



UNIVERSIDAD DE GRANADA

TRABAJO FIN DE MÁSTER UNIVERSITARIO INGENIERÍA INFORMÁTICA

Sistema de Recuperación de Información para Intranets basado en Elasticsearch

Autor

César Albusac Jorge



Escuela Técnica Superior de Ingenierías Informática y de Telecomunicación

—
Granada, 15 de septiembre de 2017

Sistema de Recuperación de Información para Intranets basado en ElasticSearch

César Albusac Jorge

Palabras clave: Sistema de recuperación de Información, buscador, Intranet, ElasticSearch, documentos, relevancia.

Resumen

Este Trabajo Fin de Máster propone una solución inteligente a un problema real existente en muchas empresas privadas y por lo tanto intranets de las mismas: la búsqueda de documentos almacenados en su servidor.

Nos encontramos en la era de la información, donde cada día se generan PetaBytes de información en internet, y donde, a su vez, también se genera mucha información en las empresas. Todos los documentos suelen almacenarse en un servidor interno al cual pueden acceder los trabajadores de la misma. Suele estar distribuido por carpetas y por departamentos, pudiendo así almacenar los archivos en el mismo, teniendo los usuarios acceso a un número limitado de carpetas en red. Pero llega un momento en el que ya existen demasiados departamentos, categorías, subcategorías, etc, y empieza a ser un problema el encontrar un documento de hace algunas semanas que se guardó en algún lugar del servidor o servidores.

Este trabajo consiste en la realización de una aplicación web que permita a los trabajadores de una empresa y a todos los que tengan acceso a la intranet, a buscar los documentos que necesitan, de manera eficiente y rápida.

Para realizar este proyecto, vamos a utilizar una biblioteca muy potente llamada ElasticSearch [1], que está basada en Lucene [2], y la adaptamos a las necesidades internas de la empresa para optimizar el funcionamiento del mismo. Así, indexaremos todos los documentos pertenecientes a un servidor o servidores, y haremos las operaciones necesarias con los datos para tratar de minimizar el tiempo transcurrido entre la consulta del usuario y la respuesta del sistema con los documentos relevantes.

De este modo, un usuario podrá buscar en cuestión de segundos dónde está el documento que desea encontrar (en caso de que exista), o alguno parecido que le pueda servir de ayuda. Podrá visualizarlo o descargarlo sin perder tiempo abriendo rutas del servidor, o abriendo documentos para ver si sus contenidos eran los esperados para el mismo.

Information Search and Retrieval System for Intranets based on ElasticSearch

César Albusac Jorge

Keywords: Information Search and Retrieval System, Search Engine, Intranet, documents, ElasticSearch, relevance.

Abstract

This Final Master Project proposes an intelligent solution for a real problem that is frequent in most of the private companies that have an intranet: the search of documents that are kept in their internal servers.

We are inverse in the information Area, where each day there are generate PetaBytes of Information in Internet, and at the same time, lots of information is generated by companies.

Documents are frequently stored in an internal server and which employers can access to them. It uses to be distributed by folders and departments, giving the possibility of storage the documents and giving users limited privileges to access them. But the amount of documents starts to grow and it is difficult to manage all departments, categories, subcategories, and start to be a problem to find a documents that you created a couple of weeks ago and you stored it in any place of the server.

This project consists on develop a web application that allow employers of a company and those who has permissions and internet access to search documents that they need, in an efficient and quick way.

To do this project, we are going to use a very powerful library called ElasticSearch [1], that is based on Lucene [2], and we will adapt it to the internal company needs to optimize the functioning. We will index all documents belonging to a server or servers, and we will implement the necessary operations with data to try to minimize time between the user query and the response of the system with the relevant documents.

In this mode, a user will be able to search in seconds where is the documents that need to find (if exists), or any similar one that can help. The user could see and download it without lose time opening server routers or opening documents to check if the contain if what he/she expected.

Índice de Figuras

Figura 1 - Indexación.....	25
Figura 2 - Gráfico del ranking de Buscadores.....	35
Figura 3 - Ranking de puntuación de buscadores.....	35
Figura 4 - Diagrama de Casos de Uso.....	54
Figura 5 - Diagrama de Gantt Inicial.....	55
Figura 6 - Diagrama de Clases.....	58
Figura 7 - Arquitectura Hardware de la aplicación.....	59
Figura 8 - Arquitectura de la aplicación web.....	60
Figura 9 - Código para crear el cliente de ES.....	62
Figura 10 - Código para obtener sugerencias.....	64
Figura 11 - Código de Búsqueda General.....	66
Figura 12 - Código de Búsqueda de Imágenes.....	67
Figura 13 - Página principal del SRI.....	74
Figura 14 - Ejemplo Autocompletar.....	74
Figura 15 - Resumen Resultados.....	75
Figura 16 - Ejemplo de archivos encontrados.....	75
Figura 17 - Ejemplo Búsqueda de Audio.....	76
Figura 18 - Ejemplo Búsqueda de Imágenes.....	77
Figura 19 - Ejemplo Búsqueda de Vídeos.....	78
Figura 20 - Página de Configuración.....	79
Figura 21 - Temporización Gantt Final.....	80
Figura 22 - Diagrama de Gantt Final.....	80

Índice de tablas

Tabla 1 - Comparación BBDD vs RI.....	27
Tabla 2 - Actor Administrador.....	45
Tabla 3 - Actor Usuario final.....	45
Tabla 4 - Caso de Uso Indexar Documentos.....	47
Tabla 5 - Caso de Uso Actualizar Documentos.....	48
Tabla 6 - Caso de Uso Borrar Documentos.....	49
Tabla 7 - Caso de Uso Búsqueda General.....	50
Tabla 8 - Caso de Uso Búsqueda Imágenes.....	51
Tabla 9 - Caso de Uso Búsqueda Audio.....	52
Tabla 10 - Caso de Uso Búsqueda Video.....	53
Tabla 11 - Planificación temporal del trabajo.....	55
Tabla 12 - Costes de personal.....	56
Tabla 13 - Costes Hardware.....	56
Tabla 14 - Costes Software.....	56
Tabla 15 - Coste total del proyecto.....	57
Tabla 16 - Prueba PRU-01.....	69
Tabla 17 - Prueba PRU-02.....	69
Tabla 18 - Prueba PRU-03.....	70
Tabla 19 - Prueba PRU-04.....	70
Tabla 20 - Prueba PRU-05.....	70
Tabla 21 - Prueba PRU-06.....	71
Tabla 22 - Prueba PRU-07.....	71
Tabla 23 - Prueba PRU-08.....	71
Tabla 24 - Prueba PRU-09.....	72
Tabla 25 - Prueba PRU-10.....	72
Tabla 26 - Prueba PRU-11.....	73
Tabla 27 - Resultado de las pruebas.....	73
Tabla 28 - Temporización final real.....	80
Tabla 29 - Costes de personal final.....	81
Tabla 30 - Presupuesto final real.....	81

Índice

Resumen	7
Abstract	9
Agradecimientos	15
1. Introducción	23
1.1 Motivación	23
1.2 Contexto en el que desarrolla	23
2. Objetivos	23
3. Introducción a los SRI	24
3.1 Introducción a la RI	25
3.2 Estudio del estado del arte	28
3.3 Introducción a ElasticSearch	34
4. Solución anterior	37
5. Análisis y planificación	38
5.1 Requisitos	38
5.1.1 Interfaces Externas	39
5.1.2 Requisitos funcionales	39
5.1.3 Requisitos de rendimiento	44
5.1.4 Requisitos tecnológicos	44
5.1.5 Requisitos de Fiabilidad y Disponibilidad	44
5.1.6 Requisitos de Implementación	44
5.1.7 Otros requisitos	45
5.2 Diagramas de casos de uso	45
5.2.1 Especificación de actores	45
5.2.2 Casos de Uso	46
5.2.3 Diagrama de Casos de Uso	53
5.3 Planificación y presupuesto	55
6. Diseño	57
6.1 Diagrama de clases	57
6.2 Arquitectura	59
7. Implementación y pruebas	60
7.1 Herramientas utilizadas	60
7.2 Creación de los documentos: Crawler de documentos	61
7.3 Creación del índice de ES y mapping	62
7.4 Operaciones principales	63
7.5 Relevancia de los resultados	67
7.6 Depuración y pruebas	68
7.6.1 Depuración	68
7.6.2 Catálogo de pruebas	68
7.7 Interfaz Web	74
8. Conclusiones y trabajo futuro	79
8.1 Temporización y presupuesto reales	79
8.2 Relación con los estudios	81
8.3 Valoración personal	82
8.4 Conclusiones	82
8.5 Trabajo futuro	82
9. Comentarios sobre fuentes principales utilizadas	83
10. Bibliografía	84

1. Introducción

Vivimos en la era de la información, también conocida como era Digital [3] o era Informática, donde las tecnologías de la información y las comunicaciones juegan un papel vital en nuestra sociedad actual. Hoy en día, todo está interconectado, todas las empresas y principales aplicaciones generan una cantidad grandísima de información, superando los 2,5 quintillones de bytes al mes [4]. Esto genera dos problemas principales: cómo almacenar esa gran cantidad de datos y cómo poder acceder a los mismos de manera eficiente. Tenemos por tanto un problema con el exceso de información. Las bases de datos tradicionales tienen límites en cuanto a la cantidad de información que pueden almacenar y pueden llegar al colapso.

Así, se llega al Sistema de Recuperación de Información (SRI) como solución, es decir, una solución que utilice la RI [5]. Se propone una manera alternativa de “almacenar” los documentos que ya no es simplemente guardar los mismos en archivos y directorios.

1.1 Motivación

El presente Trabajo Fin de Máster se desarrolla en el campo de la Recuperación de Información y se pretende dar una solución a un problema cada vez más común en la era de la información en la que estamos inmersos: el manejo eficiente de los documentos y una recuperación rápida y eficaz por parte de un sistema de Recuperación de Información. Se ha realizado una solución aplicable a cualquier empresa que posea su propia red con los equipos necesarios. Se ponen a prueba conocimientos sobre RI, inteligencia artificial y programación web.

1.2 Contexto en el que desarrolla

El contexto en el que se desarrolla esta solución, es el de la necesidad de mejorar un sistema de recuperación ya implementado en el Mando de Adiestramiento y Doctrina (MADOC) [6]. Aunque este modelo no era óptimo, su funcionalidad dejaba satisfechos a todos usuarios que lo utilizaban. Si bien es cierto que a medida que ha pasado el tiempo, se han ido incrementando de manera casi exponencial el número de documentos que poseían en sus servidores. Surgía así la necesidad de crear otra solución que fuese capaz de mejorar el modelo.

2. Objetivos

Como hemos indicado al comienzo, el objetivo principal es crear una solución eficiente al problema que se planteó al comienzo de esta memoria: La gestión de los archivos internos de una empresa/Intranet para posibilitar su búsqueda. Así, los dos objetivos principales serán:

- **Implementar una solución con un motor de búsqueda que permita la gestión de los principales archivos.**

- **Comparar la eficiencia y la eficacia del nuevo modelo con el existente anteriormente.**

La solución implementada tendrá diversos módulos asociados tanto para indexaciones, borrado, actualizaciones y búsqueda. Los archivos con los que vamos a trabajar son archivos de texto, hipertexto, audio, vídeo e imagen.

Dentro del desarrollo del sistema de RI que hemos mencionado, cabe destacar otros subobjetivos que son necesarios para la gestión:

- Gestión de usuarios a los que se le permite acceder al sistema.
- Gestión de las ubicaciones donde se encuentran los documentos que deseamos que nuestro sistema tenga indexados.
- Gestión de indexación de los documentos.
- Gestión de la actualización de los documentos del índice.
- Gestión de borrado de los documentos.
- Opción de autocompletar: Objetivo deseable ya que en muchas ocasiones minimiza los tiempos de búsqueda.
- Aplicación de filtros a las búsquedas: Objetivo importante, ya que permite distinguir entre documentos similares, pero de distintos departamentos en este caso.

3. Introducción a los SRI

En esta sección vamos a tratar de definir algunos términos que son importantes en relación con este trabajo, y definiremos en qué va a consistir el mismo.

En primer lugar, debemos saber qué es un Sistema de Recuperación de Información (S.R.I.). Un S.R.I está formado por tres componentes: usuario, fuente de conocimiento e intermediario. Partiendo de la base de que el usuario tiene un problema y necesita de una fuente externa de conocimiento para resolverlo, el objetivo final será que ese problema se resuelva. Así, tenemos que facilitar en la medida de nuestras posibilidades la interacción del usuario con la información [7].

Por lo tanto, la tarea de un S.R.I será predecir, conociendo la fuente de recursos y al usuario, qué objetivos son los más adecuados para que el usuario pueda resolver su problema. Los pasos a seguir para realizar esto son:

1. Representar el problema de necesidad de información del usuario (consulta, normalmente de texto).
2. Representar y organizar el contenido de la fuente de conocimiento.
3. Comparar la consulta con los componentes del contenido.
4. Presentar los resultados al usuario para que interactúe o los juzgue.

La recuperación de información tiene dos entradas:

- > Conjunto de documentos (texto en lenguaje natural).
- > Consulta de un usuario (también en lenguaje natural).

Y una sola salida:

- > Conjunto ordenado de documentos que satisfacen mejor la consulta, cuyo ordenamiento se ha realizado conforme al cálculo de una puntuación o score, una

medida de un objeto de información con respecto a la situación del problema del usuario.

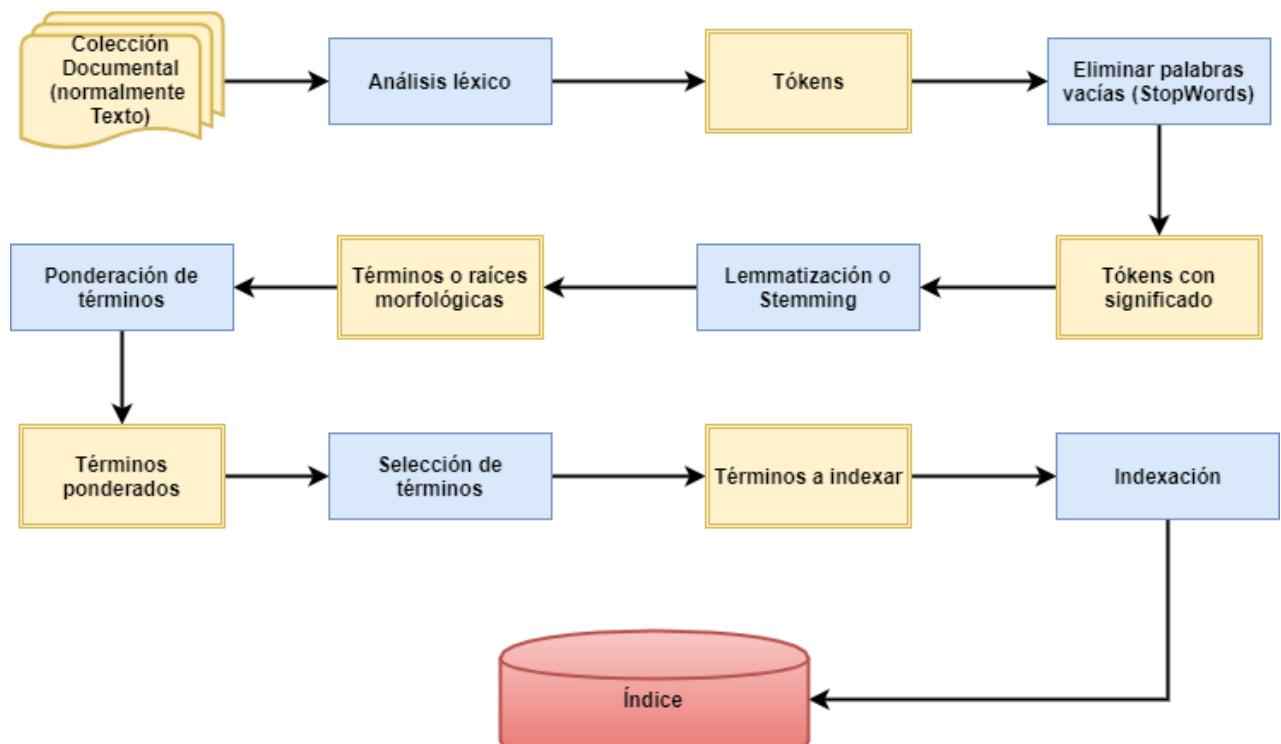
Las características deseables de estos documentos obtenidos con la búsqueda es que sean útiles, es decir, que sean relevantes para nuestra consulta, y se haya realizado de forma eficiente, o lo que es lo mismo, que obtengamos esos documentos intentando minimizar el costo en tiempo.

3.1 Introducción a la RI

Según Manning, Raghavan, 2008 [8] “*RI es la disciplina que trata de encontrar material (típicamente documentos) de una naturaleza desestructurada (típicamente texto) que satisface una necesidad de información en una colección grande (típicamente almacenada en ordenadores)*”.

En la Recuperación de Información se realizan diversos procesos que veremos a continuación.

El primero de ellos es la Indexación [9] (Figura 1): una vez tenemos la colección de datos que deseamos indexar, debemos crear las estructuras de datos para representar los mismos de manera eficiente para realizar la búsqueda posteriormente de manera lo más rápida posible. El proceso pasa por varios pasos hasta que se forma el Índice.



Vamos a explicar los términos relacionados con un S.R.I:

- **Documento** [10]: Es la unidad básica de información que será indexada en el S.R.I.
- **Índice** [10]: Es una colección de documentos que tienen o suelen tener algo en común. Por ejemplo, si tuviésemos datos sobre una empresa de Automóviles, podemos crear un índice para Ventas, otro para Compras y otro para Alquileres. Dentro de un clúster podemos crear tantos índices como deseemos.
- **Tokenization** [12][13]: Es el proceso de romper una frase o palabra en palabras individuales o tokens. Debemos conocer los documentos y datos que tenemos para saber cómo vamos a tratar los espacios en blanco, dígitos, guiones, signos de puntuación, letras mayúsculas al principio, etc.
- **StopWords** [16][17][18]: También llamadas palabras vacías, son aquellas palabras que tienen una alta frecuencia de aparición en un idioma o del dominio en el que estemos y que no aportan ningún significado a nuestra búsqueda. Son artículos, determinantes, pronombres, preposiciones, etc. Como son muy frecuentes, estas palabras nos pueden ofrecer algunas ventajas como:
 - Reducir el número de palabras que indexamos, por lo que ahorramos espacio.
 - Mejorar la eficacia para recuperar los documentos.

Por ejemplo, algunas stopwords o palabras vacías en inglés son: *a, an, and, are, as, at, be, but, by, for, if, in, into, is, it, no, not, of, on, or, such, that, the, their, then, there, these, they, this, to, was, will, with*.

O en castellano: *un, una, unas, unos, uno, sobre, todo, también, tras, otro, algún, alguno, alguna, algunos, algunas, ser, es, soy, eres, somos, sois, estoy, está, estamos, estáis, están, cómo, en, para, atrás, porque, por qué, estado, estaba, ante, antes, siendo, ambos, pero, por, poder puede, puedo, podemos, podéis, pueden fui, fue, fuimos, fueron, etc.*

Debemos tener cuidado con las palabras vacías, ya que, al eliminarlas, estamos reduciendo la capacidad para realizar algunos tipos de búsquedas. Nos impediría realizar lo siguiente:

- Distinguir entre “happy” y “not happy”.
- Encontrar la frase de Shakespeare “to be or not to be”.
- Buscar por el código de Noruega, “no”.

