30 Digits
LINKING PEOPLE TO CONTENT

Technical White Paper

Overview

When considering a knowledge management platform for any topic, there are certain technical functionalities and architecture requirements that should not be overlooked. This document addresses these points.

It starts by covering the core features involved in the methods used in the information discovery and retrieval commonly referred to simply as "search" which are vital to getting the right information out of the system. Then it moves on to the abilities and methods used for collaboration to make sure the right people get informed and can share knowledge efficiently. Next the exceptionally important, although sadly often overlooked, process of data acquisition is covered to show how the system is fed with quality data in the first place to assure the rest of the process can proceed on a solid foundation. Finally, there is a wrap up on architecture and administration which make a large scale and secure system possible with a low total cost of ownership with minimal administration necessary.

- **Information Discovery and Retrieval / Search**
- **Collaboration**
- **Data Acquisition and Enrichment / Spidering**
- **Architecture & Administration**

Contents

Information Discovery and Retrieval / Search

Information Discovery and Retrieval or to put it simply is the art of getting to the right information quickly and easily. That sounds simple enough, but there are many aspects to consider to do it right. It is easy to have a simple algorithm that picks out words that match exactly to words in documents in the index. How do you make sure you are really getting all of the desired documents if the word forms are slightly different than what was typed? How do you prevent having to scroll through pages and pages of results? How do you catch typos and other errors early? How do you actually discover information which you didn't even know was there? In this section, we'll be explaining how we've answered these question in the IDS platform.

The Algorithm

This is a topic often debated by the geeks of the search world but of little interest to the rest of the world even though it influences the way any search system works. There are actually many algorithms out there used by commercial search systems and academics for many purposes. Bayesian Inference is one algorithm which is often enough used based around statistics. While delivering some benefits in nearness and probabilistic similarity, it often delivers results which don't make sense from a normal perspective. It also has complications in tuning due to its complex nature. The PageRank like algorithms made popular by Google can be beneficial for popularity searches but does not really meet the needs for most businesses uses.

Therefore, in choosing the algorithm for IDS, we wanted an algorithm that would be easy to understand, simple to tune for specific requirements, completely open and transparent, deliver quality results, and still be fast and scalable. The combination of the Boolean model and Vector Space Model has proven over time to accomplish all of these tasks well. With Lucene using these in way that has excellent performance, it was the obvious way to go.

A very rough explanation of this model, often referred to as TF/IDF, is provided here. Documents where a term appears frequently (TF – Term Frequency) are important for searches looking for that term. Term that show up in the entire index very often are not very important (IDF – inverse document frequency). When taking these two items into account, one gets back documents pertaining to the terms searched from documents where these words appear often and words that are more rare have a higher relevancy to the results.

If you want to get more into the math and exact details of the formulas displayed below, the following link provides a nice explanation of both the conceptual and practical formulas: http://lucene.apache.org/java/2_9_1/api/core/org/apache/lucene/search/Similarity.html

$$\text{score}(q,d) \;=\; \text{coord-factor}(q,d) \cdot \text{query-boost}(q) \cdot \frac{V(q) \cdot V(d)}{|V(q)|} \cdot \text{doc-len-norm}(d) \cdot \text{doc-boost}(d)$$

Lucene Conceptual Scoring Formula

$$\text{score}(q,d) \;=\; \text{coord}(q,d) \cdot \text{queryNorm}(q) \cdot \sum_{t \text{ in } q} \Big(\text{tf}(t \text{ in } d) \cdot \text{idf}(t)^{2} \cdot t.\text{getBoost}() \cdot \text{norm}(t,d) \Big)$$

Lucene Practical Scoring Function

Also, the following site provides a presentation with a good overview on the history and development of the algorithm and Lucene: http://www.scribd.com/doc/18004805/Lucene-Algorithm-Paper

Before moving on to the next point, it is worth noting that we are not alone in choosing the Lucene. It is also used by Wikipedia, CNet, LinkedIN, Monster, IBM’s OminFind , and many many more: http://wiki.apache.org/lucene-java/PoweredBy.

Linguistics

Algorithms are great but even the best ones break down when dealing with languages. Hence, there needs to be some consideration on this side to assure the best quality.

