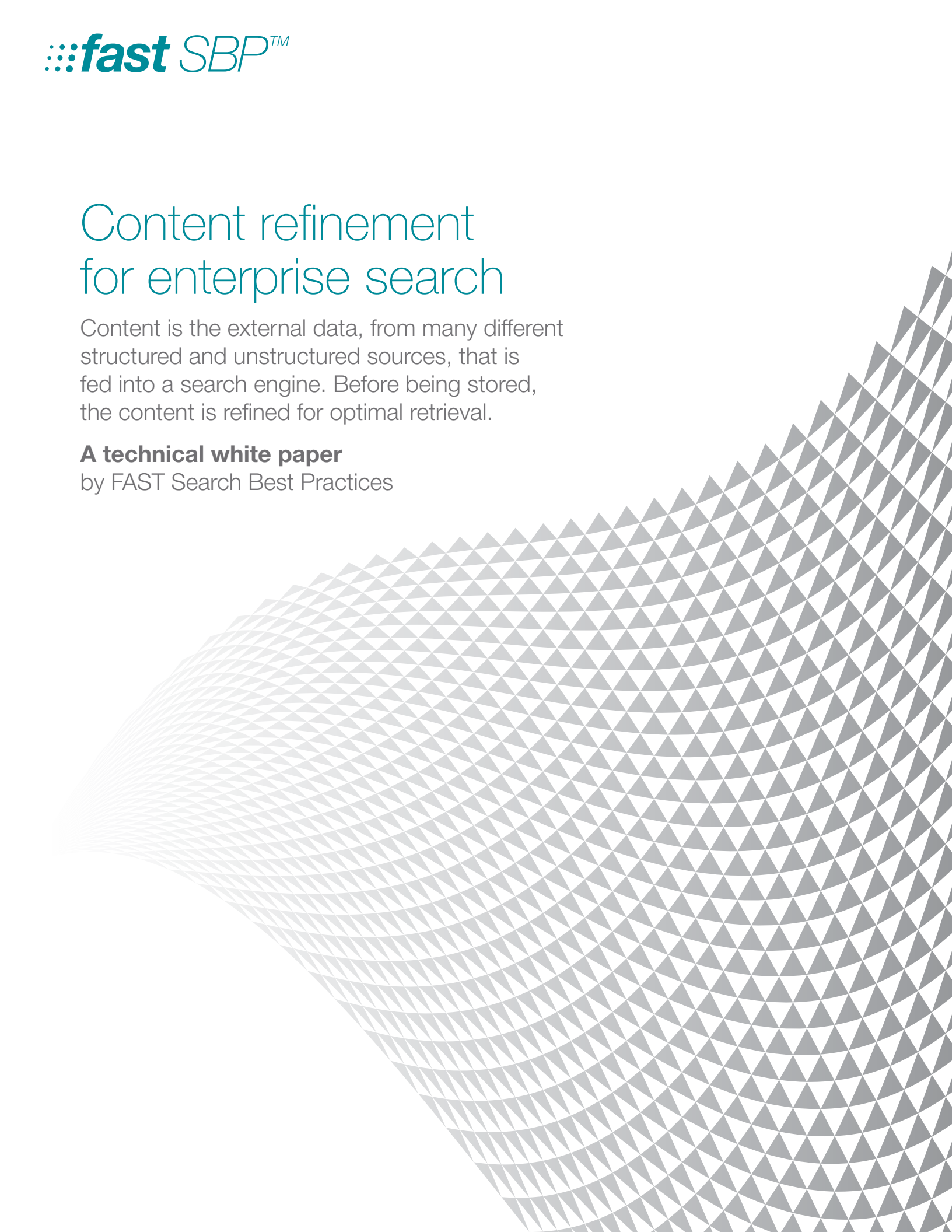
fast SBP™

Content refinement for enterprise search

Content is the external data, from many different structured and unstructured sources, that is fed into a search engine. Before being stored, the content is refined for optimal retrieval.

