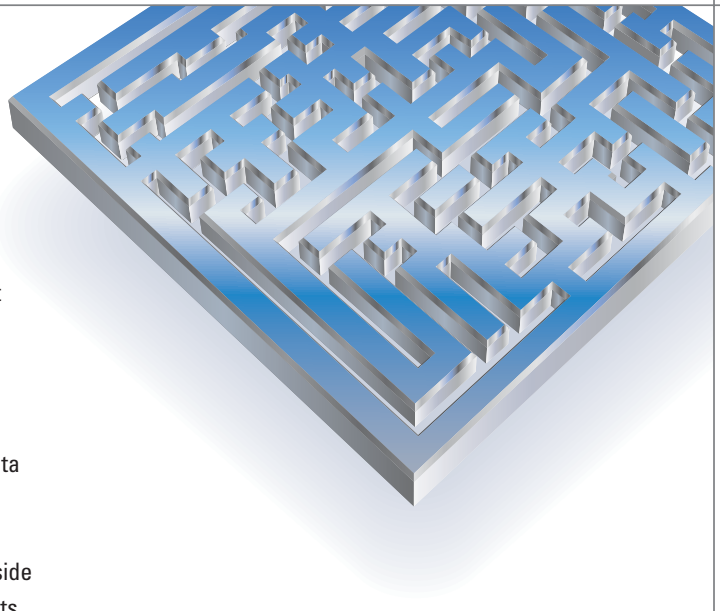
dtSearch®

A leading supplier of text retrieval software, dtSearch Corp. develops, manufactures and sells the dtSearch text retrieval product line. dtSearch products have been the Smart Choice for Text Retrieval® since 1991.

METHODS OF INTEGRATING FULL-TEXT AND METADATA SEARCH



This article addresses methods of integrating metadata with full-text indexed search, with the aim of providing more relevant search results. The discussion relies on the dtSearch® Text Retrieval Engine for its specific examples, although the general concepts have broader applicability.

Document Metadata. The simplest option for integrating metadata and full-text searching is to use existing fields in documents. For example, MS Office, XML, PDF, HTML, and other documents as well as emails all contain metadata fields. Using the fields inside these documents has the advantage of making each document its own self-contained data unit.

The diversity of document types and the size of a document collection can, however, make adding fields to each document prohibitively time consuming. The metadata data itself may also require a more complex table or hierarchical data structure than the underlying documents' fielded data options support.

Database Metadata. Another alternative is to store metadata for each document within a separate database such as SQL or XML. The documents themselves can either remain outside the database with only a filename or other identifier in the database. Or the documents can be inside a BLOB field in the database.

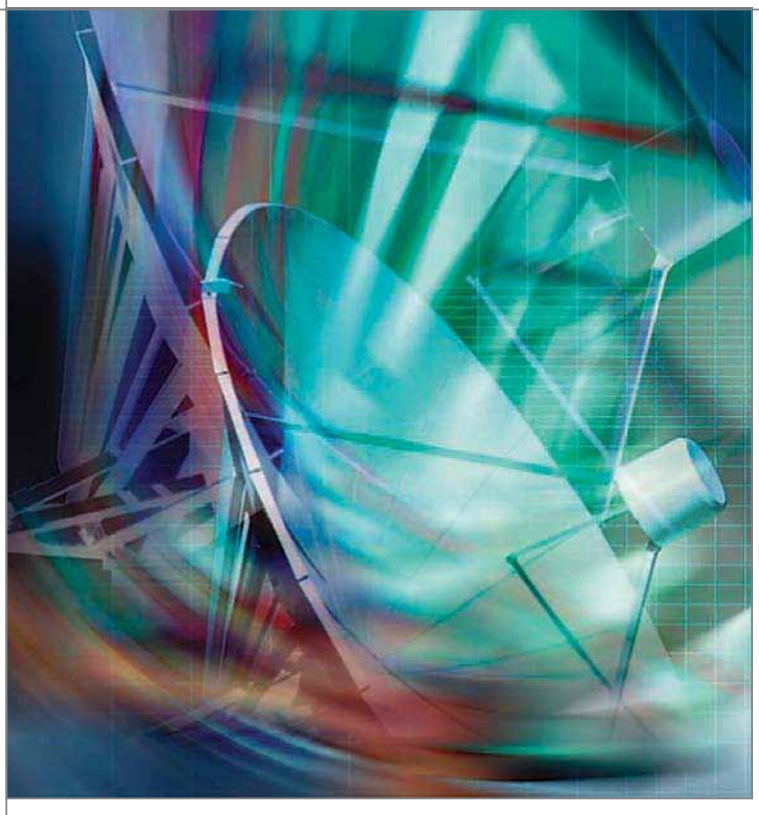
Because a structured database holds the fields, the database approach supports a more complex relational metadata structure. For example, searching can support the hierarchical field structures in XML data, enabling highly refined nested field queries.

The database approach also preserves the original document text intact, including original fields inside the documents for searching. And the database approach has the advantage of integrating with programming languages such as .NET, C++, and Java.

Indexing and Searching Metadata. Regardless of where the metadata resides in relation to the underlying documents, the search index combines the full-text document content with the metadata content. In the indexing process, the metadata joins the search index dynamically, as the search engine generates the index.

After indexing, full-text and metadata search typically take less than a second, even over a terabyte or more of data. The combination of metadata and full-text search capabilities in a single index makes it easy to integrate the two types of search in a single query. Example:

```
(apple w/5 pear) and (Author
contains John Smith) and
(PublicationDate contains 2007)
```



Search options such as fuzzy (to find *alphaqet* in a search for *alphabet*), phonic, Boolean, proximity, directed proximity, variable term weighting, and even hit-highlighting in search results work equally well with full-text and metadata searching.

Metadata-Enhanced Search Results. Another benefit of adding metadata to a full-text index is that the metadata can appear in search results. For example, search results can display a document subject field along with the highlighted hit in context. The search engine designates metadata for inclusion in search results as a "stored" field during the indexing process.

In addition to display of a document subject, for example, document properties such as filename, date, author, and the like can appear as stored fields in search results. Stored fields can also provide quick access to information such as row ids and security categories.

Search Filters. In some cases, an application requires the integration into full text searching of another type of selection that is too complex to express (efficiently) as a query. For example, in certain high-security settings, a complex set of security categories often applies to the execution of searching.

Search filter objects are an efficient means of implementing complex security classifications. They can also implement non-security related classification schemes, such as limiting a search to a specific geographical location. In either case, a search filter object works by limiting search results to a subset of documents in a document collection.

A search filter can correspond to external or internal document metadata. Or a search filter can result from a series of one or more queries against a full-text index. When a user submits a query, the application selects the relevant filter and attaches it to the search. The search filter uses an in-memory object, consisting of a table of bit vectors to minimize overhead.

When the search executes, it returns only documents that the search filter permits. In this way, multiple users with different classification settings can search the same document collection, without having to maintain separate indexes corresponding to each classification level.