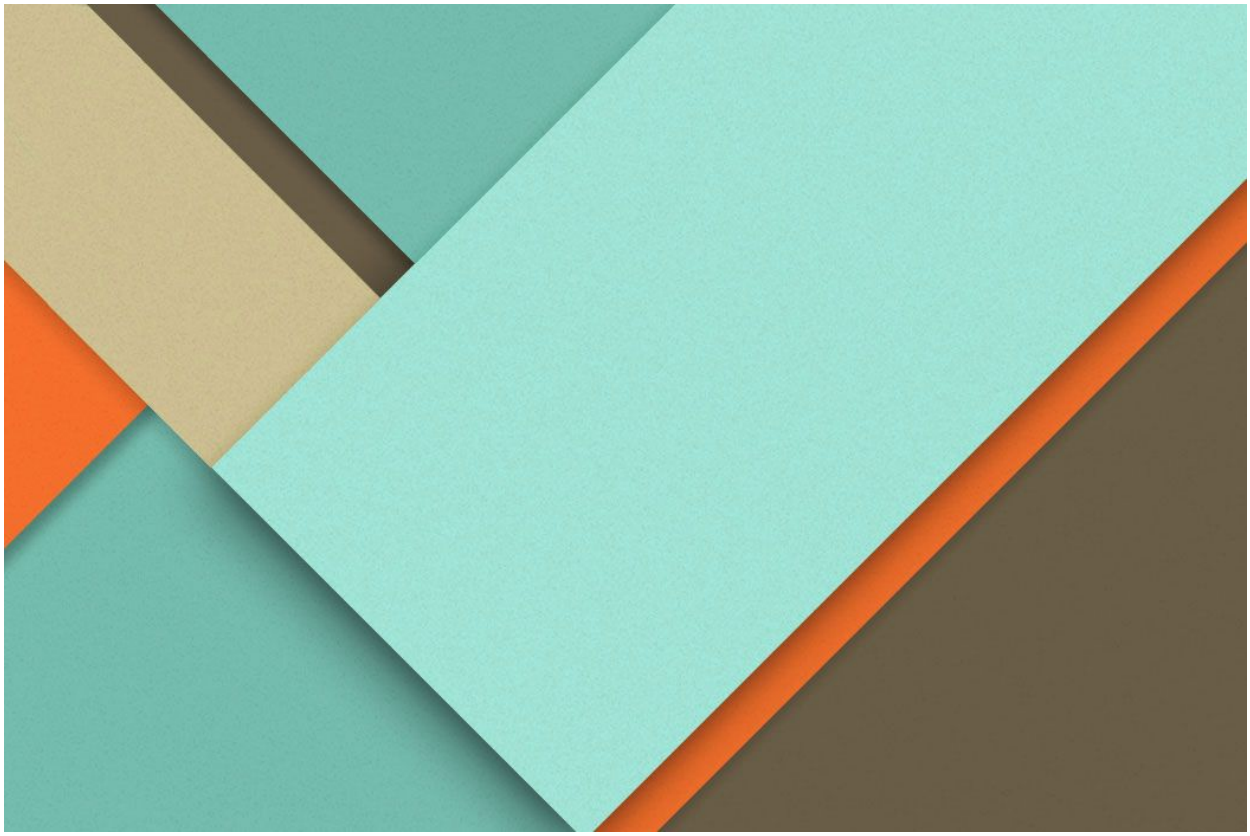


Apache Stanbol

Complementos de Bases de Datos

19 de Mayo de 2018

Escuela Técnica Superior de Ingeniería Informática
41012, Sevilla

Gordo Aguilar, Jorge

Rodríguez Artacho, Antonio

Índice

Qué es Apache Stanbol

Apache Stanbol es un conjunto de componentes reusables para la gestión de contenido semántico (CMS). Empezó, y a día de hoy sigue siendo, un proyecto *open source* desarrollado en java por *Apache Software Foundation*.

