Revista Digital Universitaria
1 de noviembre 2012 • Volumen 13 Número 11 • ISSN: 1067-6079

ARTÍCULO

APACHE SOLR, UN MOTOR DE BÚSQUEDA DE CÓDIGO ABIERTO

Luis Miguel Estrada Ramos

Resumen

En este artículo se da una breve introducción a Apache Solr, un popular motor de búsquedas de código abierto desarrollado por la Fundación Apache. Esta herramienta se utilizó en el desarrollo de dos importantes proyectos dirigidos por el departamento de Publicaciones Digitales de la Dirección General de Tecnologías de la Información y Comunicación de la UNAM. En el artículo se exponen las características de la plataforma, se realiza una comparación con la tecnología de bases de datos, se explica su funcionamiento a grandes rasgos y finalmente se dan a conocer algunas APIs de programación que pueden ser de ayuda para el lector en la integración de la herramienta a sus propios desarrollos.

Palabras clave: Apache Solr, Motor de búsqueda, Índice, Navegación por facetas, Bases de datos, REST

Abstract

This article is intended as a brief introduction to Apache Solr, a popular open source search engine developed by the Apache Software Foundation. This software was used on the development of two important projects directed by the Digital Publications Department of the Directorate General of Information and Communication Technologies at the UNAM. This article presents a review of the features of Solr and makes a comparison between this platform and database technology, also usage is explained and finally two programming APIs are introduced. These APIs can be useful for the reader in their own development projects.

Keywords: Apache Solr, Search engine, Index, Faceted browsing, Databases, REST

Introducción

Todo desarrollador de aplicaciones debe ser consciente de la importancia que tiene una buena experiencia de búsqueda para el usuario final. Factores como la demora o inexactitud en la entrega de resultados, la imposibilidad de realizar búsquedas complejas o la falta de opciones de navegación entre los resultados suelen ser causa de una mala recepción de la aplicación. En la actualidad es común que las aplicaciones estén respaldadas por una base de datos sobre la que se implementan las búsquedas. Sin embargo, con este enfoque, la implementación de funcionalidades complejas de búsqueda puede llegar a ser complicada debido a las limitantes de las bases de datos, sin mencionar que las búsquedas pueden ser ineficientes si el volumen de información es muy grande o la forma en que está estructurada la información no es la adecuada. Es en escenarios como estos cuando una plataforma de soporte para las búsquedas puede resultar una alternativa más práctica y eficiente.

1 de noviembre 2012 • Volumen 13 Número 11 • ISSN: 1067-60710

En el mercado existen varias alternativas para este tipo de soluciones, en este artículo daré una breve introducción a una de las más populares y completas: Apache Sorl. Esta plataforma se utilizó en dos proyectos en los que participé activamente en el equipo de desarrollo, ambos dirigidos por el departamento de Publicaciones Digitales de la Dirección General de Tecnologías de la Información y Comunicación de la UNAM. Estos proyectos son: Portal de Portales Latindex y el Repositorio Institucional de la UNAM .

A grandes rasgos, estos sistemas recolectan metadatos de recursos académicos digitales disponibles en otros sistemas y los almacenan en una base de datos local sobre la que se implementan los servicios de búsqueda y recuperación de recursos digitales.

En el caso de Portal de Portales Latindex, a medida que la cantidad de información recolectada se acercaba a medio millón de registros, la aplicación se volvía considerablemente más lenta. La solución a este inconveniente fue Apache Solr, plataforma sobre la que actualmente se implementa el servicio de búsquedas de Portal de Portales Latindex. A pesar de que el número de registros ha llegado a casi 1,300,000, el tiempo de respuesta es más que aceptable. En el caso del Repositorio Institucional de la UNAM, Solr permitió la implementación de mejoras para la interfaz de búsqueda con nuevas funcionalidades que facilitarían al usuario la localización de recursos de una manera más rápida y organizada.

¿Qué es Apache Solr?

Apache Solr o simplemente Solr (pronúnciese *solar*), es un popular motor de búsquedas de código abierto que proporciona potentes funcionalidades de búsqueda y navegación por facetas, es decir, explorar la información desde diversas perspectivas. Tales funcionalidades pueden ser difíciles de implementar sobre una base de datos relacional. Solr es capaz de manejar complejos criterios de búsqueda, corrige ortografía en las consultas, permite realzar resultados (*highlighting*), configurar la relevancia de términos, entre otras funcionalidades. Solr funciona como un servidor independiente de búsquedas a texto completo dentro de un contendedor de servlets.

Solr es un producto maduro utilizado por grandes empresas en sus servicios de búsqueda. Tal es el caso de la popular plataforma de video bajo demanda, Netflix. Solr está implementado en Java, y este mismo lenguaje se puede utilizar para extender sus funcionalidades mediante la implementación de simples interfaces.

