Unified Search and Result Presentation from Disparate Process Manufacturing Data Systems using Service Oriented Architecture

PROJECT REPORT

An Internship Report Submitted In Partial Fulfillment Of The Requirements For Master Of Technology (Petro Informatics) Of University Of Petroleum And Energy Studies, INDIA

Submitted By:
Anurag Sinha
Karthik Viswanathan

ACKNOWLEDGEMENT

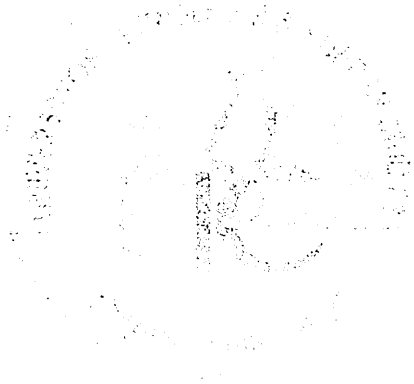
This is to acknowledge with thanks the help, guidance and support that we have received during the internship.

We express a deep sense of gratitude to the management of **SATYAM COMPUTER SERVICES LIMITED** for giving us an opportunity to pursue our summer internship.

We express our gratitude and obligation to our mentors **Mr. Sivananda Sarma** and **Mr. K Vinay Nayak** for their able guidance and support.

We would especially like to thank **Mr. Ramesh Gorugantu** for his valuable support.

Anurag Sinha
Karthik Viswanathan



Satyam

25 August 2005

To Whomsoever It May Concern

This is to certify that the internship report on "**Unified Search and Result Presentation From Disparate Process Manufacturing Data Systems Using Service Oriented Architecture**" submitted to the University Of Petroleum & Energy studies (Delhi) by Anurag Sinha, in partial fulfillment of the requirement for the award of the degree of **Masters Of Technology (Petro Informatics)**, is a bonafide work carried out by him under my supervision and guidance. This original work was carried during 13/6/05 to 25/8/05 at **Satyam Computers Services Limited**, Hyderabad, Andhra Pradesh.

G. V. Sivananda Sarma

Mr. Sivananda Sarma
Sr. Systems Analyst
VBU Energy & Utilities
Satyam City Center,
Ashoka Raghupathy Chambers,
Begumpet, Hyderabad

Satyam

August 25, 2005

To Whomsoever It May Concern

This is to certify that the internship report on "Unified Search and Result Presentation From Disparate Process Manufacturing Data Systems Using Service Oriented Architecture" submitted to the University Of Petroleum & Energy studies (Delhi) by Karthik Viswanathan, in partial fulfillment of the requirement for the award of the degree of Masters Of Technology (Petro Informatics), is a bonafide work carried out by him under my supervision and guidance. This original work was carried during 13/6/05 to 25/8/05 at Satyam Computers Services Limited, Hyderabad, Andhra Pradesh.

G.V. Sivad S.

Mr. Sivananda Sarma
Sr. Systems Analyst
VBU Energy & Utilities
Satyam City Center,
Ashoka Raghupathy Chambers,
Begumpet, Hyderabad

TABLE OF CONTENTS

INTRODUCTION

1. Introduction

An effective search tool on an intranet can make an enormous difference to its usability. In fact, usability expert Jakob Nielsen found that "Poor search was the greatest single cause of reduced usability across intranets". A good search engine ensures that users find what they're looking for, first time, regardless of the format or location of the information. This means that a wide variety of information can be effectively dispersed and made available to staff, without the need for complex navigation systems or filing conventions.

Most intranets evolve over time, and search functionality need not be a daunting task. A search tool can be implemented quickly, and then refined as the intranet grows and the needs of the organization change. Importantly, a flexible search engine that is cost-effective and expands to suit the growing requirements can be a much better investment than a cheap product with no ability to scale or an expensive solution where the majority of its functionality is wasted.

It is important to recognize that every intranet is different, with its own objectives, requirements and environment.

1.1 OBJECTIVES

A good start to understanding what an organisation need from an intranet search engine is to firstly understand the role of the intranet for the organization. What is the objective of the intranet? Is it to provide human resources information to staff? To provide information to help desk? Or to support the sales team? Perhaps the intranet is designed to meet a number of different objectives. Once these are clearly defined, it is easy to determine how a search engine can best add value.

The functionality that is required from a search engine will depend on the intranet objectives.

1.2 DATA SOURCES & FILE TYPES

Once the organisation objectives are clearly defined, the type of file formats and data sources the search engine will need to support can be worked out. The file types that are used in creating the information on the intranet usually fall into one of three categories:

1.2.1 Unstructured formats are file formats that contain primarily text-based information. These include text files, word processor files, PDFs, emails and formats used to create most documents. There is no real structure to these file formats and few relationships exist between elements within them.

1.2.2 Semi-structured formats are file formats that contain a mixture of text-based and data based information, with a basic structure. These include file types such as HTML, spreadsheets, XML. There may be relationships between elements within these files, however they are not as rigidly defined as they are in structured formats, and there may be sections of textual information where no structure exists.

1.2.3 Structured formats are file formats where the information is contained in a well-defined structure, such as a relational database. Many enterprise systems have a structured architecture, such as ERP and CRM systems, as well as many legacy databases.

For example, simply making available all of the HTML content on the intranet may start the search engine implementation. Then PDF and word documents can be included. Eventually, access to customer database or to XML data created in other systems can be provided. The intranet search engine needs to be able to support the full spectrum of file types that the organisation requires, even in multiple languages, if that is a concern for the organization. In addition, it should be considered how many files would be in the intranet data repository. This can affect search engine performance.

The organisation needs to think about publishing scope – depending on the sources of the information the organisation want to make available via the intranet, a search engine that provides umbrella

functionality over many different data sources may be needed. Some search engines are designed for specific data types, such as structured data, or are limited to single data repositories, such as search engines within content management systems. If organisation needs to incorporate legacy data into the intranet, a search engine that can support this will be needed. Finally, it must be considered whether organisation wants to incorporate external data sources into the intranet. Spidering functionality enables external websites to be integrated into the intranet data repository. Organisation can include the websites of suppliers or partners, for example, into the intranet search functionality, so that staff can search that content directly from the intranet.

1.3 TECHNOLOGY ENVIRONMENT

Organisation options will be determined by the internal technology infrastructure. All of the systems and software that will interact with the intranet search engine must be listed out. These include the intranet web server operating system and application software, the networking software and architecture, and the file system security environment. Any other systems that need to integrate with the intranet search, such as content management systems, ERP, CMS, email servers or legacy systems must be thought out beforehand.

1.4 FEATURES

For most intranets, there will be a wide spectrum of users, from very basic all the way through to highly technical power users. The search function needs to cater for all of these people, with a simple yet powerful interface that provides options for advanced searching if required.

There should be three steps to the search process, and a range of features work to streamline each of these steps; the important ones are described below. But it should be remembered that the

organisation may have specific objectives for the intranet that require certain features rather than others and it should be kept in mind.
The three steps in the search process are
(1) Entering the Query, or asking the initial question,
(2) Getting the Search Results or receiving the list of found documents back from the search engine
(3) Finding the Right Answer, or examining and refining the search results to find the information you were looking for.

1.4.1 Entering the Query. When a user enters their query, they should have the option to do this using a natural language approach; that is, by simply entering the question as they would ask it. Such as "What is our policy on blending?" There should also be the option to build queries using Boolean operators, so that users who know exactly what they want can be extremely specific with their search. For example, "policy within 10 words of blending but not mixing". The user interface must be checked to make sure it is intuitive for basic users, but also provides powerful advanced search functionality for more experienced users.
A good search engine should enable a person to group logical chunks of information together so that searches can be conducted on specific areas of interest. For example, an engineer may only be interested in searching technical documentation and might not want to search the HR policies that are also available on the intranet. The search engine needs to be able to group information separately to enable this to happen.
There should also be tools such as thesaurus functionality, where the search engine picks up common words with similar meanings, and synonyms, a thesaurus that can be custom-defined to include terms relevant to your particular organization. Spelling errors should be catered for with a 'sounds like' function, which enables users to find other words that sound similar to the one that they are typing. This compensates not only for spelling errors when the user enters the

query, but also for errors within the data itself, ensuring that such data is not lost forevermore.

1.4.2 Getting the Search Results If the organisation has very specifically defined data, such as legal documents, a high degree of precision may be required to identify and return specific information. In other situations, however, it may be better to return a wider range of documents for a given query. The accuracy that is required depends on the role of the search engine and the nature of the data.

If organisation want to make available a large volume of data on its intranet, providing a fast search engine is important. Otherwise users find it frustrating to wait for the search engine to bring back the search results. With smaller amounts of data this will be less of a concern; it all depends on the volume of data that one intends to make available on the intranet.

Any good search engine should use some form of intelligent relevancy determination. This is where the search engine, based on the query entered, makes a judgment about which results will be the most relevant, and ranks them accordingly.

1.4.3 Finding the Right Answer. The search process doesn't stop once the user receives the list of results. They then need to refine and manipulate the results list until they find exactly what they were looking for. There are many features that can assist in this task, some of which include:

1.4.3.1 Document summary information – the display of useful document attributes such as file type, file size, date last changed, relevancy rating and the number of 'hits' (key words found) in the document. The display of an extract of the document, say several lines above and below the first hit, is helpful for determining the context in which the document has been returned.

1.4.3.2 Re-sorting – the ability to re-sort the results list using different criteria, such as title, number of hits, relevancy, and date changed; file type or any other criteria that makes sense for your organization.

1.4.3.3 Hit-to-hit navigation – the provision of navigation buttons enabling users to go directly to the first hit in the returned document, and thereon to the next or previous hit as required. This means users avoid having to read through pages and pages of document before finding the relevant section, making it much more efficient.