A technical white paper

by FAST Search Best Practices



5 things you should know about content refinement

1. Clean and normalize content to achieve the best possible relevancy during query time
2. Normalize content – ideally data (especially structured data) should be consistent and without duplication
3. Appreciate that ingestion of content will be affected by the amount and number of different types of data, in addition to the latency of the source systems
4. Optimize document processing – remove all unnecessary document processing components and choose the right processor for the content type and task at hand
5. Marry content with the appropriate document processing – language detection, synonyms, spell checking, lemmatization, taxonomy classification, custom plug-ins, etc.

Today's public search applications have taught users to find information in the shortest time possible. They have specific information requirements that must be fulfilled. However, they aren't perfect. Users often have to launch multiple queries before they find the information they want.

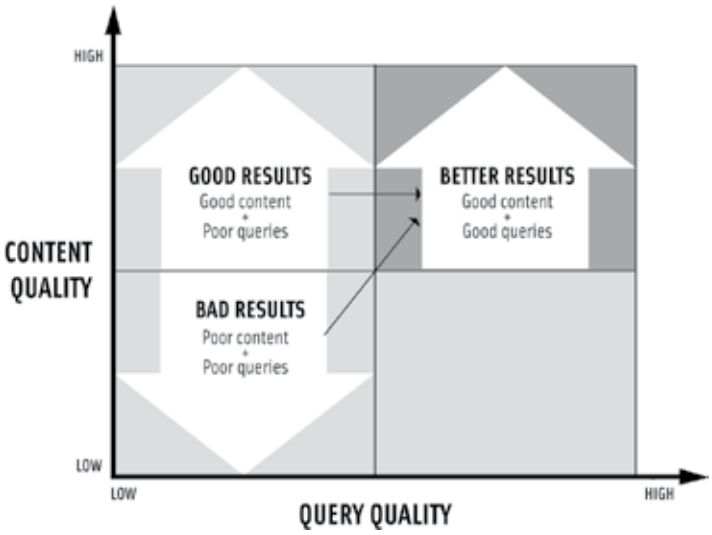
One reason for this is that queries are typically one or two terms in length and fairly generic. Another reason is substandard content quality – low quality content and poor queries will result in bad hits. If the content quality improves, or the content and the queries improve, users will get better results. Generally, search applications have little or no control of the information being fed into the search system, so query results can be poor.

Content preparation is often an integral component of the overall workflow that supports business processes. Use subject matter knowledge to best prepare the data for later retrieval.

Content owners and business managers can control the quality of content before it is pushed to the search application - although many aren't yet aware that this is possible.

This paper highlights the business need and impact of content preparation, the interconnected topics of content aggregation (getting information into the search system), and document processing (analysis, transformation, and enrichment of original content for the purpose of indexing).

For good results, first improve content



Content preparation

The old adage "garbage in, garbage out" springs to mind when considering content quality. Content preparation means selecting the right content, appropriately transforming and tagging it, cleansing or normalizing (regular and consistent, appropriate spelling or style) the content, and reducing the complexity of disparate data types. Information sources may include: Web/intranet (HTML, XML, multimedia); file system and content management systems (Doc, XLS, PDF, text, XML, CAD, etc.); e-mail (e-mail text, attachments); and databases (structured records).

This data will comprise free and semi-structured text, structured data (XML), binary files, and highly structured text and numeric data in the case of databases. Content preparation on each can vary from simply adding meta tags to deep cleansing of content.

Typical search application use will have limited content preparation. It will include multiple data sources and types, Web crawls (with no special content preparation) intranet crawls, file system content, CMS data with standard metadata applied to all documents, and pushed or pulled access to database content that has been cleansed.

Q: Our IT group developed document processing-type rules and code from a previous search solution. Now we have changed providers. Can we reuse the solution?

A: If document processing code already exists – in Java or C++, for example – you can use it by writing a stage in the document processor to call out to it. Don't waste time rewriting code. Although document processing stages need to be written in a scripting language, you can leverage this existing IP as scripting language with the ability to invoke other languages.

Organizations may leverage some automated tools to assist content preparation as a first step. Advanced organizations will look to do it all, but will have a consistent metadata model applied to all data, including Web content. They will also add specific keywords or industry terms to documents to improve relevancy and search. Content preparation will be performed by editors (actual staff) or by automated workbenches to correctly tag up information.