- **Stemming** [14][15]: Es la técnica que nos permite reducir las palabras a sus raíces. Esta técnica aumenta considerablemente el número de documentos que se pueden encontrar con la búsqueda. Por ejemplo, si aplicamos Stemming en nuestro S.R.I y buscamos “Floricultura” y “Florecer”, obtendremos todos los resultados que tengan flor (la raíz). Hay que tener cuidado con esta técnica, pues podemos tener dos problemas: Understemming y Overstemming:
 - **Understemming** [14]: es el error que cometemos cuando no reducimos las palabras con el mismo significado a la misma raíz. Por ejemplo, si tenemos las palabras en inglés “jumped” y “jumps”, se reducirán a jump, mientras que

“jumping” se reducirá a “jumpi”. Este problema hace que se devuelvan menos documentos relevantes.

- **OverStemming** [14]: Es el error que se comete al reducir a la misma raíz dos palabras diferentes. Por ejemplo, “general” y “generate” son reducidos a “gener”. Se reduce así la precisión porque nos devolverá documentos que no sean relevantes para nuestra búsqueda.
- **Lematización** [14] [15]: Al igual que Stemming, la lematización trata de agrupar las palabras que están relacionadas entre sí, pero va un paso más allá e intenta agrupar las palabras por el significado de las mismas, es decir, el lema al que pertenece. La misma palabra puede representar varios significados. Por ejemplo, la palabra “banco” puede significar una entidad bancaria, un conjunto de peces o un lugar para sentarse. Mientras que la lematización intentaría distinguir entre los sentidos de la palabra, Stemming las combinaría de forma incorrecta.

A continuación, podemos ver una comparación [19] entre el modelo “clásico” de Bases de Datos y un Modelo de Recuperación de Información:

	Bases de datos	Recuperación de Información
Tipo de Datos	Estructurados	No estructurados
Campos	Semántica clara	No hay campos (sólo texto)
Consultas	Definida (Álgebra relacional, SQL)	Texto libre (lenguaje natural, booleano)
Recuperación	Crítica (control de concurrencia, operaciones atómicas)	Minimizar
Comparación de resultados	Exacta (Resultados son siempre “correctos”)	Imprecisa (se debe considerar la efectividad, ranking)

Tabla 1 - Comparación BBDD vs RI

Vamos a comentar brevemente la tabla anterior:

Tipos de datos: en las Bases de datos clásicas (modelo jerárquico, de red y relacional), los datos van siguiendo siempre algún tipo de orden, haciendo uso de atributos o campos. Por lo contrario, en la RI los datos son mayormente no estructurados: Los datos son conjuntos de documentos que contienen texto.

Campos (Semántica de datos): En las bases de datos, la semántica de datos es clara. Hay un esquema por base de datos, que define el tipo de datos, nombre, relaciones y otro atributo. Por el contrario, en RI se trata con datos desestructurados donde la semántica de los datos es más ambigua

Consultas: Las consultas se realizan en lenguaje SQL y álgebra relacional, mientras que en la RI se hace uso del lenguaje natural y booleano para obtener los documentos que deseamos.

Recuperación: En las BBDD la recuperación es crítica, además de que debemos controlar la concurrencia ya que las operaciones son atómicas y pueden provocar errores. Por el

contrario, en la RI, en la recuperación se trata de minimizar el tiempo en el que se obtienen los resultados.

Resultados: Mientras que las BBDD ofrecen una coincidencia exacta, la RI ofrece un ranking de relevancia ordenado por una puntuación calculada internamente. Es una búsqueda más subjetiva.

3.2 Estudio del estado del arte

Por todos es conocido que, desde hace ya unos años, el sistema más grande y potente para la recuperación de información hoy en día es Google, y, probablemente, lo seguirá siendo. Es un sistema que rastrea toda la web en busca de contenido de diferente naturaleza y la indexa en su buscador para hacerlo accesible a todos los usuarios. Lo ideal sería tener un buscador como Google dentro de la empresa. Es lo que vamos a intentar realizar, siendo conscientes de las limitaciones que tenemos. Hay que tener en cuenta que nuestro objetivo es encontrar un motor de búsqueda abierto y gratuito. A continuación, se van a explicar varias alternativas (algunas de pago) que actualmente existen en el mercado, describiendo por encima sus características y veremos una pequeña tabla resumen con una comparación entre algunas de ellas:

1. **Searchxml** [20]:

Se describe como “la primera base de datos todo en uno para datos estructurados y no estructurados”:

- Tienen una interfaz de usuario sencilla y está diseñada tanto para entornos cloud como para utilizarlo en el lugar físico de la empresa.
- Posee interfaces de comunicación HTTP/HTTPS, servicios web y WEB DAV [21].
- Está automatizada: producciones automáticas de flujos de trabajo.
- Mantenimiento por sí misma.
- Capacidades multi-cliente: Conceptos de jerarquía (sitio, usuario, rol), múltiples bases de datos de usuario (una por sitio), interfaces separadas.
- Proporciona diferentes estrategias de búsqueda para datos estructurados y no estructurados: Combina consultas de XML, búsqueda de proximidad XML y búsqueda full-text para incrementar la escalabilidad y velocidad.

2. **DBSight** [22]:

Es una plataforma para utilizar una búsqueda full-text de alto rendimiento. Los usuarios sólo especifican los comandos SQL y seleccionan las plantillas de renderizado. **Está basado en la librería de Lucene**, con facetas, ordenamiento, clasificación y almacenamiento en caché integrada. Es mucho más rápida que una búsqueda en una base de datos, y dicen estar a la altura o por encima de cualquier otra solución de Lucene. Entre sus características:

- Extracción eficiente SQL: Sólo se necesitan conocimientos en SQL para crear un buscador. Se generarán los esquemas basados en el SQL y no se necesita escribir Java o XML. El extractor del motor SQL es eficiente debido a la caché y a la selección de lotes.

- Construcciones potentes de Plantillas: es personalizable con un potente sistema de scaffolding. Las soluciones preconstruidas incluyen el motor de búsqueda, dataGrid, JSON/JSONP, Excel, CSV, XML, etc.
- Permite realizar una búsqueda múltiple en una página a diferentes índices.
- Posibilidad de realizar sugerencias mientras el usuario está escribiendo.
- Permite “particionar” el índice cuando aumenta de tamaño y realizar las búsquedas en los shards.
- Permite el uso de las facetas para acotar la búsqueda, ya sea por fechas, categoría, rangos de valores, etc.

3. Exorbyte [23]:

Se describen como líderes del mercado de búsqueda: proporcionan búsqueda en tiempo real tolerante a fallos utilizando un sólo campo de búsqueda. Las principales prestaciones que este buscador proporciona son:

- Búsqueda de identidades: permite buscar a personas con identidad mediante su DNI, nombre o fecha de nacimiento.
- Búsqueda en central de llamadas: búsqueda tolerante a fallos en tiempo real donde se integran todos los datos del cliente.
- Búsqueda comercial: se realiza en las “tiendas inteligentes” con muchas funciones, rankings, rangos, etc.
- Búsqueda en web: Búsqueda en toda la web de contenidos y documentos. El contenido puede ser rastreado, leído directamente o excluido.

4. Indica [24]:

Proporciona a las empresas un control sobre sus datos en tiempo real, así como el descubrimiento de datos y búsqueda en la empresa. Proporciona:

- Visualizaciones: Plantillas construidas para permitir captar idea sobre los datos.
- Soluciones preparadas para trabajar, sin necesidad de leer manuales.
- Fácil de usar: interfaz intuitiva.
- Resultados avanzados: Poseen un algoritmo patentado para encontrar relaciones entre los datos estructurados y los datos no estructurados.

5. SearchBlox [25]:

Ofrecen búsqueda para la empresa, análisis de sentimientos y análisis de textos. Las posibilidades que ofrece esta solución son:

- Búsqueda en el sitio: Crea una colección para rastrear tu página web, añade un campo de búsqueda en tu página y configura los resultados de búsqueda de tu empresa.
- Búsqueda en comercio electrónico: Crea una colección para buscar en su catálogo de productos disponibles en una base de datos o un archivo CSV/Excel. Se pueden aplicar diferentes filtros, como marca, precio, disponibilidad, etc.
- Búsqueda en Intranet: Ayuda a los usuarios de una intranet a encontrar información que se localiza en unidades compartidas, bases de datos, etc. Ofrece búsqueda por facetas para encontrar los datos con una sola consulta. Éste sería una de los objetivos de este proyecto.
- Búsqueda en Cloud: Puedes añadir el buscador a tu infraestructura cloud fácilmente con Amazon Web Services.

- SearchBlox posee una API para que lo integres en una aplicación o página web. Permite crear tu colección e indexar urls o datos. También permite eliminar o actualizar rápidamente los mismos.
- Búsqueda Big Data: permite buscar en grandes cantidades de datos que se encuentran ya indexados.

6. **Xapian** [26]:

Es una librería Open Source de motores de búsqueda. Está escrita en C++, aunque también se permite utilizar Perl, Python, PHP, Java, C#, R, etc.

Esta librería posee las siguientes características:

- Portable: puede funcionar en Linux, Mac OS X, Windows, Solaris, OpenBSD, HP-UX, etc.
- Búsqueda por ranking probabilístico: las palabras más importantes tienen más peso y pueden mejorar los resultados.
- Feedback: Dado uno o más documentos, Xapian puede sugerir otros documentos relacionados u ofrecer términos para expandir la consulta.
- Frase de búsqueda y de proximidad: Se puede realizar búsqueda exhaustiva por frases exactas o que se parezcan.
- Operadores booleanos para la búsqueda, cuyos resultados serán ordenados por pesos probabilísticos. Los filtros booleanos pueden aplicarse también para restringir una búsqueda probabilística.
- Puede aplicar stemming a los términos buscados. Esto ayuda a encontrar documentos relevantes que podrían no tenerse en cuenta.
- Búsqueda con comodines.
- Uso de sinónimos.
- Uso de búsquedas con facetas.
- Soporta archivos de Bases de datos más grandes de 2 GB.
- Permite actualización y búsqueda de manera simultánea.

7. **CrateDB** [27]:

SQL para los datos de las “cosas” (IoT). Guardan y analizan grandes cantidades de datos en tiempo real. Ofrece SQL standard, consultas en tiempo real y soporte (JSON) de manera simple y escalable de forma horizontal. Está más orientada a sensores, Internet de las Cosas, Logs, analíticas de eventos, analítica geoespacial y de series temporales.

8. **Amazon CloudSearch** [28]:

Solución de Amazon para búsquedas en un sitio web o aplicación. Además de poseer características cruciales como Escalabilidad y Seguridad, también es sencillo, totalmente administrativo y entre las características a realizar en la búsqueda podemos encontrar las siguientes:

- Búsqueda de texto libre, booleanas y facetadas.
- Sugerencias de autocompletar.
- Clasificación por importancia.
- Ponderación de campos.
- Búsqueda geoespacial.
- Resaltado (Highlight).
- Soporte 24 idiomas.
- **Es de pago.** Tarifas reducidas por hora y pago por los recursos que se utilicen.

9. Algolia [29]:

Proporciona una solución de búsqueda para las empresas rápida y con exactitud, de manera escalable y segura. Sus características son las siguientes:

- Tolerancia a fallos de escritura: Los usuarios encontrarán el resultado que desean encontrar, aunque se equivoquen en algunos caracteres.
- Búsqueda Geográfica.
- Utilización de sinónimos: con el uso de los mismos podemos ayudar a los usuarios a encontrar aquello que buscan, aunque hayan utilizado otra palabra cuyo significado sea el mismo.
- Posibilidad de utilizar diferentes idiomas.
- Utilización de filtros y facetas.

10. Google Search Appliance [30]:

Solución del gigante Google para empresas, que consta de las siguientes características:

- Ordenación por metadatos: por ejemplo, por fecha, autor, precio, facilitando así al usuario encontrar lo que desean.
- Búsqueda con comodines: si los usuarios no saben cómo se busca algo o desean buscar términos similares a la vez.
- Reconocimiento de entidades: se utiliza para estructurar el contenido analizando fechas, autores, etc.
- Corrector ortográfico y sinónimos: sugiere las formas más comunes de las palabras y amplía las consultas realizándose también con los sinónimos.
- Traducción: posibilidad de traducir los documentos a otros idiomas.
- Marcador de autoaprendizaje: crea perfiles para los usuarios para así poder ajustarlas a futuras búsquedas y obtener mejores resultados. En el algoritmo de ordenación tiene importancia los clicks o hits que reciben los enlaces.
- Búsqueda de expertos: facilita la búsqueda de expertos utilizando palabras clave.
- Resultados añadidos por el usuario: los usuarios pueden interactuar con el sistema para añadir resultados a búsquedas que se realicen.
- Vista previa del documento, dando la posibilidad de saber si el documento que se está visualizando satisfará al usuario o no.
- Función de autocompletar.
- Posibilidad de tener colecciones ilimitadas.
- Seguridad: mantiene el control de acceso mediante los permisos de los usuarios, permite autenticación rápida y sencilla con protocolos como Kerberos, NTLM y LDAP.
- De pago.

11. Microsoft Azure Search [31]:

Servicio de Azure de búsqueda en la nube para el desarrollo de aplicaciones web y móviles. Proporciona:

- Escalabilidad.
- Conectividad de los resultados con las empresas.
- Procesamiento del lenguaje natural.
- Carga y actualización con indexadores integrados.
- Búsqueda geoespacial.

- Búsqueda de texto complejo y análisis de texto: Se pueden hacer las consultas con sintaxis de consulta simple, con operadores lógicos, operadores de búsqueda de frase, operadores de sufijo y de procedencia. También, mediante la sintaxis de búsqueda de Lucene se puede realizar búsqueda aproximada, por prioridad y con expresiones regulares.
- Compatibilidad de idiomas, mediante la utilización de analizadores de Microsoft y de Lucene.
- Integración de datos: se pueden insertar datos JSON para llenar un índice, aunque también se pueden utilizar otros indexadores como Azure SQL database, Azure Cosmos DB, etc. Se permite indexar documentos de los principales formatos, entre ellos documentos de Microsoft Office, PDF y HTML.
- Experiencia de búsqueda: también ofrece la posibilidad de realizar sugerencias de búsqueda, búsqueda por facetas, aplicación de filtros, resultado de las referencias, puntuación simple, clasificación, paginación, etc.
- De pago.

12. Sphinx [32]:

Es un motor de búsqueda de texto completo, bajo la licencia GPL v2. Se puede conseguir licencia comercial realizando una petición. Entre sus características:

- 3 métodos de acceso: protocolo de red MySQL de Sphinx, mediante la API nativa de búsqueda, o bien mediante servidor MySQL con una extensión (SphinxSE).
- La API tiene implementaciones para varios lenguajes: PHP, Python, Ruby, Java. Aunque también hay plugins para Perl, C#, Haskell y Ruby on Rails.
- Indexación y búsqueda de alto rendimiento.
- Indexación avanzada y herramientas de consulta, como tokenizadores, diferentes modelos de ranking, etc.
- Post procesamiento (Select con expresiones, WHERE, ORDER BY, GROUP BY,etc..).
- Escalabilidad.
- Integración fácil con SQL y datos XML.
- Consultas booleanas, con frases, por proximidad de palabras...
- Uso de stopwords, diccionarios, Stemming, etc.

13. MarkLogic [33]:

Es una base de datos empresarial transaccional y operativa que integra datos NoSQL (Not only SQL) de forma rápida y menos costosa. Algunas características:

- Modelo de datos flexible: no hay que preocuparse por los esquemas predefinidos de los datos.
- Facilidad de obtención de datos: indexa todos los datos, incluyendo JSON, XML, textos, etc, pudiendo obtenerlos en tiempo real y obtener resultados rápidamente.
- 100% de confianza: Permite realizar transacciones ACID [34], alta disponibilidad y recuperación ante desastres. También proporciona seguridad.
- Una vez realizada la aplicación puedes desplegarla en cualquier entorno, tanto cloud, como virtualizado o en las instalaciones. Puede desplegarse en AWS, Google Cloud o Microsoft Azure.
- De pago.

14. Splunk [35]:

Monitorea y analiza datos procedentes de cualquier fuente para entregar inteligencia operacional para optimizar el IT, la seguridad y el rendimiento del negocio. Utilizan herramientas de análisis intuitivo, machine learning, etc. Entre las características:

- Recolecta e indexa logs y datos de ordenadores de cualquier fuente.
- Búsqueda potente, análisis y visualización para cualquier tipo de usuario.
- Aplicaciones para seguridad, análisis del negocio, IT ops.
- Proporciona escalabilidad, disponibilidad y seguridad para cualquier negocio.
- Modelo tanto cloud como software.
- Versión de prueba gratuita, pero solución para empresas de pago.

15. Apache Solr [36]:

Es una de las mejores soluciones y de código abierto. Está construido sobre Apache Lucene. Es altamente fiable, escalable y tolerante a fallos, proporcionando indexaciones distribuidas, replicaciones y consulta de carga equilibrada, recuperación, etc. Entre sus características:

- Posee API REST.
- Posibilidad de indexación a través de JSON, XML, CSV o archivos binarios a través de HTTP.
- Consultas con HTTP GET y se reciben los mismos tipos.
- Capacidades avanzadas de búsqueda full-text: frases, comodines, joins, agrupamiento, etc
- Optimizado para el alto tráfico de volumen.
- Interfaces de administración comprensivas.
- Monitorización sencilla.
- Altamente escalable y tolerante a fallos.
- Flexible y adaptable con configuración sencilla.
- Indexación casi en tiempo real.

16. Apache Lucene [2]:

Es la librería por excelencia de la Recuperación de Información. Los mejores buscadores de código abierto lo usan por debajo. Esta biblioteca proporciona un motor de búsqueda de texto completamente escrito en Java. Entre las características que proporciona:

- Escalable, indexación de alto rendimiento: más de 150 GB por hora en hardware moderno. No necesita mucha RAM.
- Algoritmos de búsqueda potentes: Búsqueda por ranking, numerosos tipos de consultas (por frases, comodín, rangos, etc), búsqueda por campos, ordenaciones, actualización simultánea, facetting, sugerencias eficaces y rápidas, modelos de clasificación, etc.
- Aunque está escrito en Java, está disponible para utilizarla en otros lenguajes compatibles (C++, .NET, C, PHP, Perl, Python, Lisp..).

17. Elasticsearch [1]:

Al igual que Solr, está construido sobre Lucene. Proporciona una capa de abstracción para que se haga más sencillo el utilizarlo. Explicaremos más adelante las características del mismo ya que ha sido el elegido y veremos qué ventajas nos ofrece. Es de código abierto, aunque algunos módulos que añaden funcionalidades son de pago.

3.3 Introducción a ElasticSearch

¿Por qué ElasticSearch? En primer lugar, fue la solución que propusieron implementar para el problema del Mando de Adiestramiento y Doctrina.

Así, con más de 750.000 documentos en sus servidores, había que crear una nueva solución. Ésta debía ser escalable, robusta, segura, resistente a fallos y rápida en cuanto a la recuperación de los documentos. Dado el gran auge de ElasticSearch se propuso el mismo para ello ya que cumplía todos estos requisitos. Se empezó así a investigar sobre el mismo y a migrar poco a poco de una BBDD relacional al nuevo sistema de ES, y más tarde perfeccionando las consultas del mismo conforme al gusto o preferencias de las personas allí presentes.

Pero además de ello, tenemos una serie de ventajas con respecto a otras soluciones, que hicieron que nos decidieramos por esta solución. En primer lugar, como hemos visto de manera resumida anteriormente, es **libre**, por lo tanto, es **gratuito**, una de las primeras condiciones que debía tener la solución que estábamos buscando.

En segundo lugar, no depende del exterior, es decir, nadie del exterior de la empresa/intranet gestiona o administra ningún elemento de nuestra implementación (al contrario que, por ejemplo, Google Search Engine). Otro punto a favor es la capa de abstracción que posee: aunque utiliza Lucene debajo, la capa facilita el uso para el usuario proporcionando un mejor entendimiento de las operaciones y una mayor facilidad para aprender a utilizarlo.

