UNIVERSIDAD DE SEVILLA
ESCUELA TÉCNICA SUPERIOR DE INGENIERÍA INFORMÁTICA

COMPLEMENTOS DE BASES DE DATOS
(4º curso)

APACHE STANBOL

Asistencia para las bibliotecas

Manuel Fco. López Ruiz
Miguel Rodríguez Caballero

21 de mayo de 2017

Índice

Tabla de Comandos

Tabla de Imágenes

1. Introducción

1.1. Web semántica

La web semántica es un conjunto de actividades desarrolladas en la web, destinadas a la creación de tecnologías para publicar datos legibles por aplicaciones informáticas. Se basa en la idea de añadir metadatos semánticos y ontológicos a la web. Esa información adicional describe el contenido, el significado y la relación de los datos, que además debe ser proporcionada de manera formal, para que así sea posible evaluarlas automáticamente por máquinas de procesamiento. El objetivo de la web semántica es mejorar internet ampliando la interoperabilidad entre los sistemas informáticos usando "agentes inteligentes", los cuales son programas encargados de buscar información sin operadores humanos.

Se basa en dos puntos fundamentales. El primero de ellos es la descripción del significado, donde se articula la semántica, los metadatos y las ontologías, y el segundo, es la manipulación automática de estas descripciones, que se realiza mediante lógica y motores de inferencia.

- **La semántica**. Es el estudio del significado de los términos lingüísticos.
- **Los Metadatos.** Son datos que describen otros datos, en este contexto los datos que describen recursos de la web. La distinción entre datos y metadatos es relativa, pues depende de la aplicación. Los metadatos de una aplicación pueden ser los datos que maneja otra aplicación.
- **Las Ontologías**. Una ontología es una jerarquía de conceptos con atributos y relaciones, que define una terminología consensuada para definir redes semánticas de unidades de información interrelacionadas y con ciertas reglas.

La web semántica es una ampliación de la Web, por medio de la que se intenta realizar un filtrado de manera automática pero precisa de la información. Además, es necesario hacer que la información que anida en la web sea entendible por las propias máquinas. En concreto se atiende a su contenido, independientemente de la estructura sintáctica. A través de esta modalidad de web semántica se pueden obtener soluciones a problemas habituales en la búsqueda de información gracias a la utilización de una infraestructura o proceso común, mediante la cual, es posible compartir, procesar y transferir información de forma sencilla.

En la actualidad, la Web está basada principalmente en documentos escritos en HTML, un lenguaje de marcado que sirve para crear hipertexto y el cual es válido para adecuar el aspecto visual de un documento e incluir objetos multimedia, pero no para categorizar los elementos que configuran el texto más allá de las típicas funciones estructurales. La web semántica se ocuparía de resolver estas deficiencias. Para ello dispone de tecnologías de descripción de los contenidos, como RDF y OWL, además de XML, el lenguaje de marcado diseñado para describir los datos. Estas tecnologías se combinan para aportar descripciones explícitas de los recursos de la web, de forma que el contenido queda desvelado, como los datos de una base de datos accesibles por web, o las etiquetas inmersas en el documento (normalmente en XHTML, o directamente en XML, y las instrucciones de visualización definidas en una hoja de estilos aparte). Esas etiquetas permiten que los gestores de contenidos interpreten los documentos y realicen procesos inteligentes de captura y tratamiento de información.

1.2. XML

Es un metalenguaje que permite definir lenguajes de marcas desarrollado por el World Wide Web Consortium (W3C) utilizado para almacenar datos en forma legible. Aporta la sintaxis superficial para los documentos estructurados, pero sin dotarles de ninguna restricción sobre el significado.

1.3. XML Schema

XML Schema es un lenguaje de esquema utilizado para describir la estructura y las restricciones de los contenidos de los documentos XML de una forma muy precisa, más allá de las normas sintácticas impuestas por el propio lenguaje XML. Se consigue así una percepción del tipo de documento con un nivel alto de abstracción. Fue desarrollado por el World Wide Web Consortium (W3C).

1.4. RDF

Es un modelo de datos para los recursos y las relaciones que se puedan establecer entre ellos. Aporta una semántica básica para este modelo de datos que puede representarse mediante XML. Permite la interoperabilidad entre aplicaciones que intercambian información comprensible por la página web, para proporcionar una infraestructura que soporte actividades de metadatos.

El modelo de datos RDF es similar a los enfoques de modelado conceptual clásicos como entidad-relación o diagramas de clases, ya que se basa en la idea de hacer declaraciones sobre los recursos (en particular, recursos web) en forma de expresiones sujeto-predicado-objeto. Estas expresiones son conocidas como tripletas en terminología RDF. El sujeto indica el recurso y el predicado denota rasgos o aspectos del recurso y expresa una relación entre el sujeto y el objeto.

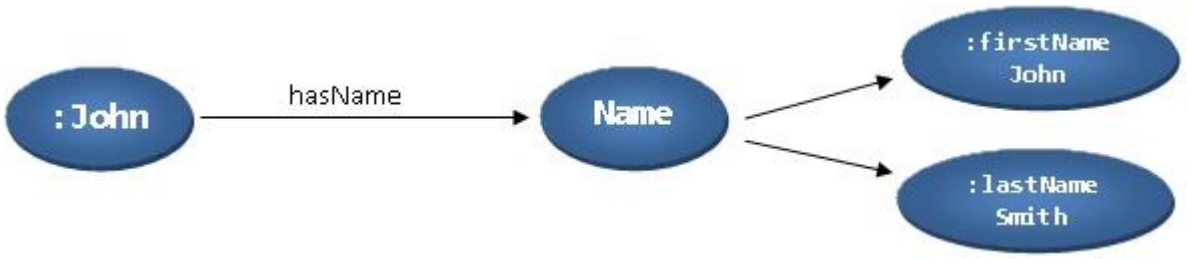

Imagen 1. Ejemplo SPARQL

1.5. RDFS

Es un vocabulario para describir las propiedades y las clases de los recursos RDF, con una semántica para establecer jerarquías de generalización entre dichas propiedades y clases.

1.6. OWL

Es un lenguaje para definir ontologías mediante la descripción detallada de propiedades y clases: tales como relaciones entre clases (p.ej. disyunción), cardinalidad (por ejemplo "únicamente uno"), igualdad, tipologías de propiedades más complejas, caracterización de propiedades (por ejemplo simetría) o clases enumeradas.

Tiene como objetivo facilitar un modelo de marcado construido sobre RDF y codificado en XML. También extiende RDFS para permitir la expresión de relaciones complejas entre diferentes clases RDFS, y obtener una mayor precisión en las restricciones de clases y propiedades específicas. Permite describir recursos para limitar las propiedades de clases con respecto a un número y tipo, los recursos para inferir qué elementos, con varias propiedades, son miembros de una clase en particular, los recursos para determinar si todos los miembros de una clase tendrán una propiedad en particular, o si puede ser que sólo algunos la tengan, los recursos para distinguir entre relaciones uno-a-uno, varios-a-uno o uno-a-varios, etc.

1.7. SPARQL

Es un lenguaje de consulta de conjuntos de datos RDF, siendo una tecnología clave en el desarrollo de la Web Semántica que se constituyó como Recomendación oficial del W3C. Además en dicha

especificación también se incluye un formato XML que detalla el modo en el que se estructuran los resultados obtenidos.