Organizations must understand the benefits and trade-offs of using prepared content versus the raw form, and the benefits of using people over automated tools. People will be more accurate in the initial stages, but over time will be less consistent than an automated system, and will take much longer. There will be a cost overhead to factor into search estimations. Organizations will delegate a certain amount of the tagging to the content owners to enhance the "correctness" of content.

As part of the preparation, organizations must close the loop of content and monitor the effectiveness of search and navigation, and manipulate it to align with overall business objectives. Understanding the ratio of content preparation to information usage will help determine where to focus efforts. Over time, content preparation can enhance the value of future searches dramatically.

Best practices in content preparation include:

Planning ahead - deciding which content needs to be prepared, by whom and at what quality level. You need to factor staff-driven operations into your resources, work and time estimates.

Aiming to increase relevancy – people use a search platform to find the information they need, when they need it. Focus efforts on increasing the relevancy of the results returned.

Normalizing content – ideally data (especially structured) should be consistent and without duplication.

Logically partitioning multi-lingual and localized content – isolating documents on a per-site or language basis.

Striving to normalize acronyms – they can be easily expanded in the search system (i.e. IBM →I.B.M → International Business Machines).

Considering directed search by preparing content to provide multiple navigation points – product line, model family and price are all complimentary navigation facets against the same data.

Automating where possible, since information is produced and consumed at incredible rates. Use automated preparation tools to save time and reduce error rates.

Common mistakes include expecting good results from bad data, failing to consider how to process the content for highest utilization, and ignoring the potential for error introduction by automatic data preparation. Not all systems are perfect 100% of the time.

Organizations are urged to take a step-by-step approach. Do not try to prepare all content types at once; learn what works and then introduce no more than one or two content types per review cycle.

The remainder of this paper will discuss content aggregation and document processing.

Enhancing search with content aggregation

Content aggregation is the bringing together of content from multiple sources for the purposes of later retrieval. It is used to consolidate search results into a comprehensive whole.

Federation of search is also important within content aggregation. Benefits of federated search include the potential for more complete result sets, and no need for an increased index size or for the associated hardware.

Organizations must balance the increased flexibility with the tradeoffs of such an approach. Over-simplifying the search experience does not add value.

Content is available to the search and filter/alert engine via the content API which acts as a broker of information. It performs the tasks of pulling content from the data source (database, CMS application, etc.) during scheduled calling requests and also handles the pushing the content into the search engine.

Q: We have a corporate taxonomy. How can we use it to help us with searching for documents?

A: During document processing, you can assign sets of taxonomic information to documents, prior to their being indexed. Subsequent searches can use this information as a filter and a UI drill-down mechanism.

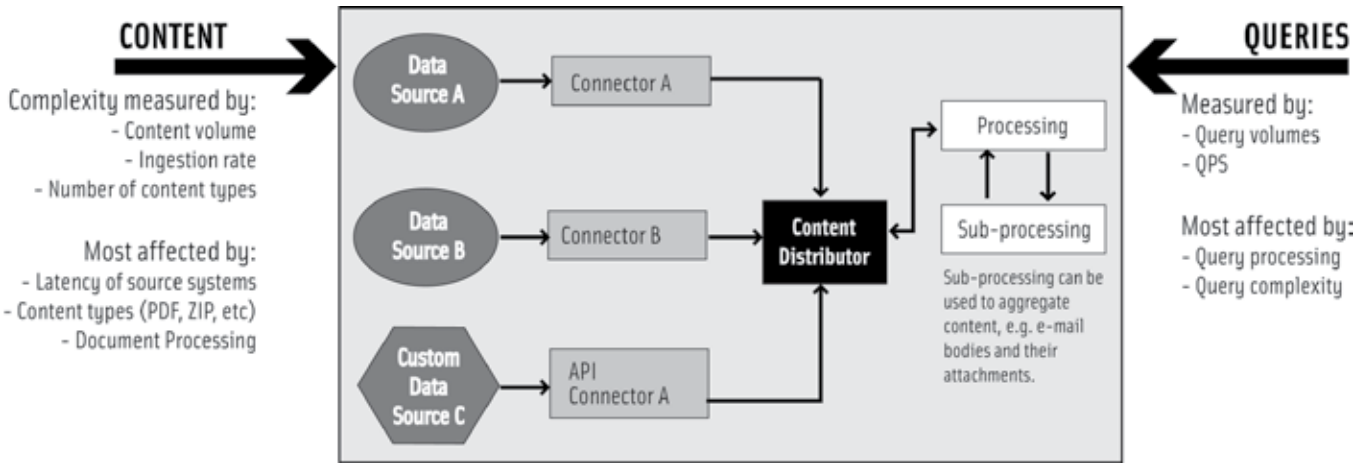
The content pull approach leverages data connectors to retrieve the information via standard APIs or interfaces. This is the core technology of most search solutions, and it includes retrieval of Internet-based information, database information, or file server-based documents. The data connectors do not require integration programming towards the target data repositories although in some cases they may not provide the required real-time performance. In these cases API integration may be preferred.