También la documentación disponible es crucial: un sistema de búsqueda muy bueno que no posea documentación en la web, no es muy aconsejable. Por lo contrario, ElasticSearch ha ido creciendo en cuanto a documentación y su comunidad ha ido creciendo a la misma vez. A continuación, podemos ver un gráfico [37] (Figura 2), con los motores descritos anteriormente y su puntuación (que describiremos a continuación):

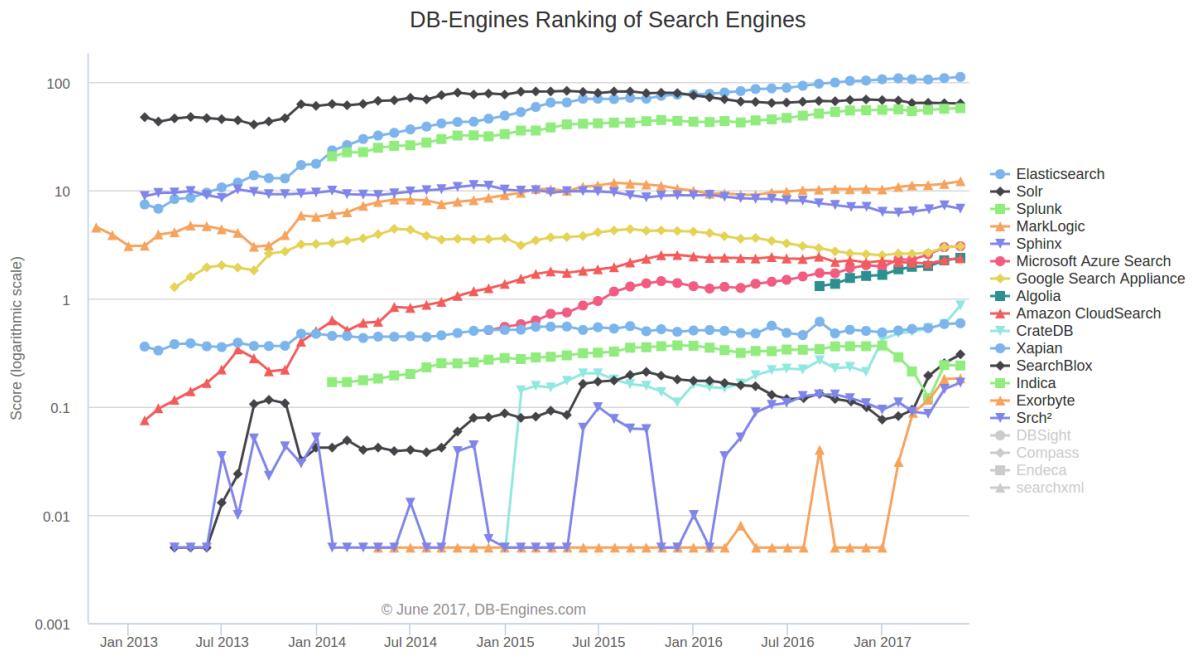


Figura 2 - Gráfico del ranking de Buscadores

Vemos que Elasticsearch se encuentra en primera posición, seguido de Solr y Splink. En la siguiente tabla [38] (Figura 3), vemos que lleva ya tiempo en la primera posición:

17 systems in ranking, June 2017									
Rank				DBMS	Database Model	Score			Jun 2017
	Jun 2017	May 2017	Jun 2016			Jun 2017	May 2017	Jun 2016	
1.	1.	1.	Elasticsearch	Elasticsearch	Search engine	111.56	+2.74	+24.14	Jun 2017
2.	2.	2.	Solr	Solr	Search engine	63.61	-0.16	-0.46	May 2017
3.	3.	3.	Splunk	Splunk	Search engine	57.52	+0.82	+12.29	Jun 2016
4.	4.	4.	MarkLogic	MarkLogic	Multi-model	12.13	+0.65	+2.56	
5.	5.	5.	Sphinx	Sphinx	Search engine	6.77	-0.47	-1.56	
6.	6.	↑ 8.	Microsoft Azure Search	Microsoft Azure Search	Search engine	3.05	+0.06	+1.62	
7.	7.	↓ 6.	Google Search Appliance	Google Search Appliance	Search engine	3.04	+0.06	-0.38	
8.	↑ 9.		Algolia	Algolia	Search engine	2.38	+0.13		
9.	↓ 8.	↓ 7.	Amazon CloudSearch	Amazon CloudSearch	Search engine	2.35	+0.08	-0.07	
10.	↑ 11.	↑ 11.	CrateDB	CrateDB	Multi-model	0.86	+0.28	+0.65	
11.	↓ 10.	↓ 9.	Xapian	Xapian	Search engine	0.59	+0.01	+0.03	
12.	12.	12.	SearchBlox	SearchBlox	Search engine	0.31	+0.05	+0.18	
13.	13.	↓ 10.	Indica	Indica	Search engine	0.24	-0.00	-0.09	
14.	14.	↑ 15.	Exorbyte	Exorbyte	Search engine	0.18	+0.00	+0.18	
15.	15.	↓ 13.	Srch ²	Srch ²	Search engine	0.17	+0.02	+0.06	
16.	16.	↓ 14.	DBSight	DBSight	Search engine	0.07	-0.01	+0.06	
17.	17.	↓ 15.	searchxml	searchxml	Multi-model	0.00	±0.00	±0.00	

Figura 3 - Ranking de puntuación de buscadores

La puntuación de ésta gráfica la ha realizado la web <https://db-engines.com>, que es una web experta en modelos de bases de datos así como de buscadores, teniendo presente los siguientes 6 puntos a tener en cuenta [39]:

1. Número de menciones del sistema en páginas webs (número de resultados en buscadores sobre esos sistemas, en concreto, Google, Bing y Yandex).
2. Interés general del sistema: Usando la frecuencia de búsquedas de Google.
3. Frecuencia de discusiones técnicas sobre el sistema: Preguntas relacionadas en páginas muy conocidas por informáticos como Stack Overflow o DBA Stack Exchange.
4. Número de ofertas de trabajo en las que el sistema se menciona (Indeed y Simply Hired).
5. Número de perfiles en redes profesionales en las que se ha mencionado el sistema (LinkedIn y Upwork).
6. Relevancia en redes sociales: número de veces que se ha mencionado en Twitter el sistema.

Podemos ver que las medidas utilizan información de las páginas más utilizadas hoy en día y son mundialmente conocidas.

Veamos ahora otros conceptos básicos que debemos conocer para trabajar con ElasticSearch:

- **Documento**: Ya lo hemos comentado anteriormente. En ElasticSearch, los documentos se presentan en formato JSON (JavaScript Object Notation) [11].
- **Tipo** [10]: Dentro de un índice se pueden definir diferentes tipos. Es una categoría que se puede diferenciar de otra categoría. Por ejemplo, si tuviésemos creado un índice llamado “documentos”, dentro del mismo podríamos tener los tipos “texto”, “hipertexto”, “audio”, “vídeo” e “imagen”. Algo igual a estos tipos utilizaremos en este trabajo.
- **Nodo** [10]: Es un servidor que forma parte de un clúster, almacena los datos y colabora en las capacidades de indexación y búsqueda del clúster.
- **Clúster** [10]: Está formado por uno o más nodos (servidores) que almacenan los datos y proporcionan la capacidad de indexar y buscar a través de los nodos.
- **Shards** [10]: Cuando un índice crece y alcanza un tamaño muy grande, ES permite dividir ese índice en partes, llamadas “shards”. Cada shard funciona por sí solo y es totalmente funcional y puede almacenarse en cualquier nodo o clúster. Es útil por dos motivos:
 - Permite escalar o reducir horizontalmente el volumen de contenidos.
 - Permite distribuir y paralelizar operaciones a través de los shards.
- **Rélicas** [10]: Es un mecanismo para tratar los errores cuando una shard o nodo no esté disponible o se eliminen por alguna razón. Para ello, se utilizan copias de los índices, llamadas “rélicas”. Es importante utilizarlas, ya que:
 - Nos proporciona alta disponibilidad.

- Nos permite escalar el volumen en paralelo ya que las búsquedas pueden ejecutarse en cualquier réplica.
- > **Mapping** [40]: Es la especificación de cómo vamos a indexar un documento y los campos del mismo. Los tipos de datos que tenemos pueden ser tipos simples como texto, keyword, fecha, long, double o boolean.

Otra ventaja de ElasticSearch es que se puede complementar con otras herramientas del grupo Elastic [41]. Algunas de ellas son:

- > **Kibana** [42]: Permite visualizar y explorar los datos de ElasticSearch, así como realizar cualquier operación en el cliente. Permite realizar interacciones interactivas, realizando histogramas, gráficos de líneas, gráficos circulares, utilizar datos Geográficos, series temporales, analizar las relaciones con un grafo, explorar anomalías con Machine Learning, etc.
- > **Beats** [43]: plataforma para transportar datos. Se instalan agentes ligeros y envías datos desde cientos de miles de máquinas a Logstash o Elasticsearch. Es ideal para la recolección de datos. Se implantan en los servidores y centralizan los datos en Elasticsearch.
- > **Logstash** [44]: Centraliza, transforma y guarda los datos. Es un canal de procesamiento de datos en la parte del servidor, código abierto, que recibe los datos de múltiples fuentes al mismo tiempo, lo transforma y lo envía al lugar que desea el usuario.
- > **X-pack** [45]: Es una extensión que proporciona seguridad, alertas, monitorización, reportes y gráficos.

Otra razón clave por la que nos decantamos por esta solución, es porque muchas de las mayores empresas mundiales lo usan, lo que equivale a decir que era una opción muy buena. Entre las empresas que utilizan ElasticSearch [46], están: DELL, Ebay, Facebook, Netflix, Slack, BBC, Cisco, Barclays, Nvidia, Uber, Microsoft, Mozilla, New York Times, IBM, Orange, Docker, Github, Soundcloud, Wikimedia, Deezer, etc...

4. Solución anterior

La solución anterior para recuperar los documentos en el MADOC como hemos comentado, era bastante completa, pero con la cantidad creciente de documentos dentro de la red Intranet, empezaba a tener algunas limitaciones y los tiempos de respuesta empezaban a ser un poco grandes.

En el modelo inicial que tenían implementado, utilizaron índices invertidos y una BBDD relacional MySQL [47], donde se almacenaban todos los campos de los archivos, y en los que se hacía uso de índices de tipos FULLTEXT para realizar la búsqueda posteriormente. Las puntuaciones eran modificadas en función de los tipos de archivos, servidores a los que pertenecían, si se encontraba la palabra en el texto, la distancia de Levenshtein [48], y otros

parámetros más, referentes a localizaciones dentro de la intranet y a los metadatos del archivo en sí.

También hacía uso de búsquedas en modo booleano [49], de manera que los usuarios podían especificar otras opciones potentes en la consulta, permitiendo el uso de las siguientes características:

- El operador +: indica que la palabra debe de estar presente en el texto.
- El operador -: Indica que la palabra no debe de estar presente en el texto.
- Operadores < y >: Incrementa o decrementa la importancia de una palabra para la puntuación total (score).
- Los operadores (y): Permite agregar palabras en subexpresiones. Pueden aplicarse a subexpresiones o a una palabra en concreto.
- Operador ~: Nos permite que la palabra que lleva antes este operador no se tenga en cuenta en el cálculo de la puntuación final. Se utiliza para disminuir ruidos.
- Operador *: Es el carácter comodín, encuentra cualquier cosa que empiece con la palabra.
- Operador “”: Todo lo que haya en el interior se busca exactamente (solamente las palabras y no los signos de puntuación)

Algunos problemas que provocaba esta solución son los siguientes:

- No se podía realizar inserciones concurrentes debido a que se llevaba un contador que se obtenía del nº de documentos insertados en la BBDD, y por lo tanto en ocasiones, al consultar dos id al mismo tiempo se generaba un fallo con la base de datos. Aunque se puede hacer, creando una “región crítica”, ya que mientras se ejecutan consultas u operaciones en la tabla de contadores, se bloquea dicha tabla, y se completan las transacciones. Hecho esto, entonces la tabla quedará liberada para ser operada/consultada. Esto produce mucha lentitud a la hora de indexar. Otra solución es confiar la asignación de los ID al SGBD, p.e. Oracle 11g explota el comando SECUENCE.
- Escalabilidad (muy importante).
- Robustez en caso de caída de un nodo (muy importante).

5. Análisis y planificación

5.1 Requisitos

Este capítulo presenta el formato de Especificación de Requisitos Software (ERS) para un motor de búsqueda de contenidos web cuya finalidad es su explotación en una red Intranet. Esta especificación se ha estructurado inspirándose en las directrices de la última versión del estándar ANSI/IEEE 830-1998. Según IEEE, un buen documento de requisitos, pese a no ser obligatorio que siga estrictamente la organización y el formato determinado por el estándar 830, sí incluirá, de una forma o de otra, la información presentada en dicho estándar.

5.1.1 Interfaces Externas

Nombre	RIE1 - Interfaces elementales
Fuente	ANALISTA
Descripción	El sistema dispondrá de interfaces fáciles e intuitivas.
Prioridad	MEDIA/DESEADO

Nombre	RIE2 - Simplificar interacción humana
Fuente	ANALISTA
Descripción	El sistema deberá construirse para que un máximo de 5 clicks del usuario sean suficientes para llegar a un punto de información buscada.
Prioridad	MEDIA/DESEADO

Nombre	RIE3 - Interfaces en Castellano
Fuente	ANALISTA
Descripción	El sistema tendrá todas sus interfaces en español.
Prioridad	ALTA/NECESARIO

5.1.2 Requisitos funcionales

Nombre	RF1 - Datos Iniciales
Fuente	ANALISTA
Descripción	El sistema permitirá al administrador gestionar los denominados datos iniciales, sobre los que se establecerán los mecanismos y procesos de indexación y actualización (RF2 y RF3 respectivamente). Los datos iniciales son: 1. Una lista de servidores web activos en la red Intranet. (alta/necesario). 2. Una lista de webs vinculantes sobre la que partir en la búsqueda de contenidos web (RF5), estas son de tipo hipertexto y son públicas en un servidor web del punto 1. (alta/necesario). 3. Una lista de directorios virtuales sobre los que buscar mediante el rastreo de directorios web (RF6). Si un servidor web no dispone de al menos un directorio virtual, el sistema podrá rastrearlo desde la raíz suponiendo que tenga habilitada la navegación de directorio contenido (baja/opcional).
Prioridad	ALTA/NECESARIO

Nombre	RF2 - Indexación de los documentos
Fuente	ANALISTA
Descripción	El sistema incorporará un mecanismo o proceso de indexación, consistente en: 1. Rastrear los contenidos web (según RF5) e incorporarlos al

Descripción	índice de ElasticSearch (según RF4) en caso de: <ul style="list-style-type: none"> • Localizarse en un servidor web registrado en datos iniciales (RF1). • No encontrarse previamente indexado en el índice de ElasticSearch (evitar duplicidades).
Prioridad	ALTA/NECESARIO

Nombre	RF3 - Actualización de los documentos
Fuente	ANALISTA
Descripción	<p>El sistema incorporará un mecanismo de actualización que considera los datos iniciales (RF1) y el resto de datos ya establecidos en el índice. El proceso consiste en:</p> <ol style="list-style-type: none"> 1. Rastrear contenidos web (según RF5), encontramos dos posibles casos: <ol style="list-style-type: none"> 1.1 Si el documento no está indexado en el índice, se procederá a su indexación (según RF2) en caso de: <ul style="list-style-type: none"> • Localizarse en un servidor web registrado en datos iniciales (RF1). 1.2 Si el documento está indexado en el índice, se procederá a su actualización (según RF3) en caso de: <ul style="list-style-type: none"> • Haber cambiado la fecha de modificación del documento (caso de disponer de esta fecha antes y durante el rastreo). 2. Si Finalizado el proceso de rastreo, algún documento que anteriormente estuviese indexado ya no existe, se modifica el campo estado de actividad a 0, ya que el archivo ya no puede ser accedido, y por lo tanto, no tiene sentido que ofrezcamos a los usuarios el mismo como resultado de su búsqueda en el sistema.
Prioridad	ALTA/NECESARIO

Nombre	RF4 – Incorporación/Actualización de contenidos Web
Fuente	ANALISTA
	Siempre que el sistema indexe o modifique documentos en el índice, como consecuencia de un proceso de carga (RF2 – solo indexaciones) o de actualización (RF3 – indexaciones y

Descripción	actualizaciones), el sistema registrará para cada documento los siguientes datos, manteniéndose intactos en caso de actualización (RF3) aquellos datos que no hayan sido modificados. 1. URL del documento (alta/necesario). 2. Servidor en el que se localiza (alta/necesario). 3. Formato o extensión (alta/necesario). 4. Nombre de archivo. 5. Autor del archivo. 6. Fecha de creación del archivo. 7. Fecha de modificación del archivo. 8. Tamaño del archivo (en kb). 9. Estado de actividad (alta/necesario).
Prioridad	ALTA/NECESARIO

Nombre	RF5 – Rastreo de contenidos web mediante directorios web
Fuente	ANALISTA
Descripción	El sistema rastreará los contenidos web mediante la navegación de directorios web habilitada en cada servidor web sobre el que se efectúe el rastreo, si algún servidor tuviera esta propiedad desactivada será omitido. Aquellos que lo tengan activo realizarán un rastreo recursivo que funcionará así: Si el servidor web no tiene directorios virtuales cargados, entonces el sistema buscará la raíz y rastreará. En este proceso, el sistema no considera las webs vinculantes y deberá evitar duplicidades en los contenidos web registrados. El sistema será capaz de construir la URL de cada elemento o contenido web recorrido mediante una sintaxis del tipo: http:// ("IP" o "alias" del servidor) /(ruta virtual de destino)
Prioridad	BAJA/OPCIONAL

Nombre	RF6 - Interfaz Web de Búsqueda
Fuente	ANALISTA
	El sistema permitirá al usuario consultar fácilmente los contenidos del índice mediante una interfaz web de conformidad con los criterios y términos de búsqueda especificados por el

Descripción	<p>usuario. Esta interfaz web debe mostrar dos tipos de búsqueda:</p> <ul style="list-style-type: none"> - Por defecto, un buscador universal consistente en una única caja de texto o “inputbox” donde realizar las búsquedas universales, cuyo ámbito de búsqueda es sobre todos los datos de tipo texto de los documentos del índice. - Posibilidad de realizar filtros por departamentos una vez realizada la búsqueda universal o por defecto, para poder así afinar la búsqueda. - Un buscador para cada tipo de archivo multimedia: audio, vídeo e imagen.
Prioridad	ALTA/NECESARIO

Nombre	RF7 – Autocompletar Búsqueda
Fuente	ANALISTA
Descripción	<p>El sistema autocompletará los términos de búsqueda a medida que estos vayan siendo introducidos por un usuario dentro de la caja de texto (inputbox) de la interfaz web de búsqueda; el sistema mostrará un máximo de 6 documentos según los criterios de ordenación establecidos. El sistema podría presentar entre otras estrategias, una línea en la interfaz por cada uno de estos contenidos web, donde aparecería el nombre del archivo y su formato.</p>
Prioridad	MEDIA/DESEADO

Nombre	RF8 – Registro de HITS
Fuente	ANALISTA
Descripción	<p>El sistema registrará los hits (clicks en los resultados de las distintas búsquedas).</p>
Prioridad	MEDIA/DESEADO

Nombre	RF9 – Filtro u Ordenación de resultados de búsqueda
Fuente	ANALISTA
Descripción	<p>El sistema podrá filtrar u ordenar los contenidos web que satisfagan una búsqueda según los siguientes criterios:</p> <ul style="list-style-type: none"> - Por número de visitas.

