Scaling strategies for mission-critical discovery and navigation applications

Contents

In the majority of enterprises, no one application contains all of the data business users need to access on a daily basis. For example, email is the main medium for business communication today, but it requires storage that often rivals or exceeds the amount of disk space used for user files and directories. CRM systems are the backbone of the sales, marketing and customer support teams and contain a wealth of business-critical information. Financial and operational data is located in ERP and BI applications, databases and other data repositories.

All of this data is growing 60-80 percent annually, according to various industry analyst estimates—and shows no signs of diminishing. New content is created every minute, yet old content is rarely purged, resulting in terabytes of data for large, information-intensive environments. Providing unified access to this massive amount of content for users to easily discover and navigate information is an absolute requirement, but finding a solution that can not only securely crawl and index all data repositories, but also quickly scale to handle the sheer amount of data is a necessity as well. This whitepaper discusses scaling strategies for building flexible, high performance and high availability solutions for mission-critical discovery and navigation applications.

What is scalable discovery and navigation?

Enterprise discovery and navigation applications generally provide the ability to query and retrieve results from existing search applications (known as meta-search), the ability to examine online information and process it so that specific information can be located quickly and the ability to combine search results from any number of these types of sources into a unified, prioritized set of search results (known as federated search). The basic tasks within a good enterprise discovery and navigation solution are the following:

- *Query processing, translation and response*—your discovery and navigation solution must provide a mechanism for enabling a user to enter their query, which is typically either composed of simple text strings or combinations of terms that use relational keywords or symbols such as AND, OR, WITHIN, and so on. For complex queries, the application must be able to convert your query into the specific syntax required by any other applications that it retrieves results from in order to submit that query to those other applications. Finally, your discovery and navigation application must be able to combine, prioritize and display the results returned by your query in a user-friendly fashion.
- *Crawling sources of enterprise information and retrieving their content*—your discovery and navigation solution must be able to contact all of the sources of information that you want your users to be able to navigate, retrieve their content and related metadata such as access control lists, and return that information to the discovery and navigation engine for subsequent processing.
- *Indexing the information retrieved by crawling*—your discovery and navigation solution must be able to build an index for the information retrieved by the crawling process. Indexing the data for which you want to provide access is the first step toward improving navigation performance, increasing query throughput by caching the content that you are navigating in a fast, easily-accessed local index. Using an index largely eliminates the need to re-retrieve remote information at query time, and therefore increases the performance of your application. However, because indexing requires significant amounts of information access, retrieval and data processing, the process of index creation can be very slow. The implemention of an index needs to consider issues such as size, index location, lookup performance and updating.
- Scalable discovery and navigation is the term we are using to describe a solution that can handle growth, increasing demand, and increasing user expectations as both its user community and the amount of data that it provides access to continues to grow. Scalable enterprise information navigation and discovery solutions must therefore provide flexible mechanisms for handling growth-related issues such as the following:

  – *Increasing system load and number of queries*—the front-end work of accepting and pre-processing queries and the back-end work of prioritizing and combining results are relatively quick operations. However, within the discovery engine, each query requires the engine to access and scan the appropriate index(es) for matching information.
  – *Increasing amounts of information that users want or need to access*—as the amount of information that you want to be able to navigate increases, the size of the index associated with that information also grows.
  – *Updates to existing indexes without interrupting service*—in most cases, the enterprise data that your users want to discover and navigate is changing, expanding or both. The engine behind your solution must be able to quickly and easily index new content while delivering uninterrupted discovery and navigation capabilities.

What are the challenges for scalable discovery and navigation solutions?

As the name imples, scalable enterprise discovery and navigation must be able to continuously provide a high-performance solution as both the amount of information that it provides access to and the number of users that depend on that discovery and navigation technology increase.

The issues listed at the end of the previous section are specific, performance-related scenarios. However, a successful and scalable enterprise discovery and navigation solution must be able to address both performance and higher-level issues such as redundancy, failover, update, and manageability to guarantee its usability and availability.