Summarization

There isn't any other item that takes up more real-estate on a result page than the summaries. They are also the window into the documents contents at an early stage and getting them right can prevent a great deal of wasted time browsing documents. Thus, we've created two summaries. The first is what one would typically find called the highlighted summary. This grabs snippets of text out of the document where the search terms are found. This gives the user a quick perspective on how the terms they've searched for are used in the document.

The second summary is one that summarizes the entire document. It does this by avoiding areas of the document not of interest and focusing on the terms in the document that occur most frequently but are not just filler words like "the", "of" or "a". The combination of these two summaries allows the user to evaluate the relevancy of the document quickly and easily from their perspective after the algorithm has done the initial listing.

Additionally, where possible, we have added thumbnails to the summaries that some sorting can take place even quicker through visual ques.

Stop words

Often "stop words" are mentioned as another feature to give the creditability of another feature. In actuality, "stop words" have fallen out of fashion. The reason being that with a reasonable algorithm that takes into account the amount of times a word appears in the entire document base as the inverse document frequency (IDF explained previously) these words fall automatically in the category of practically irrelevant terms effecting relevancy. In some cases though, they can be very beneficial in making the difference between a roughly accurate match to an exact match. Thus, we actually don't use "stop words".

Synonym Expansion

In many cases, a term searched has more than one synonym. The user typically doesn't know or want to think about all of these terms. Without these terms, the results are often leaving out many relevant documents. For example, someone might search for "USA". They will only find documents containing this phrase. IDS can suggest synonyms which the user can then select to expand their query. This would result in the original search for "USA" to be expanded to include "America" and "United States".

Lemmatization & Stemming

Stemming is the classic feature in any good search engine which handles a situation where someone searches for "runs" and also gets results with "run" and "running". This is important for any search system to work well and works best when tuned for the language. Here are just some of the languages supported: English, French, Spanish, Portuguese, Italian, Romanian, German, Dutch, Swedish, Norwegian, Danish, Russian, Finnish, Hungarian, Turkish.

Lemmatization is not that often talked about because not many systems have it. The primary reason most likely being that with English, stemming tends to do a pretty good job and the advantage of lemmatization are minimal. In a language like German though, it is vital. When a word has so many different variations which are not simple changes to ending and where words are often combinations of words, the results returned without a lemmatizer are only a fraction of the relevant results. Lemmatization makes sure that the root forms of the words are stored in the index and that these words are also searched no matter which form is originally in the document or given in the query.

Getting off to a good start / Query related functionality

Although it isn’t discussed much, there are a few things that a modern search system has to get ride to assist the user with their initial search.

Auto-complete

Auto-complete is no longer a nice-to-have. It has become a must. To make it just a bit better, the terms should be pulled from the actual index instead of just from a dictionary. This reduces the risk that a search will return no results and takes into account new terms, product names, and other terms not found in a typically dictionary. Not to be forgotten though is the removal of certain meta-information to prevent too many strange terms appearing which might confuse the end user.

Did you mean? / Spell check

The “did you mean?” functionality has also become a standard requirement even though it is missing often from many systems still. The suggestions here are determined on their similarity to the search term to other terms in the index.

Operators

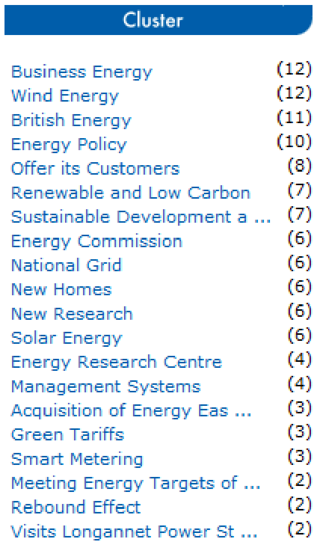
Although Boolean, NEAR, and other operators are typically only used by Power Users, these are vital to round out a good system. All of the standard operators are supported in IDS.

Navigating the Information Flood

Once the user has sent off their query and the algorithm and linguistics have done their job in delivering and sorting a comprehensive list of results, the amount of data can still be quite large. Hence, it is vital to provide the user a way to refine the list quickly and easily to come to the right document within just a few clicks. For this purpose, we have employed a number of different methods each providing its individual advantage. They can be used independently or in conjuction.