Al igual que sucede con SQL, es necesario distinguir entre el lenguaje de consulta y el motor para el almacenamiento y recuperación de los datos. Por este motivo, existen múltiples implementaciones de SPARQL, generalmente ligados a entornos de desarrollo y plataformas tecnológicas.

2. Objetivos a alcanzar

Los objetivos a alcanzar serán los siguientes:

- Conocer en profundidad el funcionamiento de la herramienta "Apache Stanbol".
- Realizar un ejemplo práctico con la ayuda de la herramienta usando los conocimientos adquiridos anteriormente.
- Proporcionar una explicación breve y concisa de la herramienta a mis compañeros mediante una presentación.

3. Tecnología – Apache Stanbol

3.1. ¿Qué es?

Apache Stanbol es una herramienta que proporciona un conjunto de componentes reutilizables para la gestión de contenido semántico. Esta herramienta pretende utilizar sistemas tradicionales de gestión de contenidos con servicios semánticos, así como otros casos de uso factibles como uso directo de aplicaciones web (por ejemplo, para extracción / sugerencia de etiquetas o finalización de texto en campos de búsqueda), flujos de trabajo de contenido inteligente o enrutamiento de correo electrónico basado en entidades extraídas, temas, etc.

Imagen 2. Relación Apache Stanbol con sistemas CMS

3.2. Componentes

Con el objetivo de ser utilizado como un motor semántico a través de sus servicios, todos los componentes ofrecen sus funcionalidades en términos de una API de servicio web RESTful. Dichos componentes serán explicados a continuación.

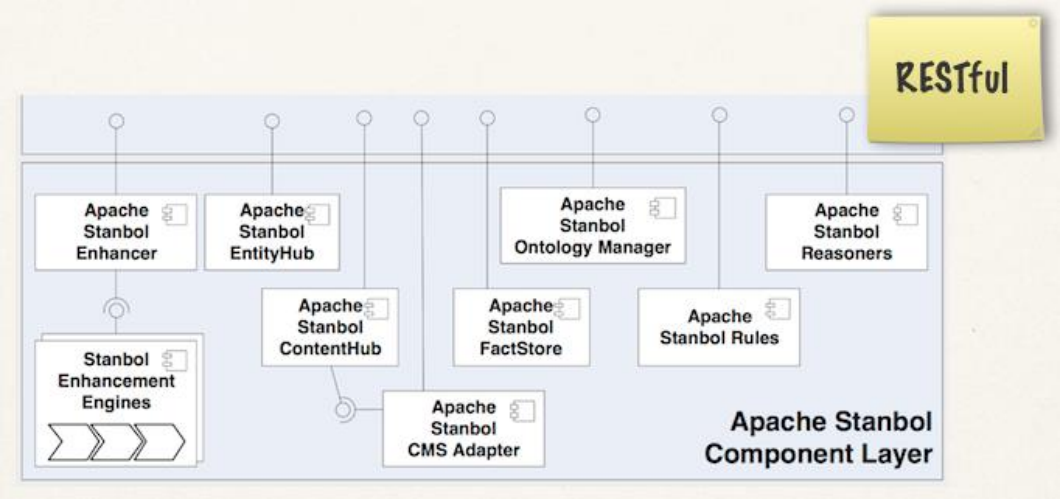

Imagen 3. Componentes Apache Stanbol

Los componentes no dependen uno del otro. Sin embargo, pueden combinarse fácilmente si es necesario, como se mostrará más adelante. Todos los componentes se implementan como paquetes, componentes y servicios OSGi.

3.2.1. Enhancer

Apache Stanbol Enhancer proporciona una API RESTful y Java que permite extraer características del contenido que se le ha suministrado. Con más detalle, el contenido pasado es procesado por los motores de mejora como se define por la llamada Enhancer Chain. Esta será la responsable de definir cómo será procesada la información suministrada al Stanbol Enhancer, indicando el motor de mejoras a usar, así como el orden en el que serán usados. Enhancer Chain no es el responsable del procesamiento de la información, solamente indica el plan de ejecución a seguir por el EnhancerJobManager para procesarla.

La siguiente figura proporciona una visión general tanto de la API RESTful, así como de la API de Java proporcionada por el Stanbol Enhancer.

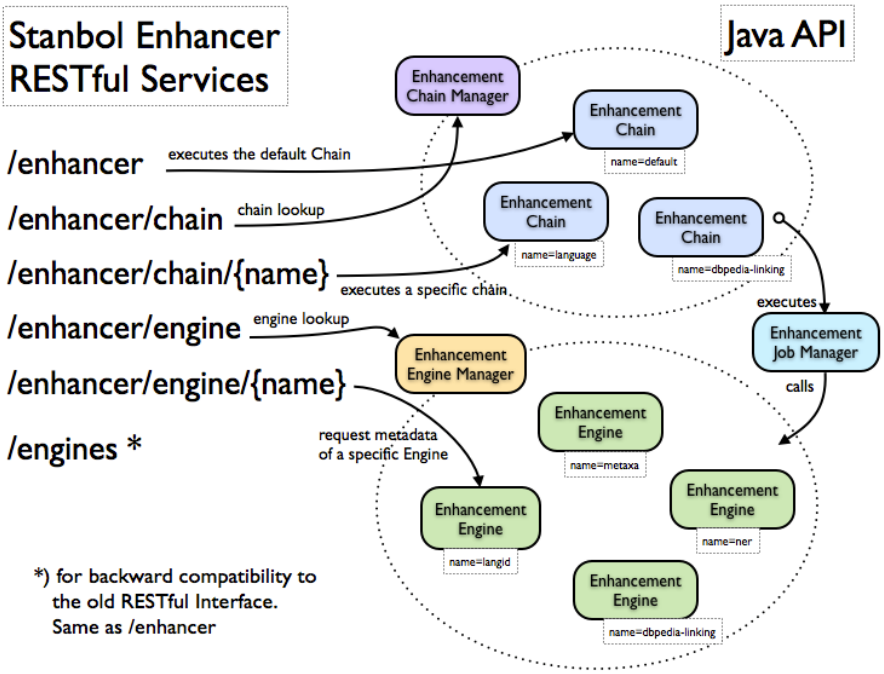

Imagen 4. Funcionamiento Apache Stanbol Enhancer

El contenido a analizar debe enviarse en una solicitud POST con el tipo mime especificado en el encabezado Content-type. A continuación, el contenido analizado se procesa mediante el Enhancer Chain elegido. La respuesta mantendrá la mejora RDF serializada en el formato especificado. Aquí podemos ver un ejemplo.