	<ul style="list-style-type: none"> - Por formato de fichero. - Por colección de formatos de fichero (texto, hipertexto, imagen, audio, video). - Por fecha de creación. - Por fecha de modificación. - Por autor. - Por servidor. - Por tamaño de fichero. - Por nombre de fichero. - Por título de fichero. - Por texto contenido en fichero (sólo para texto e hipertexto).
Prioridad	MEDIA/DESEADO

Nombre	RF10 – Ordenación por defecto de resultados de búsqueda
Fuente	ANALISTA
Descripción	<p>En la búsqueda universal “por defecto” el sistema no establecerá filtros y empleará las siguientes instancias de criterios de ordenación para los contenidos web que son resultado de una determinada búsqueda:</p> <ol style="list-style-type: none"> 1. Por coincidencia de nombre de archivo. 2. Por el TextoContenido contenido 3. Por número de visitas o hits. 4. Por fecha de modificación (de más reciente a más antiguo).
Prioridad	MEDIA/DESEADO

Nombre	RF11 – Presentación web de resultados de búsqueda
Fuente	ANALISTA
Descripción	<p>Los documentos resultantes de una determinada búsqueda se presentarán en una interfaz web “por bloques”, al igual que la mayoría de motores de búsqueda conocidos a cada bloque le corresponderá un contenido web. Cada bloque estará compuesto de hasta un total de cuatro líneas de texto:</p> <p>LÍNEA 1: El texto estará en su totalidad hipervinculado al contenido web:</p> <ul style="list-style-type: none"> - Si el contenido web es hipertexto, contendrá el título del contenido web. - En otro caso contendrá el formato del fichero entre corchetes seguido del nombre del fichero. <p>LÍNEA 2: Contendrá la URL que apunta al contenido.</p> <p>LÍNEAS 3 y 4: Contendrán información del documento:</p> <ul style="list-style-type: none"> - Si el fichero es de hipertexto, contendrá el primer texto contenido “filtrado” (Sin etiquetas u otros elementos). - Si el fichero es de texto, contendrá el primer texto contenido. - En otro caso, se omitirán estas líneas.
Prioridad	MEDIA/DESEADO

5.1.3 Requisitos de rendimiento

Nombre	RR1 – Tiempo de respuesta
Fuente	ANALISTA

Descripción	El sistema deberá tener un tiempo máximo de respuesta de 5 segundos para cualquier operación de consulta.
Prioridad	MEDIA/DESEADO

5.1.4 Requisitos tecnológicos

Nombre	RT1 – Navegador Web
Fuente	ANALISTA
Descripción	El sistema interaccionará con los usuarios sobre un navegador web, será compatible con los navegadores Internet Explorer, Mozilla Firefox y Google Chrome..
Prioridad	ALTA/NECESARIO

5.1.5 Requisitos de Fiabilidad y Disponibilidad

Nombre	RFD1 - Máxima disponibilidad
Fuente	ANALISTA
Descripción	El sistema debe estar disponible y en servicio en cualquier momento
Prioridad	ALTA/NECESARIO

Nombre	RFD2 - Recuperación del servicio
Fuente	ANALISTA
Descripción	En caso de fallo o error del sistema, este deberá recuperarse en un periodo máximo de 24 horas desde el momento de la detección o el conocimiento del fallo o error.
Prioridad	ALTA/NECESARIO

5.1.6 Requisitos de Implementación

Nombre	RIM1 - Fácil Mantenimiento
Fuente	ANALISTA
Descripción	El sistema debe mantenerse fácilmente.
Prioridad	ALTA/NECESARIO

Nombre	RIM2 - Infraestructura de red
Fuente	ANALISTA
Descripción	El sistema se implementará sobre la infraestructura de la red intranet existente.
Prioridad	ALTA/NECESARIO

5.1.7 Otros requisitos

Requisitos económicos

Nombre	RE1 - Impacto económico
Fuente	ORGANIZACIÓN
Descripción	Mínimo impacto económico en su adquisición y mantenimiento.
Prioridad	ALTA/NECESARIO

Requisitos de accesibilidad

Nombre	RAC1 - Acceso concurrente
Fuente	ORGANIZACIÓN
Descripción	El sistema debe soportar acceso concurrente.
Prioridad	ALTA/NECESARIO

5.2 Diagramas de casos de uso

5.2.1 Especificación de actores

Como se ha comentado anteriormente, podemos distinguir dos roles que interactúan con el sistema. A continuación, se presentan detalladamente cada uno de estos actores:

Actor	Administrador	ACT-01
Descripción	Será el encargado de realizar todas las operaciones que no son de búsqueda, estas son: <ul style="list-style-type: none"> • Parametrización del buscador e índices. • Gestión de las indexaciones de documentos. • Gestión de las actualizaciones de documentos. Gestión de los borrados de documentos.	

Tabla 2 - Actor Administrador

Actor	Usuario final	ACT-02
Descripción	Es todo aquel usuario cuyo propósito es encontrar información o documento dentro de una red Intranet de una empresa. Las funciones del mismo serán tanto la búsqueda como la descarga de los documentos que le interesen.	

Tabla 3 - Actor Usuario final

5.2.2 Casos de Uso

Caso de Uso	CU - Indexar Documentos	CU-01
Actores	Administrador	
Tipo	Primario	

Referencias								
Precondición	El usuario debe ser administrador y estar en la pestaña de configuración.							
Postcondición	Los documentos han sido indexados correctamente en el Índice de ElasticSearch.							
Autor	César Albusac Jorge	Fecha	28/06/2017	Versión	0.1			
Propósito	Indexar documentos en el sistema creado para que puedan ser buscados posteriormente.							
Resumen	El administrador tendrá la posibilidad de indexar documentos en el Índice creado en ElasticSearch. La indexación se realizará desde la pestaña de configuración y habrá 5 tipos diferentes de indexaciones:							
	<ol style="list-style-type: none"> Indexación general: Se indexarán todos aquellos documentos que se encuentren en los diferentes servidores de la Intranet. Indexación por servidor: Se podrá especificar de qué servidor de dentro de la empresa se desea indexar documentos. Indexación por departamento: Se podrá especificar por qué aplicación o departamento se desea indexar documentos. Indexación por tipo de archivo: Se podrá especificar qué tipo de archivo se desea indexar en el índice: hipertexto, texto, imagen, audio o vídeo. 							
	Indexación por url prioritaria: El administrador podrá especificar una url de un documento para que tal documento sea indexado en el Índice.							
Curso Normal (Básico)								
1	Administrador: Accede a la dirección web donde se encuentra el sistema de recuperación de información de la Intranet.	2	El sistema comprueba que la IP corresponde a la de un usuario con privilegios, es decir, a un administrador.					
3	Administrador: Tendrá acceso a la pestaña de "Configuración" y pinchará en la misma.	4	El sistema devuelve la página de administración del SRI.					
5	Administrador: Tendrá tres opciones en dicha página: Indexar, Actualizar y Borrar. Pinchará en Indexar y verá las 5 opciones de indexación comentadas anteriormente. Finalmente, pinchará una y en caso de ser necesario introducirá los parámetros necesarios.	6	El sistema recibe la petición de indexar documentos con los parámetros que sean correspondientes y el tipo de indexado que se desea. Comienza a indexar y una vez terminado devuelve la página con un mensaje de éxito o de error, dependiendo de si se realizó correctamente o algo falló.					
7	Administrador: Ve en pantalla si se ha realizado correctamente o no.							
Cursos Alternos								
6a	El sistema generará el mensaje de error correspondiente en caso de que se produzca algún inconveniente.							
Otros datos								
Frecuencia esperada	Depende de la cantidad de documentos que posea la empresa/Intranet. Una frecuencia esperada podría ser 1 vez por semana.	Rendimiento	Alto					

Importancia	Vital	Urgencia	Alta
Estado	Realizado	Estabilidad	Alta

Tabla 4 - Caso de Uso Indexar Documentos

Caso de Uso	CU – Actualizar Documentos			CU-02			
Actores	Administrador						
Tipo	Primario						
Referencias							
Precondición	El usuario debe ser administrador y estar en la pestaña de configuración. Para que se actualice un documento debe haber cambiado algún campo del mismo.						
Postcondición	Los documentos han sido actualizados correctamente en el índice de ElasticSearch.						
Autor	César Albusac Jorge	Fecha	28/06/2017	Versión	0.1		
Propósito	Actualizar aquellos documentos que ya existen en el índice modificando aquellos campos que hayan cambiado.						
Resumen	El administrador tendrá la posibilidad de actualizar los documentos que ya existen en el índice de ElasticSearch. La actualización se realizará desde la pestaña de configuración y habrá 5 tipos diferentes de actualizaciones:						
	<ol style="list-style-type: none"> Actualización general: Se actualizarán todos aquellos documentos que hayan cambiado y que se encuentren en los diferentes servidores de la Intranet y hayan sido indexados anteriormente en el Índice. Actualización por servidor: Se podrá especificar de qué servidor de dentro de la empresa se desea actualizar los documentos indexados. Actualización por departamento: Se podrá especificar por qué aplicación o departamento se desea actualizar documentos. Actualización por tipo de archivo: Se podrá especificar qué tipo de archivo se desea actualizar en el índice: hipertexto, texto, imagen, audio o vídeo. 						
	<u>Actualización por url prioritaria:</u> El administrador podrá especificar una url de un documento que quiere actualizar (debe estar indexado en el índice para que se actualice).						
Curso Normal (Básico)							
1	Administrador: Accede a la dirección web donde se encuentran el sistema de recuperación de información de la Intranet.	2	El sistema comprueba que la IP corresponde a la de un usuario con privilegios, es decir, a un administrador.				
3	Administrador: Tendrá acceso a la pestaña de "Configuración" y pinchará en la misma.	4	El sistema devuelve la página de administración del SRI.				
5	Administrador: Tendrá tres opciones en dicha página: Indexar, Actualizar y Borrar. Pinchará en Actualizar y verá las 5 opciones de indexación comentadas anteriormente. Finalmente, pinchará una y en caso de ser necesario introducirá los parámetros necesarios.	6	El sistema recibe la petición de actualizar los documentos con los parámetros que sean correspondientes y el tipo de indexado que se desea. Comienza a comprobar si los archivos han cambiado y a actualizarlos en caso de que así sea. Una vez terminado devuelve la página con un mensaje de éxito o de error, dependiendo de si se realizó correctamente o algo				

			falló.
7	Administrador: Ve en pantalla si se ha realizado correctamente o no.		
Cursos Alternos			
6a	El sistema generará el mensaje de error correspondiente en caso de que se produzca algún inconveniente.		
Otros datos			
Frecuencia esperada		Depende de la frecuencia con que se modifiquen los que posea la empresa/Intranet. Una frecuencia esperada podría ser 1 vez por semana.	Rendimiento
Importancia		Vital	Urgencia
Estado		Realizado	Estabilidad
Alta			

Tabla 5 - Caso de Uso Actualizar Documentos

Caso de Uso		CU – Borrar Documentos			CU-03
Actores		Administrador			
Tipo		Primario			
Referencias					
Precondición		El usuario debe ser administrador y estar en la pestaña de configuración.			
Postcondición		Los documentos han sido borrados correctamente del Índice de ElasticSearch.			
Autor		César Albusac Jorge	Fecha	28/06/2017	Versión
					0.1
Propósito					
Borrar los documentos que estén indexados en el índice.					
Resumen					
El administrador tendrá la posibilidad de borrar los documentos siempre y cuando estén ya indexados en el Índice de ElasticSearch. El borrado se realizará desde la pestaña de configuración y habrá 5 tipos diferentes de borrado:					
<ol style="list-style-type: none"> <u>Borrado general</u>: Se borrarán todos los documentos que existan en el índice de ElasticSearch. <u>Borrado por servidor</u>: Se podrá especificar de qué servidor se desean borrar todos los documentos del Índice. <u>Borrado por departamento</u>: Se podrá especificar de qué aplicación o departamento se desea borrar los documentos. <u>Borrado por tipo de archivo</u>: Se podrá especificar qué tipo de archivo se desea borrar del índice: hipertexto, texto, imagen, audio o vídeo. 					
<u>Borrado por url prioritaria</u> : El administrador podrá especificar una url de un documento que quiere borrar (debe estar indexado en el Índice para que se elimine).					
Curso Normal (Básico)					
1	Administrador: Accede a la dirección web donde se encuentra el sistema de recuperación de información de la Intranet.	2	El sistema comprueba que la IP corresponde a la de un usuario con privilegios, es decir, a un administrador.		
3	Administrador: Tendrá acceso a la pestaña de "Configuración" y pinchará en la misma.	4	El sistema devuelve la página de administración del SRI.		

5	Administrador: Tendrá tres opciones en dicha página: Indexar, Actualizar y Borrar. Pinchará en Borrar y verá las 5 opciones de borrado comentadas anteriormente. Finalmente, pinchará una y en caso de ser necesario introducirá los parámetros necesarios.	6	El sistema recibe la petición de borrar los documentos con los parámetros que sean correspondientes y el tipo de borrado que se desea. Comienza a borrar los documentos del índice. Una vez terminado devuelve la página con un mensaje de éxito o de error, dependiendo de si se realizó correctamente o algo falló.
7	Administrador: Ve en pantalla si se ha realizado correctamente o no.		
Cursos Alternos			
6a	El sistema generará el mensaje de error correspondiente en caso de que se produzca algún inconveniente.		
Otros datos			
Frecuencia esperada	Baja.	Rendimiento	Alto
Importancia	Vital	Urgencia	Alta
Estado	Realizado	Estabilidad	Alta

Tabla 6 - Caso de Uso Borrar Documentos

Caso de Uso		CU – Búsqueda General		CU-04	
Actores		Administrador, Usuario			
Tipo		Primario			
Referencias					
Precondición		Debe acceder a la página principal del SRI desde su navegador web..			
Postcondición		Se ha realizado la búsqueda satisfactoriamente.			
Autor		César Albusac Jorge	Fecha	28/06/2017	Versión 0.1
Propósito					
Realizar una búsqueda general de documentos pertenecientes al servidor Intranet de la empresa.					
Resumen					
El usuario que desea realizar una búsqueda (general, en este caso), accede a la página principal del SRI e introducirá el texto que desea buscar en los documentos del índice. La búsqueda se hace sobre los siguientes campos y tipos de documentos:					
<ul style="list-style-type: none"> Sobre los archivos de hipertexto: Se buscan coincidencias para el título del hipertexto como para el texto contenido dentro del mismo. Sobre los archivos de texto: Se buscan coincidencias para el nombre del archivo o para el texto contenido en el mismo. 					
Sobre los archivos de imagen: Se busca sobre el nombre del archivo, la ruta de la misma, y las etiquetas de la imagen.					
Curso Normal (Básico)					
1	Usuario/Administrador: Accede a la página principal del SRI de la empresa.				
2	Usuario/Administrador: Introduce el texto como palabras claves que desea buscar entre todos los	3	Es sistema realiza una búsqueda de esas palabras en aquellos archivos que sean de tipo texto, hipertexto o		

	documentos.		imagen, como hemos indicado anteriormente.
		4	Genera una lista de los archivos encontrados ordenados por la puntuación interna (aunque modificada para ajustarlo a las necesidades de la empresa).
6	El usuario es redirigido a la página con los resultados en caso de que los haya. En las imágenes encontradas, puede hacer click y se le abrirá la misma. En cuanto a la lista de resultados de texto e hipertexto, se visualiza el tipo de archivo que es, el departamento del mismo si lo tiene, el nombre del archivo y los trozos de texto por el cual se ha seleccionado ese documento como válido para la consulta. Si hace clic en cualquiera de ellos, se le abrirá o descargará el mismo.	5	Se devuelve una página con los resultados encontrados (si los hay). Las imágenes salen en primer lugar y después los archivos de texto e hipertexto aparte.
Cursos Alternos			
5a	El sistema generará el mensaje de error correspondiente en caso de que se produzca algún inconveniente		
Otros datos			
Frecuencia esperada	Alta	Rendimiento	Alto
Importancia	Vital	Urgencia	Alta
Estado	Realizado	Estabilidad	Alta

Tabla 7 - Caso de Uso Búsqueda General

Caso de Uso		CU - Búsqueda Imágenes			CU-05	
Actores		Administrador, Usuario				
Tipo		Primario				
Referencias						
Precondición		Debe acceder a la página de búsqueda de imágenes del SRI desde su navegador web.				
Postcondición		Se ha realizado la búsqueda satisfactoriamente.				
Autor		César Albusac Jorge	Fecha	28/06/2017	Versión	0.1
Propósito						
Realizar una búsqueda de imágenes pertenecientes al servidor Intranet de la empresa.						
Resumen						
El usuario que desea realizar una búsqueda de imágenes, accede a la página principal del SRI y después en "Búsqueda de Imágenes". Introducirá el texto que desea buscar en los documentos del índice. La búsqueda se hace sobre los siguientes campos de las imágenes indexadas: el nombre del archivo, la ruta de la misma, y las etiquetas de la imagen.						
Curso Normal (Básico)						
1	Usuario/Administrador: Accede a la página principal del SRI de la					

	empresa.		
2	Usuario/Administrador: Accede a la pestaña de búsqueda de imágenes.		
3	Usuario/Administrador: Introduce el texto como palabras claves que desea buscar entre todos los documentos de tipo imagen.	4	Es sistema realiza una búsqueda de esas palabras en aquellos archivos que sean de tipo imagen.
		5	Genera una lista de los archivos encontrados ordenados por la puntuación interna.
7	El usuario es redirigido a la página con los resultados en caso de que los haya. Las imágenes encontradas pueden ser pinchadas y se abrirá la misma. También se ven la altura y anchura de la misma, las etiquetas.	6	Se devuelve una página con los resultados encontrados (si los hay).
Cursos Alternos			
6a	El sistema generará el mensaje de error correspondiente en caso de que se produzca algún inconveniente.		
Otros datos			
Frecuencia esperada	Alta	Rendimiento	Alto
Importancia	Vital	Urgencia	Alta
Estado	Realizado	Estabilidad	Alta

Tabla 8 - Caso de Uso Búsqueda Imágenes

Caso de Uso	CU – Búsqueda Audio			CU-06	
Actores	Administrador, Usuario				
Tipo	Primario				
Referencias					
Precondición	Debe acceder a la página de búsqueda de audios del SRI desde su navegador web..				
Postcondición	Se ha realizado la búsqueda satisfactoriamente.				
Autor	César Albusac Jorge	Fecha	28/06/2017	Versión	0.1
Propósito					
Realizar una búsqueda de audios pertenecientes al servidor Intranet de la empresa					
Resumen					
El usuario que desea realizar una búsqueda de audios, accede a la página principal del SRI y después en “Búsqueda de Audios”. Introducirá el texto que desea buscar en los documentos del índice. La búsqueda se hace sobre los siguientes campos de los audios indexados: el nombre del archivo, la ruta del mismo, y las etiquetas asociadas					
Curso Normal (Básico)					
1	Usuario/Administrador: Accede a la página principal del SRI de la empresa.				
2	Usuario/Administrador: Accede a la pestaña de búsqueda de audios.				
3	Usuario/Administrador: Introduce el texto como palabras claves que desea buscar entre todos los	4	Es sistema realiza una búsqueda de esas palabras en aquellos archivos que sean de tipo audio.		