Dynamic Clustering

Clustering dynamically on the top results provides a way to see the top terms in the top results. Then with just one click, the user sees just those results in the selected cluster and can even refine further. With the few systems that provide this potentially powerful feature, it still often provides poorly implemented. The main cause for this is that the clusters are based on simple algorithms gathering the frequency of terms on a selection of documents. This sounds good enough, but in most environments the most frequent terms are the text included in the headers and footers, names and organizations listed in template documents that have been copied, or other irrelevant meta-data. We've thus gone a step further and made sure the clusters only work on a linguistically extracted summary from the main body of text. This avoids all the standard issues and generates usable clusters.



Cluster	
Business Energy	(12)
Wind Energy	(12)
British Energy	(11)
Energy Policy	(10)
Offer its Customers	(8)
Renewable and Low Carbon	(7)
Sustainable Development a ...	(7)
Energy Commission	(6)
National Grid	(6)
New Homes	(6)
New Research	(6)
Solar Energy	(6)
Energy Research Centre	(4)
Management Systems	(4)
Acquisition of Energy Eas ...	(3)
Green Tariffs	(3)
Smart Metering	(3)
Meeting Energy Targets of ...	(2)
Rebound Effect	(2)
Visits Longannet Power St ...	(2)

Energy Cluster

Faceting

Properly done facets take search to a completely new level. With good facets, one knows the people, places, locations, and any other entities defined within the entire result lists or even the entire document corpus. Many systems provide only poor shadow of the proper capability. Following is a list of the difference between proper facets and poor facets.

Poor Facets	Proper Facets
take into account only a subset of the documents, typically around 100	take into account every document in the result set
supply no counts on the number of documents containing the facet	have counts next to every facet showing the number of documents containing it
no way to remove previously selected facets without starting the search over from scratch	adding and removing facets and updating the results and other facets (with their accompanying counts) as the changes are made
gain facets from a small set of meta-data available only in highly structured data sources	can gain facets from meta-data found in structured as well as unstructured data and can even extract facets from the main body of text based on linguistics or lists

As our system was built to handle facets properly from the core architecture and through our spidering and entity enrichment technology (covered later in this document), we provide quality faceting.

Channels

What we refer to as channels are simply pre-defined queries typically defined by a topic expert. The technology behind this is not special. The benefit though can still be high when handled correctly. This feature allows a topic expert to define a query which returns results for a specific topic. Essentially, this allows someone who knows the material very well to prepare content for other users to get the latest and relevant data on a topic with just 1 click.

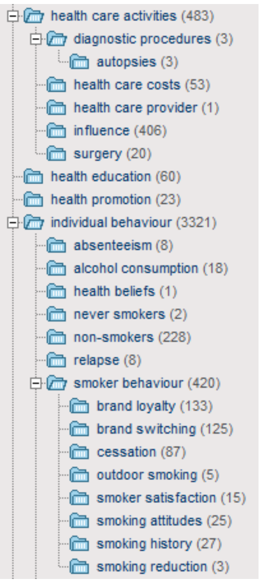
Where this feature fails by some implementation is the complexity or hurdle for the topic expert to apply their knowledge because they would also have to be a search expert to define their queries. Thus, here we've made the administration of the channels possible with simple click and drag features based around the normal search process.

Taxonomy

Depending on customer requirements and the topic area, taxonomies can be a very powerful tool in narrowing down the results. They are pre-defined hierarchical categories to a specific topic put together by topic experts. Regularly updates are also available for topics which are new are quickly changing.

We provide a way to that all content pushed into the system gets tagged by one or more nodes of the taxonomy. This in turn gives an easy to use tree like navigation method based on known categories from people in the given industry.

Many of the same poor implementation of facets are also done for taxonomies. As we tag the documents at index times, the topics actually become facets (as far as the system is concerned) allowing us to hold the same high quality standards for taxonomy navigation as held for facets.