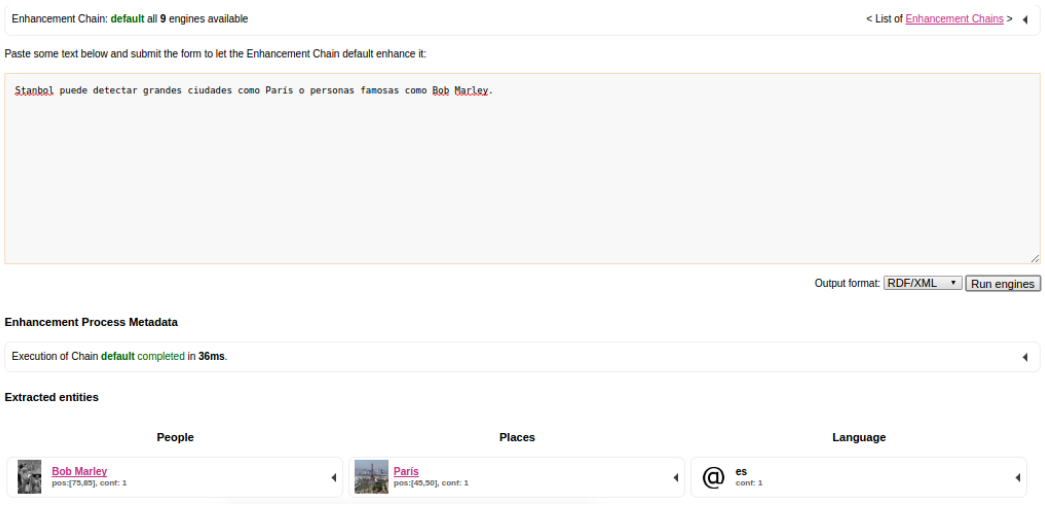

Imagen 5. Uso de Apache Stanbol Enhancer

Vemos como al introducir la frase “Stanbol puede detectar grandes ciudades como París o personas famosas como Bob Marley.” es capaz de detectar y recopilar la información sobre dicha ciudad o el cantante, además de detectar el idioma en el que está escrito el texto. Después de eso la herramienta nos proporciona los datos en el formato elegido, RDF/XML en este caso.

Raw RDF output

```
<rdf:RDF
    xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
    xmlns:j.0="http://purl.org/dc/terms/"
    xmlns:j.1="http://www.opengis.net/gml/"
    xmlns:j.2="http://www.w3.org/2003/01/geo/wgs84_pos#"
    xmlns:owl="http://www.w3.org/2002/07/owl#"
    xmlns:j.3="http://dbpedia.org/ontology/"
    xmlns:j.4="http://schema.org/"
    xmlns:j.5="http://stanbol.apache.org/ontology/entityhub/entityhub#"
    xmlns:rdfs="http://www.w3.org/2000/01/rdf-schema#"
    xmlns:j.6="http://fise.iks-project.eu/ontology/"
    xmlns:j.7="http://xmlns.com/foaf/0.1/" >
  <rdf:Description rdf:about="urn:enhancement-d7df9a03-9dd5-d111-2fcb-059f98a66f37">
    <j.6:confidence rdf:datatype="http://www.w3.org/2001/XMLSchema#double">0.9999944554430718</j.6:confidence>
    <j.6:extracted-from rdf:resource="urn:content-item-sha1-ffc021f4a2c1304f852ee5d4ad810b78a3f9577b"/>
    <j.0:created rdf:datatype="http://www.w3.org/2001/XMLSchema#dateTime">2017-05-18T14:42:18.547Z</j.0:created>
    <j.0:creator rdf:datatype="http://www.w3.org/2001/XMLSchema#string">org.apache.stanbol.enhancer.engines.langdetect.LanguageDetectionEnhancementEngine</j.0:creator>
    <j.0:language rdf:datatype="http://www.w3.org/2001/XMLSchema#string">es</j.0:language>
    <j.0:type rdf:resource="http://purl.org/dc/terms/LinguisticSystem"/>
    <rdf:type rdf:resource="http://fise.iks-project.eu/ontology/Enhancement"/>
    <rdf:type rdf:resource="http://fise.iks-project.eu/ontology/TextAnnotation"/>
  </rdf:Description>
  <rdf:Description rdf:about="urn:enhancement-1867abb9-68e7-dfdb-f60b-54ee48958705">
    <j.6:confidence rdf:datatype="http://www.w3.org/2001/XMLSchema#double">0.16072725704869648</j.6:confidence>
    <j.6:entity-label xml:lang="es">Metro de Paris</j.6:entity-label>
    <j.6:entity-reference rdf:resource="http://dbpedia.org/resource/Paris_M%C3%A9tro"/>
```

Imagen 6. Resultados de Apache Stanbol Enhancer

3.2.2. Entityhub

Entityhub es el componente de Stanbol responsable de proporcionar la información sobre las entidades relevantes para el dominio. Permite gestionar entidades locales así como entidades de importación desde “Sites” o definir asignaciones de Entidades locales a Entidades administradas por “Sites”. Una instancia de Apache Stanbol sólo puede tener un sólo Entityhub así que si qusiéramos administrar varios vocabularios habría que usar ManagedSite en su lugar. Un ManagedSite permite a los usuarios administrar una colección de entidades utilizando la API RESTful.

Por último una foto que resume el comportamiento de Entityhub.

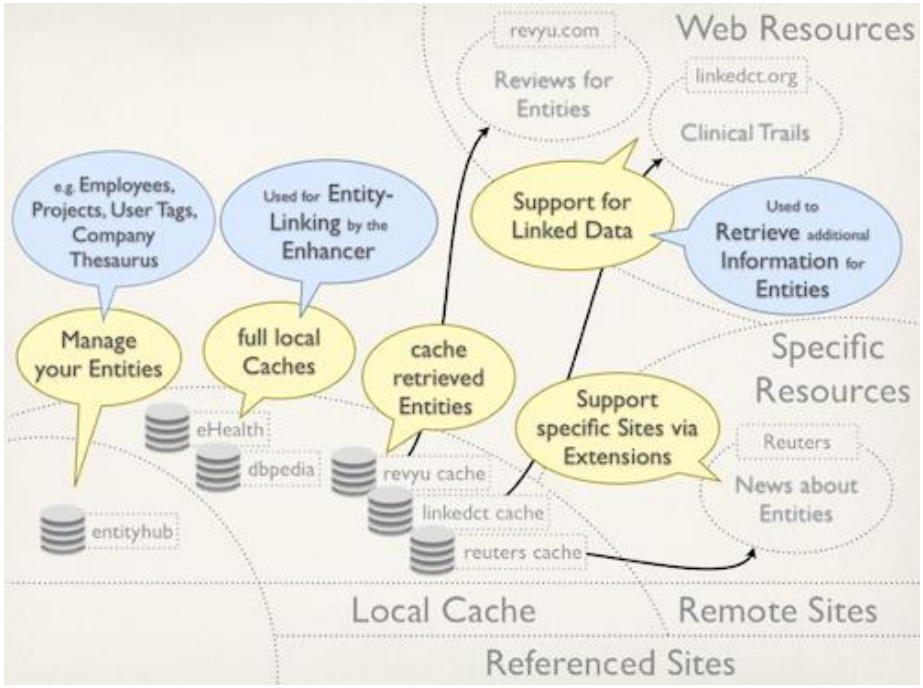

Imagen 7. Comportamiento Apache Stanbol Entityhub

3.2.3. Contenthub

Este módulo es un repositorio, que permite almacenar documentos basados en texto y recursos de búsquedas semánticas personalizables. Contenthub expone una API Java eficiente junto con los servicios RESTful correspondientes. Funciona de manera bastante similar al ya mencionado anteriormente Enhancer, con la diferencia de que Contenthub almacena los resultados de las búsquedas. Es importante destacar que está funcionalidad fue eliminada con el cambio de la versión 0.12 a la 1.0.

3.2.4. Ontology Manager

El Administrador de Ontologías de Apache Stanbol proporciona un entorno controlado para la gestión de ontologías, redes de ontología y sesiones de usuario para datos semánticos modelados a partir de ellos. Proporciona acceso completo a las ontologías almacenadas y poder administrar una red de ontología, lo cual significa que puede activar o desactivar partes de un modelo complejo de vez en cuando, para que sus datos puedan verse y clasificarse bajo diferentes directrices lógicas, funcionalidad que puede ser especialmente útil en operaciones de razonamiento. En resumen, estas son las ventajas del gestor de ontologías de Apache Stanbol:

- Instalar múltiples redes de ontologías simultáneamente, interconectando el conocimiento contenido en las mismas.
- Desactivación (dinámica) dinámica de partes de cualquier red de ontología, según lo requiera el razonamiento específico, la ejecución de reglas u otras tareas de procesamiento del conocimiento.
- Organizar ontologías en librerías, que pueden ser pobladas mediante la creación de simples gráficos RDF.