1.4.3.4 Hit highlighting – a familiar concept from searching the web, hit highlighting is when the key words, or 'hits', in a document, are highlighted in a different colour. This feature is often not available in an intranet search engine, but it really should be, as combined with hit-to-hit navigation it enables users to immediately see the relevant sections of the document.

1.4.3.4 Fast preview – the ability to preview large non-HTML documents in a basic HTML format, without the need for downloading the whole document. This function enables users to view a few lines above and below each hit, and then to expand up or down to continue reading.

1.4.3.5 Search within – the ability to search within the current set of results, to further narrow them.

Although just some of the features available in intranet search engines, these are the main features required ensuring that users have the best overall experience. Others that may be relevant to the organization might include **intelligent agents** that automatically advice users when relevant content appears in the data repository or the ability to **save or export** search results.

PROBLEM DEFINITION

2. Problem Definition:

Most enterprises unstructured content is increasing between 65 to 200 percent a year depending on the industry sector. Employees can spend up to 40% of a workday looking for the right content. Too much expensive content goes underused or must be recreated. Add to this there are multiple legacy IT systems in today's refineries to manage different varieties of data. A unified view of the different data types and facilities to search the data would be very important for the refining operations and would help the management team to take quick and accurate decisions.

From the analysis of the type of data that resides in Refinery and inputs from various people, we derive that there is really a huge chunk of useful data that is left unexplored. If this data is available to the right people at the right time, it will not only hasten the decision making process but also increase the reliability.

Keeping this in mind we have scoped out the data sources for Unified Search.
We have categorised these data sources as Structured and Unstructured.

2.1 The structured content consists of the Transactional databases, which are mainly being used as backend for

- Asset Management - Maximo and SAP
- Laboratory Management – Sample Manager and Win Bliss
- Production Planning – PIMS
- Document Management – Documentum
- Drawing Management – Documentum
- HSE – Traction
- Operating Envelope – Alarm Database

Along with all these there is **PI Hierarchical database**, which is being used for pooling in Real Time data as well as to maintain its history. This is also well within our scope.

2.2 The **Unstructured content** consists of Files and Spreadsheets, which are mainly being used for

- Utilities Reporting – PI Process Book
- Process Graphic – Process Book Files
- Availability Reporting – Spreadsheets
- Also, there are huge amount of relevant data in Shared Files and Folders in each refinery.

2.3 Sneak Preview of Benefits of Using this Unified Search

We intend to develop an architecture based over SOA, which enables us to

- Build loosely coupled web services, interoperable with leading 3rd-party search services.
- Use the "Best In Class" Search services already available over web.
- Achieve improved scalability.
- Dramatically improved query performance.
- Increased relevancy.

OVERVIEW

3. Overview

The purpose of this search service is to provide facility to search the most relevant documents for different level of employee working in a refinery, which is currently not possible with any existing search engine. This search engine needs to cater the requirement of a particular user in the refinery. So the need is to find out who the end users are, what are their requirements, which document are relevant and in need in their work and what are the sources of the document?

So, the methodology / approach used to design this search service are:

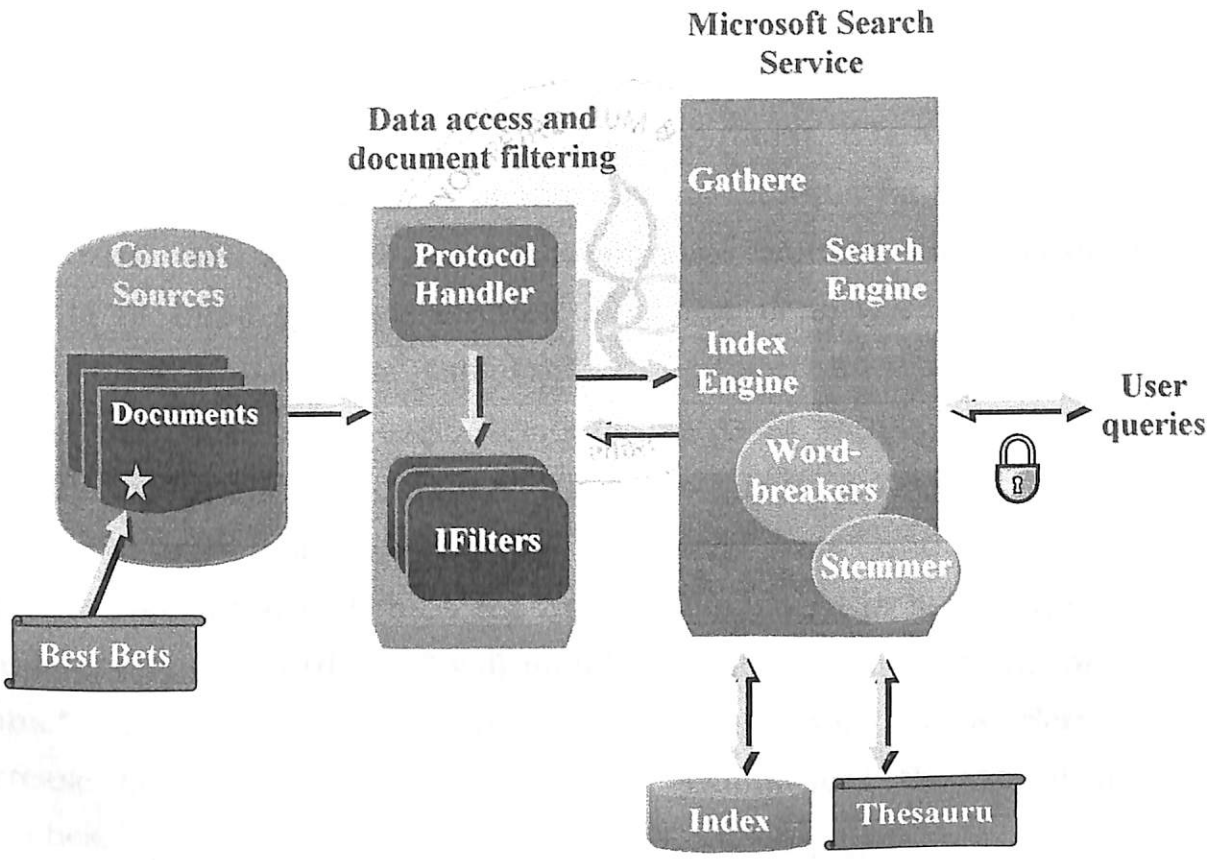
1. Identification of Business Processes in the Refinery
2. Finding out associated documents in each business process
3. Users and their requirement
4. Organisation structure and the flow of document in that
5. Categorise the documents based upon
    a. Architecture / Design
    b. Departments such as Marketing, Production, Planning, Finance etc.
    c. Documents sent to Customers / Suppliers
6. Map the documents to users

So, the flow of documents in the refinery will be found. Then the documents for a particular user can be prioritize.

4. Sharepoint Search

SharePoint search is one of the most advanced forms of information retrieval. SharePoint uses several advanced mathematical and computational techniques to increase the efficiency of information search and retrieval.

These factors combined with the ease of customization that SharePoint offers makes it ideal to base the Refinery Search Service on SharePoint.



The core search architecture of SharePoint Portal Server

Information from content sources is extracted using SharePoint data access and document filtering techniques and the information is then passed through other components like word breakers and stemmers. Individual words are mapped to their documents by creating an inverted index. The Thesaurus enables an organization to customize

the search process by allowing their abbreviations, brand names and departments within the corporate taxonomy to be included.

4.1 Data access and document filtering: Information filters (called "IFilters" in SharePoint Portal Server) extract textual information from a specific document format, such as Microsoft Word files. SharePoint Portal Server includes filters for all of the Microsoft Office document types, for TIFF and fax files, and for HTML documents and e-mail messages. Third-party filters, including a filter for Adobe PDF files, are also available.

4.2 Word-breaking and stemming: Word-breaking and stemming are software components that ready the words in a document for searching. Word breaking determines where the word boundaries are in the stream of characters in a query or document. When SharePoint Portal Server crawls documents that are in multiple languages, the customized word breaker for each language enables the resulting terms to be more accurate for that language. If no word breaker is available for a particular language, the SharePoint Portal Server neutral word breaker is used, breaking words at neutral characters such as spaces and punctuation marks. Stemming generates related forms of a single word; for example, "running," "ran," and "runs" are generated as variants of the stem "run." A document or search query that contains the word "run" will match against "running," "ran," and "runs." Word breaking and stemming in SharePoint Portal Server increase both the relevance of search results and the speed of searches.

