S.R.I. con Apache Solr y GATE

Universidad de Huelva
Ingeniería informática
Motores de búsqueda

Esta es la memoria de la Practica 1 de Motores de búsqueda la cual trata de implementar un Sistema de Recuperación de Información (S.R.I.) con Apache Solr y GATE.

Contenido

Introducción

En esta practica se ha propuesto como objetivo crear un Sistema de Recuperación de Información usando el lenguaje de programación Java y las herramientas Apache Solr y GATE.

Esta divido en dos versiones, en la primera montaremos el proyecto y usaremos la configuración básica de Apache Solr, mientras que en la segunda versión se usara GATE para realizar etiquetado y se pasara a configurar de una forma un poco más avanzada Apache Solr.

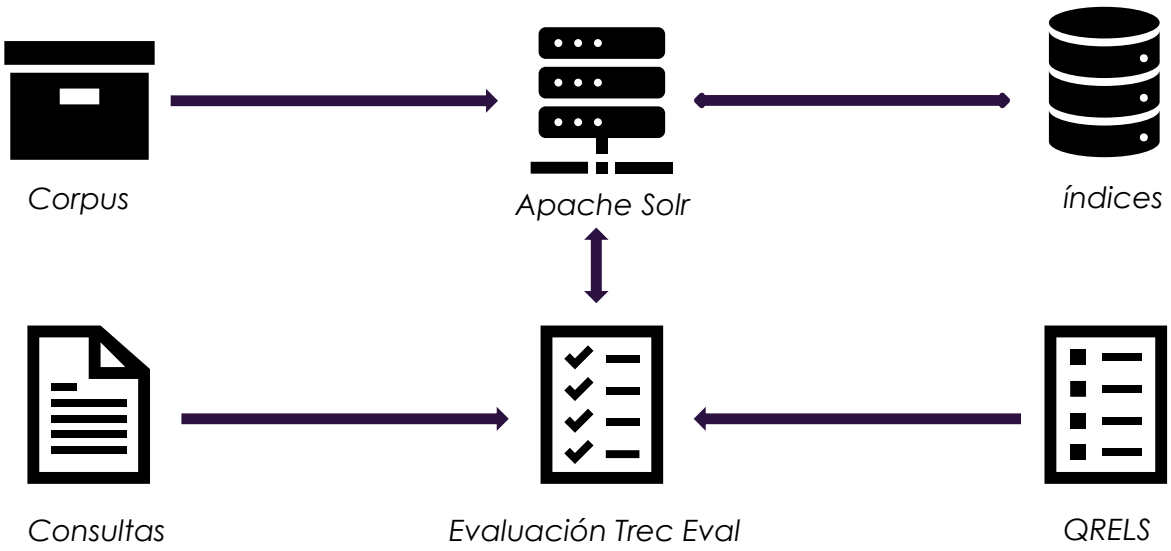
Para la prueba se usará un corpus facilitado por los profesores con aproximadamente con 6000 documentos y para la evaluación se usará la herramienta TREC_EVAL.

Nota: El proyecto de comenzó a desarrollar en Linux y se terminó en Windows 10, por lo tanto, es posible que funcione en ambos sistemas operativos, pero en el caso de Windows 10 se experimentaron ciertas dificultades que causan inestabilidad y que no se pueden resolver debido a su naturaleza intrínseca.

S.R.I. - Primera versión

Introducción Primera versión

En esta versión simplemente procesamos e indexamos los documentos del corpus en Apache Solr, obtenemos las preguntas y generamos los archivos para la evaluación.



Implementación de la primera versión

En este caso he optado por usar **Apache NetBeans 12.2** como entorno de desarrollo y he seleccionado como base un proyecto de tipo **JavaFX** con soporte de **FXML** en **ANT** para hacer uso del patrón **MVC** (Modelo-Vista-Controlador) en mi aplicación y que fuera fácil de montar el proyecto.

El proyecto tiene el código divido en dos paquetes de código fuente y todo los demás como hojas de estilo en cascada (css), imágenes y las vista en fxml están en la carpeta de recursos.

Los dos paquetes mencionados antes son **com.joseram0n.sri** y **solr_io,** el primero contiene algunos modelos y los controladores de las vistas mientras que el segundo tiene una clase para hacer test y clases para la comunicación con Apache Solr que también actúan como modelo en algunos casos.