Delivering a high performance solution

Though users may initially accept some performance and responsiveness problems in an enterprise discovery and navigation solution, the appearance of performance problems is always cause for alarm. The number of users who rely on an enterprise discovery and navigation solution and the amount of enterprise data that can be located through that system rarely decrease.

As discussed at the end of the previous section, performance problems can arise at a number of different points in the enterprise discovery and navigation process, all related to increasing demands on the system. While popularity is usually a good thing, continuously increasing demands on the solution can hamper its usability for everyone. Unless your solution provides built-in mechanisms to address to the performance problems associated with indexing large amounts of data and accessing large index files, the usability of the solution will continue to degrade as the amount of data that you want to search index and your usage increases.

Eliminating downtime

Making sure that your discovery and navigation solution is always available to your users is probably the highest priority. As more users come to depend on a discovery and navigation solution, its availability requirements also rise. Most users can tolerate some performance problems, but they will not tolerate downtime.

One of the first steps toward high availability for any system application is to remove single points of failure, such as having single systems that house both a discovery and navigation engine and the index(es) it uses. Redundancy and replication are two specific techniques that help guarantee high availability. Redundancy typically refers to providing multiple instances of a specific server process, while replication typically refers to maintaining multiple copies of a software resource. However, simply having multiple systems and copies of a resource is only part of a solution to potential problems. To make the most of redundancy and replication, your solution must provide both monitoring and failover support, so that problems in existing hardware and software are quickly detected and automatically cause other resources to be activated and used.

Delivering up-to-date information

As the amount of information that you want to locate through an enterprise discovery and navigation solution grows, identifying and indexing both new data and changes to existing data can be a complex problem. If changes to the data require complete re-indexing of that data, your discovery and navigation engine could spend most of its life simply doing index creation and recreation. It is therefore important that a scalable solution creates an index that can easily be updated without being recreated.

In addition to being incremental, index updates should also be transactional, which means that they either succeed completely or make no changes to the index. This preserves the integrity of your index during the update process, provides an opportunity to resume indexing at a given point without redoing the entire indexing process and also protects the index against problems related to system outages or failures.

Providing a flexible, manageable solution

A key to addressing items such as availability and performance problems is automation, where an enterprise discovery and navigation solution itself automatically probes and monitors associated systems, and automatically adjusts system characteristics such as load and fail-over sequences. Similarly, system administrators must have the ability to easily obtain information about the status of the applications and resources associated with a discovery and navigation solution. Providing easy access to configuration and status information makes it easy to identify and address problems as they emerge, rather than after they have become crises.

Being able to identify emerging problems is only truly useful if your solution provides a fast, easy-to-use mechanism for interacting with and reconfiguring its underlying technologies. For example, the amount of disk space and memory available on each specific system is always finite. As the storage requirements for an index approach the amount of disk space available on the system where it is stored, you need be able to head off any associated performance or usability problems. In the short term, you can simply add additional disk space to a specific system, but this typically requires downtime. In the long term, it is much more useful to have an administrative interface that enables you to dynamically manage your index(es) and the general configuration of your solution. This could enable you to easily move an index from one system to another that has additional memory or storage space and reconfigure the application to use that index, all without interrupting its availability.

A scalable discovery and navigation solution must provide flexible ways of augmenting and expanding your existing resources, as well as tailoring the configuration of your application to its hardware execution environment. The ability to add additional systems and their associated resources (processing power, memory, disk space, etc.) to your search infrastructure gives you a flexible and scalable solution that you can reconfigure on the fly to improve performance or address emerging problems. Ideally, your solution must provide interactive administrative support for a wide range of performance parameters, such as index size, location, pinning an index in memory, and so on.

Distributed indexes: A foundation for scalable discovery and navigation

As the amount of data that you want to be able to navigate grows, so does the size of the index for that data. Users often report that the size of an index is approximately one-third of the size of the data that you want to be able to navigate, but this ratio can be much worse depending on the type of information that you are indexing. Index access time, lookup performance, and storage requirements are therefore very important issues for evaluating any discovery and navigation engine. These types of issues become critical in a solution that is crucial to your business.

