

IXE, Ideare indexing Engine

Several centuries ago the human being began to study efficient ways to gather and store information, to make it easy to retrieve them later, so as to extract information out of them. Many techniques have been developed to rapidly access information contained (for instance) in a book. Just consider sophisticated tools like an *Index* or a *Table of Content*. These structures were usually created manually, by a particular kind of scientists, *librarians*. They were experts in defining categories and classifying each subject to a well-defined category.

In the computer era, document classification problem consists mainly in:

- defining efficient automatic indexing algorithms,
- writing extraction tools that provide users with fast answers, and
- developing ranking algorithms that improve the *quality* of the answer set.

Information Retrieval deals with the representation, storage, organization of and access to information items. In IR, the items we are trying to retrieve are called *documents*, and the documents are described by sets of *terms*. Usually a document is thought of as a piece of text and a term as a word or phrase which helps to describe the document, and which may indeed occur in the document, one or several times. So a document might be about holidays, and could be described by corresponding terms *travel*, *hotel*, *museum*, *excursion*, *beach* and so on.

Generally speaking, a document can be anything we want to retrieve and a term any feature that helps describe the documents. So the documents could be a collection of fossils held in a big museum collection, and the terms could be morphological characteristics of the fossils. Or the documents could be tunes, and the terms could then be phrases of notes that occur in the tunes.

What's IXE?

IXE can be seen as a C++ class library built for indexing and searching large collections of documents. The library has been designed for efficiency and scalability; it can handle multiple collections of documents, with overall size of several Terabytes. Using the library, several kinds of applications can be built, ranging from Web search engines to local disk indexing and retrieval, document management systems to specialized indexing and search of multimedia collection. A typical use of IXE could be to build a search engine, to search the Web or a company Intranet, with documents spread over the globe.

But being IXE so light and compact, it's also the perfect choice when it comes to indexing a relatively small amount of data (for instance to develop document management systems for small companies, legal firms, down to a single, personal hard disk), or to do specialized indexing and search of multimedia collection.

IXE is surprisingly fast at indexing time. Moreover, its architecture allows for parallel indexing carried out simultaneously on different systems. This feature allows the design of spidering architecture spread over the territory, where each spidering system resides near-by the domain or the collection of documents to be indexed. Each local system will spider its (portion of) domain and at the same time builds a local index; the collection of indexes can be consolidated into a large one in a central location. The advantage of such an approach is measured in terms of:

- bandwidth saving: spidering from a central location wastes a lot of bandwidth more than that needed to transfer a compressed, local index from a remote spider to the central location;
- simplification of the systems involved: IXE top performances are achieved on small Linux boxes, an order of magnitude cheaper than those used by our typical competitor;

Ideare S.p.A.

Lungarno Mediceo, 56 — 56100 PISA - Tel: (+39) 050 575300 Fax: (+39) 050 575583 <http://www.ideare.com>
Capitale Sociale L.1.000.000.000 i.v. — P. IVA 01477990509 — Iscr. Reg. Imprese Pisa n. 5934 — R.E.A. 130566

- higher speed and accuracy. To build a new index takes just a few hours so indexes can be generated more frequently, offering a great advantage in terms of freshness.

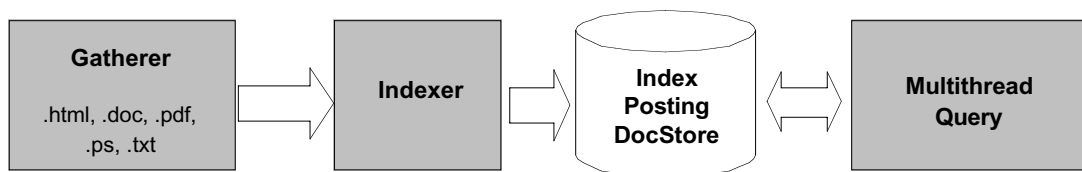
IXE represents the most exciting news in the Information Retrieval field. Its technology put IXE at the very top of the IR market in terms of performance, flexibility and ease of use. Moreover, these exciting features coupled with an innovative licensing scheme make it the perfect choice for any IR application development.

IXE has been designed to offer excellent performance on low cost Linux boxes but it is also available on Windows and on high end systems like Solaris, Tru64 UNIX, etc.

IXE Architecture

IXE overall architecture consists of:

- Gatherer: the module that collects document to be indexed
- Indexer: builds the index for a given collection of documents
- Document store: document repository
- Query server: processes queries to the index.