Toda la administración de las ontologías es llevada a cabo por OntoNet, el cual implementa la API para la gestión de OWL. Para hablar de OntoNet es necesario introducir los siguientes conceptos:

- Scope o ámbito: un "ámbito lógico" para todas las ontologías que abarcan un conjunto de conceptos relacionados con el propio gestor de contenidos. Los scopes nunca se heredan unos de otros, aunque pueden cargar las mismas ontologías si es necesario.
- Space o espacio: es un contenedor de acceso restringido a ontologías dentro de un scope. Las ontologías en un scope se cargan dentro de su conjunto de espacios. Un ámbito ontológico contiene:
    - Un espacio central, que contiene el conjunto inmutable de ontologías esenciales que describen el ámbito.
    - Un espacio personalizado (posiblemente vacío).
- Session o sesión: un contenedor de datos semánticos que necesitan ser relacionados con uno o más scopes, para la gestión de redes de ontología. Se puede utilizar para cargar instancias y razonar sobre ellos usando diferentes modelos (uno por scope).

La siguiente imagen ilustra el comportamiento de OntoNet.

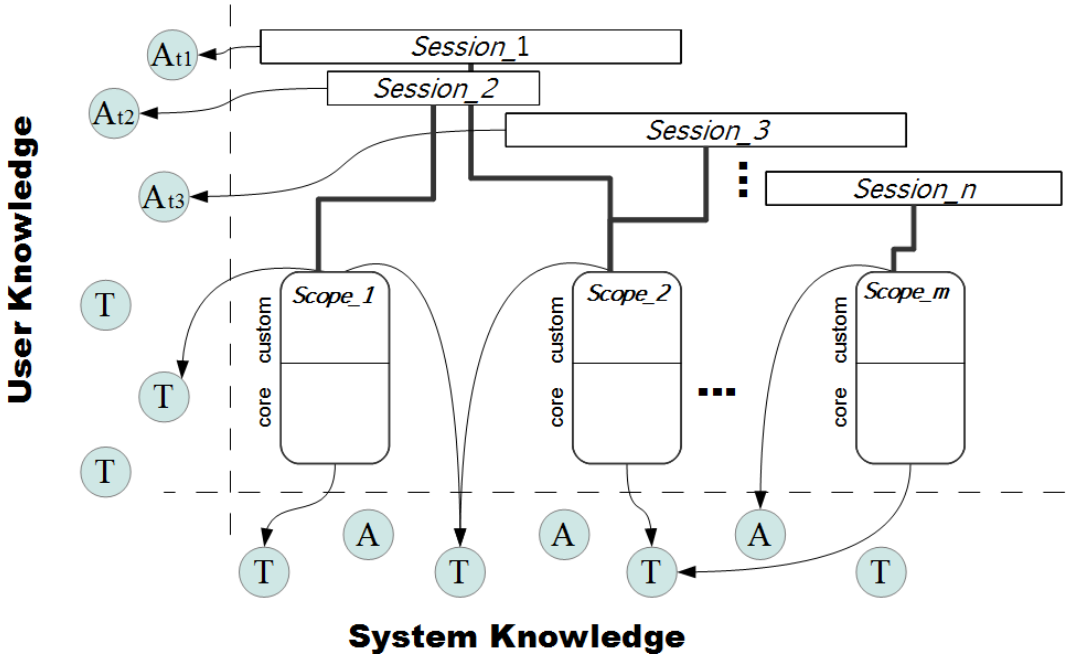

Imagen 8. Comportamiento Apache Stanbol OntoNet

3.2.5. Rules

Stanbol Rules es un componente que soporta la construcción y ejecución de reglas de inferencia. Una regla de inferencia, o regla de transformación, es una regla o función sintáctica que toma premisas y devuelve una conclusión. Este componente permite añadir una capa para expresar las lógicas de negocio mediante axiomas, que codifican las reglas de inferencia, y los cuales se pueden organizar en un contenedor llamado receta, que identifica un conjunto de reglas que comparten la misma lógica de negocio y las interpretan como un todo.

Gracias a Stanbol Rules el administrador puede definir controles de integridad para los datos obtenidos de fuentes externas para evitar formatos no deseados o datos inconsistentes. De esta manera, el administrador puede configurar su gestor de contenidos para filtrar la información recuperada a través del Enhancer, de tal forma que cumpla las restricciones impuestas por la lógica de negocio. La restricción puede ser definida por medio de reglas de Stanbol. Por ejemplo, en un gestor de contenidos que recopila conocimientos básicos sobre músicos, podría ser posible definir reglas de verificación de integridad que mantengan sólo las entidades que se escriben como músicos en DBpedia y que tengan una imagen asociada, un lugar de nacimiento y el instrumento reproducido. Se descarta cualquier otra entidad que no cumpla las restricciones.

3.2.6. Reasoners

El componente Stanbol Reasoners proporciona un conjunto de servicios que se aprovechan de los motores de inferencia automáticos. El módulo implementa una api común para los servicios de razonamiento, proporcionando la posibilidad de conectar diferentes reasoners y configuraciones en paralelo.

Cada Reasoner tiene tres funciones:

- Check: para realizar una comprobación de coherencia. Este servicio devuelve el estado HTTP 200 si los datos son consistentes, 204 de lo contrario, aunque actualmente no incluye una explicación del por qué la entrada es inconsistente.

- Classify: para materializar todas las declaraciones inferidas de rdf: type.
- Enrich: para materializar todas las declaraciones inferidas.

3.2.7. CMS Adapter

El componente Adaptador CMS actúa como un puente entre los sistemas de gestión de contenido y Apache Stanbol. Hay que tener en cuenta que todos los componentes de Apache Stanbol también ofrecen servicios RESTful que permiten acceder directamente desde el exterior. El adaptador interactúa con los sistemas de gestión de contenidos a través de las especificaciones JCR y CMIS. En otras palabras, cualquier repositorio de contenido compatible con las especificaciones JCR o CMIS puede hacer uso de las funcionalidades del adaptador CMS. Hay dos funcionalidades principales que ofrece CMS Adapter: "Bidirectional Mapping" y "Contenthub Feed".

- Bidirectional Mapping: permite que los sistemas de gestión de contenido representen su estructura en formato RDF. Esto ayuda a crear servicios semánticos encima de los sistemas de administración de contenido existentes usando su representación RDF.
- Contenthub Feed: la función de mapeo bidireccional hace posible explotar datos enlazados abiertos, que ya están disponibles en la web, en sistemas de administración de contenido. Aparte de los datos ya disponibles en la web, cualquier dato RDF puede ser asignado al repositorio de contenido. Mediante la asignación de datos RDF externos, los elementos del repositorio de contenido existentes se pueden actualizar o se pueden crear otros nuevos.

Igual que ocurre con el componente Contenthub, es importante destacar que CMS Adapter fue eliminado con el cambio de la versión 0.12 a la 1.0.