- health care activities (483)
  - diagnostic procedures (3)
    - autopsies (3)
  - health care costs (53)
  - health care provider (1)
  - influence (406)
  - surgery (20)
- health education (60)
- health promotion (23)
- individual behaviour (3321)
  - absenteeism (8)
  - alcohol consumption (18)
  - health beliefs (1)
  - never smokers (2)
  - non-smokers (228)
  - relapse (8)
  - smoker behaviour (420)
    - brand loyalty (133)
    - brand switching (125)
    - cessation (87)
    - outdoor smoking (5)
    - smoker satisfaction (15)
    - smoking attitudes (25)
    - smoking history (27)
    - smoking reduction (3)

Excerpt from Tobacco Taxonomy Navigation

Similar documents

For the case when a document is found which is of interest and the reader would like to see more documents of a similar nature, we've implemented the "similar documents" feature. This works on the principle of comparing the highest valued terms (see algorithm section) to those of other documents in the index. Those which have the highest similarity are then returned with this function.

Collaboration & Workbooks

An easy to use collaboration functionality is built into the system. This allows users to create Workbooks of projects they are working on, areas being researched, or topics they need to stay up to date with. Virtual teams can be created around Workbooks as well by inviting other users to share a Workbook. Then as users add saved queries, documents, notes, and alerts; the other team members are notified, and their shared Workbook is updated. These virtual teams can be created out of anyone that has access to the system allowing teams to be created across divisional or geographic boundaries. Security remains persistent though never allowing access to a document without the proper privileges. The information cooperation and collaboration between people, teams, and departments can bring tremendous benefits in knowledge sharing and synergies that weren't previously possible.

Active Queries & E-Mail Alerts

Queries entered which are on topics which the user wants to regularly reviit can be saved with one simple click to workbook. The queries that are saved can be made Active or set to be E-Mail Alerts. These Active Queries create a custom home page of result documents for those topics of interest identified through the saved queries made active. Without even logging in to the system, the E-Mail Alerts will push the top documents to the user at regular intervals whenever new information comes into the system on the specified topic keeping the user up to date with the latest news and changes.

Query History Analyzer

Query History Analyzer is designed to allow the user to analyze trends and changes over time to various topics. It does this through allowing the user to turn any query into a series of time slices displaying the documents that were published or indexed in those time slices.

Bookmarking documents

Documents can also be bookmarked for easy future reference at any time and for sharing with other colleagues the most important document to a particular topic. Notes can even be added to them to document why they are interesting or to remind of an action that needs to be taken.

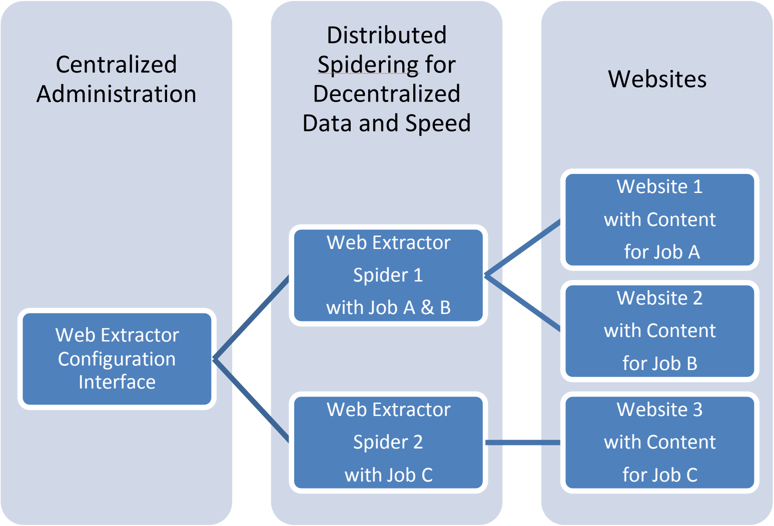
Data Acquisition / Spidering

Crawling the Internet

Web spidering is a typical and common task for which each search engine has a solution. There are even free web spiders that make their source code available on the Internet. Thus, many would ask, “Why build another web spider?” The answer is that it is easy to build a web spider that goes to a web site and pulls down the text from the pages that is readily available, but it is something entirely different to create a web spider that extracts the valuable data from a website which is needed and only that data.

Most web spiders were built with the mind-set that it would never be possible to have a web spider pull out precise information from the almost endless variety of web sites on the Internet and most likely the hope that no one would ever look too closely at the large amount of garbage information retrieved and the lack of information that was intended to be retrieved. 30 Digits development team has seen too many projects where the off the shelf web spiders just did not meet the customer’s needs. Thus, the 30 Digits Web Extractor was created.