	documentos de tipo audio.		
		5	Genera una lista de los archivos encontrados ordenados por la puntuación interna.
7	El usuario es redirigido a la página con los resultados en caso de que los haya. Se muestra el nombre del archivo, la duración y las etiquetas asociadas. Los audios encontrados se pueden reproducir y descargar.	6	Se devuelve una página con los resultados encontrados (si los hay).
Cursos Alternos			
6a	El sistema generará el mensaje de error correspondiente en caso de que se produzca algún inconveniente.		
Otros datos			
Frecuencia esperada	Media	Rendimiento	Medio
Importancia	Media	Urgencia	Media
Estado	Realizado	Estabilidad	Alta

Tabla 9 - Caso de Uso Búsqueda Audio

Caso de Uso		CU – Búsqueda Video		CU-07	
Actores		Administrador, Usuario			
Tipo		Primario			
Referencias					
Precondición		Debe acceder a la página de búsqueda de videos del SRI desde su navegador web..			
Postcondición		Se ha realizado la búsqueda satisfactoriamente.			
Autor		César Albusac Jorge	Fecha	28/06/2017	Versión 0.1
Propósito					
Realizar una búsqueda de videos pertenecientes al servidor Intranet de la empresa					
Resumen					
El usuario que desea realizar una búsqueda de videos, accede a la página principal del SRI y después en “Búsqueda de Videos”. Introducirá el texto que desea buscar en los documentos del índice. La búsqueda se hace sobre los siguientes campos de los videos indexados: el nombre del archivo, la ruta del mismo, y las etiquetas asociadas.					
Curso Normal (Básico)					
1	Usuario/Administrador: Accede a la página principal del SRI de la empresa.				
2	Usuario/Administrador: Accede a la pestaña de búsqueda de videos.				
3	Usuario/Administrador: Introduce el texto como palabras claves que desea buscar entre todos los documentos dentro del tipo video.	4	El sistema realiza una búsqueda de esas palabras en aquellos archivos que sean del tipo video.		
		5	Genera una lista de los archivos encontrados ordenados por la puntuación interna.		
7	El usuario es redirigido a la página con los resultados en caso de que los	6	Se devuelve una página con los resultados encontrados (si los hay).		

	haya. Se muestra el nombre de los videos, las duraciones y las etiquetas asociadas. Se pueden visualizar o descargar. Los videos encontrados pueden ser pinchados y se abrirán los mismos.		
Cursos Alternos			
6a	El sistema generará el mensaje de error correspondiente en caso de que se produzca algún inconveniente.		
Otros datos			
Frecuencia esperada	Media	Rendimiento	Medio
Importancia	Media	Urgencia	Media
Estado	Realizado	Estabilidad	Alta

Tabla 10 - Caso de Uso Búsqueda Video

5.2.3 Diagrama de Casos de Uso

Como hemos comentado anteriormente, existen dos actores que interactúan con el sistema, representando a los dos tipos de usuario existentes: administrador y el trabajador de la empresa. El administrador, será el encargado de indexar, actualizar y borrar los documentos del índice. El usuario normal, solamente podrá realizar los diferentes tipos de búsquedas. A continuación, se muestra un Diagrama de Casos de Uso, que describe visualmente las relaciones entre los dos actores y qué Casos de Uso pueden realizar cada uno de ellos:

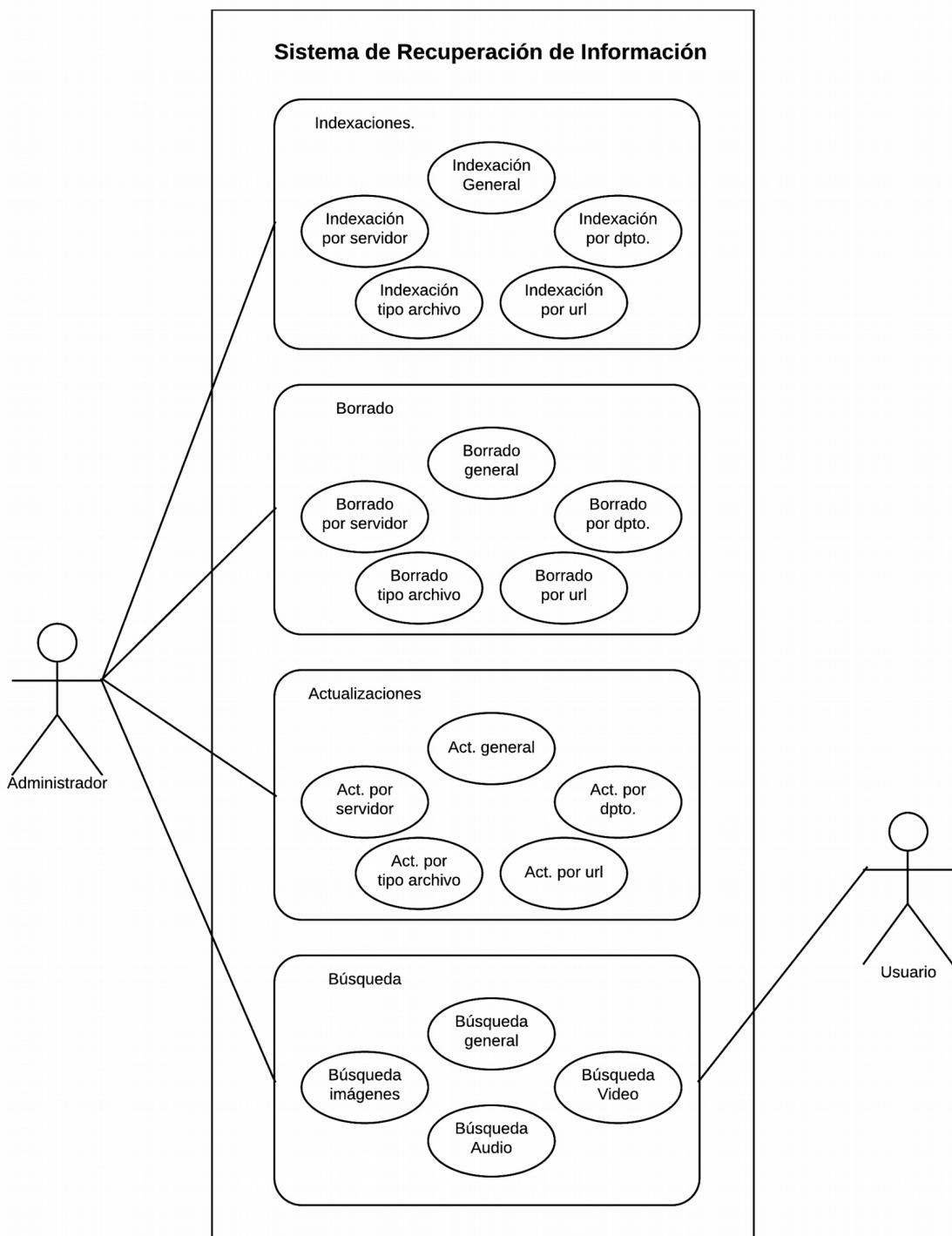


Figura 4 - Diagrama de Casos de Uso

5.3 Planificación y presupuesto

La planificación del trabajo y tareas establecidas desde el inicio de este proyecto, han sido divididas y planificadas del siguiente modo:

Establecimiento de especificaciones

- Establecimiento de requisitos (30 horas).
- Análisis (15 horas).
- Estudio de las alternativas de diseño (25 horas).

Diseño

- Diseño del modelo de datos (35 horas).
- Diseño de la aplicación Web (20 horas).

Desarrollo

- Implementación del modelo de ElasticSearch y consultas (80 horas).
- Implementación de la aplicación Web (70 horas).
- Desarrollo de algunas mejoras (20 horas).

Pruebas

- Integración del Sistema de Recuperación de Información y pruebas en la aplicación Web con usuarios reales (60 horas).

Documentación

- Redacción de la memoria (65 horas).

Reajustes a posteriori

- Correcciones y ajustes oportunos (35 horas).

Tarea	Duración
Especificaciones	70 horas
Diseño	55 horas
Desarrollo	170 horas
Pruebas	60 horas
Documentación	65 horas
Reajustes	35 horas
TOTAL	455 horas

Tabla 11 - Planificación temporal del trabajo

Su desarrollo a nivel de plan de trabajo, se realizará siguiendo el siguiente diagrama de Gantt:

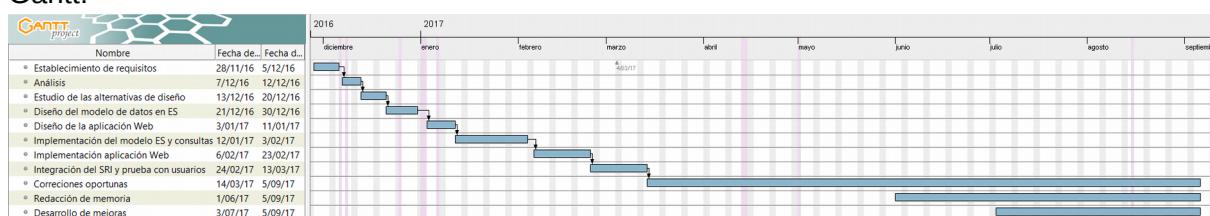


Figura 5 - Diagrama de Gantt Inicial

Como podemos ver, toda la planificación sigue un proceso secuencial, salvo la elaboración de la memoria, las correcciones y la implementación de algunas mejoras.

Estimación del presupuesto

Voy a pasar a detallar el presupuesto de este trabajo, basándome en las tareas expuestas en el apartado anterior. Se desglosa de la siguiente forma:

Costes de personal

Ocupación	Horas	Precio/Hora (€/hora)	Importe(€)
Jefe de proyecto	25	50,00	1.250,00
Ingeniero Informático	455	25,00	11.375,00
TOTAL	480	-	12.625,00

Tabla 12 - Costes de personal

Costes hardware

Concepto	Unidades	Precio por Unidad (€)	Coste para el proyecto (€)
Servidor por piezas[50]	1	1.733,94	1.733,94
PC sobremesa (i5, 8GB RAM, 2TB HDD)[51]	1	678,44	678,44
Teclado Logitech[52]	1	12,95	12,95
Ratón HP [53]	1	19,95	19,95
Monitor HP 23.8" LED IPS FULLHD [54]	1	149,00	149,00
TOTAL			2.594,28 €

Tabla 13 - Costes Hardware

Costes software

Concepto	Unidades	Precio por unidad (€)	Coste para el proyecto (€)
Windows 10 Pro[55]	1	279,00	279,00
Windows Server 2012 r2[56]	1	214,00	214,00
Visual Studio 2017, edición Community para estudiantes.	1	0	0,00
Microsoft Office 2016 edición estudiantes[57]	1	79,00	79,00
GranttProject[58]	1	0	0,00
TOTAL			572,00 €

Tabla 14 - Costes Software

COSTE TOTAL

Concepto	Precio
Costes de personal	12.625,00 €
Costes hardware	2.594,28 €
Costes software	572,00 €
TOTAL	15.791,28 €

Tabla 15 - Coste total del proyecto

6. Diseño

6.1 Diagrama de clases

A continuación, vamos a mostrar el diagrama de las clases que componen el sistema principal. Se muestran las siguientes clases:

- **Clases de documentos**: Son los tipos de documentos con los que vamos a tratar (hipertexto, texto, imagen, audio y vídeo) con sus respectivos atributos. También hay una clase que corresponde al documento que será devuelto por el buscador: **resultadoDeBusqueda.cs**.
- **Crawler.cs**: Es la clase encargada de parsear los diferentes documentos, para después hacer la operación correspondiente que se haya solicitado.
- **OperacionesElasticSearch.cs**: Aquí se define la mayoría de las operaciones importantes del proyecto. Solamente hay funciones que modifican el índice de elasticSearch. Aquí se encuentran los métodos de autocompletado, indexación, búsqueda, actualización y borrado.
- **Supercontroladora.cs**: Es la encargada de realizar la conexión con el cliente de elasticSearch y, en caso de no existir, crearlo. En la creación se especifican las características que tendrá el mismo, como veremos más adelante.

A continuación, se muestra un diagrama (Figura 6) con las principales clases del trabajo:



6.2 Arquitectura

Arquitectura

La arquitectura será centralizada, existiendo un sólo servidor para la aplicación (aunque en la práctica se ha realizado con varios servidores al mismo tiempo, uno como maestro y el otro como esclavo, teniendo varios nodos para que, en el caso de que uno cayera, el otro sirviese la petición sin ningún problema), conteniendo también el índice sobre el que se lanzarán las peticiones de los clientes. Un esquema podría ser el siguiente:

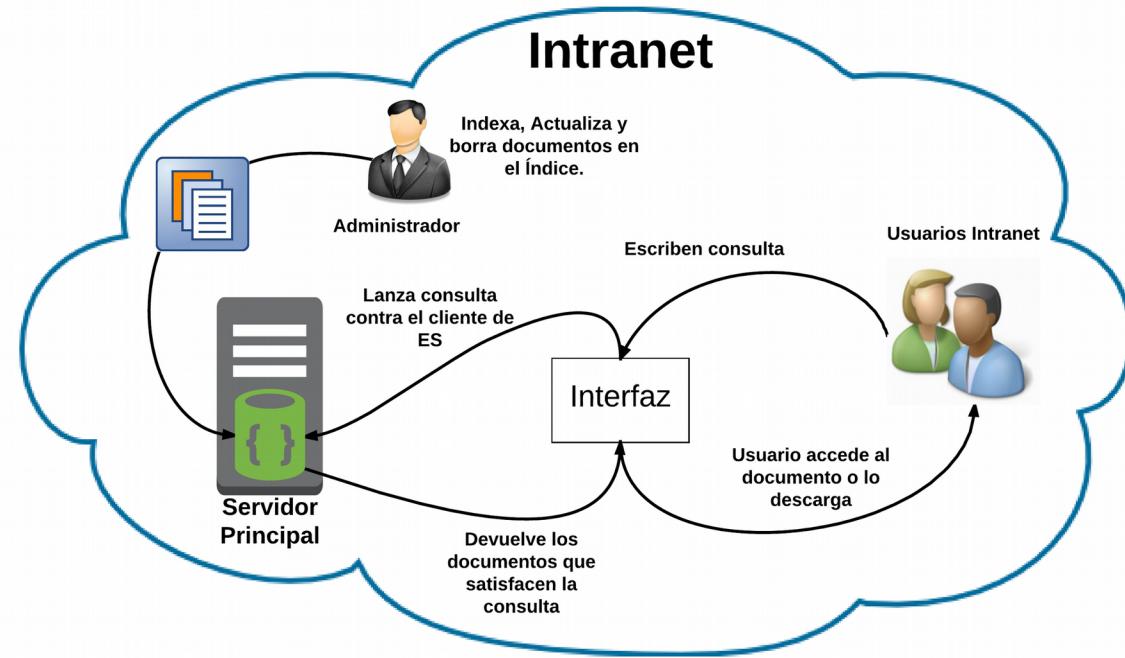


Figura 7 - Arquitectura Hardware de la aplicación

Arquitectura de la aplicación

En cuanto a la arquitectura de la aplicación web, estará formada por una serie de páginas en las que se podrán buscar diferentes tipos de archivos, y la parte de Configuración a la que podrá acceder sólo el administrador y donde se realizarán las operaciones correspondientes relacionadas con el Índice de ElasticSearch, como son las indexaciones de documentos, actualizaciones y borrado. Así, la arquitectura de la misma sería la siguiente:

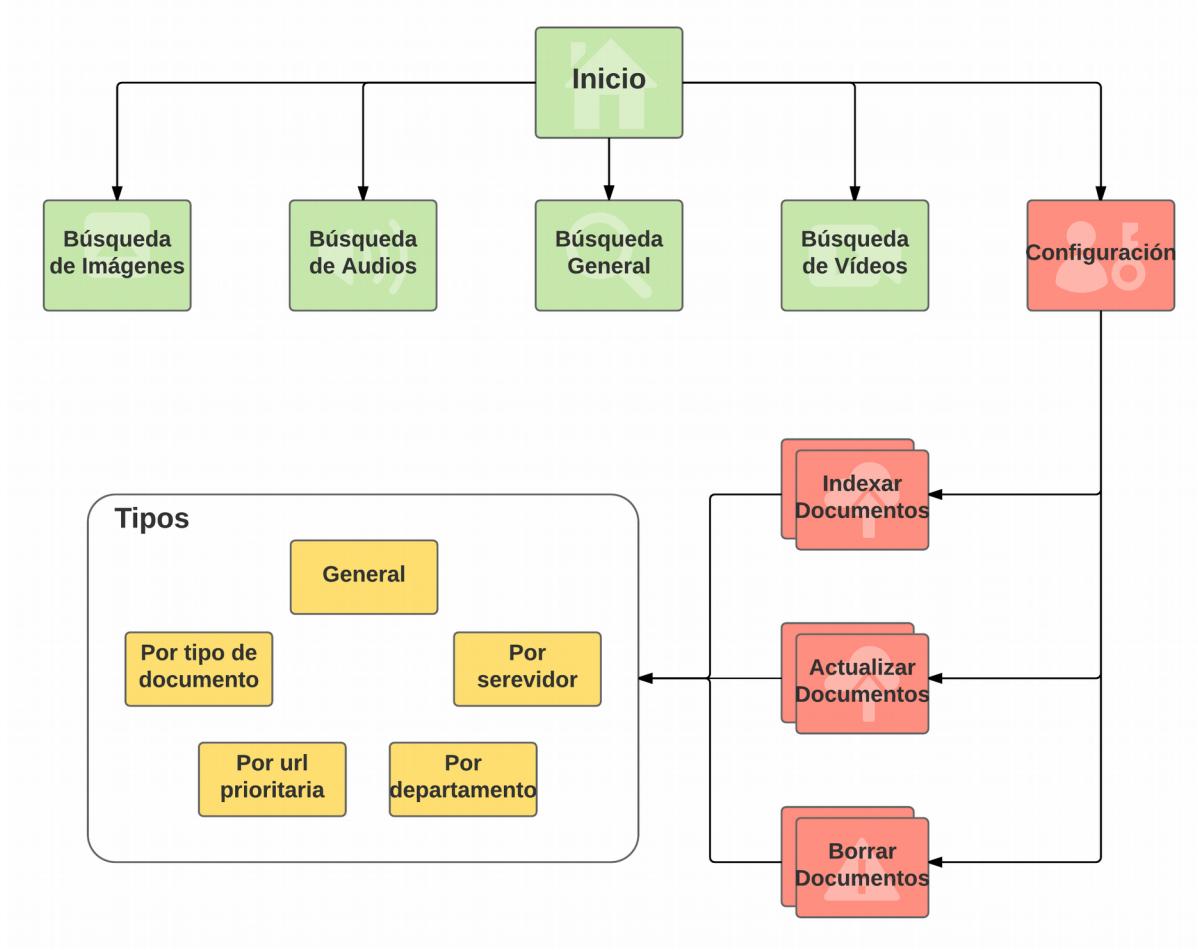


Figura 8 - Arquitectura de la aplicación web

7. Implementación y pruebas

7.1 Herramientas utilizadas

Ahora vamos a pasar a comentar las herramientas principales que se han usado. En cuanto a herramientas hardware, las herramientas necesarias para llevar a cabo este proyecto fueron un servidor, desde el que se lanzaba el sistema de Elasticsearch y servía las peticiones, y un ordenador de sobremesa desde el que se desarrolló el sistema y se realizaron las pruebas.

En cuanto al software, ya que estuve desarrollando la aplicación en mis prácticas de empresa en el MADOC, decidí utilizar las mismas herramientas para no complicar el desarrollo de la versión similar realizada en este proyecto.

Por lo tanto, se ha elegido el lenguaje de programación C#, concretamente con el modelo de desarrollo web ASP.NET [59], haciendo uso de la tecnología WebForms. En cuanto al entorno de desarrollo, ocurre igual que en el caso anterior. Los ordenadores de la empresa

donde realicé las prácticas poseen el entorno de desarrollo Visual Studio 2017 [60], así que me decanté por el mismo para desarrollar este trabajo.