3.3. Instalación y mantenimiento

La instalación de Apache Stanbol se realizó en un contenedor tipo Docker[1]. De esta forma, cualquier usuario podría usar la herramienta fácilmente, ejecutando el siguiente comando en la terminal:

```
docker run -d \
    --name stanbol \
    -p 8080:8080 \
    manuellr/apache-stanbol
```

Comando 1. Primera ejecución del contenedor de Apache Stanbol

Tras la ejecución del comando, sólo sería necesario acceder mediante un navegador web a http://localhost:8080 .

Dicho contenedor ha sido creado por los miembros del grupo. Su funcionamiento básico en la etapa de compilación es descargar el código de Apache Stanbol de su página de descarga, realizar la instalación con maven, generándose así el archivo ejecutable en formato “jar”. Este archivo se ejecutará en cada cliente al iniciarse el contenedor. Se puede encontrar el código fuente y alguna información adicional en GitHub[2] y DockerHub[3].

[1] https://www.docker.com/
[2] https://github.com/ManuelLR/docker-apache-stanbol
[3] https://hub.docker.com/r/manuellr/apache-stanbol/

3.3.1. Comandos básicos para usar el contenedor

Si se desea detener el contenedor (lo que a su vez detiene a Apache Stanbol) o volver a iniciarlo será necesario ejecutar los siguientes comandos:

```
docker stop stanbol
```

Comando 2. Detener contenedor

```
docker start stanbol
```

Comando 3. Iniciar contenedor

En caso de ser necesario ver los logs, deberá mirar dentro del contenedor, en la carpeta "/var/log/supervisord/". Un ejemplo para visualizarlos sería ejecutar el siguiente comando mientras el contenedor se encuentra en ejecución:

```
docker exec stanbol \
    cat /var/log/supervisord/stanbol_stdout.log
```

Comando 4. Ejemplo visualización de logs Stanbol

Si por cualquier otro motivo fuera necesario conectarse a la máquina podríamos obtener una sesión de terminal ejecutando el siguiente comando:

```
docker exec -ti stanbol bash
```

Comando 5. Conectarse mediante terminal al contenedor

Para obtener más información acerca de Docker y los posibles comandos puede acceder a su documentación oficial alojada en docs.docker.com.

4. Ejercicio

Para aprender a usar la herramienta en profundidad hemos realizado un ejercicio práctico, en el que conseguiremos detectar los ISBN de un texto dado. Esto sería útil, por ejemplo en las bibliotecas, cuando se realizan peticiones de libros, con el fin de obtener información de los mismos rápidamente.

Cabe destacar que este ejemplo se ha basado en uno hallado en github, debido a la complejidad de Apache Stanbol y a la pobre documentación. Se puede encontrar más información al respecto en el repositorio original ogrisel/stanbol-isbn o en el nuestro, donde se ha subido todo el código ManuelLR/apache-stanbol-example.

El ejercicio lo subdividiremos en apartados para facilitar el entendimiento de las acciones a realizar.

4.1. Arrancar Apache Stanbol

Lo primero que debemos hacer es iniciar Apache Stanbol en nuestro equipo. Para ello ejecutaremos el "Comando 1. Primera ejecución del contenedor de Apache Stanbol" y, al descargar la imagen, nos mostrará el Id del contenedor. Podremos comprobar que el contenedor está en funcionamiento al acceder mediante el navegador web a localhost:8080. Tras unos segundos visualizaremos lo siguiente:

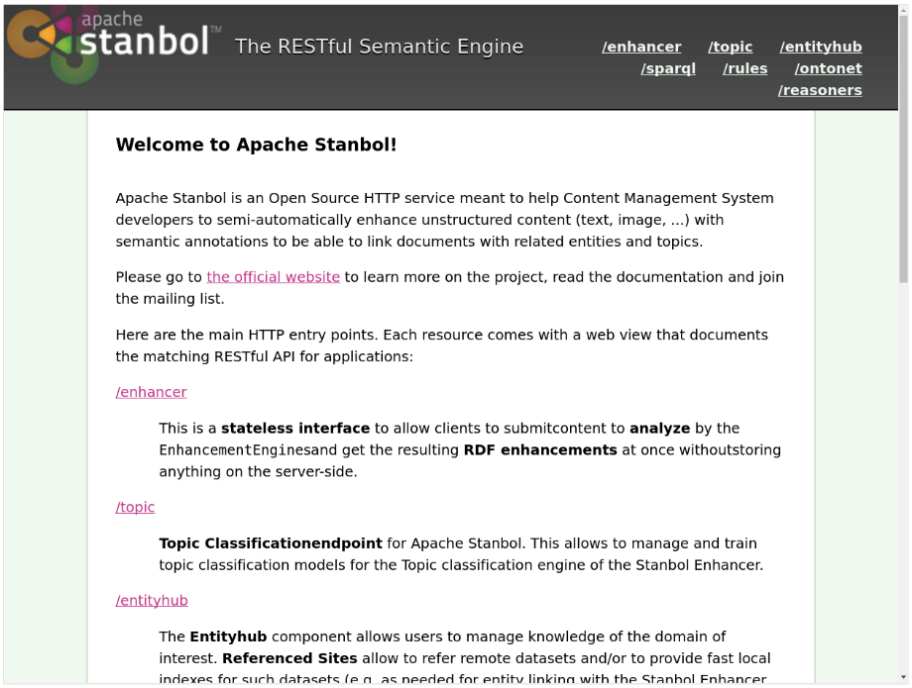

Imagen 9. Pantalla de inicio Apache Stanbol

Es posible que inicialmente no veamos todas las opciones. Esto se debe a que todavía no se han terminado de cargar todos los componentes. Sólo tendremos que actualizar la página y tras unos segundos aparecerán todos.

4.2. Apartado 1. Cargar una base de datos de ISBNs

Lo primero que debemos hacer para dar uso a nuestro sistema es cargar los datos iniciales, y para ello, será necesario obtener la herramienta de indexado que nos proporciona Apache Stanbol, por lo que descargaremos la última versión de Apache Stanbol, disponible en su web oficial. Una vez descargado lo descomprimiremos en una carpeta del equipo. Tras esto, deberemos acceder a la carpeta y compilar el proyecto con el siguiente comando:

```
mvn clean install -DskipTests
```

Comando 6. Compilación Apache Stanbol

Tras la compilación crearemos una carpeta de trabajo en otra ruta del equipo y copiaremos el archivo jar generado en la siguiente ruta de la compilación "entityhub/indexing/genericrdf/target/org.apache.stanbol.entityhub.indexing.genericrdf-*.jar" a nuestra carpeta de trabajo.

Para generar el proyecto con el que debemos trabajar ejecutaremos el siguiente comando en nuestra carpeta de trabajo:

```
java -jar org.apache.stanbol.entityhub.indexing.genericrdf-*.jar init
```

Comando 7. Creación proyecto para indexado

Ahora procederemos a configurar la herramienta de indexado. Para ello accederemos a "indexing/config" y modificaremos los archivos "indexing.properties" y "mappings.txt". En el primero, se deben definir algunas variables como el nombre del filtro, la descripción o la fuente con la que mapear los objetos. En el segundo, definiremos el mapeo, es decir, el ISBN. Se pueden encontrar estos archivos en nuestro repositorio. También tendremos que añadir la colección de objetos que deseamos que tenga indexado en local a la carpeta "indexing/resources/rdfdata". Estos datos podremos encontrarlo en la siguiente carpeta del repositorio.