Depending on the type of system on which the index is stored, scanning large index files can be slow simply due to their size. Similarly, the use of index files that are larger than the amount of memory available on a machine can compound index access time issues. When the size of an index expands beyond the amount of physical memory on the machine that is accessing the index, performance can suffer both due to the time required for swapping and the time required for the disk I/O associated with reading new portions of the index into memory. This can cause performance problems on both the system where the discovery and navigation engine is running and on the storage devices where the index is stored.

As businesses and users increase their use of and dependence on discovery and navigation capabilities, the index used by a specific solution becomes more obvious as a potential single point of both failure and performance problems. If an index is somehow corrupted due to disk problems, the system through which an index is accessed fails, or the storage on which the index is stored fills up or goes offline, your discovery and navigation capabilities grind to a halt.

Replication solves system load problems by distributing query load across multiple systems, each containing an identical index.

To provide a successful scalable solution, index access, availability, and the index(es) themselves must be scalable. Scalable indexing solutions fall into two functional classes: segmentation, which divides an index across multiple systems to solve performance and storage problems, and replication, which maintains an identical copy of indexes across multiple machines (also known as mirroring). Replication solves system load problems by distributing query load across multiple systems, each containing an identical index. It also guarantees higher discovery and navigation system availability by eliminating a single system and single index as a potential point of failure. Both segmented and replicated indexes are examples of what is commonly referred to as a distributed index.

Modern enterprise discovery and navigation solutions use distributed indexes to solve most of the performance, index size and index availability problems associated with the increasing demands of rampant data growth.

A distributed index that is split into multiple segments on multiple systems can eliminate many performance problems by enabling the size of the index segment stored on each machine to match the amount of physical available memory on that system. Using index segments in this way eliminates swapping and provides higher performance by eliminating the need to go to disk to access any portion of the index. Index segmentation also has the added benefit of substantially reducing individual system storage requirements by enabling you to distribute those requirements across multiple systems.

A distributed index that is replicated across multiple machines can eliminate most availability problems because an application can fail over to another machine if the index on one system (or the system itself) is unavailable. Replication can even be integrated with system load analysis to support load-balancing across multiple machines that store the same index(es).

Replication and segmentation are not mutually exclusive approaches to index maintenance. As discussed in the next section, the IBM® InfoSphere™ Watson Explorer is an enterprise information discovery and navigation solution that enables these approaches to be used separately or together to maximize the performance, reliability and availability of your enterprise discovery applications.

Index deployment scenarios for scalable discovery and navigation

The ability to replicate entire indexes across multiple systems and the ability to divide indexes into multiple segments help satisfy many of the critical requirements of an always-available enterprise discovery and navigation application. Solutions that take advantage of these capabilities can easily handle increasing loads without sacrificing performance, provide continuous service to enterprise users and can extend discovery and navigation capabilities to constantly increasing amounts of information.

How each site distributes its index files is largely dependent on the number of computer systems available to your discovery and navigation infrastructure and the memory, storage and performance characteristics of each. General infrastructure support systems, such as uninterruptible power supplies, backup systems and the load and other processes running on existing servers will also have an impact on how you deploy discovery and navigation collection indexes and the engines that use them.

The next few sections illustrate some of the more common ways in which replicated and segmented indexes are used to provide scalable enterprise discovery and navigation solutions. All of these configurations are supported in the latest generation of Watson Explorer and can easily be configured using its graphical administrative interface.

Replicating an index for availability and load balancing

As discussed earlier, performance and availability are among the most common concerns when developing and deploying a scalable enterprise discovery and navigation solution. Configuring a second system to host a mirror of the index for a collection is probably the most common scenario when initially deploying a discovery and navigation solution, as shown in Figure 1, "Replicating a Single Index."