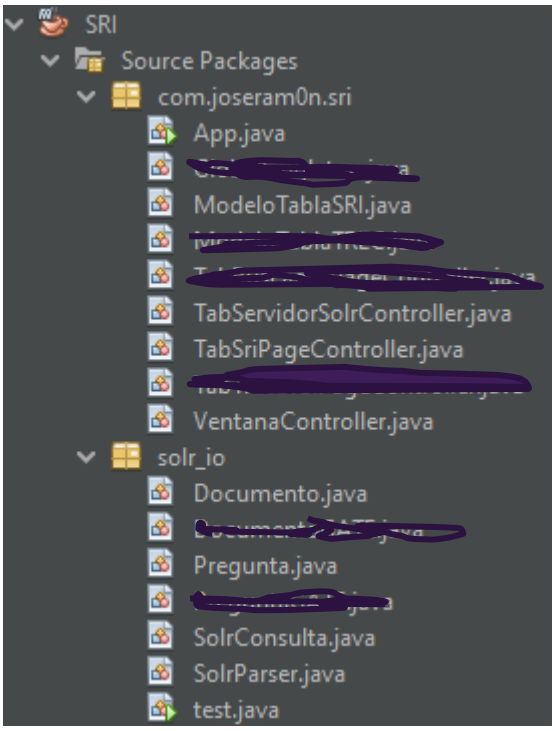


Ilustración 1 - Estructura

Sin entrar en mucho detalle en el código (para eso mejor echar un ojo a las fuentes) se puede apreciar en la ilustración 1 de una forma intuitiva, que función tiene cada .java del proyecto.

La clase SolrConsulta maneja todo lo relacionado a las queries en Apache Solr.

La clase SolrParser procesaba los documentos y las queries de búsqueda.

Los controles eran muy básicos, permitían encender/apagar el servidor e indexar y hacer solicitudes, además la evaluación de Trec Eval era necesaria hacerla a mano.

Nota: Las clases tachadas no estaban presentes en la primera versión.