Una vez que tenemos todo configurado procederemos a crear el índice. Para ello ejecutaremos en la carpeta de trabajo el comando:

```
java -Xmx1024m -jar org.apache.stanbol.entityhub.indexing.genericrdf-
*.jar index
```

Comando 8. Indexado del proyecto

Tras esto se habrán generado dos archivos en la carpeta "indexing/dist". Un ZIP, que contiene los objetos del índice, y un "jar". Para usarlos en nuestra instancia de Apache Stanbol debemos copiar primero el ZIP y posteriormente el "jar", puesto que al invertir el orden deberíamos reiniciar Apache Stanbol, con el fin de que detecte los objetos del índice. Para copiar el ZIP al contenedor deberemos ejecutar el siguiente comando:

```
docker cp indexing/dist/*.solrindex.zip stanbol:/stanbol/datafiles
```

Comando 9. Copiar objetos del índice al contenedor

Para subir el "jar" a nuestra instancia de Apache Stanbol deberemos acceder al menú de gestión oculto mediante el siguiente enlace localhost:8080/system/console/bundles. Es posible que nos solicite credenciales de acceso, en cuyo caso serán:

- Usuario: admin
- Contraseña: admin

Una vez hemos accedido al sistema, accederemos a la sección "OSGi">"Bundles" donde seleccionaremos la opción "Install/Update", la cual nos abrirá una ventana en la que deberemos seleccionar "Start Bundle" y el "jar" que se generó en la carpeta "indexing/dist" previamente.

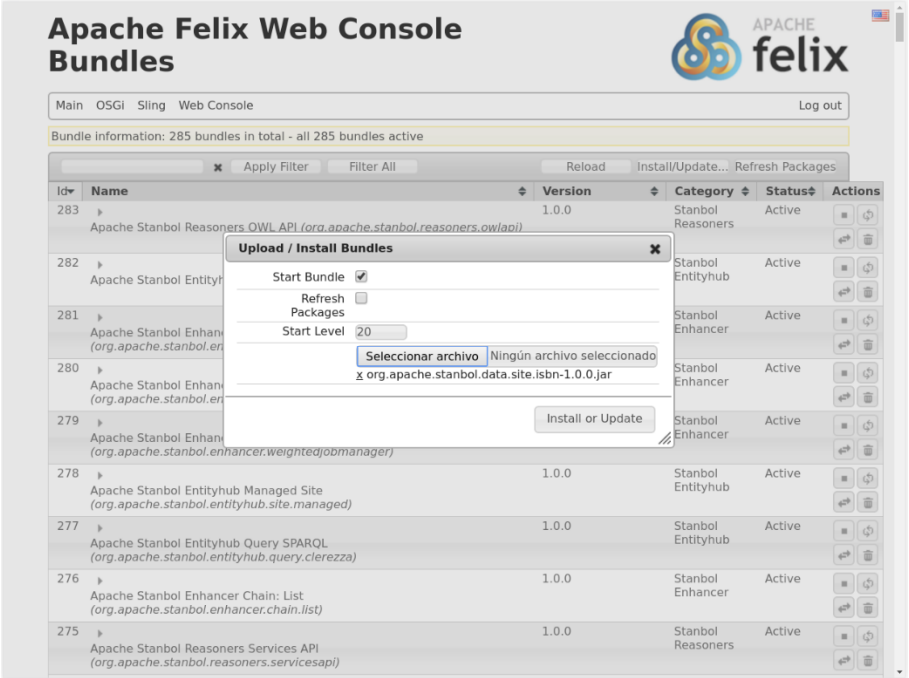

Imagen 10. Añadiendo nuevo indexado de sitio

De esta forma ya hemos añadido nuestros datos indexados a la instancia de Apache Stanbol. Dichos datos son considerados un “Site” de Stanbol y son accesibles mediante la sección EntityHub. Para comprobar que efectivamente funcionan como es debido podremos ejecutar el siguiente comando y comprobar su resultado:

```
curl "http://localhost:8080/entityhub/site/isbn/entity?id=http://dbpedia.org/resource/Mockingjay"
```

Comando 10. Comprobación correcto funcionamiento "site"

4.3. Apartado 2. Enlazar los ISBN con el “Keyword Linking Engine”

Una vez que tenemos los datos cargados en un "Site”, queremos que estos sean reconocidos mediante el escáner de contenidos Enhancer. Para ello, deberemos enlazar ese “Site” con un procesador de contenidos, concretamente con “Keyword Linking Engine”. Para lo cual, volveremos a acceder al panel oculto de configuración mediante el siguiente enlace localhost:8080/system/console/configMgr, y en la sección “OSGi”>”Configuration”, buscaremos la opción “Apache Stanbol Enhancer Engine: Keyword Linking” añadiendo otro mediante el botón a su derecha “+”.

Ahora rellenaremos los campos con las siguientes opciones (dejando igual el resto):

- Name: *isbn_keyword_linking*
- Referenced Site: *isbn*
- Label Field: *dbp-ont:isbn*
- Redirect Mode: *Follow Redirect*

Quedando la configuración así:

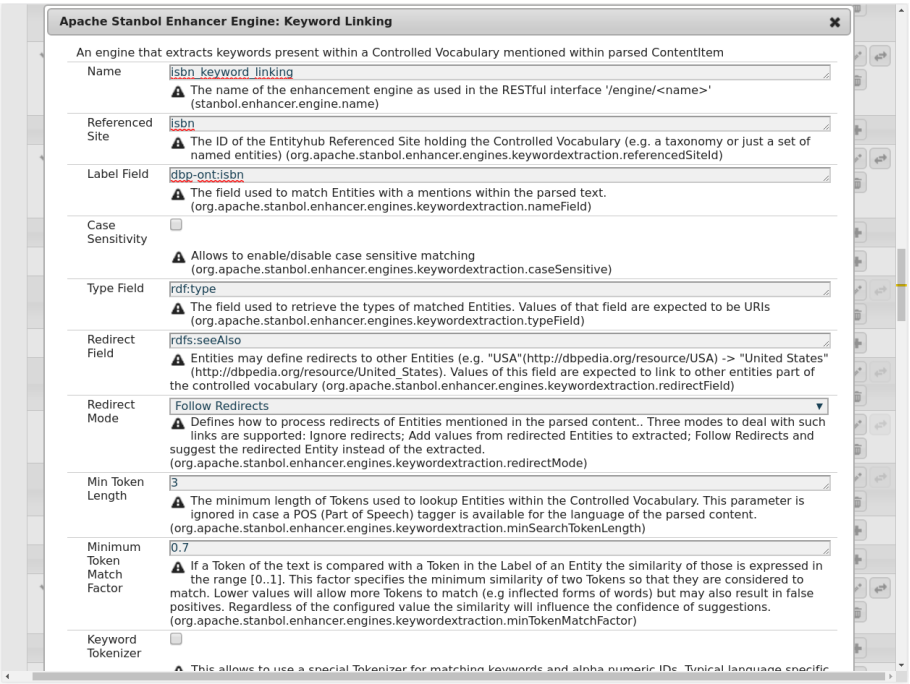

Imagen 11. Configuración "Keyword Linking Engine"

A continuación, enlazaremos este motor con un “Chain”. Este “Chain” le indicará al Enhancer los pasos a seguir para procesar el contenido. Para crearlo volveremos a acceder al panel oculto de configuración mediante el siguiente enlace localhost:8080/system/console/configMgr, y en la sección “OSGi”>”Configuration”, buscaremos la opción “Apache Stanbol Enhancer Chain: List Chain” añadiendo otro mediante el botón a su derecha “+”.