Maintaining a replica of an index increases availability by protecting against outages resulting from the failure of a single system. Replication also reduces the load on a single server by distributing query load across all servers that host a replica of an index. In high-availability environments, replicating a single index can also be used to allow you to do hardware repairs or single-user system administration of any of the systems on which an index is replicated without scheduling downtime. Any of these systems that contain an index or a replica of that index can be taken down for administrative purposes without disrupting the availability of your solution.

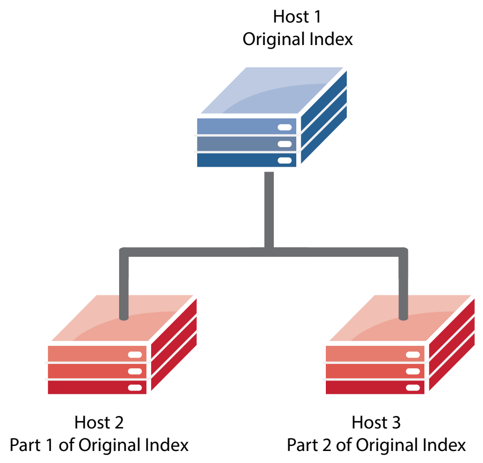

Figure 1: Maintaining a replica of an index increases availability by protecting against outages resulting from the failure of a single system.

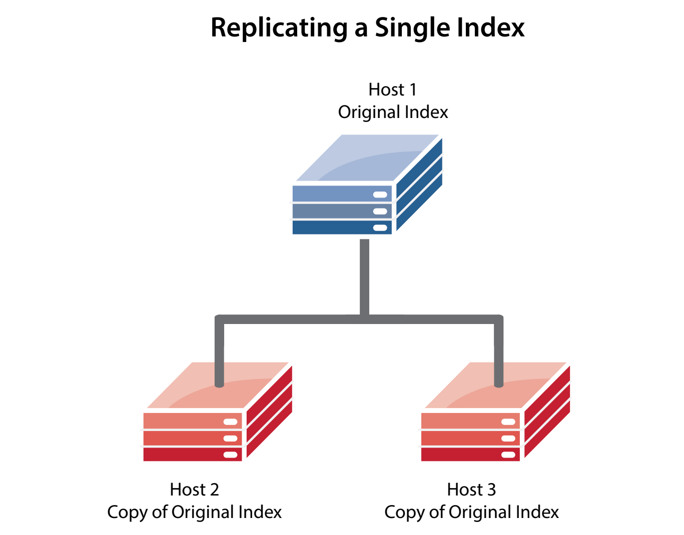

Figure 2: Segmenting an index is the process of dividing that index into multiple parts which are stored on different servers.

Segmenting an index for higher performance

Segmenting an index is the process of dividing that index into multiple parts which are stored on different servers. Index segmentation provides a number of performance improvements for enterprise discovery and navigation solutions. Figure 2, “Segmenting a Single Index” shows a simple segmentation setup.

The most significant performance improvement associated with index segmentation comes from the fact that index lookups are done most quickly if an index fits into the amount of physical memory that is available on the system where it is stored. When the size of an index expands beyond the amount of physical memory on the machine that is accessing the index, performance can suffer both due to the time required for swapping and the time required for the disk I/O associated with reading new portions of the index into memory. Segmenting an index therefore enables an administrator to divide the index files for a discovery and navigation solution into segments that match the amount of memory available on each system, providing higher overall performance. Segmenting an index across multiple hosts also provides a level of built-in load balancing, because queries targeted for specific portions of the index will be directed to the host(s) that contain that portion of the index. Distributing the query load in this way can improve the overall performance of your solution.