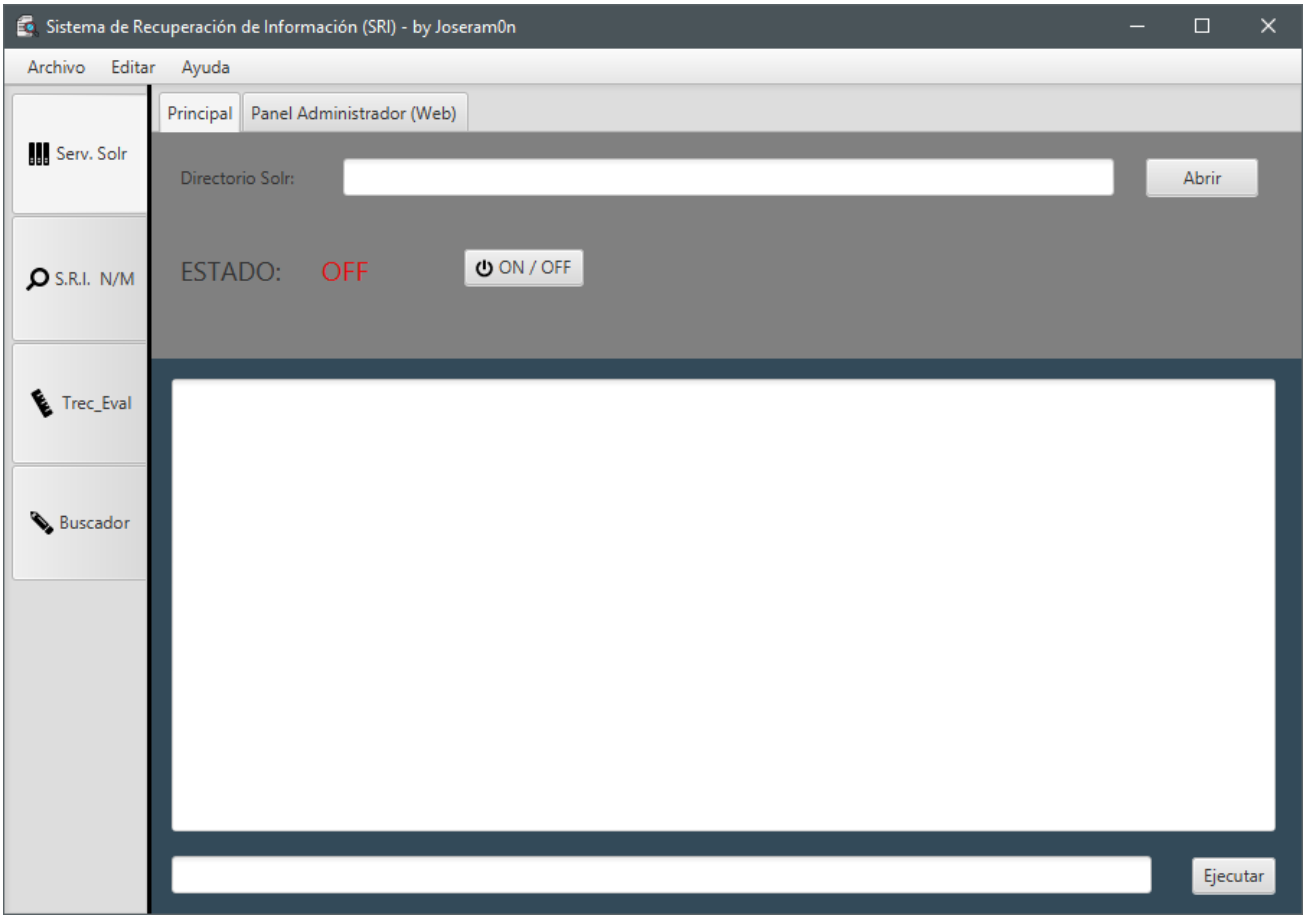


Ilustración 2 - Pantalla de Inicio

En la parte derecha identificamos cuatro pestañas principales, aunque en la primera versión solo las dos primeras, **Serv. Solr** y **S.R.I N/M** estaban disponibles.

En la primera tenemos dos subpestañas, debemos indicar la ruta de la carpeta de Apache Solr para poder manipular el servidor mediante procesos (Process) y en la otra una vez encienda el servidor jetty de Apache Solr, nos podremos conectar a su interfaz de administración.