Ahora rellenaremos los campos con las siguientes opciones (dejando igual el resto):

- Name: *isbn_listChain*
- Engines: *isbn_keyword_linking*

Quedando la configuración así:

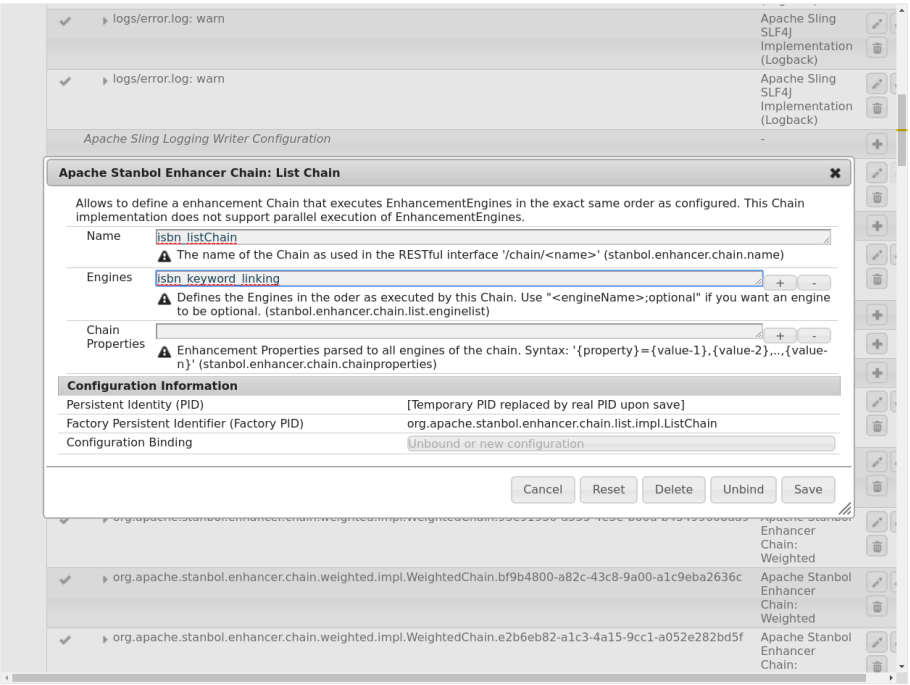

Imagen 12. Configuración "List Chain Enhancer"

Tras esto, podremos acceder a la sección Enhancer de nuestra instancia de Apache Stanbol y seleccionar entre los posibles "Enhancement Chains" el que acabamos de crear (o mediante el siguiente enlace). Ahí podremos introducir un ISBN de los contenidos en nuestro "Site" y será encontrado, como muestra el siguiente ejemplo:

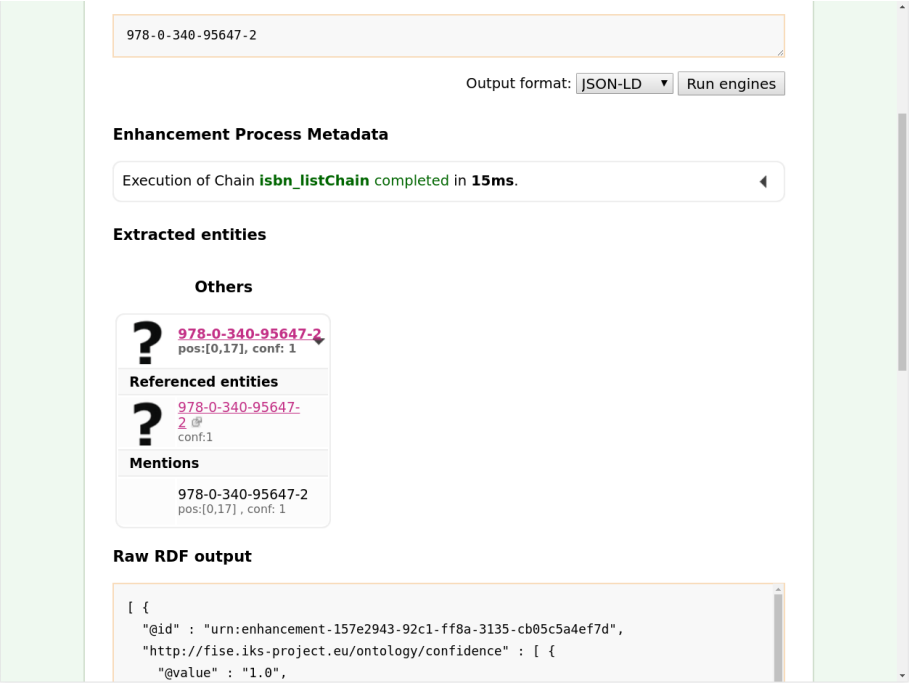

Imagen 13. Funcionamiento Apartado 2 del Ejemplo

4.4. Apartado 3. Detectar ISBN no cargados en nuestra base de datos

En este apartado vamos a crear un motor de reconocimiento ("Enhancement Engine") personalizado, con el cual detectaremos los ISBN, debido a que tienen una forma única. Para ello, usamos un proyecto base adaptado y actualizado a nuestras necesidades que se puede descargar aquí. Una vez descargado, accederemos a él y ejecutaremos el siguiente comando:

```
mvn install
```

Comando 11

En caso de que deseemos usarlo desde eclipse, mediante la función de importación de proyecto de maven, deberemos ejecutar además este comando antes de importarlo a eclipse:

```
mvn eclipse:eclipse
```

Comando 12. Adaptar proyecto maven a Eclipse

En este proyecto definiremos el pattern a buscar, qué hacer cuando lo encuentra y como procesarlo posteriormente. Podemos encontrar el proyecto debidamente completado aquí.

Una vez que hemos rellenado todo lo necesario será el momento de importarlo a nuestra instancia de Apache Stanbol. Para ello ejecutaremos el siguiente comando:

```
mvn install -DskipTests -PinstallBundle \
    -Dsling.url=http://localhost:8080/system/console
```

Comando 13. Instalación motor de reconocimiento personalizado

Con esto ya podríamos encontrarlo en el panel oculto de configuración, mediante el siguiente enlace localhost:8080/system/console/configMgr y buscando por el nombre definido en el archivo "metatype.properties" en la sección resources del proyecto que acabamos de instalar en la instancia de Apache Stanbol. Al darle a editar podremos ver el nombre que le ha puesto al motor. Lo guardaremos, en nuestro caso fue "isbn-detector".