Características

Algunas de las características más sobresalientes de Solr son:

Servidor con interfaz tipo REST (interacción vía HTTP, XML, JASON, CSV, etc.)

Esquema de datos configurable.

Utiliza varios caches para agilizar las búsquedas.

Interface Web de administración.

Navegación de resultados por facetas.

Escalable a varios servidores para búsquedas distribuidas.

Módulos de importación de datos desde bases de datos, e-mail y archivos de texto enriquecido (PDF, Word, RTF).

Análisis de texto (Tokenización, normalización, etc.)

Ventajas sobre tecnología de base de datos

Es cierto que la tecnología de base de datos más reciente ya integra capacidades de indexado de texto, entonces, ¿por qué no usar esta misma tecnología para la implementación de búsquedas en vez de integrar un nuevo elemento en la arquitectura de nuestros sistemas haciéndolos más complejos? Las ventajas de delegar el manejo de búsquedas a Solr, en vez de a una base de datos, resultan más claras si se hace una comparación entre ambas tecnologías.

La principal diferencia entre estas dos tecnologías es el modelo de datos. La característica que define a las bases de datos relacionales es su modelo de datos basado en tablas y la posibilidad de relacionarlas entre sí mediante llaves. De este modo, para una consulta que involucre varías tablas, es necesario el uso de la operación *JOIN* para unir las tablas implicadas, aunque dicha operación puede ser muy lenta si las tablas son muy grandes. En contraste, Solr utiliza un modelo de datos orientado a documentos que pueden pensarse como una base de datos de una sola tabla, por lo que una operación análoga a *JOIN* es inexistente. Un documento en Solr es simplemente un conjunto de campos, como una túpla en una tabla de una base de datos, con la diferencia de que cada columna puede ser multi valuada. Otra diferencia importante entre Solr y una base de datos es que en Solr no existe la operación equivalente a la operación UPDATE. Si cierta parte de un documento necesita ser actualizada, el documento debe ser eliminado y agregar en su lugar el documento actualizado.

1 de noviembre 2012 • Volumen 13 Número 11 • ISSN: 1067-60710

Si bien el paso de un esquema relacional a un esquema no relacional requiere un cambio en la estructura de la información, en la mayoría de los casos este proceso es directo y consiste en un proceso de normalización de los datos. Sin embargo, en ocasiones puede resultar difícil o imposible, especialmente en casos donde existen relaciones “muchos a muchos” con múltiples campos susceptibles a búsqueda.

Para una base de datos, la búsqueda sobre texto no es tan importante en comparación a otros aspectos, como el manejo de transacciones ACID, eficiencia en la inserción, actualización y recuperación de la información o el control de acceso. Con tecnología de base de datos solo es posible realizar búsquedas de subcadenas, es decir, búsquedas del estilo: *SELECT * FROM mitabla WHERE nombre LIKE ‘%Libros%’*. Solr, en cambio, es capaz de realizar búsqueda de términos (palabras), por lo que tiene incluso la capacidad para encontrar variaciones de una palabra, plurales y singulares, por ejemplo. También existe la posibilidad de calificar los resultados de acuerdo a un criterio configurable, lo cual permite identificar cuáles son los resultados que mejor cumplen con el patrón de búsqueda, lo cual no es posible de con tecnología de base de datos, ya que el patrón de búsqueda simplemente se cumple o no.

Basta revisar la lista de características de Solr para darse cuenta de que un sistema manejador de base de datos carece de muchas de estas funciones o no es tan bueno realizándolas.

Indexación y eliminación de documentos en Solr

Previamente a indexar documentos en Solr es necesario definir los campos que conforman los documentos que se indexarán y especificar el tipo de dato de cada campo, la obligatoriedad, entre otras opciones. Asimismo, es necesario definir el campo que identificará al documento de forma única dentro del servidor Solr. Esta configuración se define en el archivo *schema.xml*.

Como mencioné anteriormente, Solr proporciona una interfaz tipo REST para el envío y recepción de mensajes entre Solr y el cliente. Para indexar documentos se utiliza la instrucción ADD de Solr en formato XML, vía HTTP POST.