4.3 Inverted Index: The principle of doing as much of the work ahead of time as possible is the foundation of full-text search in SharePoint Portal Server. An index is created in advance of the first user query by extracting information about all of the documents to be searched and assembling it into a more efficient format called an inverted index that maps individual words back to the documents that contain them. Index creation typically occurs during a "crawl" of all

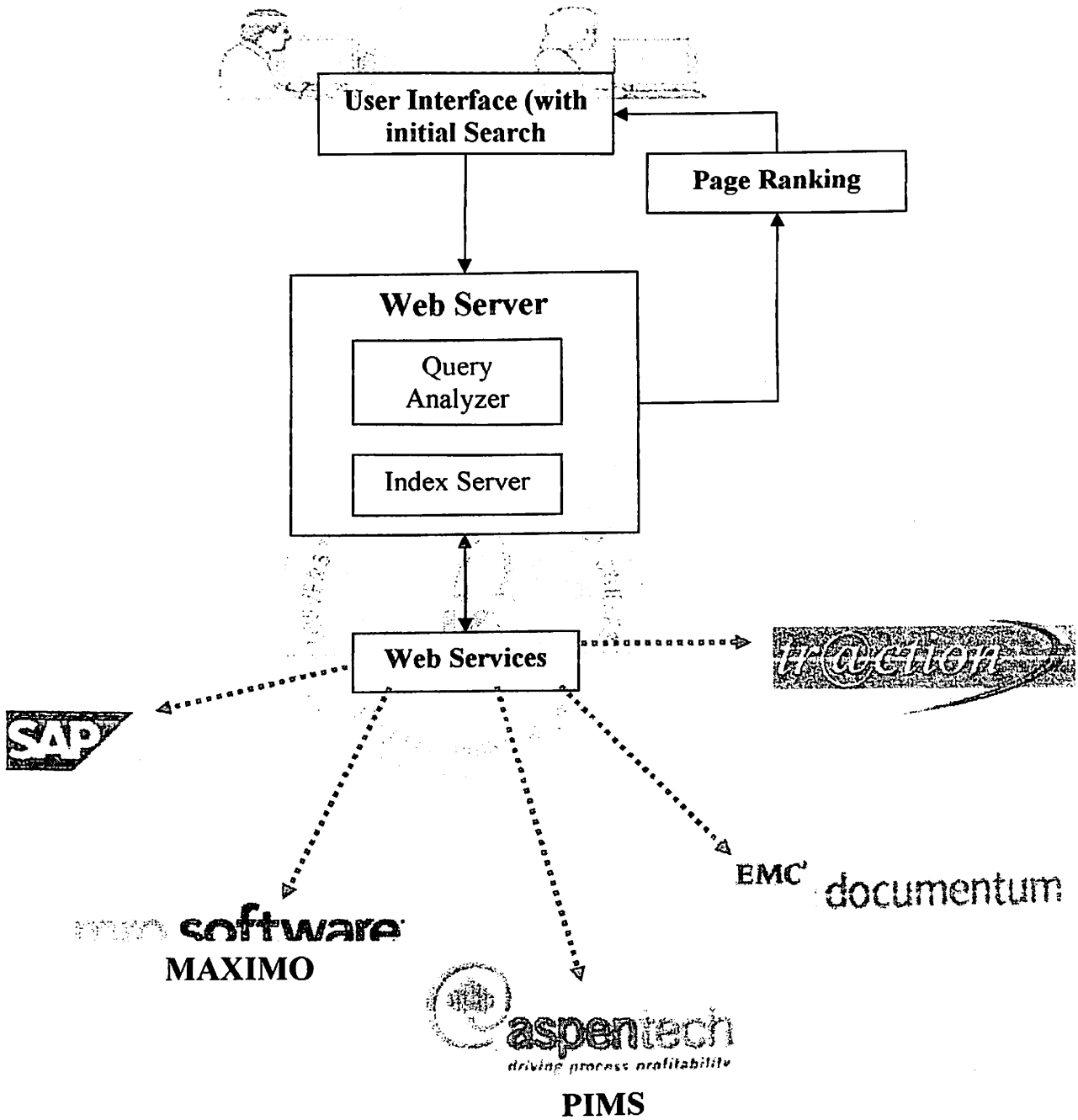
searchable documents, an automated traversal of all documents in the search space. After an index has been created, a user query can be processed by comparing the search terms against the terms in the index, not in the original documents. This removes the need to open and translate each document every time a search is performed, significantly increasing the speed with which the search engine can answer a user's query. The index is updated in the background as new documents are created or existing documents are updated or deleted. The use of an inverted index enables SharePoint Portal Server to apply advanced statistic and probabilistic formulas, such as the probabilistic relevance scoring, to more quickly compute the relevance of documents.

4.4 Thesaurus: SharePoint Portal Server enables an organization to create a thesaurus of words used in processing queries. The thesaurus capability enables users to search for information using abbreviations and previously used names that are unique to a particular organization. This is particularly useful for enabling users to find documents containing brand, department, and product names that have changed or are no longer in use. Consider a thesaurus entry that links the current name of a department, such as "HR Department," to another name for the department, such as "Human Resources Department." With this thesaurus entry in place, a query for information containing the term "HomeCare Department" will also return documents containing "Consumer Medical Products Department." The SharePoint Portal Server thesaurus is implemented using Web standard XML (Extensible Markup Language) technology and is another example of how human input and advanced technology makes searching for information easier and faster.

4.5 Best Bets: SharePoint Portal Server is designed to use human input to maximize the speed and effectiveness of searches, and Best Bets are a prime example of how human input adds value to the

search process. Best Bets in SharePoint Portal Server is a feature that enables editors to decide which documents that have been stored in the SharePoint Portal Server document repository are the most relevant result for a specific query or category. The documents chosen as Best Bets are prominently displayed at the top of the results list returned during a full-text search or when browsing categories. For example, the company's marketing plan would make an excellent "Best Bet" for intranet queries that included the term "marketing." Best Bets are particularly used for documents that do not contain the words needed to make it relevant to a particular search result, for example, a campus map that does not itself contain the word "map." SharePoint Portal Server also has the capability of automatically nominating documents to be Best Bets. As the product is taught which documents are Best Bets for a particular query, it will nominate new documents based on the presence of key terms in the document. This is just one example of how the design of SharePoint Portal Server extends the reach of human editors and document creators.

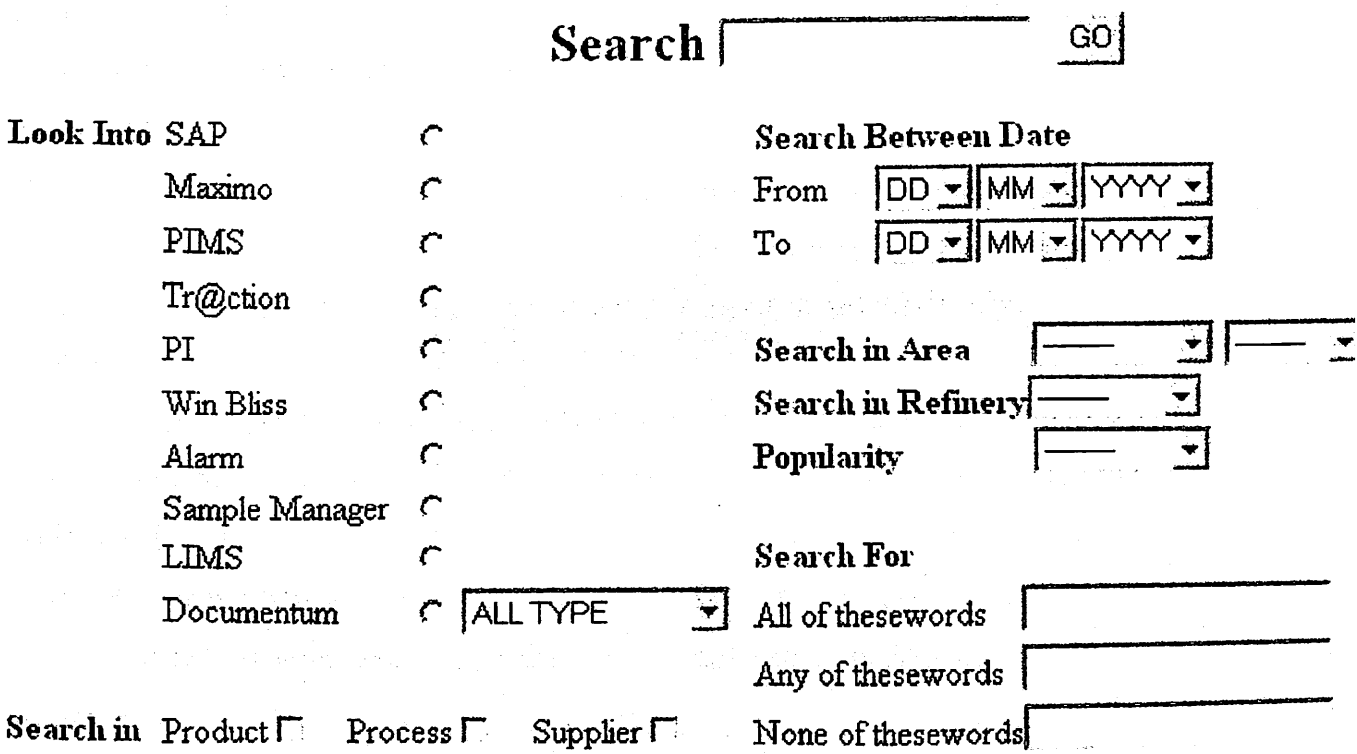
5. Overall Architecture



6. User Interface

The underlying emphasis of the Search Service is not to just list out information but also to streamline the search so as to achieve faster information retrieval rates. This can be achieved by basing the search on specific parameters.

The User Interface shown below lists out these parameters.



In most cases the User would know what his data looks like, in what format it would be in. He doesn't know where to look for.

6.1 Look Into

The user knows he is searching for a PDF file; all he has to do is just type in the word, phrase or sentence, click on documentum and select Adobe in the drop down menu. Similarly he knows his data set is in SAP, by using the SAP parameter he narrows down his search drastically.

6.2 Search between Dates

The User is interested in a Data Set applicable between certain dates; He is not interested in similar data sets before or after those dates. This option will bring down the content to be searched and limit it to a certain period.

6.3 Process, Product & Supplier

The User knows that his data set falls under Supplier related information and he also knows that his search word is applicable for a process as well as a product. In the absence of such an option he would have had to mine through the results to get his data. This parameter dictates the Search Service to only look for information pertaining to supplier Data Sets.

6.4 Location

The role of the Search Service is to Search across multiple refineries. If the User is interested in data from a specific refinery location then he can use this parameter to narrow down his search accordingly.