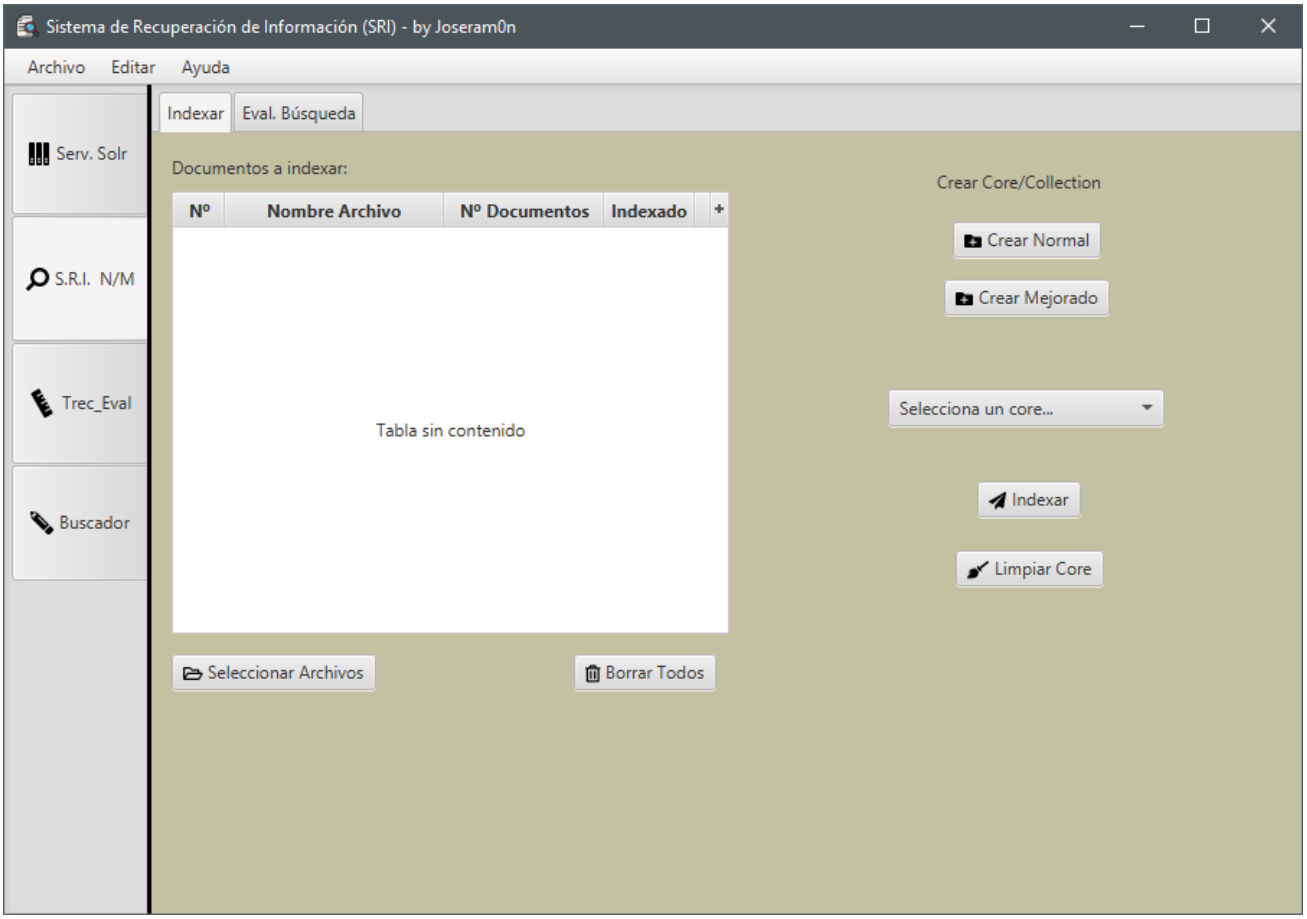


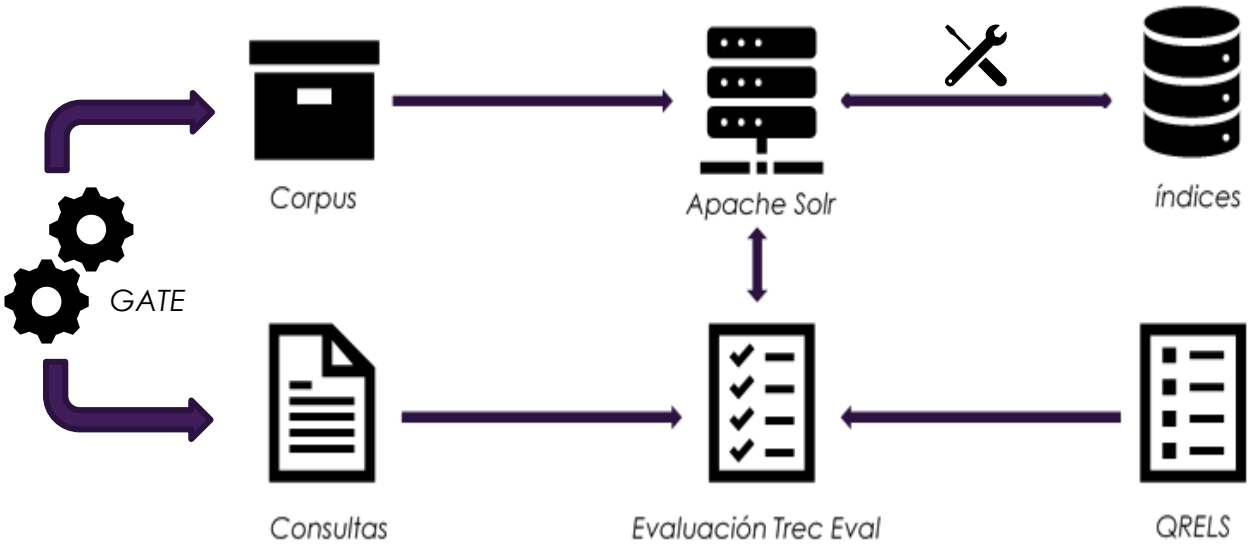
Ilustración 3 - Pestaña SRI

En la segunda pestaña principal tenemos acceso a dos subpestañas, una para indexar y otra para realizar las peticiones de búsquedas respectivamente.

En general la aplicación es más sencilla de lo que realmente parece, ya que en Apache Solr se está usando la configuración por defecto, lo único destacable que se le aplico a esta primera versión es un filtro de caracteres especiales para las queries de Apache Solr.

S.R.I. - Segunda versión

En esta versión realizamos lo mismo que en la primera, pero ahora configuramos Apache Solr para que indexar mejor los documentos y añadimos categorías con la ayuda del uso GATE.



Implementación de la Segunda Versión

Partiendo con la primera versión del proyecto como base decidí cambiar de sistema operativo (LINUX a WINDOWS10) y también la herramienta de construcción del proyecto, mas concretamente de ANT a **Maven** para facilitar la reconstrucción del proyecto al profesor o cualquier persona interesada en el mismo.

Una vez realizada la migración de entorno y la consecuente adaptación de código, se llevo a cabo una pequeña reconfiguración de Apache Solr para añadir nuevos campos a la hora de realizar las consultas y cambiar los field types a “text_en”, el cual nos brida mejores características como la aplicación de lematización y el uso de Stopswords.