El uso principal de Stanbol, y para el que fue desarrollado, es extender el modelo tradicional de gestión de contenido semántico con los servicios semánticos.

Sin embargo, existen otros usos factibles para Apache Stanbol, entre ellos realizar operaciones de enriquecimiento de contenidos. Normalmente, pueden utilizar el cruce de bases de conocimientos con el fin de complementar el texto de entrada con enlaces externos relevantes, como definiciones, imágenes, vídeos, etcétera.

Esta funcionalidad permite que, al integrar Stanbol con una aplicación web, se encuentren referencias a diferentes personas, lugares, organizaciones, entre otras cosas, que aparezcan en ella.

Con el fin de ser usado a través de sus servicios como un motor semántico, todos los componentes ofrecen sus funcionalidades a través de una API RESTful.

Cómo empezó

En 2008, la organización Salzburg Research lideró un consorcio formado por siete socios investigadores y seis socios industriales con el fin de desarrollar el proyecto IKS (Interactive Knowledge Stack).

Sin embargo, hasta enero de 2009 no fueron parcialmente financiados por la comisión europea para proveer una plataforma tecnológica de código abierto para la gestión de sistemas de contenidos semánticamente mejorados.

La duración del proyecto se estableció en 4 años y comenzó con una financiación parcial de 6.58 millones de euros proporcionados por la Unión Europea. Tras ese tiempo, el proyecto finalizó a finales de 2012.

Los mismos miembros que conformaban la junta de IKS fueron los que fundaron Apache Stanbol a finales de 2010. Surgió como medida para que los resultados, especialmente el software desarrollado, estuviese disponible para los vendedores de sistemas de gestión de contenidos (CMS) una vez que el proyecto finalizara. Los miembros de IKS decidieron iniciar el proyecto Apache Stanbol como parte del programa de incubación de *Apache Software Foundation*. Así:

- 15 de noviembre de 2010: Apache Stanbol ingresa al programa de incubación de Apache Software Foundation.
- 9 de mayo de 2012: la versión *0.9.0-incubating* es liberada.
- 10 de julio de 2012: la versión *0.10.0-incubating* es liberada.

A mediados de 2012, coincidiendo con el lanzamiento de su última versión de incubadora se demostró que Apache Stanbol tenía una comunidad muy activa y capaz de desarrollar software y lanzar nuevas versiones de acuerdo al estándar *ASF* (Alert Standard Format).

Finalmente, los responsables que conforman la dirección de *ASF* aceptaron la resolución formal de establecer Apache Stanbol como un *'top-level project'* algo que no solo demuestra la madurez del proyecto como tecnología sino también lo activa que es la comunidad que lo desarrolla y mantiene.