6.5 Area

A refinery is classified into multiple areas like Crude Area, Cracking Area, Lubes Area, Mogas Area & Offsites. Further each area is further subdivided into other sub areas. The user might specifically be interested in a data set of a certain area. This parameter dictates the search service to search for specified data sets in the specified area or sub area.

6.6 Popularity

Popularity classifies Data sets as

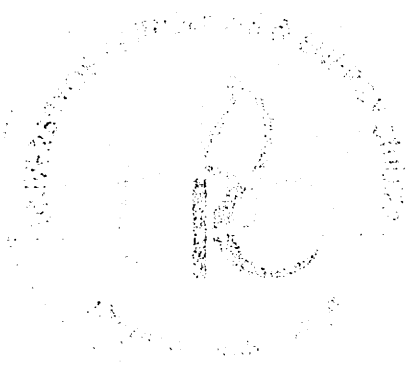
a) Most Popular
b) Moderately Popular
c) Less Popular.

This is basically a ranking parameter; The Search Service conducts the Search irrespective of how popular the data set is. While listing the results the data sets are ranked based on the parameter that the User has chosen.

In some cases the User knows for a fact that the data set he is searching for has been rarely accessed. By using this parameter he

would be able to get his Data set in the first listing instead of mining through multiple listings.

The effort is to incorporate as many parameters as possible so as to provide an effective and efficient search. In the event of the user not entering values for certain parameters then the search would be done ignoring these parameters.



7. Web Server

Search engine is the popular term for an information retrieval (IR) system. While researchers and developers take a broader view of IR systems, consumers think of them more in terms of what they want the systems to do — namely search the Web, or an intranet, or a database. Actually consumers would really prefer a finding engine, rather than a search engine.

Search engines match queries against an index that they create. The index consists of the words in each document, plus pointers to their locations within the documents. This is called an inverted file. A search engine or IR system comprises four essential modules:

- A document processor
- A query processor
- A search and matching function
- A ranking capability

While users focus on "search," the search and matching function is only one of the four modules. Each of these four modules may cause the expected or unexpected results that consumers get when they use a search engine.
The main search server consists of query analyzer and index server. It receives parameters from the user interface and connects to the database through web services.

The query processor has several parts, including the "engine" that evaluates queries and matches them to relevant documents, and the results formatter. There are many factors in determining which documents are more relevant to a query, including the popularity of

the page, the position and size of the search terms within the page, and the proximity of the search terms to one another on the page.

After evaluation and matching of the query the query analyzer sends the query to the index server. Index server is a server-side executable that does full-text extraction for creating index files for fast web-based queries. An example of Index server is Microsoft Index Server, which is an add-on service to the Internet Information Server (IIS) that enables the indexing of document properties and contents on the IIS server.

Once the IIS server documents are indexed, users can use their browsers to query the Index Server in order to search for selected document contents or properties. Responses to user queries are presented back to the client browser as a series of HTML links to the documents containing the query properties or contents.

7.1 Document Processor

The document processor prepares, processes, and inputs the documents, pages, or sites that users search against. The document processor performs some or all of the following steps:

- Normalizes the document stream to a predefined format.
- Breaks the document stream into desired retrievable units.
- Isolates and metatags subdocument pieces.
- Identifies potential indexable elements in documents.
- Deletes stop words.
- Stems terms.
- Extracts index entries.
- Computes weights.

- Creates and updates the main inverted file against which the search engine searches in order to match queries to documents.

Steps 1-3: Preprocessing. While essential and potentially important in affecting the outcome of a search, these first three steps simply standardize the multiple formats encountered when deriving documents from various providers or handling various Web sites. The steps serve to merge all the data into a single consistent data structure that all the downstream processes can handle. The need for a well-formed, consistent format is of relative importance in direct proportion to the sophistication of later steps of document processing. Step two is important because the pointers stored in the inverted file will enable a system to retrieve various sized units — site, page, document, section, paragraph, or sentence.

Step 4: Identify elements to index. Identifying potential indexable elements in documents dramatically affects the nature and quality of the document representation that the engine will search against. In designing the system, we must define the word "term." Is it the alphanumeric characters between blank spaces or punctuation? If so, what about non-compositional phrases (phrases in which the separate words do not convey the meaning of the phrase, like "skunk works" or "hot dog"), multi-word proper names, or inter-word symbols such as hyphens or apostrophes that can denote the difference between "small business men" versus small-business men." Each search engine depends on a set of rules that its document processor must execute to determine what action is to be taken by the "tokenizer," i.e. the software used to define a term suitable for indexing.

Step 5: Deleting stop words. This step helps save system resources by eliminating from further processing, as well as potential matching, those terms that have little value in finding useful documents in response to a customer's query. This step used to matter much more

than it does now when memory has become so much cheaper and systems so much faster, but since stop words may comprise up to 40 percent of text words in a document, it still has some significance. A stop word list typically consists of those word classes known to convey little substantive meaning, such as articles (*a, the*), conjunctions (*and, but*), interjections (*oh, but*), prepositions (*in, over*), pronouns (*he, it*), and forms of the "to be" verb (*is, are*). To delete stop words, an algorithm compares index term candidates in the documents against a stop word list and eliminates certain terms from inclusion in the index for searching.

Step 6: Term Stemming. Stemming removes word suffixes, perhaps recursively in layer after layer of processing. The process has two goals. In terms of efficiency, stemming reduces the number of unique words in the index, which in turn reduces the storage space required for the index and speeds up the search process. In terms of effectiveness, stemming improves recall by reducing all forms of the word to a base or stemmed form. For example, if a user asks for *analyze*, they may also want documents which contain *analysis, analyzing, analyzer, analyzes*, and *analyzed.* Therefore, the document processor stems document terms to *analy-* so that documents which include various forms of *analy-* will have equal likelihood of being retrieved; this would not occur if the engine only indexed variant forms separately and required the user to enter all. Of course, stemming does have a downside. It may negatively affect precision in that all forms of a stem will match, when, in fact, a successful query for the user would have come from matching only the word form actually used in the query.

Systems may implement either a strong stemming algorithm or a weak stemming algorithm. A strong stemming algorithm will strip off both inflectional suffixes (*-s, -es, -ed*) and derivational suffixes (*-able, -aciousness, -ability*), while a weak stemming algorithm will strip off only the inflectional suffixes (*-s, -es, -ed*).

Step 7: Extract index entries. The document processor now extracts the remaining entries from the original document.

The output is then inserted and stored in an inverted file that lists the index entries and an indication of their position and frequency of occurrence. The specific nature of the index entries, however, will vary based on the decision in Step 4 concerning what constitutes an "indexable term."

Step 8: Term weight assignment. Weights are assigned to terms in the index file. The simplest of search engines just assign a binary weight: 1 for presence and 0 for absence. The more sophisticated the search engine, the more complex the weighting scheme. Measuring the frequency of occurrence of a term in the document creates more sophisticated weighting, with length-normalization of frequencies still more sophisticated. Extensive experience in information retrieval research over many years has clearly demonstrated that the optimal weighting comes from use of "tf/idf." This algorithm measures the frequency of occurrence of each term within a document. Then it compares that frequency against the frequency of occurrence in the entire database.

Not all terms are good "discriminators" — that is, all terms do not single out one document from another very well. A simple example would be the word "the." This word appears in too many documents to help distinguish one from another. A less obvious example would be the word "antibiotic." In a sports database when we compare each document to the database as a whole, the term "antibiotic" would probably be a good discriminator among documents, and therefore would be assigned a high weight. Conversely, in a database devoted to health or medicine, "antibiotic" would probably be a poor

discriminator, since it occurs very often. The TF/IDF weighting scheme assigns higher weights to those terms that really distinguish one document from the others.

Step 9: Create index. The index or inverted file is the internal data structure that stores the index information and that will be searched for each query. Inverted files range from a simple listing of every alpha-numeric sequence in a set of documents/pages being indexed along with the overall identifying numbers of the documents in which the sequence occurs, to a more linguistically complex list of entries, the tf/idf weights, and pointers to where inside each document the term occurs. The more complete the information in the index, the better the search results.

7.2 Query Processor

Query processing has seven possible steps, though a system can cut these steps short and proceed to match the query to the inverted file at any of a number of places during the processing. Document processing shares many steps with query processing. More steps and more documents make the process more expensive for processing in terms of computational resources and responsiveness. However, the longer the wait for results, the higher the quality of results. Thus, search system designers must choose what is most important to their users — time or quality. Publicly available search engines usually choose time over very high quality, having too many documents to search against.

The steps in query processing are as follows (with the option to stop processing and start matching indicated as "Matcher"):

- Tokenize query terms.
  Recognize query terms vs. special operators.

  ————————> Matcher

- Delete stop words.

- Stem words.
- Create query representation.

————————> Matcher

- Expand query terms.
- Compute weights.

————————> Matcher

Step 1: Tokenizing. As soon as a user inputs a query, the search engine — whether a keyword-based system or a full natural language processing (NLP) system — must tokenize the query stream, i.e., break it down into understandable segments. Usually a token is defined as an alphanumeric string that occurs between white space and / or punctuation.

Step 2: Parsing. Since users may employ special operators in their query, including Boolean, adjacency, or proximity operators, the system needs to parse the query first into query terms and operators. These operators may occur in the form of reserved punctuation (e.g., quotation marks) or reserved terms in specialized format (e.g., AND, OR). In the case of an NLP system, the query processor will recognize the operators implicitly in the language used no matter how the operators might be expressed (e.g., prepositions, conjunctions, ordering).

At this point, a search engine may take the list of query terms and search them against the inverted file. In fact, this is the point at which the majority of publicly available search engines perform the search.