Luego se usó **GATE** para etiquetar el corpus y las consultas con “organización” y “localización”.

Nota: Aquí se observó que por defecto con **ANNIE** una buena cantidad de las etiquetas con organización no era muy acertadas.

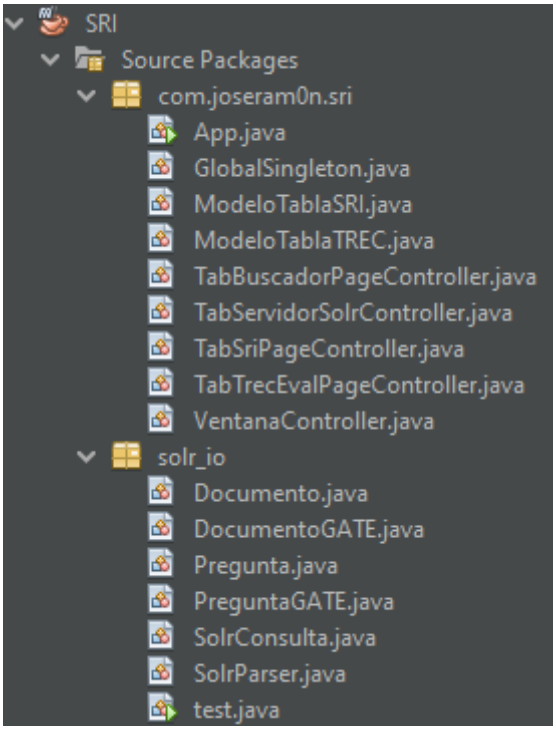


Ilustración 4 - Estructura Final

Al igual que en la primera version (Ilustracion 3) seguiemos teniendo los dos paquetes principales, pero ahora podemos ver (Ilustracion 4) que tenemos mas controladores, lo cual indica nuevas pestañas y en el paquete de **solr_io** ahora hay documentos y preguntas con las etiquetas de GATE.

Tambien se puede apreciar un nuevo modelo de tabla que se usara para la evaluacion y una clase singleton que contiene varables que pueden compartir los controladores entre si, quizas no sea la solucion mas sofisticada pero es funcional.

Los métodos que ahora requieran de los nuevos campos (etiquetas) también han sido adaptados.