The content push approach requires that data repositories, applications or messaging middleware send the data directly to the search application via the content APIs. This omits the latency of crawling but it requires a closer relationship between the content application and search engine.

A traditional search approach typically implies long latency from the time the data is modified until the modification is reflected in the searchable index. This means that the search engine does not handle dynamic data and may not be sufficient for processing real-time information.

Some enterprise search solutions remove this limitation by scheduling frequent updates to ensure that the information is made searchable in short timeframes. The system takes this functionality further by integrating the real-time filter engine that matches information against pre-defined queries as it becomes available.

Content Aggregation and Processing Diagram



Typically, organizations will deal with multiple content sources, their structured and unstructured data will be semantically related through the appending of correlation IDs for grouping, and they will need to ingest and sub-process data as a single unit (e-mails and attachments). An advanced scenario will potentially display results based on grouping of content – as in the case of an e-directory displaying a mixture of Web content (description, name, etc.) and database content (for example, opening hours).

Organizations need to appreciate that ingestion of content will be impacted by the amount and number of different types of data in addition to the latency of the source systems. Different types of content (doc, pdf, zip, etc.) will process within different timeframes. The complexity of the document processing (the numbers of individual stages and their roles) will impact the speed at which content can be ingested. External factors such as network performance, repository speed and crawling/spidering windows will all have an impact on ingestion speeds.

Enhancing search with document processing

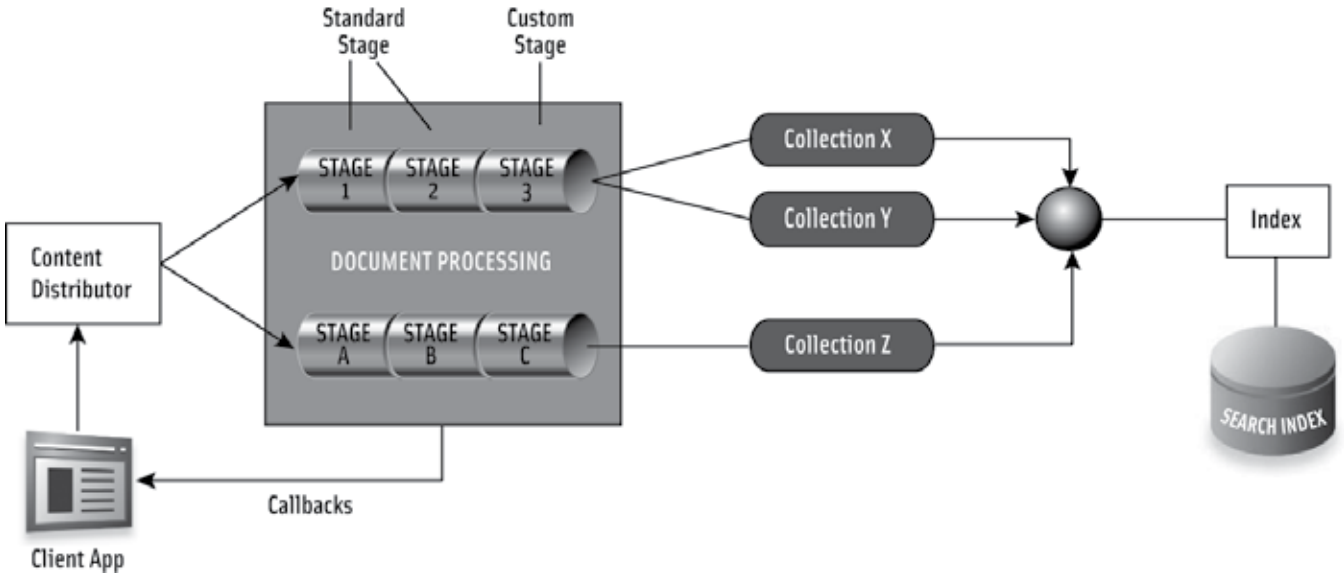
Document processing is the analysis, conversion, transformation, and enrichment of original content for indexing and subsequent retrieval.

