, Illustra also supports R-trees (region trees) for accessing spatial data, D-trees (document trees) for text support, and a custom access method for time-series data.

All standard SQL types are included with the basic Illustra server. Illustra offers optional DataBlades to support more complex 2D and 3D spatial data, images, video, time series, text, Visual Information Retrieval (visual query-by-example on image content), Web servers, and statistical functions (S-Plus). Adding an Illustra DataBlade is done dynamically and does not require taking the database server down.

The PLS Text DataBlade

Illustra has its own Text DataBlade, originally developed as a demonstration of how Illustra can support nontraditional data. The company acknowledges, however, that

 Reproduction in whole or in part is prohibited. For reprints, call (617) 742-5200.

the Text DataBlade is not as functional as it could be. So Illustra is working with partner PLS on the PLS Text DataBlade and hopes to have other text-search vendors integrate with Illustra in the future. This offers users the choice of best-of-breed solutions.

PLS is the first foreign access method to be integrated with Illustra. The partnership has involved working together to extend the PLS interface library to integrate database-type functions, such as backup and transactions. The PLS Text DataBlade was made available at the end of 1995.

Combining the Illustra server with comprehensive text-search capabilities gives the user several advantages. A single SQL statement can be used to access data and functions across multiple DataBlades, providing integrated information access. Illustra's triggers and event alerters can be used to build document processing and distribution systems. Illustra also supports versioning, allowing users to track document revisions over time.

Conclusions

The Goal Is Integrated Access to Rich Data Types

Textbases are the first step toward integrating database management capabilities with access and manipulation of rich data types. Text is well understood by users, and full-text search is a relatively mature technology compared to content-based retrieval of images or sound. Text-search capabilities have become important to a wide range of information systems, all of which manage at least some text-based data. New solutions are now popping up from all sides. Text-search vendors are delivering distributed systems that can transparently access text in multiple locations, providing flexible deployment of the technology. RDBMS vendors are extending the relational model to integrate support for complex data types on top of their robust database management functionality. The object-relational vendors are ahead of the game with the built-in extensibility of their database servers.

Up to now, organizations have been willing to commit only a subset of their documents to an RDBMS because the database manager lacks flexibility, performance, and functionality in accessing and manipulating text. Documents in an RDBMS are primarily those created and maintained in a structured authoring environment, where the benefits of using the DBMS for control and management outweigh the costs. The issue is how to give a broad spectrum of users easy access to all relevant documents across the organization without paying the penalty imposed by the DBMS.

Choosing an Approach

The choice for users today is how to approach the information access problem: Is text functionality or broad access to information more important? One alternative is to start with a special purpose, limited-domain product that handles the richness of text extremely well and integrate that with other data-access products as required. The other is to start with the concept that text is only part of a much larger environment. The trick then is to look for a flexible, integrated platform that can

Conclusions

handle multiple complex data types, and live with the current limitations of a generic solution.

The inherent benefit of the extended RDBMS and the ORDBMS is that they offer a better model for sharing a global view of the world, and the ability to leverage mission-critical features such as reliability, availability, security, performance, and scalability. The downside is that a broad focus rather than a narrow one always carries with it extra overhead and complexity. The RDBMS vendors don't have the experience yet to compete with the focused vendors. But the RDBMS vendors are used to tough challenges. The key is to transparently integrate mature "specialized" technologies, such as text-search, into the RDBMS platform without reinventing the wheel. There is no reason the database can't succeed here, and we already see significant partnerships forming in the industry. Although the DBMS has the best shot at providing broad-based information access, performance, ease of use, and state-of-the-art functionality will be critical success factors.

Next month's *Open Information Systems* will address **Multidimensional Analysis: The New Competitive Requirement.**

For reprint information on articles appearing in this issue, please contact Donald Baillargeon at (617) 742-5200, extension 117.

Fulcrum Technologies
785 Carling Avenue
Ottawa, Ontario K1S 5H4 Canada
Phone: (613) 238-1761
Fax: (613) 238-7695
Web: http://www.fultech.com

IBM Corporation
555 Bailey Avenue
San Jose, CA 95161
Richard Dow
Phone: (408) 463-3042
Fax: (408) 463-2750
Web: http://www.ibm.com

Illustra Information Technologies
1111 Broadway, Suite 2000
Oakland, CA 94607
Sandra Bateman
Phone: (510) 873-6209
Fax: (510) 869-6388
Web: http://www.illustra.com

Informix Software
4100 Bohannon Drive
Menlo Park, CA 94025
Phone: (415) 926-6300
Fax: (415) 926-6593
Web: http://www.informix.com

InMagic
800 West Cummins Park
Woburn, MA 01801
Megan Moyer
Phone: (617) 938-4442
Fax: (617) 938-6393
E-mail: inmagic@netcom.com

Open Text Corporation
180 Columbia St. W.
Waterloo, Ontario N2L 3L3 Canada
Trevor Sharpe
Phone: (519) 888-7111
Fax: (519) 888-0677
Web: http://www.opentext.com

Oracle Corporation
500 Oracle Parkway
Redwood Shores, CA 94065
Brett Newbold
Phone: (415) 506-3188
Fax: (415) 506-1361
Web: http://www.oracle.com

Personal Library Software (PLS)
2400 Research Blvd., Suite 350
Rockville, MD 20850
David C. Macdonald
Phone: (301) 990-1155
Fax: (301) 963-9738
Web: http://www.pls.com

Verity
1550 Plymouth Street
Mountain View, CA 94043
Sue Barsamian
Phone: (415) 960-7600
Fax: (415) 960-7698
Web: http://www.verity.com

Xsoft
3400 Hill View Avenue
Palo Alto, CA 94303-1300
Susan Catalano
Phone: (415) 813-7174
Fax: (415) 813-7499
Web: http://www.xsoft.com

Correction: An error was made in *Open Information Systems* Vol. 10, No. 11, page 6. Table 1 should have the following entries for Windows NT 3.51:

	Windows NT 3.51
Max storage addressable	18,000,000 TB
Filesystem size limit	18,000,000 TB
File size limit	18,000,000 TB

We apologize for any inconvenience.

 Reproduction in whole or in part is prohibited. For reprints, call (617) 742-5200.