Por otro lado, para eliminar un documento se debe usar la instrucción *DELETE* con el identificador del elemento a eliminar, o bien, con un criterio de selección para eliminar varios registros a la vez:

```
<delete>
  <id>3007WFP</id>
</delete>
```

Finalmente, para que cualquier cambio se vea reflejado es indispensable enviar la instrucción COMMIT a Solr:

```
<commit/>
```

Lenguaje de consulta

Por último, para explotar el aspecto más interesante de esta herramienta, es necesario conocer el lenguaje de consulta usado por Solr. Este lenguaje soporta el uso de operadores booleanos, calificadores de campo útiles para aplicar un criterio de búsqueda a un campo específico, uso de comodines o wildcards, operadores de rango, soporte para el uso de funciones matemáticas, opciones de ordenamiento, por nombrar algunas de las posibilidades. Una consulta típica en Solr se hace vía un mensaje HTTP GET usando el parámetro *q* para especificar la consulta o *query*:

http://servidorsolr:8983/solr/collection1/select?q=titulo:Solr AND (autor:Miguel Estrada)

La petición HTTP anterior es el tipo de peticiones que nuestra aplicación debe generar para realizar búsquedas en Solr. En cuanto al formato de los resultados, Solr dispone de varios formatos como XML, JSON, Python, Ruby, PHP, CSV. Formatos como PHP o JSON son muy útiles si tenemos una aplicación implementada en PHP o Javascript, ya que de este modo evitamos tener que decodificar la respuestas XML. A modo de ilustración, a continuación se muestra las respuesta al mensaje anterior.

```
<response>
  <lst name="responseHeader">
    <int name="status">0</int>
    <int name="QTime">0</int>
    <lst name="params">
      <str name="wt">xml</str>
      <str name="q">titulo:Solr AND (autor:Miguel Estrada)</str>
    </lst>
  </lst>
  <result name="response" numFound="1" start="0">
    <doc>
      <str name="id">3007WFP</str>
      <str name="titulo">Introducción a Solr</str>
      <str name="tema">Optimización de búsquedas</str>
      <str name="descripcion">
        Alternativas a las bases de datos para la construcción de servicios de búsqueda
      </str>
      <str name="autor">Miguel Estrada</str>
      <arr name="palabras_clave">
        <str>optimización</str>
        <str>búsqueda</str>
      </arr>
    </doc>
  </result>
</response>
```

1 de noviembre 2012 • Volumen 13 Número 11 • ISSN: 1067-60710

Integración

El esquema en el que se combina el uso de base de datos con Solr suele ser el más adecuado para la mayoría de las aplicaciones. Este es el esquema usado por los sistemas Portal de Portales y Repositorio Institucional UNAM. En este esquema, Solr es simplemente el apoyo de la aplicación para búsquedas y no la fuente principal de la información, que sigue siendo la base de datos, es decir, como respuesta a una búsqueda, Solr regresa un conjunto de campos entre los que se encuentra la llave primaria de los elementos que conforman los resultados. Una vez identificada la información requerida, es recuperada de la base de datos usando las llaves primarias devueltas por Solr y desplegada al usuario. Ver figura 1.

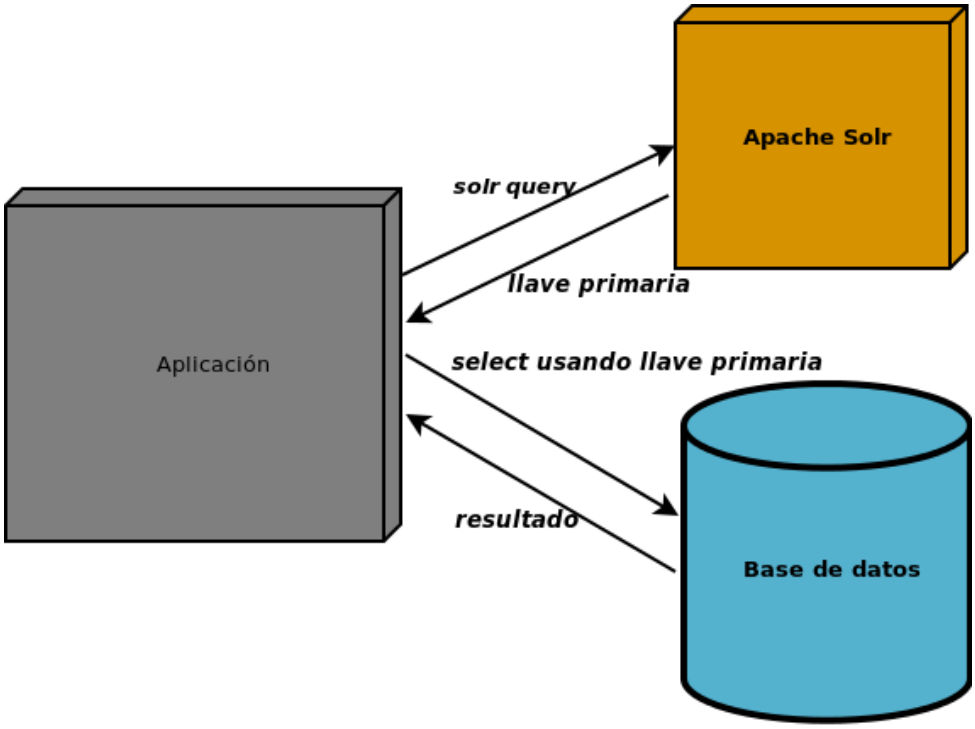

Figura 1: Esquema combinado de Base de Datos con Apache Solr

Solamente un análisis detallado de las necesidades de búsqueda de nuestra aplicación nos ayudará a tomar la mejor decisión en cuanto a la adopción de una tecnología como Solr. Si el sistema es, esencialmente, de consulta, es recomendable considerar el uso de esta herramienta.