The document processing component of a search application is shown below. Content flows in from the left of this schematic, with content ingestion rate measured by documents per second per server and by the total ingestion volume (number of documents handled). Document ingestion rates are affected by hardware capacity (the number of nodes, the RAM on each node, disk capacity and latency, CPU usage, I/O wait time, etc.), the amount of work the performed, and external lookups, where a particular stage may make database lookups or calls out to a Web service.

Content flows from outside the system into the content distributor. The content distributor dispatches content to the proper component. A pipeline processes content (serially) on behalf of one or more collections. The content is pushed out in post-processing as XML and is indexed with respect to the configuration of the index.

Document processing can comprise one or more document processing stages (language detection, synonyms, spell checking, lemmatization, taxonomy classification, custom plug-ins, etc.). These stages analyze the content and add or remove or transform data accordingly. Using this type of linguistic processing is vital to improving the search experience. Content can be normalized using language-specific document processing, language/industry specific synonyms (by defining dictionaries), etc. It is also possible to remove unnecessary content that doesn't need to be indexed, such as menus, frames, etc., before it hits the searchable index. (See the Linguistics and Search paper for detailed information on this topic.)

Modular processing stages for flexible content refinement

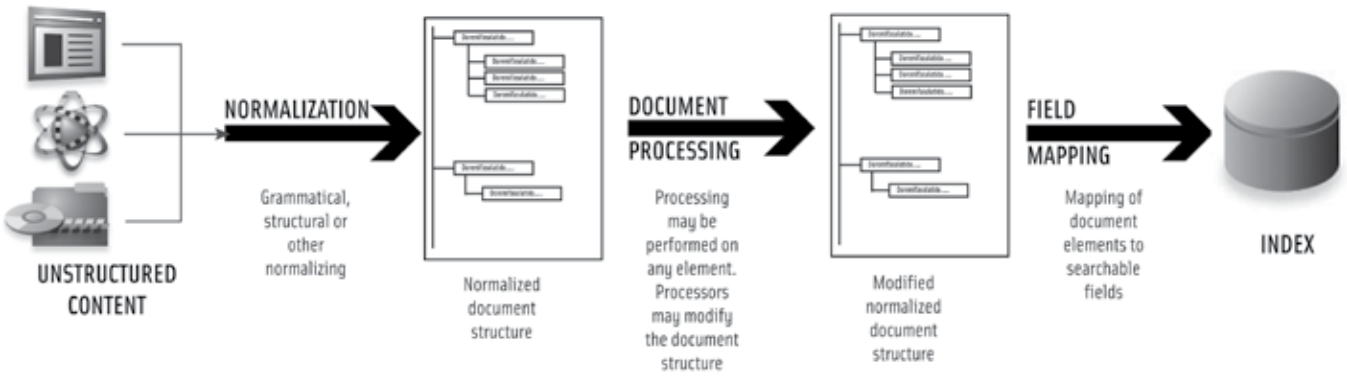


Enterprise search applications usually ship various pre-built stages into different processing components. They are designed to accept certain types of data and operating processing – Web data, XML information, news, etc. Organizations can use the standard stages as is, or create new stages based on existing ones, or augment existing pipelines with stages that they write themselves, combining these techniques as they see fit. Combining processing stages provides search customers with document processing platforms that meet their specific industry needs.

collections, use a combination of standard and custom document processing stages, and possibly use a combination of content connectors and the content API for submitting information to the system.

