www.ijecs.in
International Journal Of Engineering And Computer Science ISSN:2319-7242
Volume 4 Issue 5 May 2015, Page No. 11866-11870

iSearch-Enterprise Search Solution

Seema Mahajan[1], Soumya Bidarkote[2], Rohit Miragane[3] , Pooja Kulkarni[4] ,Vhatkar K. N.[5]

[1,2,3,4]Solapur University, N. K. Orchid College of Engg. & Tech. Solapur,
Tale Hipparaga, Solapur -413002
mahajanseema81@gmail.com
bidarkotesoumya@gmail.com
rohitmiragane@gmail.com
4kukarnipooja@gmail.com
kapilnv@gmail.com

Abstract: iSearch (Intelligent Search) is an enterprise search solution developed using Apache SOLR. iSearch system index data and documents from a variety of sources such as: file systems, intranets and document management systems. As the data in enterprise is becoming massive so iSearch helps people to seek the specific information they need of any rich text file format from anywhere inside their company. iSearch identifies and enables specific content across the enterprise to be efficiently searched, and displayed only to authorized users.

Keywords: iSearch , Search Engine , Enterprise Search , SOLR , Efficient Search , RichText Documents , Unstructured Data.

1.Introduction

Content without access is worthless. So search is an art and science of making content easy to find. One might also consider "findability", Findability is far more than just typing something into a search box and getting a result. It's also about discovering things about a topic that you didn't necessarily know you were looking for, and thus includes elements of browsing and discovery as well.

1.1 INTRODUCTION TO SEARCH

Search engines are programs that search documents for specified keywords and return a list of the documents where the keywords were found.
The art also refers to development of user interface that makes the retrieval process intuitive and responsive.

1.1.1 Search Engine Categories

Search Engines are categorized as follows:

a) Web Search Engines

Search engines that are expressly designed for searching web pages, documents, and images. They are engineered to follow a multi-stage process: crawling the infinite stockpile of pages and documents to skim the figurative foam from their contents, indexing the foam/buzzwords in a sort of semi-structured form (database or something), and at last, resolving user entries/queries to return mostly relevant results and links to those skimmed documents or pages from the inventory.

b) Database Search Engines

Searching for text-based content in databases allow pseudo-logical queries which full-text searches do not use. There is no crawling necessary for a database since the data is already structured. However, it is often necessary to index the data in a more economized form to allow a more expeditious search.

c) Mixed Search Engines

Sometimes, data searched contains both database content and web pages or documents. Most mixed search engines are large Web search engines, like Google. They search both through structured and unstructured data sources, which makes it exceptionally difficult to know what you are looking for and how to get to it. Pages and documents are crawled and indexed in a separate index. Databases are indexed also from various sources. Search results are then generated for users by querying these multiple indices in parallel and compounding the results according to "rules."

1.2 INTRODUCTION TO ENTERPRISE SEARCH ENGINE

"Enterprise Search" is used to describe the application for searching information within an enterprise (though the search function and its results may still be public). Enterprise search can be contrasted with web search, which applies search technology to documents on the open web, and desktop search, which applies search technology to the content on a single computer.
Enterprise search systems index data and documents from a variety of sources such as: file systems, intranets, document management systems, e-mail, and databases. Many enterprise search systems integrate structured and unstructured data in their collections. Enterprise search systems also use access controls to enforce a security policy on their users. Enterprise search can be seen as a type of vertical search of an enterprise.

1.2.1 What is it?

Enterprise search is how your organization helps people seek the information they need from anywhere, in any format, from anywhere inside their company – in databases, document management systems file systems, intranets etc.
Enterprise search is the practice of identifying and enabling specific content across the enterprise to be indexed, searched, and displayed to authorized users.

a) Content Ingestion

Content ingestion (or "content collection") is usually either a push or pull model. In the push model, a source system is integrated with the search engine in such a way that it connects to it and pushes new content directly to its APIs. This model is used when real-time indexing is important. In the pull model, the software gathers content from sources using a connector such as a web crawler or a database connector. The connector typically polls the source with certain intervals to look for new, updated or deleted content.

b) Content Processing And Analysis

Content from different sources may have many different formats or document types, such as XML, HTML, Office document formats or plain text. The content processing phase processes the incoming documents to plain text using document filters. It is also often necessary to normalize content in various ways to improve recall or precision.

As part of processing and analysis, tokenization is applied to split the content into tokens which is the basic matching unit.

c) Indexing

Next, a searchable index is created and other value-added processing, such as metadata extraction and auto-summarization, may take place. These functions group information into logical categories that in turn can be searched and return results to users based on how the particular search engine has categorized them.

d) **Query Processing And Matching**

Once this index is created, queries can then be accepted. Queries are terms or phrases that represent whatever you're looking for, typed into the search box.