APIs disponibles

Debido a su interfaz tipo REST y amplia variedad de formatos de salida Solr, es fácil de integrar en una variedad de ambientes. Puede integrarse de forma relativamente sencilla a desarrollos en JavaScript, Java, PHP, Ruby, Python y .NET.

Para aplicaciones desarrolladas en Java, Solr incluye SolrJ. SolrJ es una API de programación que proporciona un sencillo modelo de objetos para representar la interacción con Solr. Adicionalmente, Solr tiene la habilidad de especificar peticiones y respuestas en un formato Java binario conocido como *javabin*, que es mucho más rápido y menos pesado que XML, con lo cual se evita la carga extra de procesamiento para decodificar el XML. De forma predeterminada, SolrJ usa *javabin* para la interacción con el servidor Solr. Adicionalmente, SolrJ permite la indexación de POJOS (Plain Old Java Objets).

A pesar de que existen varias opciones de APIs de integración para aplicaciones escritas en PHP, en el caso de los proyectos Portal de Portales Latindex y Repositorio Institucional de la UNAM, las herramientas de software sobre la que se construyeron estos sistemas no utilizan ninguna de estas APIs.

El trabajo realizado en estos sistemas se enfocó mayormente en explotar las funcionalidades que proporciona Solr para implementar un servicio de búsqueda que facilitara al usuario la localización de la información de su interés, sin cambiar las interfaces de las herramientas subyacentes.

No obstante, a continuación muestro un ejemplo de integración usando la librería Solarium, disponible para su descarga en http://www.solarium-project.org

```php
<?php
require('../library/Solarium/Autoloader.php');
Solarium_Autoloader::register();

// crear una instancia del cliente
$client = new Solarium_Client();

$query = $client->createSelect();

//especificamos el criterio de búsqueda
$query->setQuery('titulo:Solr');

// ejecutamos la consulta
$resultset = $client->select($query);

// desplegamos los resultados
foreach ($resultset as $document) {
    echo 'documento:\n';
    foreach($document AS $field => $value)
    {
        // separamos los valores del campo por comas
        if(is_array($value)) $value = implode(', ', $value);

        echo 'campo: $field valor: $value';
    }
}
?>
```

1 de noviembre 2012 • Volumen 13 Número 11 • ISSN: 1067-60710

Como se observa en el ejemplo anterior, la integración con Solr se reduce a interacciones entre objetos con lo cual la interfaz tipo REST de Solr queda encapsulada.

Conclusión

Apache Solr representa una alternativa fácil de integrar a nuestros desarrollos con grandes beneficios para la calidad de servicios de búsqueda, proporcionándonos funcionalidades de búsqueda compleja y velocidades de respuesta mayores a los de una base de datos. Si bien la introducción de esta herramienta nos obliga a replantearnos la arquitectura de nuestro sistema y ver la información de una manera diferente, los beneficios pueden ser notables y la implementación de funcionalidades de búsqueda puede agilizarse significativamente si se utiliza alguna de las APIs existentes.

Referencias

Smiley, David. *Apache Solr 3 Enterprice Search Server*.Packt Publishing, 2012.

Smiley, David, Solr 1.4 *Enterprice Search Server*. Packt Publishing, 2009.

Apache Solr, [en línea]. [Consultada: 28 de octubre de 2012] Disponible en Internet:

[http://lucene.apache.org/solr/4_0_0/tutorial.html]

Chávez Sánchez, Guillermo, Ortiz Camilo, Miguel Ángel. *Un portal de acceso abierto a la literatura científica en Iberoamérica* Revista Digital Universitaria [en línea]. 1 de octubre de 2012, Vol. 13, No.10 [Consultada: 29 de octubre de 2012] Disponible en Internet: [http://www.revista.unam.mx/vol.13/num10/art104/index.html]

Muñetón Pérez, Patricia. *RAD (Red de Acervos Digitales) de la UNAM. Entrevista con la Dra. Isabel Galina Russell* Revista Digital Universitaria [en línea]. 1 de abril de 2012, Vol. 11, No. 4 [Consultada: 29 de octubre de 2012] Disponible en internet: [http://www.revista.unam.mx/vol.11/num4/art41/