In an advanced scenario, the customer would have a setup similar to the typical scenario. However, there would be more nodes involved as part of the installation, and the level of customization of the processing components would be higher. In addition, an application submitting content would use the callback features of the API and would contain logic for automatic error detection and recovery.

Some Examples of Content Refinement Stages



Queries are submitted against the index (right hand side of diagram), with the critical metric here being QPS (queries per second). This is affected mostly by query volumes, query size, and complexity, and by query transformations such as spell checking and query re-writes. Taking a closer look at normal document processing, the raw content is normalized and turned into structured information – specifically, name-value pairs. These name value-pairs are then sent to waiting document processors.

Once the document processors have completed their task, the output of one document processor serves as the input to its downstream neighbor. At the end of this lifecycle, the document's name-value pairs are mapped to a field in the index. The index acts in much the same way as a database schema in the RDBMS world, defining the overall structure and constraints of the data it stores. The advantage here is that the scope fields will offer a more flexible search experience compared to the database.

In a typical scenario, the customer will have deployed a multi-node solution with distributed document processing. The document processing nodes would be separate and distinct from the indexing-and-search nodes. In this case, the customer would employ multiple

Different industries, different solutions

Different business drivers apply to different industry domains, and the supporting structure and quality of content is vastly different from organization to organization. A flexible and tunable search application is needed to pull disparate data sources together, to cleanse and process it before making it available for consumption via the index.

Let's consider an e-directory. Today, e-directories are facing intense competition from sites such as Google and Yahoo! Many want to protect their positions in the market by leveraging search applications, and by moving their traditional print models online. Some are enhancing and differentiating their content offering by crawling and integrating local Web content, providing mapping capabilities, federating to user reviews (restaurants, hotels, etc.), and including other directory content such as White Pages and city guides.

E-directories must overcome content quality (internal database and Web) as well as aggregation and processing problems in order to succeed. Improving the traditional

Yellow Pages interface is vital to provide simple digestion of additional content. Using navigators and sentiment analysis (for reviews) and exposing database content will provide improved usability. This further highlights the ability to process and correlate structured IYP and Web content. Enhancing the search experience and the increased exposure of advertisers will potentially draw more eyeballs to the site and will enable the e-directory to offer alternative pricing models.

Q: We want to integrate and expand our current e-directory offering to include multiple document types from multiple sources. What should I be aware of?

A: Aggregating content at ingestion time requires carefully correlating the source documents. Aggregating results from multiple sources requires relevancy tuning, benchmarking, and index reconstruction, all of which are time-consuming. It's best to take a phased approach and understand the impact that additional sources will have.

In a knowledge discovery environment, such as an oil and gas provider's intranet, there may be more human intervention in the preparation of content based on the complexity of content and use of specific industry terms. This can be combined with entity extraction tools to enhance the usability of the content and search experience.

Support for multiple types of documents is a must with anticipated aggregation from file systems, CMS applications, and databases. Custom applications can be included in this environment by using content APIs. Multiple document processing stages will be necessary to correctly categorize unstructured and structured data; the use of synonym expansion and concept extraction can assist scientific researchers.

OEM integrations with the document processing approach will leverage the search application's in-built connector – in this case, the file traverser for loading content from the OEM's file system. Using the out-of-the-box connectors provides the ability to rapidly support many repository types.

Custom-built applications use the search application's indexing API to push data to the search engine. This requires all original data connections to be performed using the calling application, which controls scheduling, interfacing protocols, and data structures.

The API supports error-logging callbacks, where actions may be triggered when documents fail to be processed or do not make it into the index. It should also support updates and deletions if the system has dynamic content.