As far as the gatherer is concerned, a simple one is supplied with the library, but for a large-scale search service IXE can be interfaced to more sophisticated spidering systems.

IXE library has been designed with a modular architecture centered on a flexible relational database table, as described in Figure 1.

The Document Store contains document descriptions, including document attributes like name, title and abstract, date, and size as well as document contents if desired. The store is designed to handle up to several terabytes of material.

A library of C++ classes provides access to the functionality of document gathering, extraction/analysis, indexing and search.

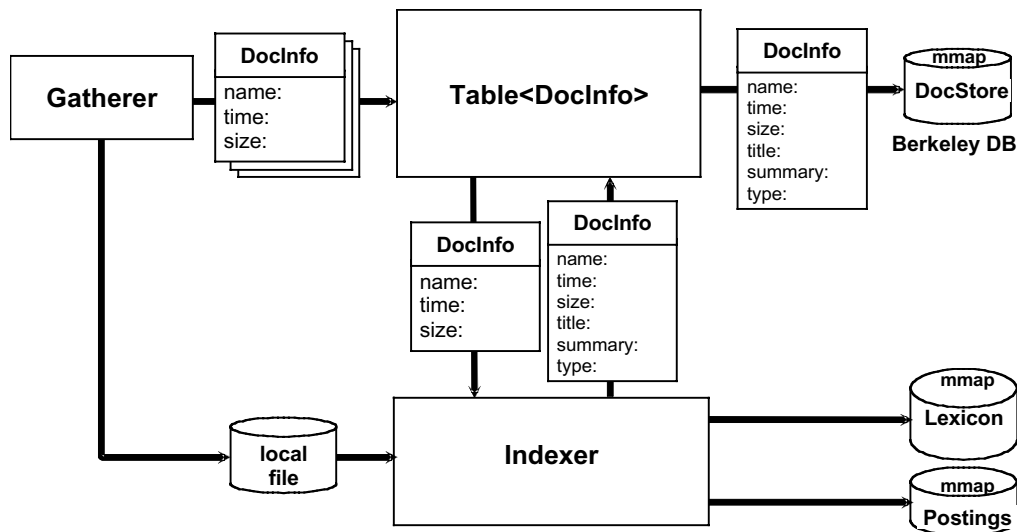


Figure 1. IXE Architecture

Indexer

Class Indexer performs full text indexing on several columns of a table of documents. Each full-text column has its own index, which is created or updated when a new document is inserted in a table.

Whenever a document is inserted in the table, for each full-text column it contains, the indexer is called supplying the **DocInfo**, the column number, the text source and the index type. The index type can be either `Field::fulltext` or `Field::external`. In the former case, the string in `SOURCE` is indexed and stored in the document store, while in the latter case, `SOURCE` contains the pathname of the file with the document contents and just the pathname is stored.

An indexer receives a short description (class **DocInfo**) of a document to be indexed. From such information (for instance the filename extension) it creates and invokes an appropriate document reader (class **DocReader**), which extracts occurrences of each term from the document and calls method `hit()` on the indexer for the appropriate column (**ColumnIndex**). Moreover a reader may extract information from the document to be added in appropriate fields in **DocInfo**.

IXE is capable of generating an index incrementally: new documents are added, old documents which appear changed, according to the equality operator in their **DocInfo** class, are replaced.

IXE indexing can be controlled either through command line options or through the configuration file `ixe.conf`.

Document Readers

The architecture is modular and new document readers can be added as necessary to deal with other document formats. The task of a reader is to extract not just terms from documents, but full hit information, i.e. *case* (upper, lower, capital) and *coloring*, i.e. within which document structure or region the term appears. For instance, the reader for HTML documents is capable of determining the tag or the element class within which a word appears.

Each reader is supplied with a specialized **DocInfo** structure, i.e. an instance of a class derived from **DocInfo**, which it can fill from information extracted from the document, e.g. its title and its summary, which it adds to its. For instance, the HTML reader uses a **PageInfo** structure, as defined above.

Such extended **DocInfos** are then stored in a **Table** implemented by means of a Sleepycat Berkeley Database. This dB is used when answering queries, providing summary information about selected documents.