Finalmente, para comprobar que efectivamente funciona como se espera, lo enlazaremos al "Chain" definido previamente. Para eso accederemos nuevamente al panel oculto de configuración mediante el siguiente enlace localhost:8080/system/console/configMgr y, en la sección "OSGi">"Configuration", buscaremos la opción "Apache Stanbol Enhancer Chain: List Chain". Justo debajo podremos observar el "List Chain" creado en el *Apartado 2. Enlazar los ISBN con el "Keyword Linking Engine"*, por lo que procedemos a editarlo pulsando sobre el lápiz y, en la opción engines, añadimos otro pulsando sobre el símbolo "+" con el nombre anteriormente apuntado ("isbn-detector" en nuestro caso). Quedaría de la siguiente forma:

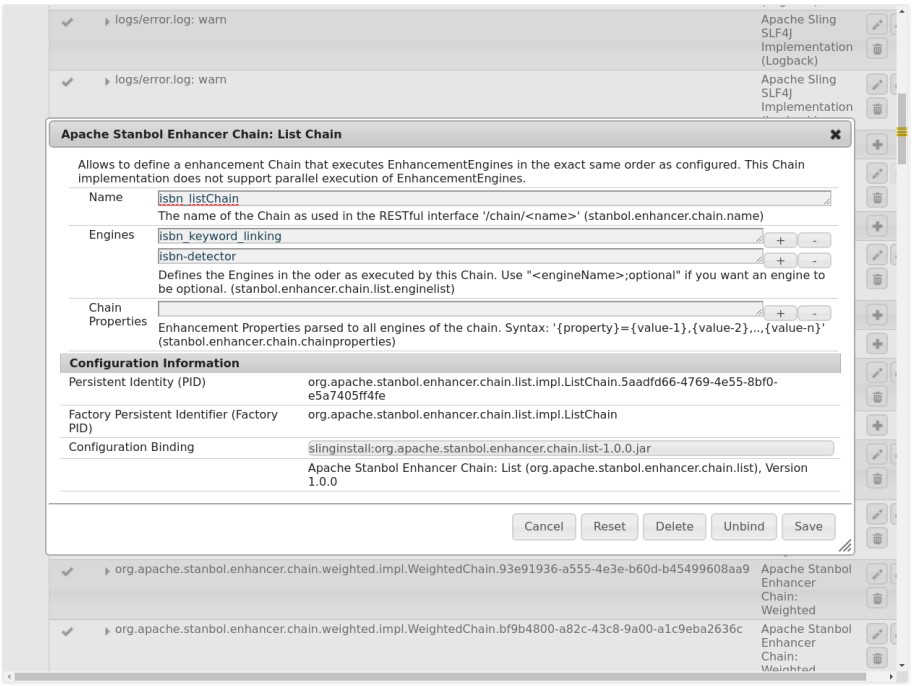

Imagen 14. Configuración 2 "List Chain Enhancer"

Ahora podremos acceder a la sección Enhancer de nuestra instancia de Apache Stanbol y seleccionar entre los posibles "Enhancement Chains" el que acabamos de modificar (o mediante el siguiente enlace). Ahí podremos introducir un ISBN no contenido en nuestro "Site" y será detectado como muestra el siguiente ejemplo:

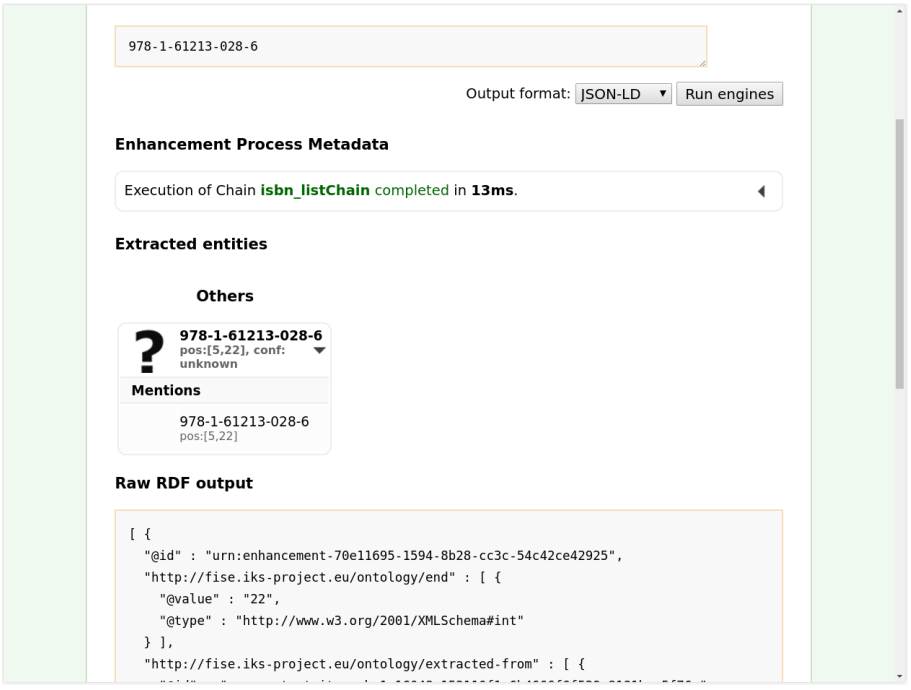

Imagen 15. Funcionamiento Apartado 3 del Ejemplo

5. Conclusiones

Las conclusiones las podemos dividir en dos grandes grupos, positivas y negativas. Entre las positivas podemos destacar que:

- Pensamos que lo que pretende realizar Apache Stanbol es una buena idea, puesto que pretende usar la Web Semántica para enriquecer la información que ya poseemos.
- Es aplicable a muchos campos no quedando limitado a un único ámbito, debido a su gran modularidad. Por ejemplo, se puede unir con algoritmos de inteligencia artificial para, tras leer historiales clínicos, predecir que enfermedad podría tener el paciente o, como en nuestro caso, realizar una búsqueda de información partiendo de un determinado ISBN.
- Es Software Libre, pudiendo así ser desarrollado y mantenido por una comunidad. Esto no impide que también sea mantenido por una empresa (como es el caso), pero sí que permite que el código sea auditado por cualquier persona.

En contra podemos destacar los siguientes aspectos:

- La documentación publicada en su página oficial es pésima. En otras palabras, demasiados tecnicismos, da por hecho demasiados conceptos de la aplicación, no sigue un orden lineal, las definiciones de términos o componentes no son claras, puesto que se referencian entre ellas para definirse, o hacen referencias a documentación antigua, donde se explican componentes que han sido modificados, sin que en ningún momento se haya actualizado la documentación para explicar su nuevo funcionamiento. Además, al no existir otra documentación en internet, es la única documentación en la que es posible basarse para entender y usar la aplicación.
- La aplicación es poco intuitiva. Existen menús ocultos de los que no se deja constancia en la documentación y que son muy necesarios, al igual que tampoco está orientada a un usuario final puesto que es muy técnica.
- No existe ningún tipo de soporte. Hay lugares habilitados para hacer preguntas o reportar errores pero nunca se responden o corrigen. Además no hay actualizaciones ni movimiento de código en su repositorio.
- Tampoco existe comunidad, puesto que no hay movimiento de ningún tipo, ya que no se responden preguntas, no hay ejemplos de uso publicados, ni tampoco existe código en plataformas de código como GitHub, donde hagan uso de la herramienta.
- Existen muchos enlaces y recursos caídos. En la documentación de la herramienta hay enlaces a páginas webs que ya no tienen esa información o que no existen. Por ejemplo, otras webs enlazan a un antiguo ejemplo centrado en la detección de enfermedades mediante el procesado de historiales clínicos, pero ya no está disponible. Pasa lo mismo con la descarga de algunos componentes o similares.

6. Bibliografía

- M. T. Gómez, “La Web Semántica”, Presentación de clase, Dep. LSI, Univ. Sevilla, Sevilla, 2016-2017.
- https://es.wikipedia.org/wiki/Web_sem%C3%A1ntica
- https://es.wikipedia.org/wiki/Extensible_Markup_Language
- https://es.wikipedia.org/wiki/XML_Schema
- https://es.wikipedia.org/wiki/Resource_Description_Framework
- https://es.wikipedia.org/wiki/RDF_Schema
- https://es.wikipedia.org/wiki/OWL
- https://es.wikipedia.org/wiki/SPARQL
- https://stanbol.apache.org/docs/trunk/