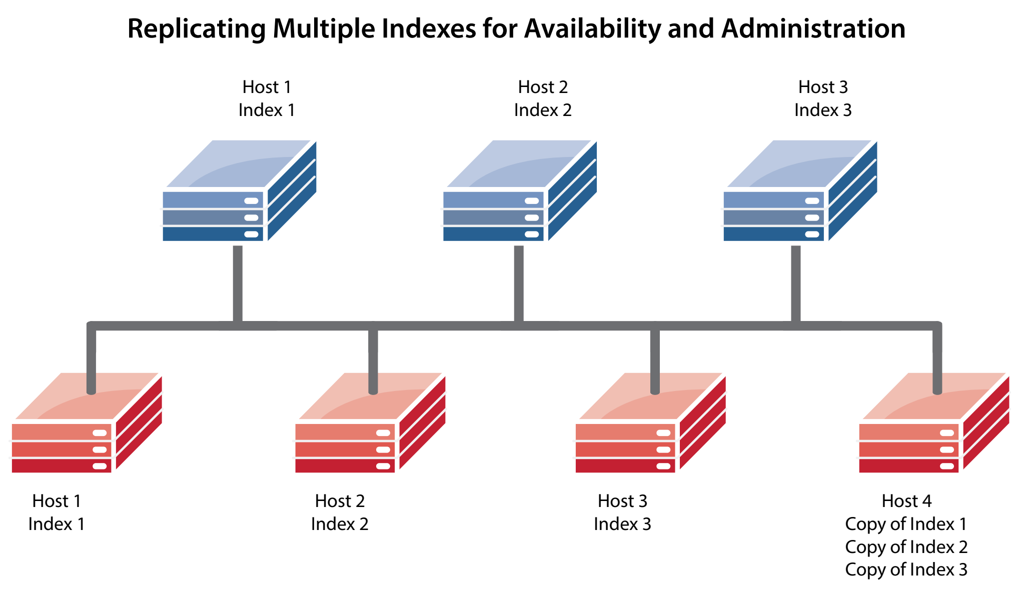

Figure 3: As with replicating a single index across two or more hosts, replicating the indexes for multiple collections to a single host can help ensure availability in the event of hardware problems.

Replicating multiple indexes to a single host

As with replicating a single index across two or more hosts, replicating the indexes for multiple discovery and navigation collections to a single host can help ensure availability in the event of hardware problems. Figure 3, "Replicating Multiple Indexes for Availability and Administration" illustrates this type of replication.

The advantage of replicating multiple indexes to a single host is that this provides increased availability guarantees without requiring as much hardware as single-host/single-index replication does. The downside of this approach is that hosting multiple indexes on a single host will probably not provide the performance benefits that singlehost/single-index can. It is likely that a single host will not have enough memory to keep all of them in core, and the host will therefore need to swap different indexes in and out of memory depending upon use requirements.

As with any use of replication, replicating multiple indexes to a single host can also provide opportunities for hardware repairs or single-user system administration, without scheduling system downtime. As long as some system that contains an index or a replica of that index is available, your discovery and navigation system is available.

Combining segmentation and replication

Segmenting an index can provide performance improvements by splitting that index into chunks that are small enough to fit in system memory and therefore support the fastest possible lookups. Replicating an index supports higher availability by providing multiple locations for an index so that the failure of a single system does not make the index unavailable.

The example in Figure 4, “Using Replication and Segmentation Together” shows a single index for a single collection that has been split into three segments, all of which are then replicated across three different hosts.

The advantage of this sort of configuration is that it provides both performance improvements and redundancy. When all three hosts are available, all three hosts are used, load-balancing occurs across all three hosts, and the segmentation of the index can provide performance benefits. Should any one of these hosts become unavailable, either due to hardware failure or regular maintenance requirements, the complete index is still available on two other hosts, and load-balancing will still occur across the remaining hosts. Even if two hosts become unavailable, the complete index is still available on a single machine ensuring zero downtime.