Currently available readers include:

GenericReader	reads text documents, and is capable of extracting text from Microsoft Office documents, and to skip encapsulated PostScript
HTMLReader	<p>parses HTML pages, extracts title and summary and detects word colors, i.e. occurring within an HTML tag specified as color in the configuration file or within the CONTENT attribute of a metatag. In this case the color is the NAME attribute of the metatag and META must be included among the specified tag colors.</p> <p>The title consists in the words between the first <TITLE> ... </TITLE> pair. If no such title is present, the title is simply the file (not path) name.</p> <p>The summary consists of the first 2048 bytes of the document, stripped of HTML tags.</p>
SoifReader	reads files in SOIF format, produced by Harvest.
MemReader	reads text from a string.

Document Store

The document store is a relational database table, where documents are stored. DocInfo objects (i.e. instances of class DocInfo or its derivatives) describe documents. A DocInfo contains the minimal set of attributes required to identify and retrieve a document from the store: i.e. a document ID, a size in bytes and a modification time. A comparison operator must be supplied for each kind of DocInfo, which is used to determine whether two documents are the same or the document has changed. For instance, if the size and modification time of the document has changed, it needs to be re-indexed. Specialized versions of DocInfo might include an MD5 code of the contents to determine equality.

Search facilities

IXE supports Boolean search, phrase search, prefix search, attribute search and color search.

Phrase search and proximity

Phrase search can be expressed by either `body matches "network computing"` or by an infix `body matches network-computing`

`body matches "network computing"`

`body matches network-computing`

Proximity search is expressed as follows:

`body matches proximity 8 ((john smith) & (associate professor))`

Attribute search

Searching for words within a specified document attribute (column) can be expressed as:

`site matches "it.unipi.*" and body matches network computing`

The search will return all documents containing both words `network` and `computing` appearing in URLs from domain `unipi.it`, including its subdomains.

Color search

It is possible for instance to search for words appearing within a certain HTML element:

body matches title = software AND linux

will search for documents that contain the word software in the title and the word linux anywhere. Also meta-tags are used as colors:

body matches distribution=global academic-year

body matches author=john-smith web-design

Which tags are to be indexed and whether to add meta-tags as colors is controlled by entries in the configuration file (ixe.conf).

Search Applications

IXE has been developed as a library and an API toolkit that allow easy integration and development of custom search modules. Ideare has already made available the full range of search modules developed and marketed in its SearchTone Search Engine, namely:

- SearchTone **Search Engine Web**
- SearchTone **Automatic Classification Engine**
- SearchTone **Search Engine Audio**
- SearchTone **Search Engine Video**
- SearchTone **Search Engine Image**
- SearchTone **Search Engine Usenet**
- SearchTone **Chat Engine**
- SearchTone **Shopping Engine**

The customer has the choice whether to buy one or more of this modules or to develop its own custom application.

Benefits

IXE is a break-through technology. A team of professionals with a long and successful history in Computer Science industry has developed it. IXE contains a great number of architectural innovations and these have a positive fall-out on performances, flexibility and user-friendliness. Its been designed to offer excellent performance when executed on small Linux systems but it perfectly fits on large number-cruncher servers as well.

When it comes to performance, everybody can claim to be the fastest in the world: we accept the challenge! Let s the users compare IXE performances with anything they like: it s faster at indexing time, faster at search time. And our innovative licensing scheme and low prices make IXE the most valuable Information Retrieval product in the market.

And, talking about features, please check if other index engines can offer the following altogether:

- Programmability: C++ library, API
- Excerpts management

Ideare S.p.A.

Lungarno Mediceo, 56 — 56100 PISA - Tel: (+39) 050 575300 Fax: (+39) 050 575583 <http://www.ideare.com>
Capitale Sociale L.1.000.000.000 i.v. — P. IVA 01477990509 — Iscr. Reg. Imprese Pisa n. 5934 — R.E.A. 130566



- Proximity Rank
- Ranking through Link Popularity
- Paragraph Indexing
- Color search
- Column search
- Maximum document size: no limit!

System requirements

IXE runs on Linux (RedHat 7.1), Windows 32-bit, Solaris and Tru64 UNIX. Hardware requirements vary according to the number of documents to be indexed.

Ideare S.p.A.

Lungarno Mediceo, 56 — 56100 PISA - Tel: (+39) 050 575300 Fax: (+39) 050 575583 <http://www.ideare.com>
Capitale Sociale L.1.000.000.000 i.v. — P. IVA 01477990509 — Iscr. Reg. Imprese Pisa n. 5934 — R.E.A. 130566