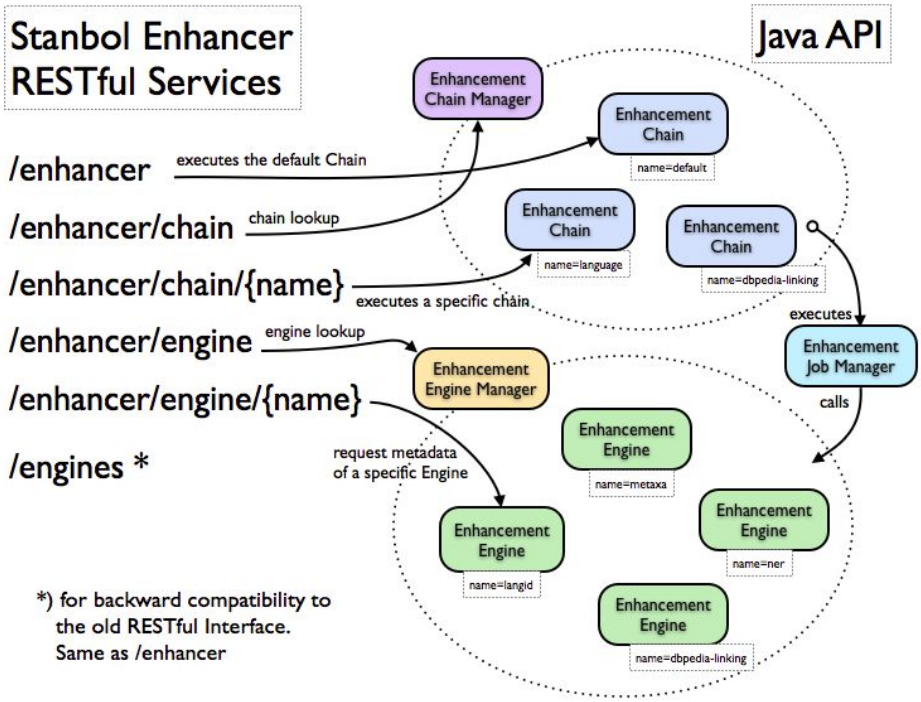
Principales características

Las principales características de Apache Stanbol son:

- Content Enhancement (*Mejora del contenido).*
- Reasoning (*Razonamiento).*
- Knowledge Models (*Modelos de conocimiento).*
- Persistence (*Persistencia).*

1. **Content Enhancement:** Servicios que añaden información semántica a piezas de contenido no semántico. Apache Stanbol provee tanto una API RESTful como una API Java que permite extraer características del contenido que pasemos.

   El contenido que pasemos será procesado por un motor de mejora definido por la cadena de mejora.



   En la imagen se pueden ver los servicios que provee la API RESTful y la API Java para usar en la característica de ***Content Enhancer.*** Cada uno de ellos proporciona las siguientes características:

- /enhancer y /enhancer/chain: ejecutan la cadena cadena ***default.*** Esta es la cadena que viene asignada por defecto para su uso en caso de no especificar ninguna cadena. Incluye una lista con los motores principales (engines) que serán usados para realizar las búsquedas de información semántica:
    - ***Tika Engine***: Detecta el tipo de contenido en contenido parseado. También extrae el texto plano del contenido parseado y lo añade al *ContentItem (es el objeto que crea Stanbol para todo el proceso, generado a partir del texto plano introducido)* como parte del contenido.
    - ***Language Detection Engine:*** Determina el lenguaje del texto.
    - ***OpenNLP Sentence Detection Engine:*** Añade frases a la parte de AnalyzedText (*un modelo de dominio Java diseñado para describir los resultados NLP).* En caso de que no esté presente este motor se encargará de crearlo.
    - ***OpenNLP Tokenizer Engine:*** añade tokens a la parte de AnalyzedText. Si el contenido no existe, entonces se añade a *ContentItem.*
    - ***OpenNLP POS Tagging Engine:*** añade etiquetas a los tokens de la parte de AnalyzedText.
    - ***OpenNLP Named Entity Extraction Enhancement Engine:*** Este motor lee el texto contenido en el *ContentItem* de tipo "text/plain" escrito en inglés. Usa una herramienta de búsqueda para detectar nombres de personas, lugares y organizaciones.
    - ***Named Entity Tagging Engine:*** busca entidades basadas en anotaciones de texto.
    - ***Entity Linking Engine:*** es un motor que consume resultados los procesos NLP del *AnalyzedText* y usa esa información para generar links de esas entidades.
    - ***Entity Dereference Engine:*** su responsabilidad es recuperar información sobre las entidades referenciadas por los resultados de mejora y los añade a los metadatos del *ContentItem.*
- /enhancer/chain/{name}: ejecuta una cadena específica que indicaremos a través de la URI. Dicha cadena usará los motores que tenga referenciados.

- /enhancer/engine: solicita los metadatos de la lista de motores que se encuentren activos por defecto.
- /enhancer/engine/{name}: solicita los metadatos de un motor específico que se introduce a través de la URI.