Para el sistema de recuperación de información, como se ha comentado anteriormente, se ha utilizado ElasticSearch, en concreto la versión 5.5.0. con el cliente de alto nivel NEST [61], la librería existente para .NET. Además, para realizar algunas pruebas también se ha utilizado Kibana 5.5.0 [42].

7.2 Creación de los documentos: Rastrear de documentos

Para obtener los documentos que vamos a indexar posteriormente, tenemos que recorrer las diferentes localizaciones del servidor web. Para ello, se ha realizado un rastreo recursivo en el cual se recorren todas las carpetas y subcarpetas para acceder a todos los documentos. Una vez que tenemos los mismos y dependiendo del formato que posea, los trataremos de una manera u otra. Veamos los diferentes tipos que se aceptan ahora mismo y que realizamos con los mismos:

- Tipo Hipertexto: Serán todos aquellos archivos web, con extensión .html. Se parsea el documento para obtener el título y el texto contenido dentro del mismo, así como los metadatos que hemos mencionado ya anteriormente (autor, fecha de creación, fecha de última lectura, tamaño, servidor, etc).
- Tipo Texto: Serán aquellos documentos de tipo texto, que en este caso serán archivos con extensión .pptx, .docx, .xlsx, .pdf y .txt. Cada uno de ellos se inspecciona de manera diferente, pues están compuestos de forma distinta:
 - Pdf: Se utiliza una clase para parsearlo, en concreto Pdfparser.cs.
 - Microsoft Powerpoint (pptx): Hacemos uso de la biblioteca Microsoft.Interop.Powerpoint para obtener el contenido del mismo.
 - Microsoft Word (docx): Hacemos uso de la biblioteca Microsoft.Interop.Word para obtener el contenido del mismo.
 - Microsoft Hoja de cálculo (xlsx): Hacemos uso de la biblioteca Microsoft.Interop.Excel para obtener el contenido del mismo.
 - Documento de texto plano (.txt): Se extrae directamente pues el entorno ASP.NET nos lo permite.
- Tipo Imagen: Aquellos documentos con extensión .png y .jpg. Todos se tratan de igual manera ya que sólo obtenemos los metadatos de las mismas.
- Tipo Audio: Los documentos con extensión .mp3. Obtenemos la duración de los mismos y los metadatos a los que podamos acceder.
- Tipo Vídeo: Documentos con extensión .mp4. Al igual que en el caso anterior, obtenemos la duración, así como los metadatos que tenga disponibles.

7.3 Creación del índice de ES y mapping

Para crear el índice contra el que lanzaremos las consultas más adelante, debemos especificar una serie de opciones y parámetros que vamos a comentar a continuación.

En primer lugar, debemos especificar el nodo (url) donde se encuentra el cliente de elasticSearch (por defecto el puerto es 9200). Después, crearemos una variable de configuración (número de servidores, número de nodos, número de réplicas, etc.) para ese nodo y por último crearemos el cliente con la configuración [62] (en nuestro caso, por defecto):

```
var node = new Uri("http://localhost:9200");
var settings = new ConnectionSettings(node);
var cliente_ES= new ElasticClient(settings);
```

Figura 9 - Código para crear el cliente de ES

Una vez que ya tenemos el cliente, tenemos que crear un índice en el que vamos a indexar los documentos. Las características del mismo van a ser las siguientes:

- Vamos a tener 1 shard.
- No vamos a tener réplicas, ya que en este caso no tenemos más servidores u ordenadores para realizarlo.
- En cuanto al análisis que vamos a realizar de los campos de texto, la configuración será la siguiente:
 - Un tokenizador estándar [63].
 - Filtro para minúsculas [64].
 - Stemmer para el castellano [65].
 - Stopwords en castellano [18].
 - Conversión de palabras con acentos a palabras normales [66].

Ya hemos visto la configuración que tendrá nuestro índice. Ahora, vamos a especificar el Mappings. El Mappings es donde definimos cómo se van a indexar y almacenar los documentos y sus campos dentro del índice. Aquí definimos qué campos de tipo string se tratarán como campos de full text, qué campos como números, fechas, palabras clave, etc. Aunque en un principio se dejó esta configuración por defecto, se modificó para mejorar los resultados. Los campos especificados del mapping para los 5 tipos de documentos son:

Hipertexto:

- Hemos especificado los campos formato y departamento como palabras clave (keyword) [67]. Se utiliza este tipo para datos estructurados como códigos postales, nombres, etiquetas, etc. Solamente se pueden encontrar si se busca el valor exacto.
- El campo títuloHipertexto y textoContenido como tipo texto. Esto indexará todo el contenido del campo. Como en nuestro caso hemos configurado el índice para que realice stemming y utilice las stopwords, estas operaciones se realizarán sobre el texto de los campos. Además, para estos dos campos, guardamos los términos, posiciones de los mismos y desplazamiento de los caracteres para poder hacer posteriormente el subrayado (**highlight**) [68].

Texto: Igual que para hipertexto.

Imagen:

- Al igual que los anteriores, formato y departamento como tipo keyword. Pero también lo será el campo etiquetas, para que podamos buscar las etiquetas si especificamos una etiqueta que tenga asignada.
- El campo url será de tipo texto, para que se intente buscar también por la url, ya que, en muchas ocasiones, la palabra puede estar en la ruta de la url.
-

Audio y vídeo: Exactamente igual que el tipo de documento Imagen.

7.4 Operaciones principales

Indexación

Una vez especificado cómo queremos que se traten los diferentes campos de los diferentes documentos, ya estamos en disposición de indexar los mismos. Para indexarlos lo hacemos con la API “index”, de manera por defecto. Aunque se puede especificar el id que tendrá el documento una vez indexado, lo dejaremos por defecto, ya que una ventaja de ES es que crea automáticamente los mismos. Los 5 tipos de Indexaciones que se han implementado son:

- **Indexación General:** Se indexarán todos los archivos de todos los tipos que se han descrito en el punto 7.2.
- **Indexación por servidor:** Se especificará de qué servidor se desea indexar documentos y, una vez indicado, se realizará indexación general sobre el mismo.
- **Indexación por departamento:** Se especificará sobre qué departamento se desea indexar documentos y se realizará una indexación general sobre el mismo.
- **Indexación por tipo de archivo:** Se especificará qué tipo de archivo se desea indexar y recorrerá todos los servidores indexando los mismos: Los posibles valores son: hipertexto, texto, imagen, audio y vídeo.
- **Indexación por url prioritaria:** El administrador podrá especificar la url dentro de la intranet donde se encuentra el documento que desea indexar y se indexará siempre que sea de los formatos aceptados.

Una vez indexado el documento, podrá ser encontrado por los usuarios.

Actualización

Aunque muchos documentos no suelen ser modificados, es necesario que si alguno lo hace, se actualice en el índice para que los usuarios sean capaces de acceder al contenido del mismo.

Para llevar a cabo esta acción, recorreremos con el Crawler los documentos del servidor y compararemos si han cambiado algunos campos críticos para la búsqueda, como el textoContenido, la fechadeModificación, etc.

Al igual que para la indexación, tenemos los mismos 5 tipos. Se actualizarán unos documentos u otros dependiendo de la acción que se seleccione.

Borrado

Esta acción elimina el documento del Índice de manera que ya no podrá ser encontrado por el usuario utilizando el buscador. Para esta operación se han implementado las 5 variables como en la Indexación y en la Actualización.

Búsquedas

Es la operación principal de este trabajo, ya que, sin ella, los objetivos no podrían llevarse a cabo. Vamos a indicar cómo se han realizado las diferentes operaciones de búsqueda en las diferentes partes de la aplicación Web. También se ha implementado dentro del campo de texto de búsqueda un autocompletado, donde, una vez que el usuario haya escrito al menos 3 caracteres, se mostrarán un máximo de 6 documentos que contengan esos caracteres en el nombre del archivo. Por ejemplo, para obtener las sugerencias del buscador General, se utiliza la siguiente consulta:

```
cliente_ES.Search<dynamic>(s => s
    .Size(6)
    .Type(Types.Type("texto", "hipertexto"))
    .Source(fs => fs
        .Includes(inc => inc.Field("nombreArchivo")))
    .Query(q =>
        (q.Bool(bo => bo
            .Filter(fil => fil
                .Term(te => te
                    .Field("estadoActividad")
                    .Value(1)
                )
            )
        )
        && +q.MatchPhrasePrefix(m => m
            .Field("nombreArchivo")
            .Query(pTextoBusqueda)
        )
    )
);
});
```

Figura 10 - Código para obtener sugerencias

Búsqueda General

Una vez que se entra en la dirección base de la aplicación, se podrá realizar esta búsqueda introduciendo el texto que se desea buscar. Esta búsqueda devolverá documentos de tipo Hipertexto, Texto e Imagen, siempre y cuando cumplan los criterios de la búsqueda.

Así, se realiza en primer lugar una búsqueda de texto e hipertexto y otra para las imágenes. En la primera buscamos un máximo de 200 documentos de tipo dinámico, dentro de los tipos hipertexto o texto. La consulta está formada por una condición booleana que se debe cumplir: estadodeActividad debe ser 1, es decir, el archivo debe existir en el servidor. Después, debe satisfacer una serie de consultas match Phrase [69]: son consultas en las cuales se analiza el texto de entrada y se crea una frase de consulta con el texto analizado

(haciendo uso del analizar, stemmer, stopwords, etc), y le añadimos un slop que hemos calculado anteriormente dependiendo del nº de palabras de la consulta.

El parámetro slop [70] indica a la consulta matchPhrase cómo de lejos se permite que se encuentren los términos y siga coincidiendo la consulta, es decir, cuántas veces se necesita mover un término para hacer que la consulta y el documento coincidan. Veamos un ejemplo sencillo: para que la consulta “perro blanco” coincida en un documento que contenga “perro grande blanco” se necesitaría un slop de 1:

	Pos1	Pos2	Pos3

Documento:	perro	grande	blanco
Consulta:	perro	blanco	
Slop 1:	perro		↳ blanco

Así, debe satisfacer la consulta en el títuloHipertexto (en caso de hipertextos), en nombreArchivo o el texto contenido. Después, añadimos un Highlight, que sirve para obtener los fragmentos de texto donde se encuentra el texto que se ha consultado, pudiendo también resaltar las mismas. Así, obtenemos un máximo de 3 fragmentos de tamaño 150 caracteres. La consulta que acabamos de comentar es la siguiente:

```

cliente_ES.Search<dynamic>(s => s
    .Size(200)
    .Type(Types.Type("hipertexto", "texto"))
    .Query(q => q
        .Bool(bo => bo
            .Filter(fi => fi
                .Term(te => te.Field("estadoActividad").Value(1))
            )
        )
        && +(q.MatchPhrase(m => m
            .Field("textoContenido").Query(pTextoBusqueda).Slop(slop)
        )
        || q.MatchPhrase(m => m
            .Field("tituloHiperTexto").Query(pTextoBusqueda).Slop(slop)
        )
        || q.MatchPhrase(m => m
            .Field("nombreArchivo").Slop(slop).Query(pTextoBusqueda)
        )
    )
    .Highlight(h => h
        .PreTags("<strong style=\"color:#FF4000;\">")
        .PostTags("</strong>")
        .Fields(fs => fs
            .Field("textoContenido")
            .Order("score")
            .Type(HighlighterType.Unified)
            .FragmentSize(150)
            .NumberOfFragments(3)
            .NoMatchSize(150)
            .ForceSource()
        )
    )
);

```

Figura 11 - Código de Búsqueda General

Ahora faltaría la parte de consulta de imágenes. Para ello, ahora buscamos en el tipo imagen, documentos que cumplan lo siguiente:

- Una consulta booleana donde estadoActividad debe ser 1, como en la consulta anterior.
- Una consulta Multimatch: Permite realizar una consulta Match sobre varios campos diferentes: en este caso buscaremos el texto introducido por el usuario en los campos nombre del archivo, url y etiquetas.

La **consulta Match** es de tipo booleano. Se analiza el texto que se ha buscado y se construye una consulta booleana, es decir, si contiene el texto o no.

Finalmente, ordenamos los resultados por la fecha de modificación, en orden descendente. La consulta es la siguiente:

```

cliente_ES.Search<Imagen>(s => s
    .Type(Types.Type("imagen"))
    .Query(q =>
        q.Bool(bo => bo
            .Filter(fi => fi
                .Term(te => te.Field("estadoActividad").Value(1))
            )
        )
        && +(q.MultiMatch(
            m => m
            .Fields(fie => fie
                .Field(fi => fi.nombreArchivo)
                .Field(fi => fi.urlRuta)
                .Query(pTextoBusqueda)))
        || +q.Match(ma => ma
            .Field("etiquetas")
            .Query(pTextoBusqueda)))
        .Sort(ss => ss.Descending(p => p.fechaModificacionArchivo))
    );

```

Figura 12 - Código de Búsqueda de Imágenes

Después, en la interfaz, realizamos la paginación de los resultados de forma que como máximo sólo se muestran 20 documentos por página.

Búsquedas de Imagen, Audio y Vídeo

Ya hemos visto la consulta sobre imágenes y qué campos tiene en cuenta. Las búsquedas de audios y de vídeo se realizan de igual manera, ya que los archivos multimedia no tienen textoContenido sobre el que buscar.

7.5 Relevancia de los resultados

Una vez realizada la consulta, los documentos se devuelven en orden descendente de relevancia. La relevancia es el **score** [72] de cada documento representada por un número positivo en coma flotante.

La forma en la que se calcula esta puntuación varía dependiendo del tipo de consulta que se haya realizado: una búsqueda difusa calcula el score calculando cómo de similares son la entrada de la consulta del usuario y la palabra o palabras del documento. La búsqueda booleana no genera puntuación, sólo muestra aquellos documentos que cumplen el requisito.

Pero vamos a explicar el algoritmo estándar usado con ElasticSearch para calcular las relevancias o scores de los documentos. El algoritmo de similitud se llama frecuencia del término / frecuencia inversa del documento, o **TF/IDF** [73], que tiene en cuenta los siguientes factores:

- **Frecuencia del término:** Responde a la pregunta ¿Con qué frecuencia aparece un término en el campo buscado? Cuanta más frecuencia, más relevante. Un documento que contenga una vez el término será menos relevante que uno que lo contenga cuatro veces.
- **Frecuencia inversa del documento:** Responde a la pregunta ¿Con qué frecuencia aparece cada término en el índice? Cuanto más aparezca, menos relevante será el documento. Los términos que aparecen en muchos documentos tienen menos peso que los términos poco comunes.
- **Norma de campo-longitud:** ¿Cómo de largo es un campo? Cuanto más largo sea, menos probable es que las palabras sean relevantes en ese campo. Un término que aparezca en un título corto tiene más peso que el mismo término que aparezca en un campo de contenido muy grande.

Así, ElasticSearch, como es lógico, utiliza el mismo método para el cálculo de la puntuación que Lucene, llamado **practical scoring function** [74], que hemos explicado en los puntos anteriores.

7.6 Depuración y pruebas

7.6.1 Depuración

Para llevar a cabo la depuración y, por lo tanto, comprobación de que todo iba correctamente se ha hecho uso del debugger de Visual Studio 2017, así como de la consola de Kibana. Conforme se iban realizando las operaciones, se ha ido comprobando en ambas las respuestas de las diferentes operaciones y se ha comprobado si se realizaban las modificaciones oportunas en el índice de ElasticSearch.

La depuración me ha permitido detectar ciertos errores. Entre ellos:

- Autocompletado no funcionaba correctamente:
- Actualizado no funcionaba correctamente:

7.6.2 Catálogo de pruebas

Tras finalizar el desarrollo de la aplicación, el siguiente objetivo es llevar a cabo una serie de pruebas para comprobar el correcto funcionamiento del sistema. Para ello, se muestra un catálogo en el que se presentan las pruebas a realizar y sus correspondientes resultados.

ID	PRU-01
Descripción	Indexar un documento en el Índice
Pasos	<ul style="list-style-type: none"> • Administrador accede al sistema. • Accede a la pestaña de Configuración -> Indexaciones. • Pulsa la opción que desea y pincha el botón
Posibles errores	<ol style="list-style-type: none"> 1. No se han obtenido bien los datos del documento físico. 2. El cliente de Elasticsearch no está en funcionamiento. 3. En el caso de indexación por servidor: el servidor no se encuentra disponible. 4. En el caso de indexación por url prioritaria: la url no existe o no se puede acceder.
Resultado esperado	El documento se ha indexado correctamente y puede ser encontrado por una consulta en el SRI.

Tabla 16 - Prueba PRU-01

ID	PRU-02
Descripción	Actualizar un documento
Pasos	<ul style="list-style-type: none"> • Administrador accede al sistema. • Accede a la pestaña de Configuración -> Actualizaciones. • Pulsa la opción que desea y pincha el botón
Posibles errores	<ol style="list-style-type: none"> 1. No se puede acceder al documento del Índice porque el cliente de Elasticsearch no está en funcionamiento. 2. En el caso de actualización por servidor: el servidor no se encuentra disponible. 3. En el caso de actualización por url prioritaria: la url no existe en ningún documento del Índice.
Resultado esperado	Los documentos cuyos campos han cambiado se han actualizado correctamente en el Índice.

Tabla 17 - Prueba PRU-02

ID	PRU-03
Descripción	Borrar un documento
Pasos	<ul style="list-style-type: none"> • Administrador accede al sistema. • Accede a la pestaña de Configuración -> Borrado. • Pulsa la opción que desea y pincha el botón.
Posibles errores	<ol style="list-style-type: none"> 1. No se puede acceder al documento del Índice porque el cliente de Elasticsearch no está en funcionamiento. 2. En el caso de borrado por url prioritaria: la url no existe en ningún documento del Índice.
Resultado esperado	Los documentos se han borrado correctamente y ya no podrán obtenerse cuando se realiza una consulta desde la interfaz.

Tabla 18 - Prueba PRU-03

ID	PRU - 04
Descripción	Búsqueda General
Pasos	<ul style="list-style-type: none"> • Administrador o usuario acceden al sistema, introducen la consulta y pulsa en buscar.
Posibles errores	<ol style="list-style-type: none"> 1. El cliente de Elasticsearch no se encuentra en funcionamiento.
Resultado esperado	Se redirecciona a una página con los resultados de la consulta. Resultados de tipo Imagen, Hipertexto y Texto.

Tabla 19 - Prueba PRU-04

ID	PRU - 05
Descripción	Búsqueda de imágenes
Pasos	<ul style="list-style-type: none"> • Administrador o usuario acceden al sistema, y después a la pestaña de “Buscar Imágenes”, introduce su consulta y pulsa en Buscar.
Posibles errores	<ol style="list-style-type: none"> 1. El cliente de Elasticsearch no se encuentra en funcionamiento.
Resultado esperado	La aplicación redirecciona a una página con los resultados obtenidos de la consulta realizada.

Tabla 20 - Prueba PRU-05

ID	PRU-06
Descripción	Búsqueda de Audios
Pasos	<ul style="list-style-type: none"> Administrador o usuario acceden al sistema, y después a la pestaña de “Buscar Audios”, introduce su consulta y pulsa en Buscar.
Posibles errores	1. El cliente de ElasticSearch no se encuentra en funcionamiento.
Resultado esperado	La aplicación redirecciona a una página con los resultados obtenidos de la consulta realizada

Tabla 21 - Prueba PRU-06

ID	PRU-07
Descripción	Búsqueda de vídeos
Pasos	<ul style="list-style-type: none"> Administrador o usuario acceden al sistema, y después a la pestaña de “Buscar Vídeos”, introduce su consulta y pulsa en Buscar.
Posibles errores	1. El cliente de ElasticSearch no se encuentra en funcionamiento.
Resultado esperado	La aplicación redirecciona a una página con los resultados obtenidos de la consulta realizada

Tabla 22 - Prueba PRU-07

ID	PRU-08
Descripción	Autocompletar
Pasos	<ul style="list-style-type: none"> El administrador o usuario accede al sistema y empieza a introducir las letras que forman la palabra/frase que desea consultar.
Posibles errores	1. El cliente de ElasticSearch no se encuentra en funcionamiento.
Resultado esperado	Si los caracteres que está introduciendo (mínimo 3), forman parte de algún nombre de archivo de algún documento de texto o hipertexto, se mostrará debajo de la entrada de texto (hasta un máximo de 6 sugerencias).