Steps 3 and **4: Stop list and stemming.** Some search engines will go further and stop-list and stem the query, similar to the processes described above in the Document Processor section. The stop list might also contain words from commonly occurring querying phrases,

such as, "I'd like information about." However, since most publicly available search engines encourage very short queries, as evidenced in the size of query window provided, the engines may drop these two steps.

Step 5: Creating the query. How each particular search engine creates a query representation depends on how the system does its matching. If a statistically based matcher is used, then the query must match the statistical representations of the documents in the system. Good statistical queries should contain many synonyms and other terms in order to create a full representation. If a Boolean matcher is utilized, then the system must create logical sets of the terms connected by AND, OR, or NOT.

An NLP system will recognize single terms, phrases, and Named Entities. If it uses any Boolean logic, it will also recognize the logical operators from Step 2 and create a representation containing logical sets of the terms to be AND'd, OR'd, or NOT'd.

At this point, a search engine may take the query representation and perform the search against the inverted file. More advanced search engines may take two further steps.

Step 6: Query expansion. Since users of search engines usually include only a single statement of their information needs in a query, it becomes highly probable that the information they need may be expressed using synonyms, rather than the exact query terms, in the documents which the search engine searches against. Therefore, more sophisticated systems may expand the query into all possible synonymous terms and perhaps even broader and narrower terms.

This process approaches what search intermediaries did for end users in the earlier days of commercial search systems. Back then, intermediaries might have used the same controlled vocabulary or thesaurus used by the indexers who assigned subject descriptors to documents. Today, resources such as WordNet are generally available,

or specialized expansion facilities may take the initial query and enlarge it by adding associated vocabulary.

Step 7: Query term weighting (assuming more than one query term). The final step in query processing involves computing weights for the terms in the query. Sometimes the user controls this step by indicating either how much to weight each term or simply which term or concept in the query matters most and *must* appear in each retrieved document to ensure relevance.

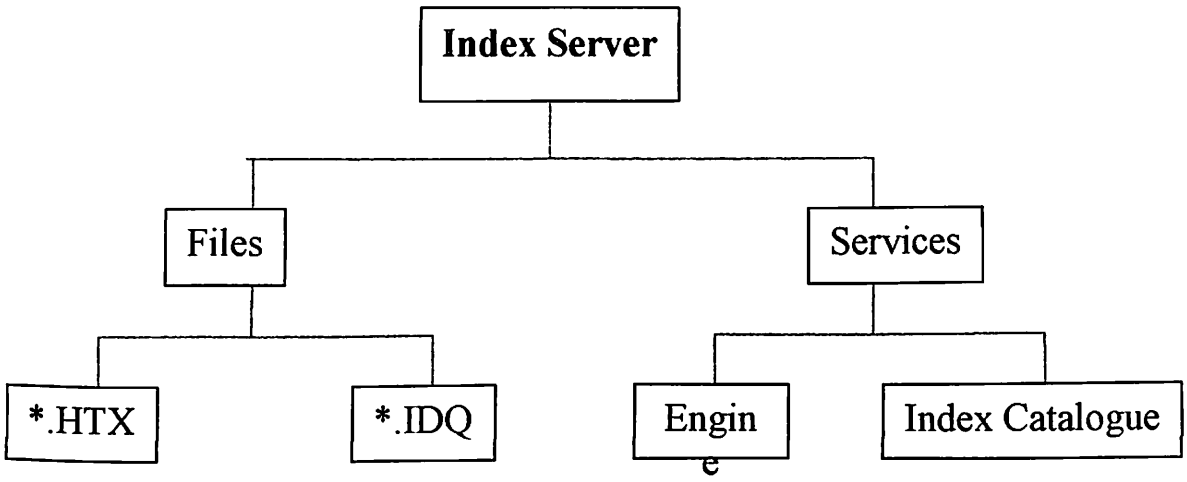
Leaving the weighting up to the user is not common, because research has shown that users are not particularly good at determining the relative importance of terms in their queries. They can't make this determination for several reasons. First, they don't know what else exists in the database, and document terms are weighted by being compared to the database as a whole. Second, most users seek information about an unfamiliar subject, so they may not know the correct terminology.

Few search engines implement system-based query weighting, but some do an implicit weighting by treating the first term(s) in a query as having higher significance. The engines use this information to provide a list of documents/pages to the user.

After this final step, the expanded, weighted query is searched against the inverted file of documents.

7.3 Index Server

Index server is a server-side executable that does full-text extraction for creating index files for fast web-based queries.



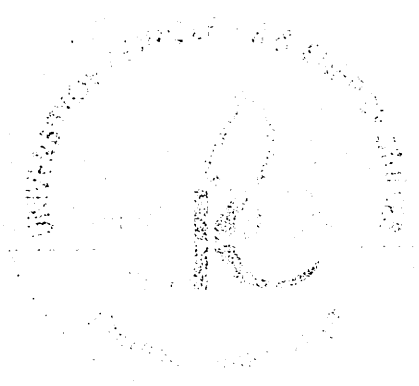
The core of Index Server that is most concerning is made up of two types of files: *.HTX - The template file. *. IDQ - This file tells Index Server what to do. It tells Index server **what to search, how to organize the data, and what to send to** the HTX file. The HTX file then takes the data and formats it so it's pretty for the user.

The core of Index server is the Index Server EXECUTABLE- The engine, which does the most work: The other file(s) is the Index CATALOGUE- Index Server makes an index catalogue that is made up of "keywords" that points to the files we / the administrator have chosen to index. Index server transparently maintains this catalogue once we've set everything up. Basically when it senses that it's got a little downtime, it will start crawling through the designated directories making indexes of keywords. It updates, adds, and deletes keywords and keeps our indexes up to date...automatically!

Currently, Index Server can perform full text extraction for all of Microsoft's file types- *.doc,*.xls, *.mdb, etc. Microsoft has also provided a filter interface for third parties to develop filters that will index other file types but these third party filters are often poorly documented and are not supported by the manufacturers. For example, Adobe makes a filter that extracts keywords from PDF files.

7.3.1 Process:

We type in a search word or phrase and click "submit" on the asp / html page. The information is passed to the IDQ file, which defines the properties that the Index Server executable will use. This information gets passed to the Index Server engine, which crunches the information we want against its index catalogue. The information Index Server finds is then passed to the HTX file which takes all the information and filters it to make it pretty to deliver us the useful HTML/ASP file you want with hyperlinks to the information! Now, we have a page of useful data that we can integrate into any ASP application!



8. Service Oriented Architecture

Services-Oriented Architecture (SOA) is beginning to revolutionize the way IT is structured. Organizations are already moving to an SOA environment and Web Services is one of the key enabling standards.

SOA enables organizations to respond to change better, faster, cheaper, and provides agility, flexibility and cost savings. Ultimately, SOA better aligns IT with business.

Definition

SOA is an architectural style whose goal is to achieve loose coupling among interacting software agents.

Applications in SOA are built based on services. A service is an implementation of well-defined business functionality, and clients in different applications or business processes can then consume such services.

How does SOA achieve loose coupling among interacting software agents?

It does so by employing two architectural constraints:

- A small set of simple and ubiquitous interfaces to all participating software agents. Only generic semantics are encoded at the interfaces. The interfaces should be universally available for all providers and consumers.

- Descriptive messages constrained by an extensible schema delivered through the interfaces. No, or only minimal, system behavior is prescribed by messages. A schema limits the vocabulary and structure of messages. An extensible schema

allows new versions of services to be introduced without breaking existing services.

The paucity of generic interfaces means that we have to include application specific semantics in the messages.

There are a few rules to follow before terming an architecture as service oriented.

- The messages must be descriptive, rather than instructive, because the service provider is responsible for solving the problem
- Service providers will be unable to understand your request if your messages are not written in a format, structure, and vocabulary that is understood by all parties.
- Extensibility is vitally important. If messages are not extensible, consumers and providers will be locked into one particular version of a service.
- An SOA must have a mechanism that enables a consumer to discover a service provider under the context of a service sought by the consumer. The mechanism can be really flexible, and it does not have to be a centralized registry.

SOA and Web Services

SOA and web services are two different things, but web services are the preferred standards-based way to realize SOA.
Interoperability is the most important principle of SOA. This can be realized through the use of web services, as one of the key benefits of web services is interoperability, which allows different distributed web services to run on a variety of software platforms and hardware architectures.

Although the concepts behind SOA were established long before web services came along, web services play a major role in a SOA.

This is because web services are built on top of well-known and platform-independent protocols. These protocols include HTTP, XML, UDDI, WSDL, and SOAP.

It is the combination of these protocols that make web services so attractive. Moreover, it is these protocols that fulfill the key requirements of a SOA. That is, a SOA requires that a service be dynamically discoverable and can be invoked. This requirement is fulfilled by UDDI, WSDL, and SOAP.

SOA requires that a service have a platform-independent interface contract. This requirement is fulfilled by XML.

SOA stresses interoperability. This requirement is fulfilled by HTTP.

This is why web services lie at the heart of SOA.

9. Web Services

For years, IS teams have faced a thorny problem: how to flexibly modify, increment, and connect heterogeneous applications to meet the requirements of business. The Web services paradigm has emerged as a powerful mechanism for integrating disparate IT systems and assets. Combining the best aspects of component-based development and the Web, Web services leverage the Service-Oriented Architecture (SOA) concept.