2. **Reasoning:** los componentes de razonamiento de Stanbol proveen un conjunto de servicios que usan los motores de inferencia. Estos servicios son capaces de recuperar información semántica adicional sobre los contenidos basados en información semántica recuperados a través de la funcionalidad explicada anteriormente, es decir, a través de *Content Enhancement (Mejora de contenido).*

   Su funcionalidad se podría resumir de forma informal en "mejorar la mejora de contenido", es decir, añade nueva información semántica a la información semántica ya obtenida anteriormente.

3. **Knowledge Models:** se compone de un conjunto de servicios que son usados para definir y manipular los modelos de datos que son usados para almacenar información semántica. Un ejemplo puede ser el *Ontology Manager*, que proporciona un un ecosistema controlado para gestionar ontologías (definiciones formales de tipos, propiedades y relaciones entre entidades), redes ontológicas y sesiones de usuario para datos semánticos modelados después.

   Proporciona acceso completo a las ontologías almacenadas en la capa de persistencia de Stanbol.

   ¿Qué significa gestionar una red ontológica? Significa que puedes activar o desactivar partes de un modelo complejo de tal forma que los datos almacenados pueden ser vistos y clasificados bajo diferentes vistas lógicas, algo que es especialmente útil en operaciones que usen la funcionalidad *Reasoning* encargada de la adición de nueva información semántica adicional.

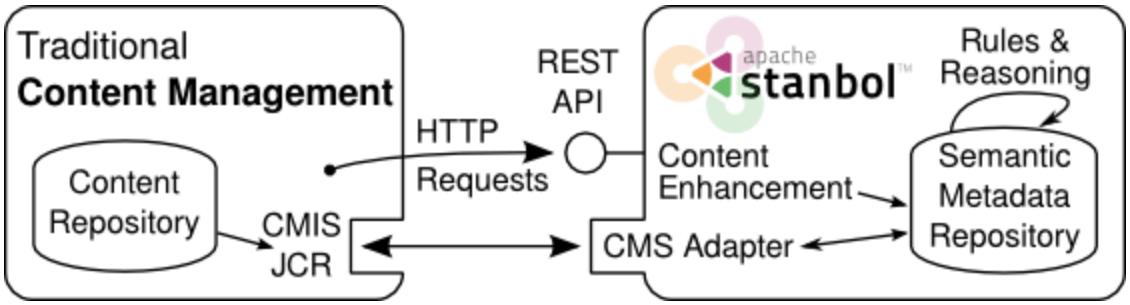
4. **Persistence:** Conformado por los servicios que almacenan o cachean información semántica, por ejemplo, contenido mejorado, entidades, datos y permitir su búsqueda.

   El componente que proporciona almacenamiento de documentos de persistencia cuyo *'back-end'* es Apache Solr se denomina ***Contenthub***.

Contenthub es un repositorio de documentos basado en Apache Solr que permite el almacenamiento de documentos de texto y personalizar los servicios de búsqueda semántica.