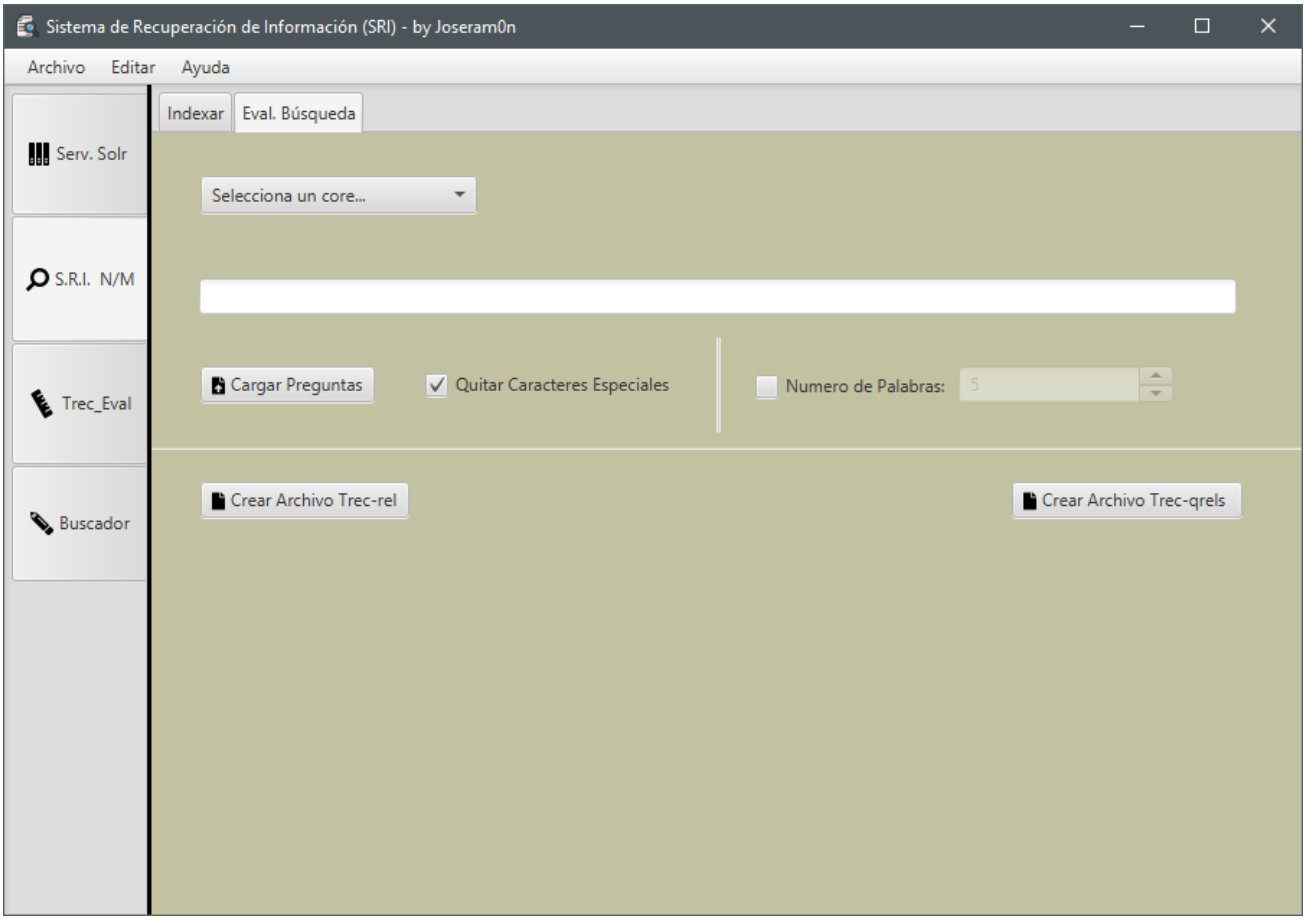


Ilustración 5 - SRI PARTE 2

En la Ilustración numero cinco podemos observar que puede cargar preguntas y crear los dos tipos de ficheros necesarios para la evaluación con la herramienta Trec Eval.

También faltó mencionar que en la subpestaña de indexar cuando creas un core mejorado, automáticamente se crean los nuevos campos con todas las modificaciones necesarias.