At this point, the search engine processes the query by passing over the index, finding the information that matches the particular term or subject entered, and sending that information to some sort of processor, which then sorts the information by relevancy or other measure, clusters it based on the categorization, applies some other logic (such as "best bets" or "recommended best").Last comes the formatting, which presents the results page that you're used to seeing, in whatever format you've chosen.

Key Elements of Enterprise Search That Must be Addressed

Key elements of Enterprise search are as follows:

a) Federation

The search needs to reach each enterprise repository and index its content, so that a user can search one, some, or all enterprise content through a single search.

Federated search can be quite complicated, requiring capabilities such as advanced authentication, ranking of relevance across multiple repositories, and disaggregation of results from repositories with unique content.

b) Comprehensiveness

In addition to content location, search must be able to index critical content types. This includes files in file systems, documents and content management systems, structured data in databases even business data in business applications. Specific file types to index include text files, databases, desktop applications output, voice, video, compressed files, etc.

c) Relevance

Relevance measures how closely search results match user expectations. A search with high relevance will successfully return the documents the user intended when specifying the search term. Enterprise content also can have unique meanings for terms that vary from division to division, or even person to person. So, a search must also be tunable to ensure that the right results reach the user first.

d) Security And Access Control

It is not the role of enterprise search to set control access policies but search must ensure that its activities enforce those policies to ensure corporate security and the privacy of individuals. It must integrate with each repository's authentication scheme. To ensure security, you must control access not only to source documents but also to the search index that centralizes and summarizes them. Otherwise, search can become a weak link in the IT security chain.

e) **Scale And Scope**

Be prepared for the scope of your enterprise search problem to grow, and for more uses and users to surface. This issue also relates to scale an enterprise search solution must be able to scale to the needs of your enterprise.

2. Introduction To iSearch

The advent of Big Data causes enterprises to rethink how they handle search analysis, and decision making.

May be a person is looking for information internally in enterprise. He knows it exists but he is not quite sure where. The information lies across silos and it's a mix of structured and unstructured. iSearch (Intelligent Search) is an enterprise search solution developed using Apache SOLR which helps people to seek the specific information they need of any rich text file format from anywhere inside their company. iSearch identifies and enables specific content across the enterprise to be searched, and displayed only to authorized users.

iSearch is an intelligent search engine which lets you extract documents as per its content .

iSearch is a system with the following key characteristics:

1. Indexing multiple, disparate content repositories, heterogeneous in nature, and typically topologically distributed around the corporate network.
2. Serving a range of user requirements from deep research to simple fact checking, and supporting business-critical processes with customized search capabilities.

2.1 System Architecture

Fig. 2.1 shows the system architecture of the application.

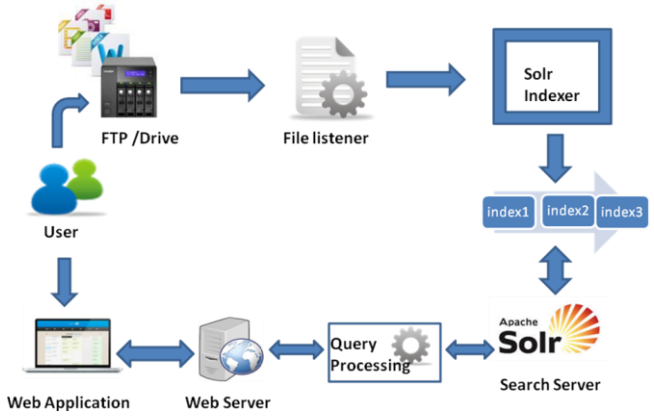

Figure 2.1: System Architecture Of iSearch

- **File Server (FTP/Drive) :**

File server is storage device dedicated to storing files which will be indexed and searched by user and database is used to store the user information such as user name, password and access level of user in order to maintain security in the system.

- **Document Processor (File Listener) :**

It fetches file from storage and give it to the indexer when any file related event occur like delete, modify etc for indexing.

- **Indexer:**

Indexer in search engine performs indexing for given file and prepare inverted index to store in index document.

- **SOLR Server :**

SOLR Server accepts query object find required document using index file and reply result back to query processor.

- **Query Processor:**

It is responsible for handling query given by browser and giving result back to the browser.

- **Web Server:**

Web server provide interface to browser for searching document in FTP server as it is connected to search engine. Also it replies back search result to browser.

- **Client Browser:**

Client Browser provide interface to user for entering keyword for searching relevant file .It send this keyword to web server to send it to search engine for searching .Also it display result of entered query to user.

2.3 HOW IS iSearch FLEXIBLE?

- It is reliable and gives quick response.
- It provides, highlighting, faceted search, spell checking.
- Diverse Content: Ability to index and search diverse content repository.
- Secured Search: Ability to make content accessible to only authorized people and/or groups.
- User Interface: Ability to provide faceted user interface (UI) components to serve end users with precise results.
- It lets the user to find a specific document.
- Result contains all the documents of specified keyword.