Tabla 23 - Prueba PRU-08

ID	PRU-09
Descripción	Aplicar filtro por departamento
Pasos	<ul style="list-style-type: none"> El administrador o usuario acceden al sistema, introducen la consulta y pulsa en buscar. Si se encuentran resultados que satisfagan su búsqueda, se muestran los resultados y un desglose de cuántos documentos coinciden para cada departamento. El administrador o usuario pincha en el departamento en el cual está interesado de que el documento pertenezca
Posibles errores	<ol style="list-style-type: none"> El cliente de ElasticSearch no se encuentra en funcionamiento. No se ha actualizado correctamente la interfaz con los resultados actuales.
Resultado esperado	Se muestran sólo los documentos del departamento que el usuario ha seleccionado y que cumplen los requisitos de la búsqueda realizada.

Tabla 24 - Prueba PRU-09

ID	PRU-10
Descripción	Acceder a un documento (Búsqueda general)
Pasos	<ul style="list-style-type: none"> El administrador o usuario acceden a la página principal, escriben su consulta y pulsa en buscar. Si hay resultados, se muestran ordenados en una nueva página. El usuario pincha sobre el enlace del título del documento.
Posibles errores	<ol style="list-style-type: none"> El servidor de documentos no se encuentra accesible. La url no está bien compuesta. El archivo se ha eliminado recientemente del índice.
Resultado esperado	Se abre una nueva pestaña con el documento al que se ha accedido o se descarga en caso de ser de ciertos tipos (PowerPoint, Word, Excel, etc).

Tabla 25 - Prueba PRU-10

ID	PRU-11
Descripción	Visualizar documento (búsqueda de Audio, Vídeo e Imagen)
Pasos	<ul style="list-style-type: none"> Administrador o usuario acceden a algún tipo de búsqueda multimedia (Audio, Vídeo o Imagen), introducen su consulta y pinchan en buscar. Se redirecciona a la página con los resultados obtenidos (en caso de que los hubiese) y los visualiza.
Posibles errores	<ol style="list-style-type: none"> El servidor de documentos no se encuentra accesible. La url no está bien compuesta. El archivo se ha eliminado recientemente del servidor.
Resultado esperado	El usuario puede ver los contenidos y reproducirlos en caso de audio y vídeo. En las imágenes se pueden abrir en una nueva pestaña si se pincha sobre la misma.

Tabla 26 - Prueba PRU-11

Resultado de las pruebas:

ID Prueba	Resultado
PRU-01	Superada
PRU-02	Superada
PRU-03	Superada
PRU-04	Superada
PRU-05	Superada
PRU-06	Superada
PRU-07	Superada
PRU-08	Superada
PRU-09	Superada
PRU-10	Superada
PRU-11	Superada

Tabla 27 - Resultado de las pruebas

Como vemos en la tabla anterior, todas las pruebas han sido superadas.

7.7 Interfaz Web

A continuación, se muestran algunas imágenes para mostrar cómo es la interfaz.

En primer lugar, veamos la página principal (Figura 13):

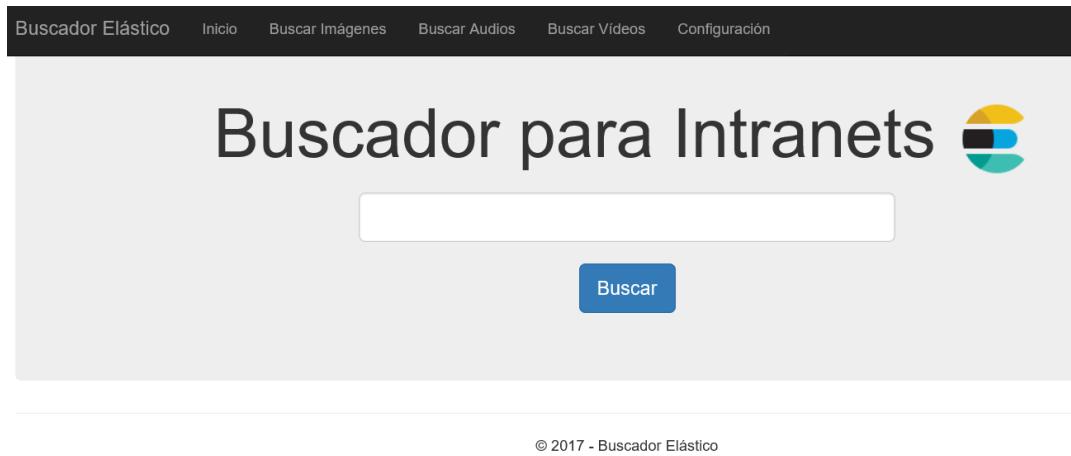


Figura 13 - Página principal del SRI

Podemos ver que en la parte superior tenemos las diferentes pestañas correspondientes a los diferentes tipos de búsquedas. Y en la parte central la caja de texto donde el usuario introducirá su consulta.

Cuando empieza a escribir, se activa la función de autocompletado para sugerir documentos (hasta seis) que tengan esas letras en el nombre del archivo, por ejemplo (Figura 14):

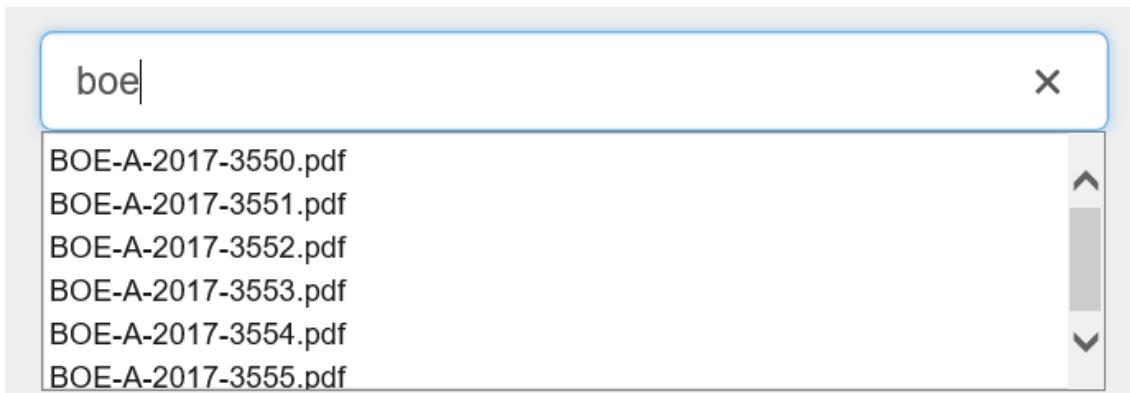


Figura 14 - Ejemplo Autocompletar

Una vez el usuario pincha el botón de buscar, se redirige a la página de los resultados encontrados, donde se muestra un pequeño resumen del resultado (Figura 15): el texto que se ha buscado, el tiempo que ha tardado, el nº de resultados totales para esa búsqueda y los resultados que se están mostrando en la página actual. Así mismo también se muestra en qué departamentos se han encontrado esos documentos y se permite mostrar sólo los del dpto. que nos interese:

Buscador para Intranets

Buscar

Resultado de la búsqueda:

Texto buscado: albusac

Tiempo de búsqueda: 2959 milisegundos.

3 resultados.

Mostrando 3 de 3 resultados

Departamentos:

Algebra(1)

Arquitectura y tecnologia de computadores(1)

Teoria de la senal, telematica y comunicaciones(1)

Figura 15 - Resumen Resultados

Los documentos encontrados se muestran bloques donde se puede ver (Figura 16): el icono del tipo de archivo, el dpto. al que pertenece (si lo tuviese), el nombre del archivo en el cual se puede pinchar para acceder, la url del archivo, la puntuación obtenida para la consulta y los trozos de texto por los cuales se ha seleccionado ese documento como relevante.

 [Arquitectura y tecnologia de computadores] [Autorización.pdf](#)

 <http://localhost/servidorIntranet/Arquitectura%20y%20tecnologia%20de%20computadores/Autorizaci%C3%B3n.pdf>

Puntuación: 18

Fdo.: Miriam **Albusac** Jorge.[...]

[...] Microsoft Word - Autorozación.docx Yo, Miriam **Albusac** Jorge, con DNI 45715301H, autorizo a mi hermano César **Albusac** Jorge a retirar de la sección de títulos

 [Teoria de la senal, telematica y comunicaciones] [1. INSTALACIÓN BÁSICA BUSCADOR ELÁSTICO.docx](#)

<http://localhost/servidorIntranet/Teoria%20de%20la%20senal,%20telematica%20y%20comunicaciones/1.%20INSTALACI%C3%93N%20B%C3%81SICA%20BUSCADOR%20EL%C3%81STICO.docx>

Puntuación: 13

pretty { "doc": { "name": "Cesar **Albusac** Jorge", "age": 24 } } 13.[...]

[...] pretty { "name": "Cesar **Albusac**" } / Y para visualizarlo: GET /customer/external/1? [...]

[...] pretty { "doc": { "name": "Cesar **Albusac** Jorge" } } Ahora, añadiendo otro dato al documento: POST /customer/external/1

 [Algebra] [1. INSTALACIÓN BÁSICA BUSCADOR ELÁSTICO.docx](#)

<http://localhost/servidorIntranet/Algebra/1.%20INSTALACI%C3%93N%20B%C3%81SICA%20BUSCADOR%20EL%C3%81STICO.docx>

Puntuación: 13

pretty { "doc": { "name": "Cesar **Albusac** Jorge", "age": 24 } } 13.[...]

[...] pretty { "name": "Cesar **Albusac**" } / Y para visualizarlo: GET /customer/external/1? [...]

[...] pretty { "doc": { "name": "Cesar **Albusac** Jorge" } } Ahora, añadiendo otro dato al documento: POST /customer/external/1

Figura 16 - Ejemplo de archivos encontrados

En cuanto a la página de resultados de buscar audios (Figura 17), podemos ver un resumen de la búsqueda, como en el caso anterior, y después los audios en bloques, en los cuáles se muestra la duración, url, el nombre del archivo, las etiquetas asignadas y el icono con el formato correspondiente del mismo:

The screenshot shows a search interface with the following elements:

- Header:** "Buscar Audios" with a red musical note icon.
- Search Bar:** An empty input field.
- Search Button:** A blue button labeled "Buscar".
- Section Header:** "Resultado de la búsqueda:"
- Text Info:** "Texto buscado: route", "Tiempo de búsqueda: 2 milisegundos.", "1 resultados.", "Mostrando 1 de 1 resultados".
- Result Item:**
 - Icon:** MP3 file icon.
 - Name:** "[Teoria de la senal, telematica y comunicaciones] Axwell Λ Ingrosso - Tomorrowland 2017 [Recorded Live].mp3"
 - URL:** <http://localhost/servidorIntranet/Teoria%20de%20la%20senal,%20telematica%20y%20comunicaciones/Axwell%20%20%CE%9B%20Ingrosso%20-%20Tomorrowland%202017%20%5BRecorded%20Live%5D.mp3>
 - Duration:** Duración: 3349 segundos.
 - Player Controls:** Includes a play button, duration (0:00:00), a progress bar, and a volume icon.
 - Tags:** Etiquetas: audio,sonido
 - Tag Input:** A text input field with placeholder "Etiquetas separadas por ,".
 - Add Tag Button:** A green button with a plus sign (+).

Figura 17 - Ejemplo Búsqueda de Audio

Para el resultado de las imágenes (Figura 18), podemos ver la altura y anchura de la imagen, el nombre de la imagen, la url y las etiquetas, así como añadir más etiquetas a esa imagen:

Buscar Imágenes

Buscar

Resultado de la búsqueda:

Texto buscado: imagen

Tiempo de búsqueda: 2 milisegundos.

10 resultados.

Mostrando 10 de 10 resultados



Figura 18 - Ejemplo Búsqueda de Imágenes

Igual para el resultado de la búsqueda de vídeos (Figura 19):

Buscar Vídeos

Buscar

Resultado de la búsqueda:

Tiempo de búsqueda: 52 milisegundos.

1 resultados.

Mostrando 1 de 1 resultados

Tomorrowland Belgium 2017 Axwell Λ Ingrosso.mp4

Duración: 3348 segundos.



Etiquetas: video

Figura 19 - Ejemplo Búsqueda de Vídeos

Y finalmente, la sección de Configuración (Figura 20), la cual sólo podrá ver el administrador:

Configuración

Indexaciones

Actualizaciones

Borrado

Figura 20 - Página de Configuración

Donde, dentro de cada opción, aparecen 5 posibilidades:

- Indexación general.
- Indexación por servidor.
- Indexación por departamento.
- Indexación por tipo de archivo.
- Indexación por URL prioritaria.

8. Conclusiones y trabajo futuro

8.1 Temporización y presupuesto reales

La temporización y consecuentemente el presupuesto han sufrido algunas variaciones en cuanto a temporizaciones. A continuación, se muestran la temporización real y el presupuesto real final tras realizar los ajustes oportunos:

Establecimiento de especificaciones

- Establecimiento de requisitos (30 horas).
- Análisis (15 horas).
- Estudio de las alternativas de diseño (25 horas).

Diseño

- Diseño del modelo de datos (35 horas).
- Diseño de la aplicación Web (20 horas).

Desarrollo

- Implementación del modelo de ElasticSearch y consultas (80 horas).
- Implementación de la aplicación Web (80 horas).
- Desarrollo de algunas mejoras (35 horas).

Pruebas

- Integración del Sistema de Recuperación de Información y pruebas en la aplicación Web con usuarios reales (60 horas).

Documentación

- Redacción de la memoria (50 horas).

Reajustes a posteriori

- Correcciones y ajustes oportunos (55 horas).

A continuación, se muestra una tabla-resumen con las diferentes tareas y las respectivas duraciones (Tabla 28):

Tarea	Duración
Especificaciones	70 horas
Diseño	55 horas
Desarrollo	195 horas
Pruebas	60 horas
Documentación	50 horas
Reajustes	55 horas
TOTAL	485 horas

Tabla 28 - Temporización final real

Su desarrollo real final a nivel de plan de trabajo, se ha realizado siguiendo el siguiente diagrama de Gantt:

Nombre	Fecha de inicio	Fecha de fin
• Establecimiento de requisitos	28/11/16	5/12/16
• Análisis	7/12/16	12/12/16
• Estudio de las alternativas de diseño	13/12/16	20/12/16
• Diseño del modelo de datos en ES	21/12/16	30/12/16
• Diseño de la aplicación Web	3/01/17	13/01/17
• Implementación del modelo ES y consultas	16/01/17	10/02/17
• Implementación aplicación Web	13/02/17	2/03/17
• Integración del SRI y prueba con usuarios	3/03/17	20/03/17
• Correcciones oportunas	21/03/17	18/09/17
• Redacción de memoria	1/06/17	1/09/17
• Desarrollo de mejoras	3/07/17	5/09/17

Figura 21 - Temporización Gantt Final

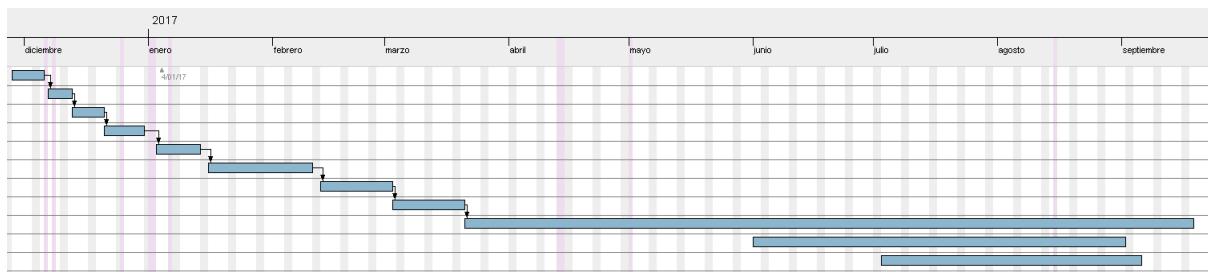


Figura 22 - Diagrama de Gantt Final

Presupuesto real final

El presupuesto se ha visto modificado debido al incremento de horas. Por lo tanto, la parte de personal ha quedado así:

Costes de personal

Ocupación	Horas	Precio/Hora (€/hora)	Importe(€)
Jefe de proyecto	25	50,00	1.250,0
Ingeniero Informático	485	25,00	12.125,0
TOTAL	510	-	13.375,00

Tabla 29 - Costes de personal final

Como las demás partes del presupuesto no han cambiado, el presupuesto final ha quedado así finalmente (Tabla 30):

COSTE TOTAL

Concepto	Precio
Costes de personal	13.375,00
Costes hardware	2.594,28
Costes software	572,00
TOTAL	16.541,28 €

Tabla 30 - Presupuesto final real

Como podemos ver, se ha visto incrementado en 750 €, con respecto a la previsión presupuestal inicial (15.791,28 €).

8.2 Relación con los estudios

La realización de este trabajo ha estado relacionada con los contenidos de diversas asignaturas tanto del Grado como del Máster en informática. A continuación, menciono alguna de ellas:

GIW: La realización de este trabajo está estrechamente relacionado con esta asignatura, pues en la misma se han impartido lecciones relacionadas con los Sistemas de Recuperación de Información: Indexación, los diferentes modelos existentes de Recuperación de Información, cómo se realiza la evaluación de la RI, RI en la Web, técnicas avanzadas de RI, además de realizar alguna práctica de la asignatura haciendo uso de la librería comentada anteriormente Lucene.

PGPI: También me ha sido útil esta asignatura ya que en la misma aprendimos a proyectar, calcular y diseñar productos, procesos e instalaciones en todos los ámbitos de la ingeniería informática. Además de todos los aspectos descritos en la memoria, entre los que están:

- Análisis y síntesis: Encontrar, analizar, criticar (razonamiento crítico), relacionar, estructurar y sintetizar información proveniente de diversas fuentes, así como integrar ideas y conocimientos

- Capacidad de organización y planificación, así como capacidad de gestión de la información.
- Capacidad para modelar, diseñar, definir la arquitectura, implantar, gestionar, operar, administrar y mantener aplicaciones, redes, sistemas, servicios y contenidos informáticos.
- Capacidad para analizar las necesidades de información que se plantean en un entorno y llevar a cabo en todas sus etapas el proceso de construcción de un sistema de información.

8.3 Valoración personal

Este trabajo me ha servido para varios fines, todos ellos bastante positivos. En primer lugar, fue parte del trabajo que realicé en mis prácticas de empresa, por lo que pude tener mi primera experiencia de trabajo dentro de una compañía. En segundo lugar, conocer el entorno de programación ASP. NET (C#), el cual no conocía, y el IDE Visual Studio, ya que tiene gran uso en las empresas privadas. También, el desarrollo de este proyecto me sirvió para conocer en profundidad los sistemas de recuperación de información y cómo funcionan, y aprender a adaptarlos a las necesidades específicas de un entorno, así como ver que en la mayoría de las ocasiones, la mejor solución es escoger una buena librería de recuperación de información y adaptarla a tus necesidades, y no crear un SRI desde cero, ya que lleva mucho tiempo y probablemente no llegue a ser mejor que uno existente, a no ser que sea muy muy específico y nada de lo que existe en el mercado se adapte a las necesidades.