The Web Extractor approaches spidering as an integral part of the data retrieval process. To this end, the spider has been designed to deal with various layouts of pages and structure of websites and to treat them individually. It can even parse a single page in different ways to extract multiple sections of valuable information. One example of this would be a page with information on a product and the associated reviews. The spider can retrieve all of the product data, and store that separately from each of the reviews which can also be split in to multiple documents even if they are all on one web page. The Web Extractor is essentially designed to cut out the particular areas of interest from a web site and deliver them in the appropriate text form whether that be for a search engine or a content platform that will analyze the data or reformat it for integrated display into another platform.



Centralized Administration
Distributed Spidering for Decentralized Data and Speed
Websites
Web Extractor Configuration Interface
Web Extractor Spider 1 with Job A & B
Web Extractor Spider 2 with Job C
Website 1 with Content for Job A
Website 2 with Content for Job B
Website 3 with Content for Job C

The Web Extractor is cable of handling an unlimited number of sources through its distributed architecture.

The 30 Digits development team has also built in a number of other features to make the spider stand out above the rest. Features like data qualifications, live spidering analysis, and regular expression based parsing are just the beginning.

Precision / Quality

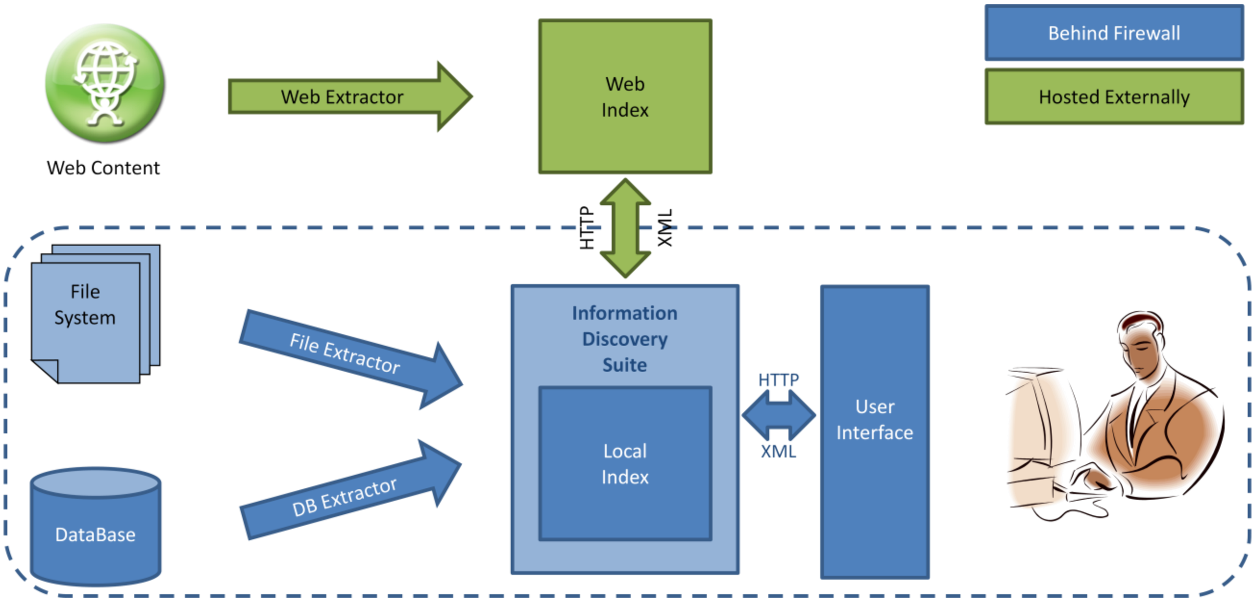
Key to making the overall search and discovery process is assuring as little useless information gets into the system. The Web Extractor does this very well by excluding not only parts of web sites but also parts of individual pages. The result is that the data that ends up in IDS solutions are purely the document information without advertisements, header & footer data, or menus.