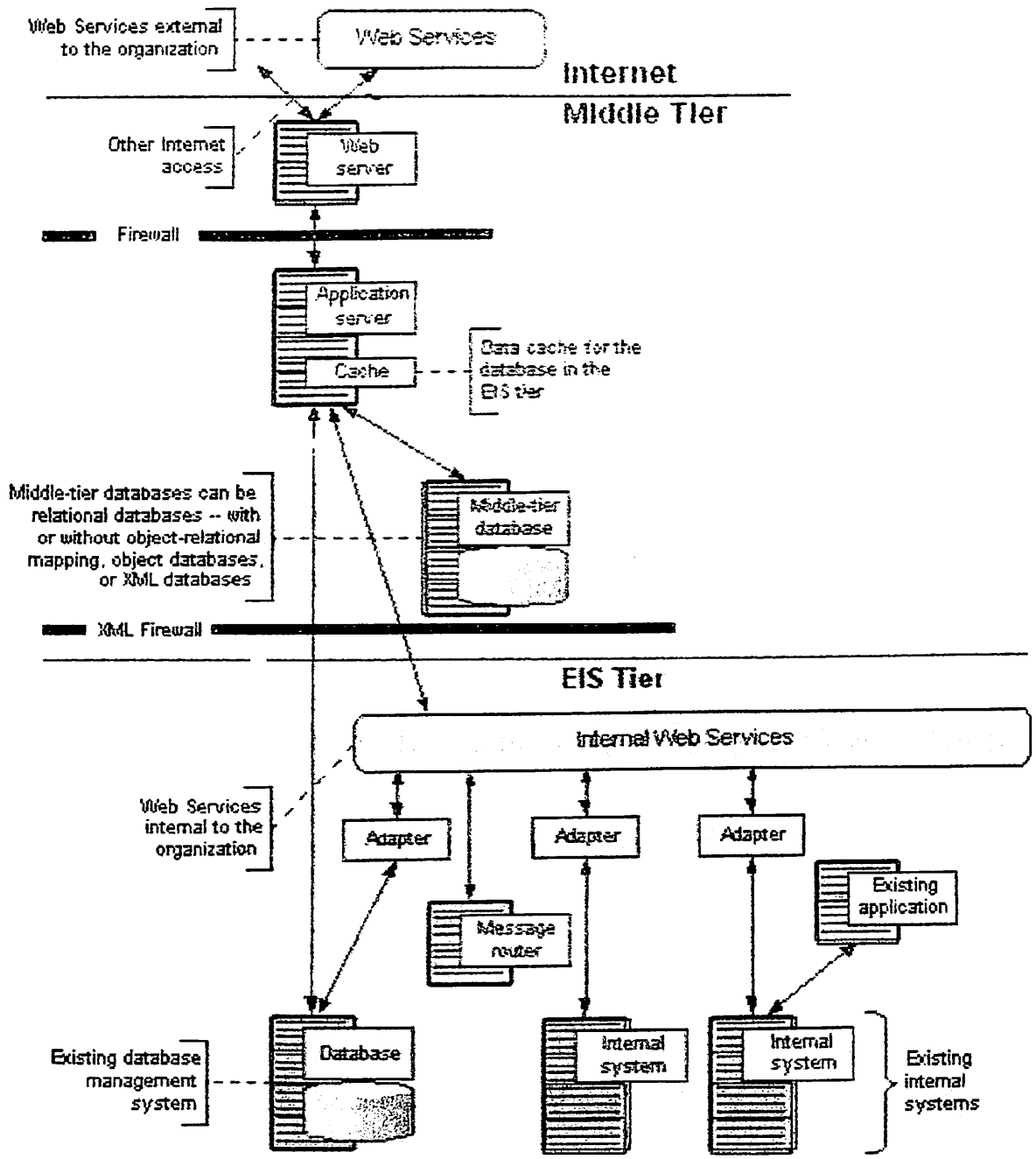
A "service" is a network-enabled component. Like components, services in general (and Web services in particular) represent functionality that can be easily reused without knowing is the details of how the service is implemented. And, since Web protocols are completely independent across vendor, platform, and language implementations, the result is an application that integrates well with the rest of the enterprise while being flexible enough to modify, as business needs change.

Applications designed using SOA can provide the same functionality as that found in a Monolithic architecture coupled with the following additional benefits:

- Easier extension of legacy logic to work with new business functionality
- Greater flexibility to change without the need to constantly re-architect for growth
- Cost savings by providing straightforward integration

8.1 What are Web Services?

A Web service is an application service (i.e. a network accessible function) that can be accessed using standard Web protocols. Web services are applicable to any type of networking environment and can support business-to-consumer, business-to-business, department-to department, or peer-to-peer interactions.



"Web services" indicate the set of standards that ensure interoperability between services, especially when those services are meant to be accessible over standard Web protocols.

Key features of creating and using Web services are:

- They are accessible using standard Internet protocols such as HTTP or SMTP. Web services communicate through platform-independent Web protocols, facilitating the integration of heterogeneous environments.

- They are distributed, which means that a Web service will usually reside on a different server than the applications that use the service.
- They can be centralized to a single source. This means that once coded it can be accessed many times from many projects. In other words, Web services allow increased code reuse.
- A single Web service isn't a full application, but rather a standalone function that can be called by many different applications.
- A Web service can be self-describing. This enables businesses and applications to find and use Web services through automated processes and Internet registries (an electronic yellow pages to let other programs find the service).
- Web services standards define an interface and communication protocol that can be invoked from an application client or provided through a server.
- The Web Services Definition Language (WSDL) adds a layer of abstraction between implementation and interface, providing a loosely coupled application that results in future flexibility.

All of these features add up to make a Web service a reusable component that can broadcast its functionality across an Internet network.

Central to the Web services approach is that it is based on widely accepted technologies and commonly used standards. This enables companies to transition to a SOA incrementally with minimal risk and at low cost

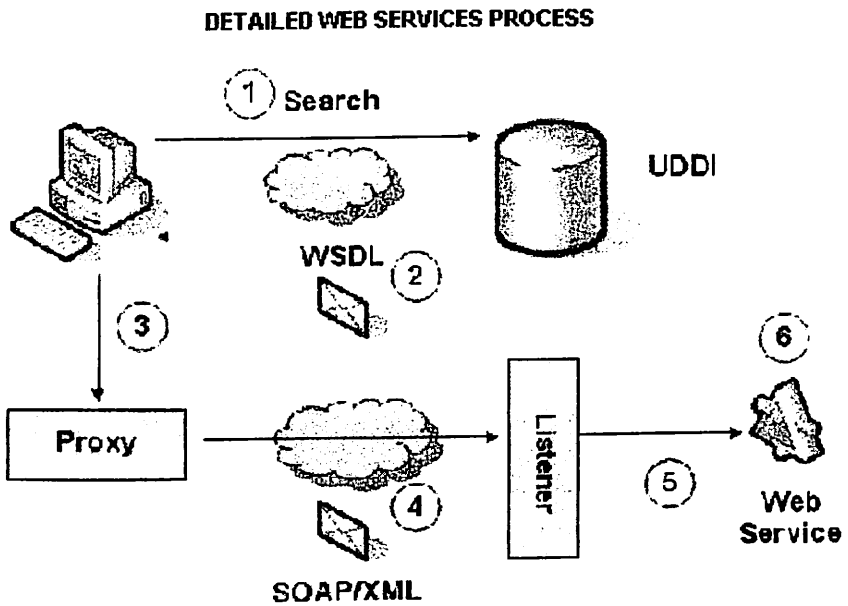


Figure 1: The process flow of a Web service

Web Services and Standards

Web services are built upon open, often already widely adopted standards such as HTTP and eXtensible Markup Language (XML).
A few of the major Web services standards groups are listed below:

- **W3C (World Wide Web Consortium)**
- **OASIS**
- **WS-I (Web Services Interoperability Organization)**

Specifications that play a vital role in Web services

- XML: eXtensible Markup Language

XML refers to the specifications surrounding custom document types using a human-readable format. XML specifications enforce document rules, similar to the English language, using words, punctuation and an accepted alphabet. Web services communicate by using XML to describe their interfaces and to encode their messages.
XML-based Web services communicate using standard Web protocols like Simple Object Access Protocol (SOAP), Universal Description, Discovery and Integration (UDDI), and Web Services Description Language (WSDL) to define the interaction. These standards use XML interfaces and messages that any application can interpret. XML allows developers to create their own customized tags, enabling the definition, transmission, validation and interpretation of data between applications and between organizations.

- SOAP: Simple Object Access Protocol

SOAP is a standard that represents a lightweight "envelope" containing the message payload as it moves between service producers and consumers. It is an XML-based standard that describes the contents of a message and how to process it, and offers a transport binding for exchanging messages. Adjuncts to the envelope and binding framework include a set of encoding rules for expressing instances of application-defined data types and a convention for representing remote procedure calls and responses.

- WSDL: Web Services Description Language

WSDL is an XML format for describing network services as a set of endpoints operating on messages containing either document-oriented or procedure-oriented information. The operations and messages are described abstractly, and then bound to a concrete network protocol and message format to define an endpoint. Most often, these messages are bound to the SOAP protocol and the HTTP transport, but these are not the only set of bindings supported. WSDL's abstract nature for describing services makes it very flexible for describing complex Web services applications.

- UDDI: Universal Description, Discovery and Integration

UDDI represents a set of protocols and a public directory for the registration and real-time lookup of Web services and other business processes. In many ways, UDDI models a "Yellow Pages," providing a listing of services available within a network. The primary benefit of a UDDI server is to provide a single point of reference to all available services within an enterprise. The UDDI server allows organizations to:

- Host multiple versions of a service
- Create aliases to services
- Limit access to specific services

While UDDI is a ratified standard, it is still gaining adoption and has been changing frequently. There are many other commercial products, known as "Web Service Management" tools, which perform many of the same tasks as UDDI while providing added benefits such as metering and inspection of the services. UDDI and Web Service Management tools help keep the connection between services loosely coupled.