Understanding the impact of …

… Content aggregation

Public search applications have created demanding search users who expect highly relevant search experiences with comprehensive access to all content, no matter where the source data originates. As a result, there is a pressing need for flexible content aggregation tools.

So the search provider must bring disparate content together for augmentation by the document processing component of the search application. OEM providers or integrators would develop various document processing stages that support specific application, content and business needs, and allows the manipulation of results.

Key decisions must be made related to how and when to aggregate – ingestion time, query time, or both (federation or merging of indexed content, for example). Performance and hardware costs need evaluation. Other tradeoffs to consider are the aggregation of content versus raw ingestion (and correlating via IDs), and aggregating at ingestion time versus at result-processing time. It's best to perform content aggregation before the content is processed and passed to the index. This improves speed and accuracy of results and increases user satisfaction.

…Document processing

The primary objectives of document processing are to apply business logic to the original content, making it easily searchable, and to augment content with business semantics to add value when it is retrieved. Organizations need to consider relevancy, advanced linguistics and other capabilities, including custom stages. Adding additional stages to document processing will impact the efficiency of indexing content, but is dependent on the complexity of each stage that is added.

The impact of document processing is highly dependent on the quality of the content processed. An important factor for customers to consider is the cost required to clean and prepare the content outside of the document processing versus the degraded performance ingestion rates.

Customers have unique content needs that will require specific custom processing to ensure that it aligns with their business goals. Organizations need to make such decisions on a case-by-case basis, based on their preferences, skill sets, and available time.

Mini case study

Global publisher uses content aggregation and document processing tools to provide comprehensive scientific search site

Who
Global scientific publishing provider

Challenge
To provide the most intuitive science reference site and to allow scientists to process and retrieve content to support their needs. The publisher has a wide variety of content sources and large data volumes (90 million Web pages, 15 million journals) and provision of entity extraction and classification for simple navigated search.

Solution
Single indexing of unstructured and structured content at high content volumes and ingestion rates. Advanced document processing and linguistic capabilities enabled results to be blended from a variety of target systems.

Technology
Advanced document processing stages with advanced linguistics and entity extraction/classification for navigation. Document processors can also call out to Java and C++ custom processors. Content aggregation via Web crawling, files, database connectivity, and tuning of result processing framework.

Guidelines and recommendations

The perceived relevancy of content and of search success is fairly subjective. User surveys can help an organization understand the needs and concerns of its customers. The next section of this paper highlights the metrics needed to measure the success of content aggregation and document processing as well as the potential mistakes to avoid.

Content aggregation

The metrics for measuring content aggregation are straightforward: total content volume, format, and speed. One key metric to consider is the time it takes to reproduce, re-aggregate, and re-index if the data was lost due to a hardware failure, for example. An e-commerce site would need to prepare for the profound business impact of bad or missing data in the index. In this case, raw content should be stored prior to indexing.

Often organizations try to aggregate too much data into one document processing stage. It may be useful to aggregate content at processing time. This may have a slight latency impact, but it can speed the ingestion and processing of original content.

Document processing

When measuring the effectiveness of document processing, customers should consider the number of documents per second per server for each node and across all nodes. This will help customers determine the efficiency of each system. In addition, customers should monitor error rates; high error rates indicate problems with the source content, the document processing stages, or the configuration of the index profile.

Q: My OEM application is suffering from submission errors. How can I fix this?

A: When using the content API, use callbacks to detect and respond to error conditions. This enables your program to monitor progress, report problems and attempt to automatically recover from submission errors.

Common mistakes include failure to maximize the document processing stages, not understanding the processing data model, not appreciating certain index configurations, and not understanding which processing stages add value. Many customers waste valuable time duplicating code and functionality that comes standard with search applications.

Best-practice guidelines for improving search

Content quality has a profound impact on the usefulness of search applications. Not only should organizations incorporate the cleansing and preparation of content into search initiatives, but they should look to leverage powerful content aggregation tools. Sophisticated document processing capabilities must be leveraged to their full effect to provide a best-in-class search application. Here are the best-practice guidelines:

Content aggregation

Plan the aggregation strategy. Understand the data, and consider ingestion time versus query time aggregation.