3. TECHNOLOGY

iSearch uses the Solr technology for preparing the index file for searching

3.1 SOLR :

The following figure shows the working of the Solr

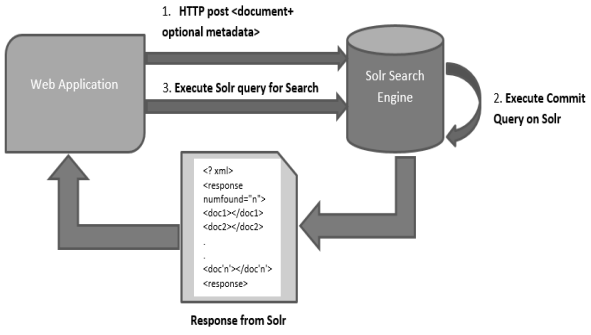

Figure 3.1 Working of Solr

- Solr is an open source enterprise search platform from the Apache Lucene project. Its major features include full-text search, hit highlighting, faceted search, dynamic clustering, database integration, and rich document (e.g., Word, PDF) handling.
- Providing distributed search and index replication, Solr is highly scalable.
- Solr is the most popular enterprise search engine.
- Solr is written in Java and runs as a standalone search server within a servlet container such as Apache Tomcat or Jetty.
- Solr uses the Lucene Java search library, and has REST-like HTTP/XML and JSON APIs that make it usable from most popular programming languages.
- Solr's powerful external configuration allows it to be tailored to many types of application without Java coding, and it has plugin architecture to support more advanced customization.

3.1.1 PHP

PHP is a server-side scripting language used for web development.

a) Solr Extension

- The Solr extension allows you to communicate effectively with the Apache Solr Server in PHP.
- The Solr extension is an extremely fast, light-weight, feature-rich library that allows PHP developers to communicate effectively with Solr server instances.
- There are built-in tools to add documents and make updates to the solr server.
- It also contains tools that allow you to build advanced queries to the server when searching for documents.

b) FileListener Connector

- FileListener Connector includes Solrj API as well as Apache Tika extract handler methods.
- Tika extract handler extracts the metadata from the documents.
- Solrj is used to send the parsed XML documents to solr server.

3.2 SOLR ARCHITECTURE

The following figure shows the architecture of Solr

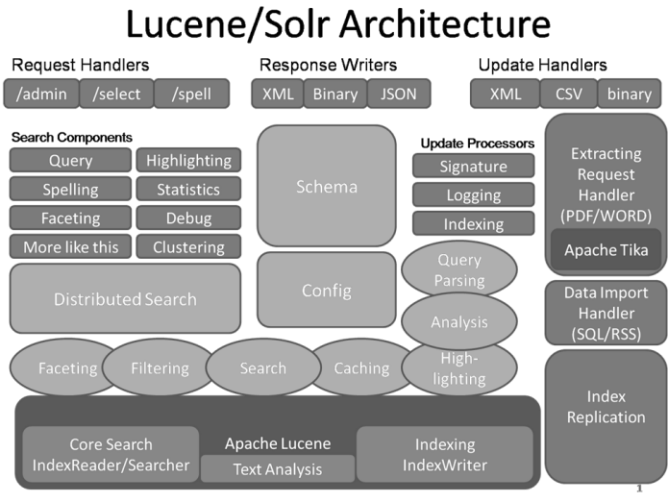

Figure 3.2: Solr Architecture

- **Request Handler**

Search request handler is a plugin that defines the logic to be used when Solr processes a request.

The search request handler includes query parser and response writer. To process a search query, a request handler calls a query parser that is responsible for parsing the textual query and converting it into corresponding Lucene query objects.

- **Response Writer**

The response writer component builds the query response object in the required format for the final presentation, generally an XML/JSON object is returned.

- **Update Handler**

Solr uses the "unique Key" to determine the "identity" of a document.While adding a document to the index with the same unique Key as that of an existing document means the new document will replace the original.

An "update" is actually performs two steps, internally:

- Delete the document with that id.
- Add the new document.

- **Extract Request Handler**

Extracting request handler is also called as solar cell. it uses Apache Tika to allow users to upload binary file to solr and let solr extract text from it and then index it.

- **Data Import Handler**

The DataImportHandler (DIH) provides access to structured data in relational databases (the Database data source in LucidWorks uses DIH under the hood).

It is a tool that can aggregate data from multiple database tables, or even multiple data sources to be indexed as a single Solr document.

- **Indexing**

Solr prepares the inverted index of the documents.

While indexing Solr defines an internal document ID which is used in the postings list for each term. During indexing, each field is analyzed to identify unique terms and their frequency in each document.

- **SolrConfig.xml**

The extract request handler is configured with solrconfig.xml.