8.4 Conclusiones

Por último, y lo más importante y satisfactorio, el proyecto ha servido para dar una solución eficiente y completa a un problema real que existía en el MADOC y que, por lo tanto, puede tener cualquier empresa/ Intranet: La búsqueda de documentos internos en ocasiones puede ser complicada.

Aunque no se han podido realizar pruebas reales para comparar las dos soluciones, la versión realizada con ES era tan superior a la anterior que no se tardó en migrar toda la información y adoptar el nuevo modelo.

8.5 Trabajo futuro

Aunque el trabajo cumplió los requisitos propuestos al comienzo y se dio por buena la solución en el entorno de trabajo, siempre se pueden aplicar mejoras para optimizar al máximo el funcionamiento de este SRI implementado. Algunas mejoras como trabajo futuro podrían ser:

- Añadir un sistema de recomendación a este sistema de recuperación de información, de manera que cada usuario, dependiendo de las búsquedas que ha realizado, pertenecerá a un grupo de usuarios. Así, conociendo el perfil de cada usuario, en la página principal podríamos mostrar los documentos que podrían interesarle dependiendo de las búsquedas que haya realizado su vecindario de usuarios.

- Añadir un método para agregar las etiquetas automáticamente mediante un algoritmo de reconocimiento de imágenes. Así, al cargar las imágenes, se analizaría cada una en busca de los elementos que aparecen en la misma y se añadirían esos descriptores para poder buscar en un futuro por los mismos.

9. Comentarios sobre fuentes principales utilizadas

Las principales fuentes que se han consultado para llevar a cabo este trabajo han sido las siguientes:

- **Documentación Online de la página oficial de ElasticSearch [80]:** Al comienzo de mi proyecto, no conocía Elasticsearch, aunque sí que me sonaba el nombre. Por lo tanto, lo primero que hice fue acceder a la página oficial y empezar a leer documentación. Dado que no conocía mucho sobre el campo de la Recuperación de Información, empecé a asimilar tanto los conceptos teóricos que Elasticsearch lleva consigo (clúster, nodo, índice, tipo, documento, shard, réplica), como los conceptos concretos de un S.R.I y las técnicas y características del mismo (Stopwords, filtros, stemming, score, distancia de Levenshtein, highlight, etc). También describen cómo utilizar ES, dónde y con qué lenguajes es posible.
- **Elasticsearch: The Definitive Guide. A distributed Real-time search and analytics Engine [81].** Ha sido un libro que ha sido vital, puesto que la documentación de la página oficial es un poco escasa y había conceptos que no terminaba de entender. Aunque no describe nada sobre ninguna API, sí que describe de forma teórica y general cómo funcionan todas las partes de Elasticsearch y proporcionan ejemplos para que se entienda de mejor manera. Es un poco extenso pero muy fácil de leer y de entender.
- **Documentación de la API de ElasticSearch para C#, NEST [66]:** Ha sido otra de las fuentes más consultadas, pues, había que traducir los conocimientos adquiridos para empezar a crear el buscador en un lenguaje determinado, en este caso el que se utilizó en el MADOC, .NET (C#). Aunque la información también es un poco escasa, para ejemplos muy básicos y captar las ideas sí que es útil. En la mayoría de las ocasiones, he realizado pruebas de ensayo-error, hasta conseguir los objetivos que se deseaban, pues las documentaciones no tenían las mismas características que nuestro problema, como los tipos de documentos, la cantidad, la forma de procesarlos, la forma de acceder a los documentos, etc.
- **ElasticSearch Cookbook - Second Edition [82]:** Antes de empezar con "Elasticsearch: The definitive Guide" comencé a mirar este libro. El libro contiene "recetas" para creaciones de índices, indexaciones de documentos, borrados, etc. Es bastante completo, pero todas las operaciones las realiza por consola mediante CURL, cosa que no nos soluciona nuestro problema, ya que queremos utilizar

instrucciones en C#. Aunque sí que sirvió para hacer pruebas iniciales de creaciones de índices, indexaciones de colecciones, búsqueda de documentos, etc.

10. Bibliografía

Para todas las referencias contenidas en este trabajo, se ha utilizado la guía de estilo de la Universidad de Granada [78].

- [1]. Elastic, “ElasticSearch: RESTful, Distributed Search & Analytics”, [Fecha de consulta: 30/08/2017]. Disponible en: <https://www.elastic.co/products/elasticsearch>
- [2]. The Apache Software Foundation, “Apache Lucene”, [Fecha de consulta: 30/08/2017]. Disponible en: <https://lucene.apache.org/>
- [3]. Castells, M, “The Information Age. Economy, Society and Culture”, Cambridge (Mass.); Oxford: Wiley-Blackwell. (1999)
- [4]. IBM, “2.5 quintillion bytes of data created every day. How does CPG & Retail manage it?”, [Fecha de consulta: 30/08/2017]. Disponible en:
<https://www.ibm.com/blogs/insights-on-business/consumer-products/2-5-quintillion-bytes-of-data-created-every-day-how-does-cpg-retail-manage-it/>
- [5]. Cacheda, F.; Fernández Luna, J.M.; Huete, J.F., *Recuperación de Información. Un enfoque práctico y multidisciplinar*, España, RA-MA EDITORIAL, 2011.
- [6] Ministerio de Defensa, “Mando de Adiestramiento y Doctrina”, [Fecha de consulta: 30/08/2017]. Disponible en: <http://www.ejercito.mde.es/unidades/Granada/madoc/>
- [7]. García Figuerola, Carlos; Zazo, Ángel Francisco; Alonso Berrocal, José Luis, “La interacción con el usuario en los sistemas de recuperación de información: realimentación por relevancia”, [Fecha de consulta: 30/08/2017]. Disponible en:
<http://www.ibersid.eu/ojs/index.php/scire/article/viewFile/1160/1142>
- [8]. D. Manning, Christopher; Raghavan, Prabhakar; Schütze Hinrich, “Introduction to Information Retrieval”, United States, Cambridge University Press, 2008.
- [9]. Baeza Yates, Ricardo; Ribeiro-neto, Berthier; Mills, Don, “Modern Information Retrieval”, England, ACM Press, 1999.
- [10]. Elastic, “ElasticSearch: Basic Concepts”, [Fecha de consulta: 30/08/2017]. Disponible en: https://www.elastic.co/guide/en/elasticsearch/reference/current/_basic_concepts.html
- [11]. Lengstorf, Jason, “JSON: What It Is, How It Works, & How to Use It”, [Fecha de consulta: 30/08/2017]. Disponible en: <https://www.copterlabs.com/json-what-it-is-how-it-works-how-to-use-it/>
- [12]. Y. Choi, Freddy, “Advances in domain independent linear text segmentation”, [Fecha de consulta: 30/08/2017]. Disponible en: <http://www.aclweb.org/anthology/A00-2004>
- [13]. Elastic, “Elasticsearch Anaysis - Tokenizer”, [Fecha de consulta: 30/08/2017]. Disponible en: <https://www.elastic.co/guide/en/elasticsearch/reference/current/analysis-tokenizers.html>
- [14]. Elastic, “Reducing Words to their Root Form”, [Fecha de consulta: 30/08/2017]. Disponible en: <https://www.elastic.co/guide/en/elasticsearch/guide/current/stemming.html>
- [15]. Cambridge University, “Stemming and lemmatization”, [Fecha de consulta: 30/08/2017]. Disponible en: <https://nlp.stanford.edu/IR-book/html/htmledition/stemming-and-lemmatization-1.html>
- [16]. Cambridge University, “Dropping common terms: stop words”, [Fecha de consulta: 30/08/2017]. Disponible en: <https://nlp.stanford.edu/IR-book/html/htmledition/dropping-common-terms-stop-words-1.html>

- [17]. Elastic, "Stopwords: Performance Versus Precision", [Fecha de consulta: 30/08/2017]. Disponible en: <https://www.elastic.co/guide/en/elasticsearch/guide/current/stopwords.html>
- [18]. Alvin Alexander, "Lucene - Spanish stopwords", [Fecha de consulta: 30/08/2017]. Disponible en: https://alvinalexander.com/java/jwarehouse/lucene/contrib/analyzers/common/src/resources/org/apache/lucene/analysis/snowball/spanish_stop.txt.shtml
- [19]. <http://myungha.weebly.com/blog/ir-vs-db>
- [20]. SchönerWelt, "Searchxml", [Fecha de consulta: 30/08/2017]. Disponible en: <http://www.searchxml.net/>
- [21]. TechTarget, "What is WebDAV (World Wide Web Distributed Authoring and Versioning)", [Fecha de consulta: 30/08/2017]. Disponible en: <http://searchmicroservices.techtarget.com/definition/WebDAV-World-Wide-Web-Distributed-Authoring-and-Versioning>
- [22]. DBSight, "DBSight - Full-Text Search for Database", [Fecha de consulta: 30/08/2017]. Disponible en: <http://www.dbsight.com/>
- [23]. Exorbyte, "Enterprise Search", [Fecha de consulta: 30/08/2017]. Disponible en: <http://www.exorbyte.com/products/enterprise-search/>
- [24]. Indica, "Enterprise Search", [Fecha de consulta: 30/08/2017]. Disponible en: <https://indica.nl/>
- [25]. SearchBlox Software, "Enterprise Search, Sentiment Analysis, Text Analysis", [Fecha de consulta: 30/08/2017]. Disponible en: <https://www.searchblox.com/solutions/enterprise-search/>
- [26]. Xapina, "The Xapian Project", [Fecha de consulta: 30/08/2017]. Disponible en: <https://xapian.org/>
- [27]. Create.io, "CreateDB - Put machine data to work. Scalable, Open Source SQL Database", [Fecha de consulta: 30/08/2017]. Disponible en: <https://crate.io/>
- [28]. Amazon Web Services, "Amazon CloudSearch", [Fecha de consulta: 30/08/2017]. Disponible en: <https://aws.amazon.com/es/cloudsearch/>
- [29]. Algolia, "The Most Reliable Platform for Building Search", [Fecha de consulta: 30/08/2017]. Disponible en: <https://www.algolia.com/>
- [30]. Google, "Google Search Appliance 7.6", [Fecha de consulta: 30/08/2017]. Disponible en: <https://enterprise.google.es/intl/es/search/products/gsa.html>
- [31]. Microsoft Azure, "Azure Search", [Fecha de consulta: 30/08/2017]. Disponible en: <https://docs.microsoft.com/es-es/azure/search/search-what-is-azure-search>
- [32]. Sphinx Technologies Inc, "Sphinx - Open Source Search Engine", [Fecha de consulta: 30/08/2017]. Disponible en: <http://sphinxsearch.com/>
- [33]. MarkLogic, "Best Database for Integrating Data from Silos", [Fecha de consulta: 30/08/2017]. Disponible en: <http://www.marklogic.com/>
- [34]. [Fecha de consulta: 30/08/2017]. Disponible en: <http://searchsqlserver.techtarget.com/definition/ACID>
- [35]. Splunk Inc, "Splunk: The platform for Digital Transformation", [Fecha de consulta: 30/08/2017]. Disponible en: <https://www.splunk.com/>
- [36]. The Apache Software Foundation, "Apache Solr", [Fecha de consulta: 30/08/2017]. Disponible en: <http://lucene.apache.org/solr/>
- [37]. DB-Engines, "Historical trend of search engines popularity", [Fecha de consulta: 28/06/2017]. Disponible en: https://db-engines.com/en/ranking_trend/search+engine
- [38]. DB-Engines, "DB-Engines Ranking-Popularity ranking of Search Engines", [Fecha de consulta: 28/06/2017]. Disponible en: <https://db-engines.com/en/ranking/search+engine>

- [39]. DB-Engines, “DB-Engines Ranking-Method”, [Fecha de consulta: 28/06/2017]. Disponible en: https://db-engines.com/en/ranking_definition
- [40]. Elastic, “Mapping”, [Fecha de consulta: 30/08/2017]. Disponible en: <https://www.elastic.co/guide/en/elasticsearch/reference/current/mapping.html>
- [41]. Elastic, “Powering Data Search, Log Analysis, Analytics”, [Fecha de consulta: 30/08/2017]. Disponible en: <https://www.elastic.co/products>
- [42]. Elastic, “Kibana: Explore, Visualize, Discover Data”, [Fecha de consulta: 30/08/2017]. Disponible en: <https://www.elastic.co/products/kibana>
- [43]. Elastic, “Beats: Data Shippers for Elasticsearch”, [Fecha de consulta: 30/08/2017]. Disponible en: <https://www.elastic.co/products/beats>
- [44]. Elastic, “Logstash: Collect, Parse, Transform logs”, [Fecha de consulta: 30/08/2017]. Disponible en: <https://www.elastic.co/products/logstash>
- [45]. Elastic, “X-Pack: Extend Elasticsearch”, [Fecha de consulta: 30/08/2017]. Disponible en: <https://www.elastic.co/products/x-pack>
- [46]. Elastic, “Stories from Users Like You”, [Fecha de consulta: 30/08/2017]. Disponible en: <https://www.elastic.co/use-cases>
- [47]. Oracle Corporation, “MySQL”, [Fecha de consulta: 30/08/2017]. Disponible en: <http://www.mysql.com/>
- [48]. Levehnstein, “The Levehnstein Algorithm”, [Fecha de consulta: 30/08/2017]. Disponible en: <http://www.levenshtein.net/>
- [49]. Oracle Corporation, “Boolean Full-Text Searches”, [Fecha de consulta: 30/08/2017]. Disponible en: <https://dev.mysql.com/doc/refman/5.7/en/fulltext-boolean.html>
- [50]. PcComponentes, “Configurador de Servidor”, [Fecha de consulta: 30/08/2017]. Disponible en <https://www.pccomponentes.com/configurador/D8F23a650>
- [51]. PcComponentes, “Configurador de PC”, [Fecha de consulta: 30/08/2017]. Disponible en: <https://www.pccomponentes.com/configurador/69A015Ae6>
- [52]. PcComponentes, “Logitech Keyboard K120”, [Fecha de consulta: 30/08/2017]. Disponible en: <https://www.pccomponentes.com/logitech-keyboard-k120>
- [53]. PcComponentes, “HP 200 Ratón Inalámbrico 1000 DPI”, [Fecha de consulta: 30/08/2017]. Disponible en: <https://www.pccomponentes.com/hp-200-raton-inalambrico-1000-dpi>
- [54]. PcComponentes, “Monitor HP 24w 23.8” LED IPS FullHD”, [Fecha de consulta: 30/08/2017]. Disponible en: <https://www.pccomponentes.com/hp-24w-238-led-ips-fullhd>
- [55]. Microsoft, “Windows 10 Pro”, [Fecha de consulta: 30/08/2017]. Disponible en: https://www.microsoft.com/es-es/store/d/windows-10-pro/DF77X4D43RKT/48DN?icid=Cat-Windows-mosaic_banner1-pro
- [56]. Amazon, “Windows server 2012 r2”. [Fecha de consulta: 30/08/2017]. Disponible en: https://www.amazon.es/Microsoft-Windows-Server-Standard-2012/dp/B00GAIMXR4/ref=sr_1_1?ie=UTF8&qid=1504176468&sr=8-1&keywords=windows+server+2012+r2+standard
- [57]. Microsoft, “Office 365 Universitarios”, [Fecha de consulta: 30/08/2017]. Disponible en: https://www.microsoft.com/es-es/store/d/Office-365-University/CFQ7TTC0K5BB?icid=Cat-Office-for-Students-mosaic_mid-O365University
- [58]. Ganttpoint, “Ganttpoint”, [Fecha de consulta: 30/08/2017]. Disponible en: <http://www.ganttpoint.biz/>
- [59]. Microsoft, “ASP.NET”, [Fecha de consulta: 30/08/2017]. Disponible en: <https://www.asp.net/>

- [60]. Microsoft, "Visual Studio 2017", [Fecha de consulta: 30/08/2017]. Disponible en: <https://www.visualstudio.com/es/vs/whatsnew/?rr=https%3A%2F%2Fwww.google.es%2F>
- [61]. Elastic, "Elasticsearch.NET and NEST: The .net clients - Introduction", [Fecha de consulta: 30/08/2017]. Disponible en: <https://www.elastic.co/guide/en/elasticsearch/client/net-api/5.x/introduction.html>
- [62]. Elastic , "Configuration options" , [Fecha de consulta: 30/08/2017], Disponible en: <https://www.elastic.co/guide/en/elasticsearch/client/net-api/current/configuration-options.html>
- [63]. Elastic, "Standard tokenizer", [Fecha de consulta: 30/08/2017]. Disponible en: <https://www.elastic.co/guide/en/elasticsearch/reference/5.5/analysis-standard-tokenizer.html>
- [64]. Elastic , "Lowercase Tokenizer", [Fecha de consulta: 30/08/2017]. Disponible en: <https://www.elastic.co/guide/en/elasticsearch/reference/current/analysis-lowercase-tokenizer.html>
- [65]. Snowball tartarus, "Spanish stemming algorithm", [Fecha de consulta: 30/08/2017]. Disponible en: <http://snowball.tartarus.org/algorithms/spanish/stemmer.html>
- [66]. Elastic, "ASCII folding token filter", [Fecha de consulta: 30/08/2017]. Disponible en: <https://www.elastic.co/guide/en/elasticsearch/reference/current/analysis-asciifolding-tokenfilter.html>
- [67]. Elastic, "Keyword datatype", [Fecha de consulta: 30/08/2017]. Disponible en: <https://www.elastic.co/guide/en/elasticsearch/reference/current/keyword.html>
- [68]. Elastic, "Highlighting", [Fecha de consulta: 30/08/2017]. Disponible en: <https://www.elastic.co/guide/en/elasticsearch/reference/current/search-request-highlighting.html>
- [69]. Elastic," March Phrase Usage", [Fecha de consulta: 30/08/2017]. Disponible en: <https://www.elastic.co/guide/en/elasticsearch/client/net-api/current/match-phrase-usage.html>
- [70]. Elastic, "Closer is Better", [Fecha de consulta: 30/08/2017]. Disponible en: https://www.elastic.co/guide/en/elasticsearch/guide/current/_closer_is_better.html
- [71]. Elastic, "Match query", [Fecha de consulta: 30/08/2017]. Disponible en: <https://www.elastic.co/guide/en/elasticsearch/reference/5.5/query-dsl-match-query.html>
- [72]. Elastic, "Theory Behind Relevance Scoring", [Fecha de consulta: 30/08/2017]. Disponible en: <https://www.elastic.co/guide/en/elasticsearch/guide/current/scoring-theory.html>
- [73]. Cambridge University, "Tf- idf weighting", [Fecha de consulta: 30/08/2017]. Disponible en: <https://nlp.stanford.edu/IR-book/html/htmledition/tf-idf-weighting-1.html>
- [74]. Elastic, "Practical Scoring function", [Fecha de consulta: 30/08/2017]. Disponible en: <https://www.elastic.co/guide/en/elasticsearch/guide/current/practical-scoring-function.html>
- [75]. Elastic, "Elasticsearch Reference", [Fecha de consulta: 30/08/2017]. Disponible en: <https://www.elastic.co/guide/en/elasticsearch/reference/5.0/index.html>
- [76]. Gormley, Clinton; Tong, Zachary, *Elasticsearch: The Definitive Guide. A distributed Real-time search and analytics Engine*, United Stated, O'REILLY, 2015.
- [77]. Paro, Alberto, *ElasticSearch Cookbook*, United Kingdom, Packt Publishing, 2015.
- [78]. García Sánchez, José A.; Peinado Santaella, Rafael G.; López Fernández, Antonio, *Guía de estilo de la editorial Universidad de Granada*, Granada, España, Editorial Universidad de Granada.