Meta-Data & Data Enrichment

In the section before "Navigation of the Information Flood", we discussed the difference faceting makes and using information like the people, places, and locations to narrow in on the exact information of interest. Those facets as well as other features like date range specification or field limitations are only possible when that data is available in the first place. Thus, we've taken great efforts to get that data out of the source documents. The first step is to get the meta-data that is simply available like a filename, modification date, or title in a HTML meta-tag. With a little effort, most systems gather this information. We then go to the next step. This one is to find the structure within the chaos. We do this through our Web Extractor which is able to find, recognize, and extract patterns from site and page structure. This data is then gathered from all the different sources in their original formats and converted into a unified structure which normalizes the date i.e. dates written like Jan. 1, 2010 or 01/01/10 all end up in the system the same way. The 3rd step is to actually add meta-data to the document based on the actual flowing text within the article. This is done by either first defining what information is of interest i.e. all company names from a particular industry or using linguistic technology which extracts and categorizes nouns based on rules. If these steps are not cared for, the navigation and thus the end user experience is lacking. When this is done right, the user can easily research information and discover new data efficiently and easily allowing more time for analysis and adding value.

Benefit of Hosted Service for Internet Data

Sites change; pages change; technology develops; standards fluctuate; the Internet moves, grows, and develops at a phenomenal rate, never resting. What sites do you want to monitor? How many sites do you want to monitor 20, 50, 100? Who will keep the configurations up to date for extracting the data? Who will update the technology as the methods on the Internet change? Who will scale the system as it grows? Who will monitor it to assure it is always running and delivering time sensitive data? Those are just a few of the starting questions when one begins to consider gathering and monitoring data from the Internet. Most projects which bravely take on this task soon find that the complexity and effort involved was extremely underestimated and consign themselves to a slow death of their system as it falls more and more out of date and less useful from the initial implementation.



IDS architecture with hybrid of local data handled locally and web content hosted

Hence, we have come up with a different model. With this model, the entire web side of the solution is cared for by us. We know the technology, the Internet, and the challenges. We've tackled them and have experts to keep up the configurations, monitor the data, and keep the system running day and night. We also have the wonderful effect of "Economies of Scale" allowing us to invest the time to assure the quality is excellent because the costs are distributed over multiple uses. This is a win, win, win.

Local Data

In addition to the web content, we can also tap into internal sources. These can be Databases or application built around them, file systems from varied operating systems, websites like Intranets and wikis, and much more. Tapping into these sources provide the user with a custom local search experience combining internal knowledge with external market information through a single interface.

File Types

Binary files and the text within them can sometimes be important by web extraction but are essential for file system extraction and other local extraction processes. The Extractors can handle a large assortment of file types. Here is a list of some of the major ones: Microsoft Word, Excel, PowerPoint, PDF, HTML, TXT, RTF, MSG, XML, and ZIP (which are handled recursively to extract out the relevant data from each file within the ZIP). New file types are constantly being added to the import process to assure all kinds of files can be processed.

Architecture & Administration

Security & User Management

The Advantag4 solution is designed from top (the user interface) to the bottom (crawling of the data) to only deliver content to users which they are allowed to see. It handles authentication through its own login screen or via Single Sign-On which can be integrated with a local Active Directory or LDAP system. Document level security is then applied by retrieving the security information (i.e. ACLs) from the local source data and comparing them against the entitlements of the authenticated user. If the organization has a more complex field and or rule based security system, the IDS system can map that within its own security which has been designed for government intelligence agencies. Special user groups such as administrators and editors can also be easily added and administered whether they exist in a user directory system or not. The profile of each user can also be edited, deleted, or simply de-activated.

Scaling

The 30 Digits IDS is designed to grow with your organization. One can start with just a single instance on one server. As the number of users increase or the data being indexed ramps up, the 30 Digits IDS can be easily expanded. This can be simply done by upgrading the hardware or adding more servers. It can also be done in a more modular method depending on the specific needs of scaling.

The architecture is naturally distributable by design. Each of the connectors can sit locally with the data source or on separate servers. The platform itself can be virtually divided into multiple sections on one machine and physically divided across multiple machines allowing it to scale both horizontally and vertically.

Two or more instances can be deployed as duplicates of one another allowing for a completely mirrored solution. This could be for scaling purposes, but it can also be for redundancy in the case of a system failure. As soon as one system is no longer available, the users could be redirected to the other system with the user experience never missing a beat.

Thank you for your interest. IDS is a product of 30 Digits. Should you have any questions, feel free to contact us at contact@30digits.com. Also, for more information see our website.

TEL: +49 89 45 23 89 66
FAX: +49 89 45 23 89 70

Agnes-Pockels-Bogen 1
80992 Munich
Germany

www.30digits.com