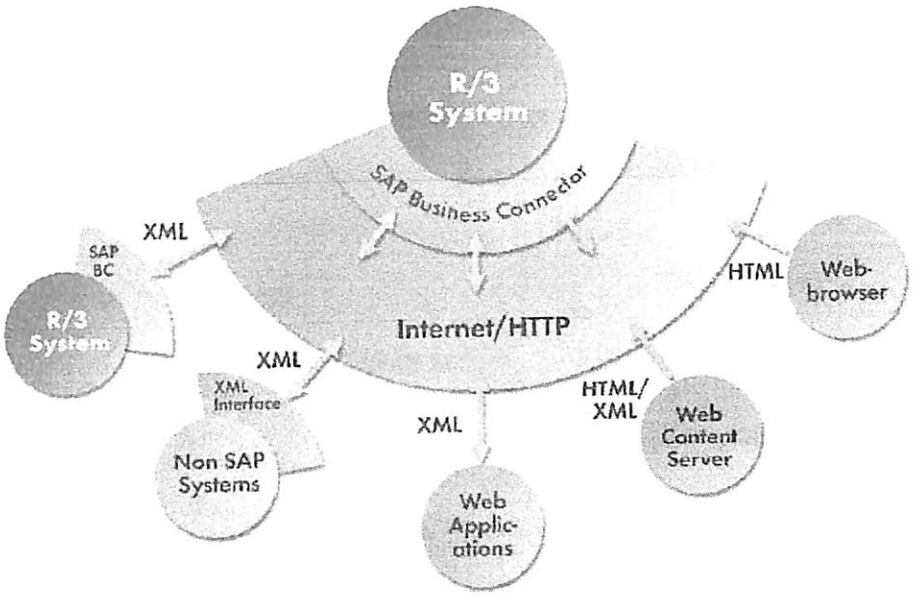
10. Generic Web Service Model

As the name suggests this model proposes the usage of a Multi-purpose web service. This model has its own unique advantages and disadvantages.

The refinery search service has a set of logic that it uses for effective and efficient information retrieval. Apart from the cost factors related to customized web services like those of SAP they also have a set of logic of their own. Will there be a logic contradiction between the two is very much a debatable issue.

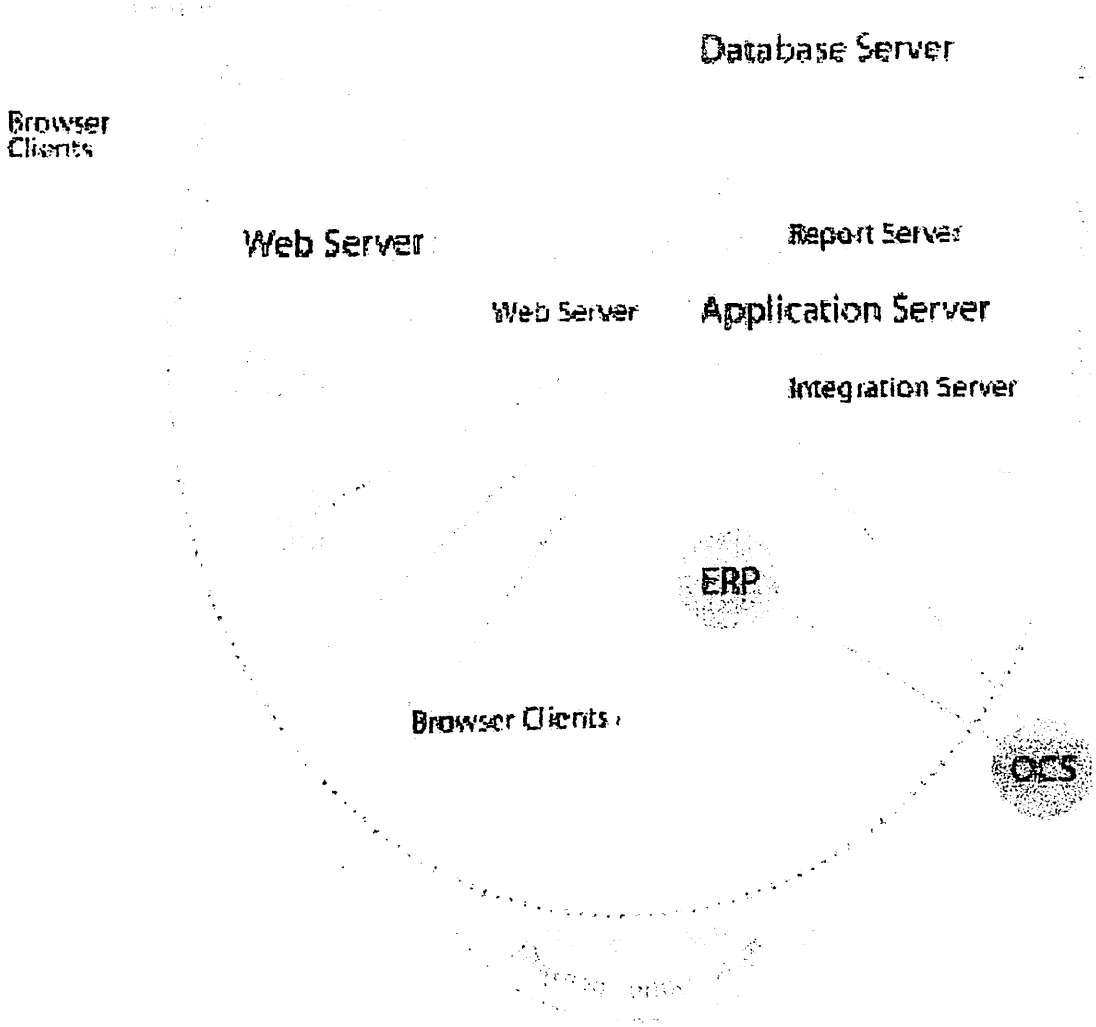
Customized Web Services will most definitely search the way they want to search. This would lead to a situation where the web Service dictates terms not the other way around.
Using a generic web service for refinery search would mean that the web service acts only as a medium and it would be dictated by the parameters specified by the user.

The latest versions of SAP are augmented by the addition of a business connector that converts XML requests into SAP requests. The Schematic below describes how this is done:



For Applications like Maximo the model does not require a connector, Maximo has this feature enabled in itself and can receive XML requests, Process the request, convert the result back to XML format and send it back.

The Schematic below explains how this model works for Maximo:



Sample manager offers a similar package like Maximo

Sample Manager's web services interface allows the LIMS to be delivered as a network service that can be integrated using common Internet standards, such as ASP, XML, SOAP and HTTP. These industry standard tools help dramatically simplify and reduce the costs of integrating Sample Manager with other laboratory and business systems, including PIMS, MES and ERP solutions.

The problem with Web Services arises when applications do not comprehend XML. Business Connectors for such applications would need to be developed.

Another viable option is by exploiting an existing ERP package. In such a competitive environment the usage of ERP package has become mandatory. One of the advantages of using an ERP package is the ease with which it integrates with other applications. Capitalizing on this feature we can integrate applications like PIMS & Tr@ction to the ERP package. The ERP package would act as a Business connector for such applications.

Advantages

- The job of a web service is to act as an interface between search tool and an application. The time and cost incurred for acquiring or developing a custom web service might not be ideal. It would be far easier and less time consuming to create a superficial layer on top of each application. The job of this layer is to convert the XML request into the request that the application comprehends.
- The advantage of using a generic web service and the usage of such interface/layers is that one can customize these layers the way they want and in the process making the search effective. By doing so the logic one has developed can be imprinted on these interfaces and not the way the application wants. It would be a comparatively difficult task to customize customized web services if it is possible.
- If and when a new application is introduced one has to only create a layer on top of the application and hence saving time and money in the pursuit of a suitable web service for that particular application.

Although some companies let you download free tools to invoke Web services, they generally don't give you much understanding of how a request is generated and usually won't go beyond showing you a raw response.

Disadvantages

1) Customized web services score over generic web services in terms of its functionality
2) Customized web services do not require any interface between it and the application.

Customized web services would ideally have a more comprehensive knowledge on the application than a generic web service.

11. Custom Web Services

Custom Web Services are those Web Services that are developed for the sole purpose of integrating with a single chosen application.
In order to remain competitive Organizations like SAP and Documentum have brought out Web Services that can interact with their applications. The advantage of using such Web Services is the ease and effectiveness of their interaction. Being an in-house product these Web Services would be having a better knowledge of their sister applications than a generic web service. This knowledge translates into identifying where to search and where not to search.
A Custom Web Service performs much more than a generic Web Service and in effect this is also one of its down side.
These Web Services perform more than just an interface; they have some in-built logics that enable them to search inside their sister applications effectively.
If a logic contradiction between the search service and the Custom web service happens then it could render the search service extremely ineffective.
In this scenario the Custom Web Services might dictate the search pattern leading to unwanted results.
Both Generic and Custom Web Services have their own advantages and disadvantages.
If there are no logic clashes or if either the Custom Web Service or the Search Service Provider can compromise on their logic then it would be better to go with a custom Web Service model.

If the Search Service Provider wants to conduct the Search purely on the intelligence of the Service then the Generic model would be a better option.

12. Search Service Intelligence

The distinction between Refinery Search Service and other Search Services is primarily its domain intelligence.

The ability to preempt User needs leads to faster information retrieval.

This intelligence is dictated by a set of Predetermined logic and it is this logic that separates it from the others

Following are the logics that have been developed.

1) User Mapping

Once the user logs on to the SharePoint portal and is authenticated, the search service immediately retrieves user information.

There are two ways in which this can be accessed

a) Accessing from Active Directory

b) Accessing from Exchange Server

SharePoint Portal provides a unique feature where it can retrieve Outlook address book information from the exchange server provided the SharePoint & Exchange server have given the necessary rights for such an activity.