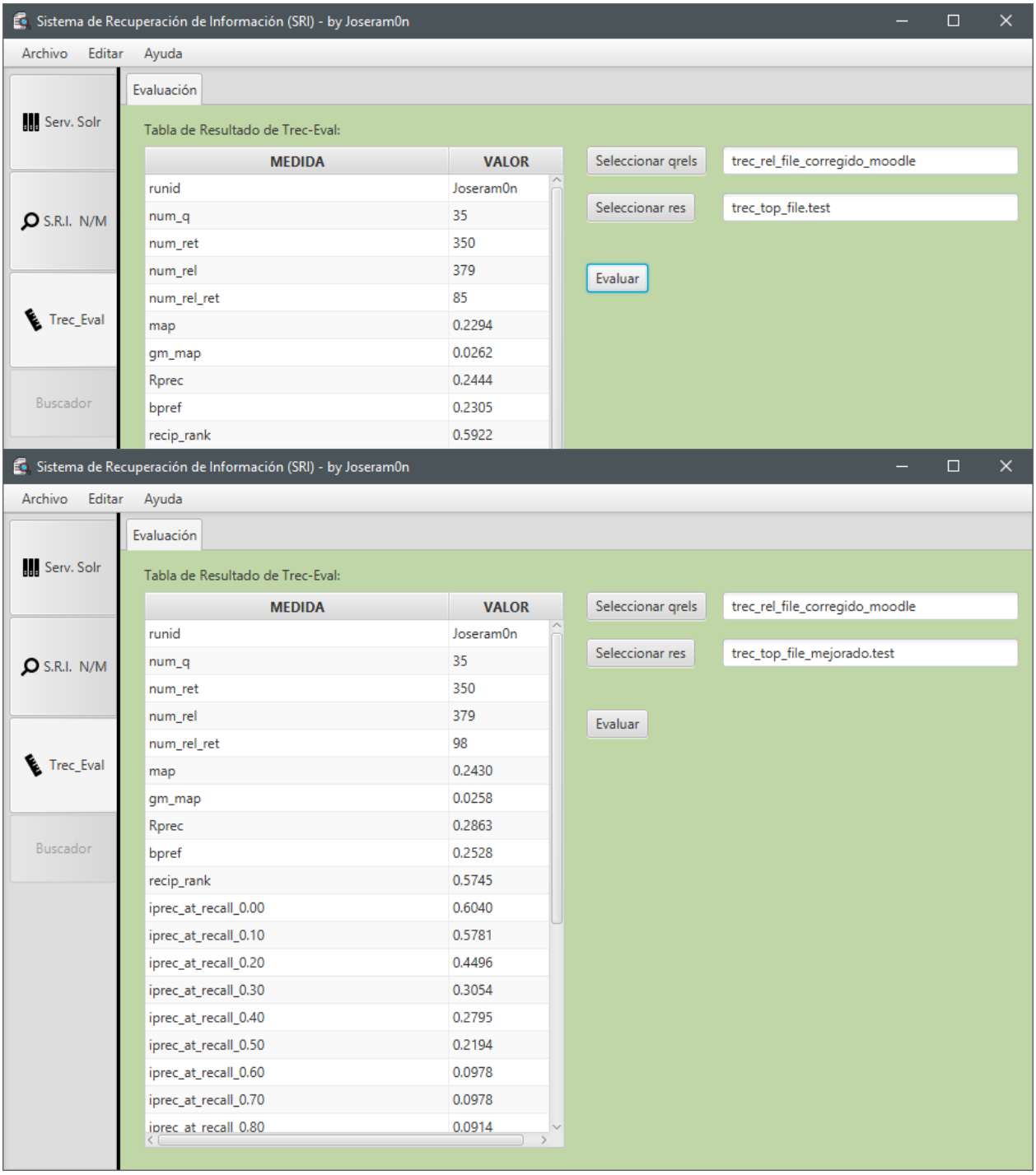


Ilustración 6 - Evaluación

La ilustración 6 corresponde a las pestañas de evaluación de Trec Eval, para ello se ha hecho uso de una wrapper externo que vincula la herramienta con java, de esta forma se puede hacer el ciclo completo dentro de la aplicación del proyecto.