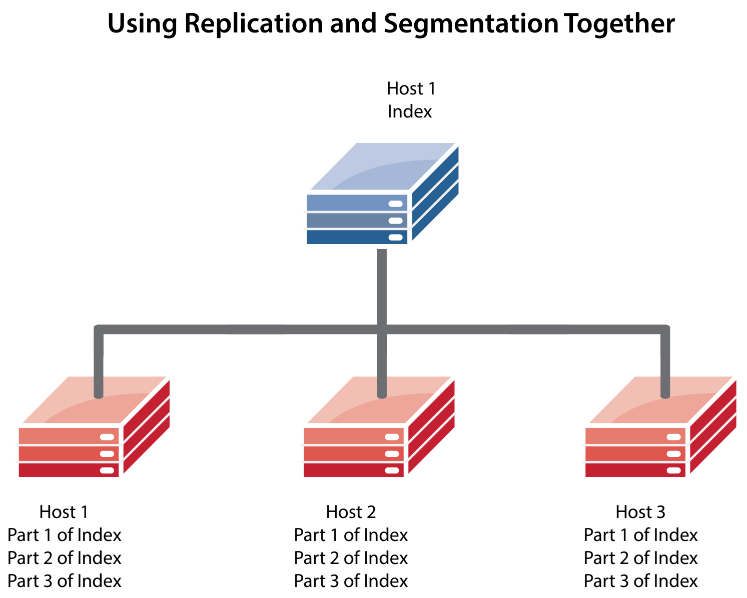

Figure 4: A single index for a single collection that has been split into three segments, all of which are then replicated across three different hosts.

Conclusion

There are many challenges in designing for maximum scalability in mission-critical discovery and navigation applications. As outlined in this whitepaper, the use of distributed indexes can address each of these challenges by improving performance, satisfying availability requirements and providing administrative flexibility as the amount of data increases throughout the organization and new resources are deployed within your infrastructure. Meeting all these needs—without incurring any penalty for downtime—ensures that your solution can support the most demanding mission-critical applications.

For information about IBM Watson Explorer, which provides a flexible solution to the search engine security requirements discussed in this paper, see http://ibm.co/1lBpw8p.

Selection and deployment checklist

So how should organizations proceed when selecting and deploying an enterprise discovery and navigation? Here are the index-related issues to consider:

- Quantify the availability requirements for your enterprise discovery and navigation solution, based on how important the application is to your users and your business. If no downtime is acceptable, then your solution must either support distributed indexes or you must prepare to physically replicate the system that is running your discovery and navigation solution and the storage that it uses. You must also select a monitoring solution that is flexible enough to probe both the health of your hardware and the state of your solution so that it can fail over whenever necessary.
- Identify the type and size of the online information that your solution must index, and use this as a metric for estimating the amount of disk space required. (If you are indexing multiple information sources and the types of data vary, you can use 30 percent as an initial number.) Make sure that the systems on which you plan to run your solution have sufficient space to store that index. If they do not, you must factor in the cost of additional storage, which can be anything from simple physical disks, to a RAID solution, to a high-availability Storage Area Network (SAN).
- Identify the amount of memory that is available on the system(s) on which you will be running your enterprise discovery and navigation solution and compare that to the estimated index size. If the index size is far greater than the amount of available memory, you will need to select a solution with index segmentation capabilities or suffer a performance penalty.
- Quantify the relative amount of daily change to the online information that your enterprise discovery and navigation solution must index. Determine whether you need a solution that can incrementally update an index, or whether your needs can be met by a solution that only offers scheduled updates.
- Test before you buy! Do not buy a product on paper; test it in a real environment to see if it can handle all of your indexing, performance and availability requirements. Experiment with different combinations of replication and segmentation, and pay great attention to the ease of configuration, the likelihood of committing configuration errors and the expertise of the vendor's support staff.
- Choose a scalable solution. Do not purchase an enterprise discovery and navigation solution that only satisfies your current requirements, because it will be unable to grow to satisfy new requirements and additional sources of information. Choose a solution that can scale to the level that you may require in three, five, or even 10 years.
- Select a reliable solution that is easy to get up and running. Enterprise discovery and navigation solutions that provide built-in support for high availability enable you to focus on developing your custom applications rather than on developing infrastructure.
- Test before you launch. Verify that the application matches your performance expectations and requirements. Before deploying your solution throughout your organization, dedicate a small group of users to query the solution simultaneously to verify both performance and load-balancing.
- Monitor the use and the performance of the application and adjust where its index(es) are stored to improve its performance and eliminate any potential single points of failure.

IMW14643-USEN-02