Consider correlation IDs. In some cases, it can be enough to use an ID field to relate content (for example, mapping zip codes or post codes from Web searches with IYP database content).

Store content locally. The search solution should automatically store a local copy of content before submitting it for processing. This is not necessary if the content can be quickly and easily regenerated from the source, or is obtained by crawling. This allows content to be reprocessed in future based on changing business needs.

Automate the process. Content acquisition and aggregation can be largely automated via scripts and applications. The less human involvement, the better!

Document processing

Prepare! Spend time evaluating document processing needs and augment content as needed. Consider your goals carefully and be deliberate about what and how you process. The index might need to be updated to accommodate your processing.

Size accordingly. Use the appropriate number of hosts for feeding and document processing. Consider the impact of ingestion rates, linguistics, document size, etc.

Use existing document processing stages. Use pre-fabricated stages where possible.

Optimize document processing. Remove all unnecessary document processing stages. Choose the right processor for the content type and task at hand. Remove all unnecessary stages that can degrade performance and cause memory leaks.

Monitor the statistics of CPU usage and I/O wait time per stage. Add processing servers where necessary to tune and overlap CPU usage and I/O time.

Submit documents in batches. When using the content API, submit document in batches of between 10 and 200. Tune appropriately to minimize processing overhead and increase overall throughput.

Update the index profile. Ensure that the document processor, index, and front-end are compatible. Some document processing changes will necessitate index changes. Align document processing with the content of the index.

Build a library of processing stages. When building custom stages, try to solve the given problem as generically as possible. You can reuse code in subsequent projects.

Frequently asked questions

Q: I don't believe that the automation of content cleansing will work for me. What do you suggest?

A: The best practice for search is to leverage automatic tools where possible. This will shorten the time it takes to take content in its raw form and publish it in the index. If you want to use staff editors for your specific vertical needs, you have to consider that normalizing relevancy from multiple results sets is time-consuming and imposes a management overhead.

Q: Is there a limit to the number of document processors and processing stages that I can use?

A: In theory there isn't, but this needs to be balanced with the latency tradeoffs that may be experienced with too many processing stages. Using more document processors will ultimately speed up the ingestion and processing of content to the index. You should map each processor to the type of content – Web, news, XML, etc.

Q: What is sub-processing?

A: A set of processing stages that work to process distinct documents from the main pipeline. Sub-processing aggregates e-mails and attachments, for instance.

Q: How can I improve the relevancy of my documents?

A: First, the content should be high quality, so it's important to put effort into getting this right. Second, look to leverage the multiple relevancy models (freshness, completeness, authority, statistics, quality, and location) built into the search application. Finally, use custom rank-tuning models to target specific content to particular queries or user groups.

Q: How should I appropriately size the document processing part of the search application?

A: If possible, use at least one extra host for feeding or document processing. If you feed more than 10 documents a second, use two hosts. If you use linguistics heavily, use two or three hosts. If you deal with large documents, use two hosts and lots of RAM (3 GB). If you do all of these things, use three to four hosts.

Q: Why should I integrate my search application with the structured data in my database?

A: Search applications provide an index architecture that is well suited to both structured and unstructured information. Integrating with a relational database is performed for two reasons: 1) relational databases are not very efficient for handling large query volumes, and 2) integrating a large number of different data sources into one index and one search bar provides a more convenient search experience. For example, an e-directory would look to publish both Web and database content, such as company description or offerings (Web) and opening hours or price catalogs (database).

About FAST SBP™ (Search Best Practices)

SBP consulting is a highly focused transfer of search knowledge and experience from FAST to its prospects and customers. SBP workshops aim to help enterprises realize the full potential of search, by creating optimal strategic, functional and technical roadmaps, delivered in the form of business model, solution and architecture designs.

Fast Search & Transfer
www.fastsearch.com
info@fastsearch.com

Regional Headquarters

The Americas
+1 781 304 2400

Europe, Middle East & Africa (EMEA)
+47 23 01 12 00

Japan
+81 3 5511 4343

Asia Pacific
+612 9929 7725

SWP.003.B.01.011206