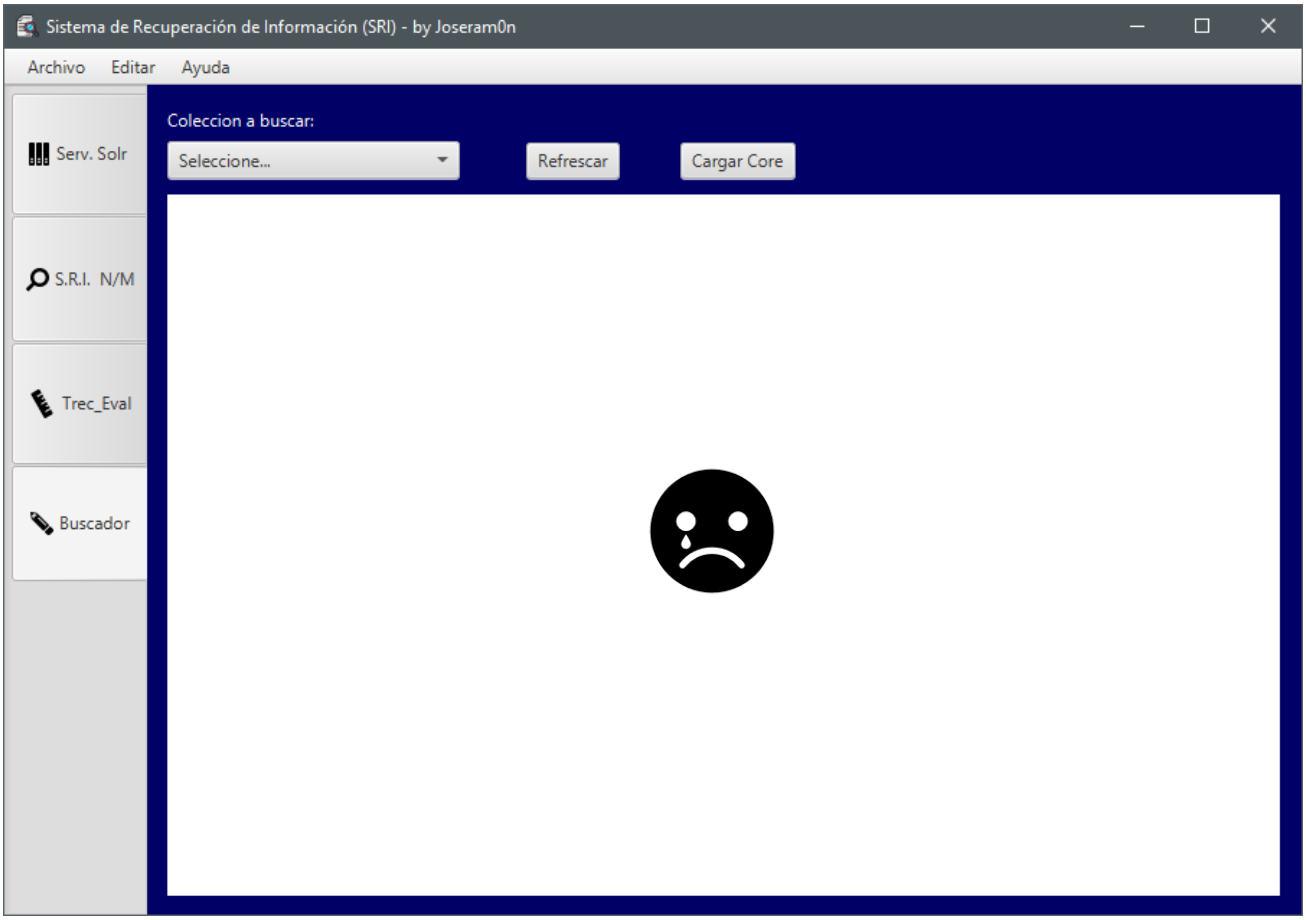


Ilustración 7 - Navegador *WIP*

La última de las 4 pestañas principales, esta pestaña es una pequeña adición al proyecto, se trata de un navegador web para los cores o colleciones. Lamentablemente bajo Windows 10 no ha sido posible su funcionamiento, a sique queda proyectado a futuro.

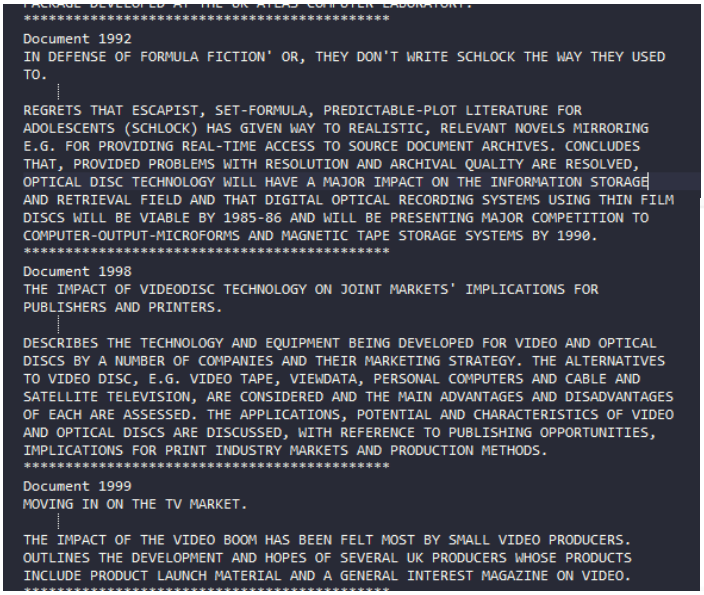
Resultados

En la ilustración 6 ya se veía un adelanto de los resultados obtenidos.

Resumidamente se puede verificar que tanto las mejoras en la configuración de Apache Solr como la etiquetación de GATE han conseguido mejorar un poco la evaluación de Trec Eval, pero debido a que el corpus y las consultas son muy pobres es complicado obtener buenas puntuaciones.

Documentos encontrados: Normal = 85 | Mejorado= 98

Cabe mencionar que faltan 5 documentos en el corpus y eso podría haber afectad “un poco” a los resultados.



Posibles mejoras

A pesar de que el corpus no es muy bueno aun se pueden retocar más parámetros tanto de Apache Solr como de GATE, pero la mejora sería muy pequeña.

Ejemplos de mejoras a realizar:

- Cambiar las Stopword list de Apache Solr.
- Modificar a ANNIE (GATE) para que reconozca mejor los tokens.

Valoración personal

Este ha sido un proyecto que personalmente me ha gustado mucho porque he podido poner en practica gran parte de los aprendido, no solo en esta asignatura sino un poco en general, debido a esto he disfrutado trabajando y espero que los conocimientos que he adquirido sean de utilidad en el futuro.

Por último, me gustaría mencionar que de haber tenido más tiempo es probable que me hubiese animado a hacer mi propio buscador con Apache Solr.

Gracias

Bibliografía

- https://stackoverflow.com/ *Respuestas a muchas preguntas de programación*
- https://lucene.apache.org/solr/guide/8_7/solr-tutorial.html
- *Documentos cedidos en la Moodle por el profesor.*