The information that it has retrieved is stored and updated. The update is done based on a specific time frame that the SharePoint administrator has set.

The search service then directly matches the login id of the user with the id in the address book and retrieves the corresponding information.

The address book lists out details like Employee name, Designation, Area and Location etc.

If the Designation of the Employee is that of a RCC Superintendent then it takes all data sources pertaining to that designation and indexes it.

2) Default Parameters:

For quick and efficient search it would be ideal if the User inputs all the parameters that have been asked.

More often than not this would not be the case. There would be a combination of many factors as to why the User ignored those parameters.

The challenge for the search service is to still get the information and list it to the User's Satisfaction.

The search service does a complete search and the importance of these default parameters only comes while ranking the data sources

Let us take a look at all the Default Parameters.

a) The Default Parameter for Area would be the Area where the User belongs.
b) The Default Parameter for Location would be the location of the User.
c) The Default Parameter for date would be the latest date
d) Based on the User information retrieved from the address book the default parameter for Process, Product and Supplier is set.
e) The Default Parameter for Popularity would be "Retrieve most popular records".

3) Ranking Methodology

This is one of the most important factors for a Search to be termed effective. It doesn't help the user if the data he is searching for is listed in the 10th page. More often than not the user would not check any data sources beyond a second list.

It is not enough if the Search is efficient; it also has to be effective.

The ranking is pretty much direct if the User has filled all the parameters or has not filled any Parameter.

The real issue arises when the User has opted only for certain parameters and chose to ignore the others.

The Search Service prioritizes those Data Sources that have satisfied the maximum number of parameters.
If two data sources end up with the same number of parameters then the values of their default parameters are added up. The data source having the maximum value is given priority.

The values that are assigned for various Default Parameters are listed below.

Default Parameter	Assigned Value
Area	9
Process, Product & Supplier	4
Refinery Location	3
Date	2
Popularity	1

4) Matching Key Words

It is also required to incorporate a whole set of Refinery abbreviations into the search service.

If the user wants to search for SKO then using the list of abbreviations the search service needs to understand that SKO means Superior Kerosene Oil.

This helps in listing out those documents where the word SKO is missing but Superior Kerosene Oil is very much present. This feature is available SharePoint Portal; SharePoint search contains a thesaurus to which the abbreviations need to be added.

Similarly If a BP user types in the word "Ultimate" then results pertaining to the Brand "BP Ultimate" should be given precedence over results where ultimate is used generically.

13. Mapping of Process with Employees:

Enterprise Asset Management:

sers→	CFO	Strategic Planner	Corporate Controller	Asset Manager	VP Operations	Project Manager	Designer	Chief Engineer	Chief Purchaser	Engineer	Sales Representative	Operations Supervisor	Maintenance Technician	Purchaser	Oper Tech
ocesses ↓															
isiness & Investigation Planning															
efine Capital Project (Scope & melines)															
btain Project Approval															
itiate RFQ (Request for Quotation)															
larify & Refine Technical Bid															
ubmit Commercial & Technical Bid															
nalyze & Award Contract															
ollaborate during Project Execution															
and over Project to Execution															
and Over Equipment to Operation															
tart up asset															
perate Asset Safely and Analyze set Performance															
erform Maintenance Activities															
rocure MRO material & Services															
arry out services															
nalyze & Optimize asset portfolio															
efine Decommissioning Project															

CASE STUDY

14. Case Study

The following Case study takes you through the entire refining search Service.

The following assumptions are made for this Case Study

1) We have taken the case of Mr. John smith.
2) Mr. Smith fits in the job description of HCC Supervisor.
3) The Search Service is based on SharePoint Portal
4) Employee details are present in Outlook Address Book
5) Outlook Address Book has its storage repository in the Exchange Server

1) When John Smith logs in:

Mr. Smith has just arrived at his workstation and is all set to resume his work; he turns on his desktop and uses his login id and password to access SharePoint Portal. Once the Id and Password are validated it sets of a trigger of activities that would simplify his day at the office.

As soon as the validations are over the Search Service tries to retrieve information pertaining to Mr. Smith. The Search Service accesses a copy of the address book stored in SharePoint and gets the required information. From the data gathered it was found that he is the HCC Supervisor.

The Search Service identifies the Structured & Unstructured Data that he would be interested in and keeps it ready for usage. The unstructured Data Content is directly retrieved from the index server where as for the Structured Data the Search Service identifies the Web Service applicable for that data source and retrieves data using that Web Service.

2) Mr. Smith wants to search:

Mr. Smith realizes that his work is incomplete without a certain bit of information. He clicks on the Search option and the screen dynamically changes to a Search Service giving him a lot of options.

He types in the word CDU, luckily for him the search service is intelligent enough to comprehend that CDU means Crude Distillation Unit. Therefore his search would not be limited to data pertaining to CDU but also data pertaining to Crude Distillation Unit.

3) Mr. Smith clicks on the location:

Mr. Smith who is the HCC Supervisor in Bulwer refinery wants to check out CDU data in Kwinana refinery, so he clicks on his choice of location.

The search service would only search for CDU data pertaining to Kwinana.

If by accident he does not click on any location then by default CDU data of all refineries would be searched and CDU data of his refinery and specifically data present in HCC would be listed first.

4) Mr. Smith wants to search inside SAP:

Mr. Smith is aware that his data is present in SAP, so he clicks on SAP. Once this is done the search service retrieves the corresponding web service applicable for SAP from the UDDI. There are two ways in which this can be done.

a) Using SAP web service
b) Using a generic web services

The Pros and Cons of both have already been discussed earlier.

A Similar approach is used for Searching inside Sample Manager LIMS

5) Mr. Smith changes his mind:

Instead of SAP if he had wanted to Search inside Maximo or PIMS, then the first step would have been obviously to click on his desired application.

A generic web service is what the search service uses for these applications. Maximo Software can ideally comprehend XML requests but in the case of PIMS, Tr@ction and other applications, which may or may not comprehend XML requests, the search service employs the usage of business connectors, which can convert XML requests to the form, which the application understands.

6) Search inside Documentum:

All Unstructured Data Content is ideally stored in side Documentum. The unstructured data could be a word file, PDF file, Spreadsheets, E Mails etc.

Now Mr. Smith decides he wants to search inside documentum. He clicks on documentum and instantly finds a drop down asking him the specific file format. If he were looking for a Word file he would click his choice. If he is unsure and does not click his choice then all unstructured content would be listed

7) Search within a Specific date:

Mr. Smith is looking for a certain Work Order applicable for a certain date and time. The Search Service gives him options to narrow down his search based on these parameters.

8) Search within a Specific Area:

As said earlier Mr. Smith who is the HCC Supervisor decides to Search for CDU information in a specific area. The areas have been classified as Crudes Area, Cracking Area, and Lubes Area etc. This is done so that he can search for data pertaining to a certain area. Further each area have been subdivided into Sub Areas, for example Cracking Area has been subdivided into ACCE, HCC, FCC etc.

So if he is searching for crude data in FCC, all he has to do is just click the appropriate parameters.

9) Mr. Smith is looking for Process related information:

Mr. Smith knows for a fact that he is not interested in Supplier related information nor product related information. His focus is primarily on the process. The search service understands this need of his and gives him options for basing his search on Supplier, Process or Product related information.

10) Mr. Smith Prefers Popularity:

As he glances down the Search Service User Interface he finds an option by the name Popularity. Mr. Smith thinks that his data would ideally be a very popular one and bases his search using this parameter. This Popularity is based on the number of times the Data has been accessed

11) Mr. John Smith is using the Service for the first time:

In this case there would not be any data in the index server, so as soon as he logs in the search service retrieves data pertaining to his area and indexes only those data.

12) Mr. Smith does not use the parameters:

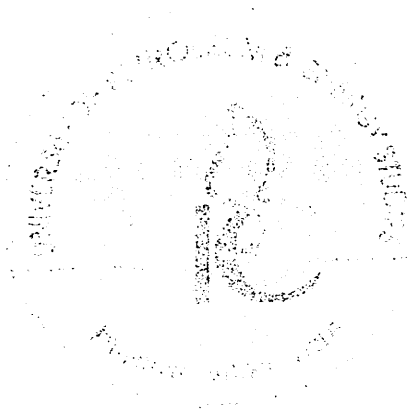
Mr. Smith does not know what to search, where to search and how to search. He is not conversant with the applications and does not know where his data lies.

Mr. Smith types in CDU and clicks on the Go button, the search service use the earlier information that it had extracted from the address book and based on that conducts the search.

The search service conducts a generic search and lists the results based on its own intelligence. CDU data satisfying all the default parameters would be given top priority. Each default parameter is set

a certain value. The ranking is based on which data source obtains a better score.

This intelligence is what separates the refinery search service from other similar offerings.



Future Scope

The ease of Customization and the inherent intelligence that it possesses has made us take up the case of SharePoint Search as our base Search.

It is very much probable and possible that during or after implementation of the Refinery Search, newer and better technologies would have evolved.

There is always scope for improvement and the Search Service very much epitomizes this fact.

A reasonable level of Refinery Logic has been incorporated into the Search Service but the level is by no means saturated.

The effectiveness of the Search Service gets enhanced with the addition of more refinery logic.

Notwithstanding the promise that Web Services offer, their ability to interact across multiple applications, the fact is that Web Services are still at a very nascent stage. There are still a lot of doubts being raised in terms of their effectiveness.

The best way to realize SOA right now is through Web Services. The future might hold better ways of realizing such architectures.

The Refinery Search Service is very much feasible, can be developed and further customized according to the need of each refinery.