- **Schema.xml**

Solr uses schema.xml to represent all of the possible fields and data types necessary to map documents into a Lucene index. This saves programming time and makes index structure easier to understand and communicate to others.

With Lucene, we need to write Java code to define fields and how to analyze those fields. So Solr adds a simple, declarative way to define the structure of our index and how we want fields to be represented and analyzed with the help of schema.xml

- **Filtering**

Used to filter results in addition to main query constraints. fq results are independently cached in Solr's filter Cache. Filter queries do not contribute to ranking scores. Filtering is commonly used for performing filtering on facets.

- **Faceting**

Faceting means arranging search result in column with numerical count of key terms.

- **Spell Checker**

solr's spellchecker supports two basic modes:

- Autocorrect-Solr can make the spell correction automatically, based on whether the misspelled term exists in the index.
- Did you mean-Solr can return a suggested query that might produce better results so that you can display a hint to your users.

- **Hit Highlighting**

When searching documents that have a significant amount of text, you can display

Specific sections of each document using Solr's hit-highlighting feature.

It is most useful for longer format documents, hit highlighting helps users to find relevant documents by highlighting sections of search results that match the user's query.

3.3 BUSINESS LOGIC

Business logic diagram shows actual functioning of the system.

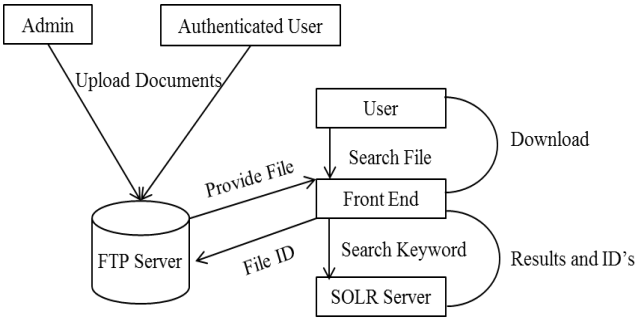

Figure 3.3: Business Logic

Figure shows business logic diagram of system. The function of administrator is to upload the documents on FTP server, which can be searched by using the SOLR server. User can enter the keyword on front end. Front end will prepare the query and send it to SOLR server. SOLR returns the result and file URL where the file located on FTP server. User can download file by following the URL of file.

4. CONCLUSION

The main objective of iSearch is to provide an efficient search for educational and college data of NKOCET members. The objective will be achieved by providing a Web Application which can be accessed by students and faculty via Local Intranet.

Thus main focus of iSearch is to provide the education related information within the minimum amount of time. To develop iSearch we are going to use Solr search engine for searching, FTP server for file storage, PHP for the frontend development.

Today almost everywhere the concept of big data is implemented, so here is a really smart application to search and retrieve the specific documents from such a massive amount of data within less amount of time.

4.1 FUTURE SCOPE

Following are the future scope of our project:

- At present our application can be used at college level, in future it can also be used in enterprise in order to search crucial data internally in the enterprise.
- Our application is able to search even the new file formats which will be invented in future.
- Along with the existing features, application can also have solr's features like Geospatial search, Advance text analysis to support most of the widely spoken languages (English, Chinese, Japanese, German, French and many more) Auto complete search and Synonym support.

REFERENCE

[1] Apache Solr Beginner's Guide, 31 Dec 2013 by Alredo Serafini

[2] Trey Grainger and Timothy Potter *Foreword by Yonik Seeley*, March 2014:Solr In Action

[3] An Integrated Approach to Software Engineering- 3rd edition: Pankaj Jalote (Narosa Publishers)

[4] Software Project management in practice-Pankaj Jalote

[5] The complete Reference, Java2 (5th edition) – Herbert Schedt et. a (Osborn)

[6] Enterprise Search Engine from: http://en.wikipedia.org/wiki/Enterprise_search
[7] Solr from: http://lucene.apache.org/solr/
[8] Apache Tika-Extract Handler from: https://cwiki.apache.org/confluence/display/solr/Uploading+Data+with+Solr+Cell+using+Apache+Tika
[9] Solr-php manual from: **http://php.net/manual/en/book.solr.php**
[10] Bootstrap from: http://getbootstrap.com

ACKNOWLEDGMENT

It was highly eventful at the department of Computer Science & Engineering, Nagesh Karajgi Orchid College of Engineering & Technology, Solapur. We would sincerely like to thank our Guide Prof. K. N. Vhatkar for giving us suggestions and taking a lot of interest in this project and whose advice and guidance helped us a lot. He has always been a source of inspiration for us right from the start of project.
Our honorable mention goes to Principal Prof. J. B. Dafedar. We would like to thank Prof. V. V. Bag Head of Department and faculty of Computer Science & Engineering for spending the valuable time for our project.
Lastly we are grateful to those people who directly and indirectly helped us during our project work.