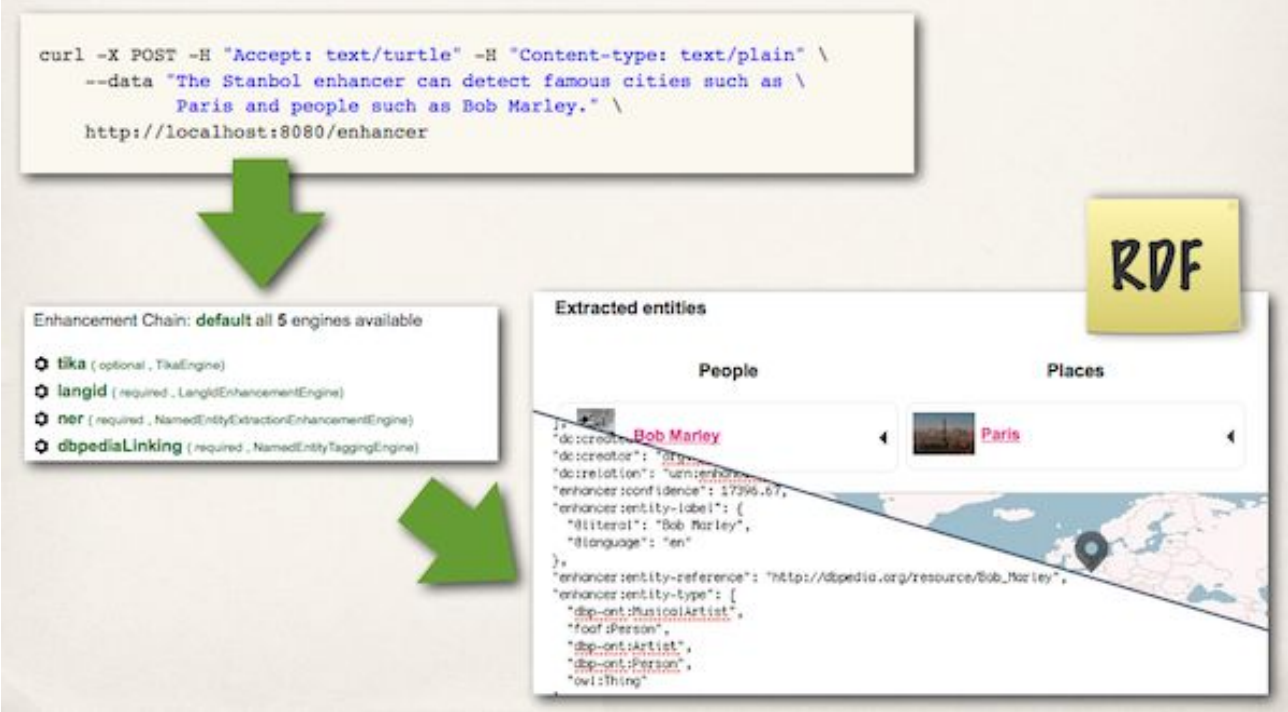
Usos posibles

Apache Stanbol está diseñado para implementar tecnologías semánticas a los sistemas de gestión de contenidos ya existentes. Para ello, sus prestaciones son fácilmente accesibles a través de una API RESTful, por lo que sólo hay que conectar el sistema a una instancia de Apache Stanbol via HTTP. De otra forma, el programa también incluye un componente de adaptador de CMS, que actúa como un puente usando las especificaciones JCR y CMIS.



Los siguientes escenarios explican en detalle cómo sacar provecho de los distintos servicios con un CMS:

- Mejora de contenido básico: analiza el contenido textual y lo mejora con entidades, mostrando sus nombres, y proporciona enlaces a las fuentes. Cuando se envía el contenido para su mejora, Apache Stanbol usará una cadena de motores de mejora para procesarlo y devolverá las características extraídas en formato RDF, usando la Estructura de Mejora de Stanbol.



El resultado se puede obtener tanto desde una llamada a la API a través de la ventana de comandos, como desde la aplicación web de Apache Stanbol.

```
curl -X POST -H "Accept: text/turtle" -H "Content-type: text/plain" \
--data "The Stanbol enhancer can detect famous cities such as Paris \
and people such as Bob Marley." http://localhost:8080/engines
```

Apache Stanbol también puede hacer una mejora de ficheros que no sean de texto plano, gracias a Apache Tika y el motor Tika-Engine, que vimos anteriormente.

La mejora la realiza usando las cadenas, que incluyen conjuntos de motores de mejora. El total de motores de mejora disponibles para Apache Stanbol se puede enontrar aquí: https://stanbol.apache.org/docs/trunk/components/enhancer/engines/list.html

- Hacer uso de las mejoras de Apache Stanbol:

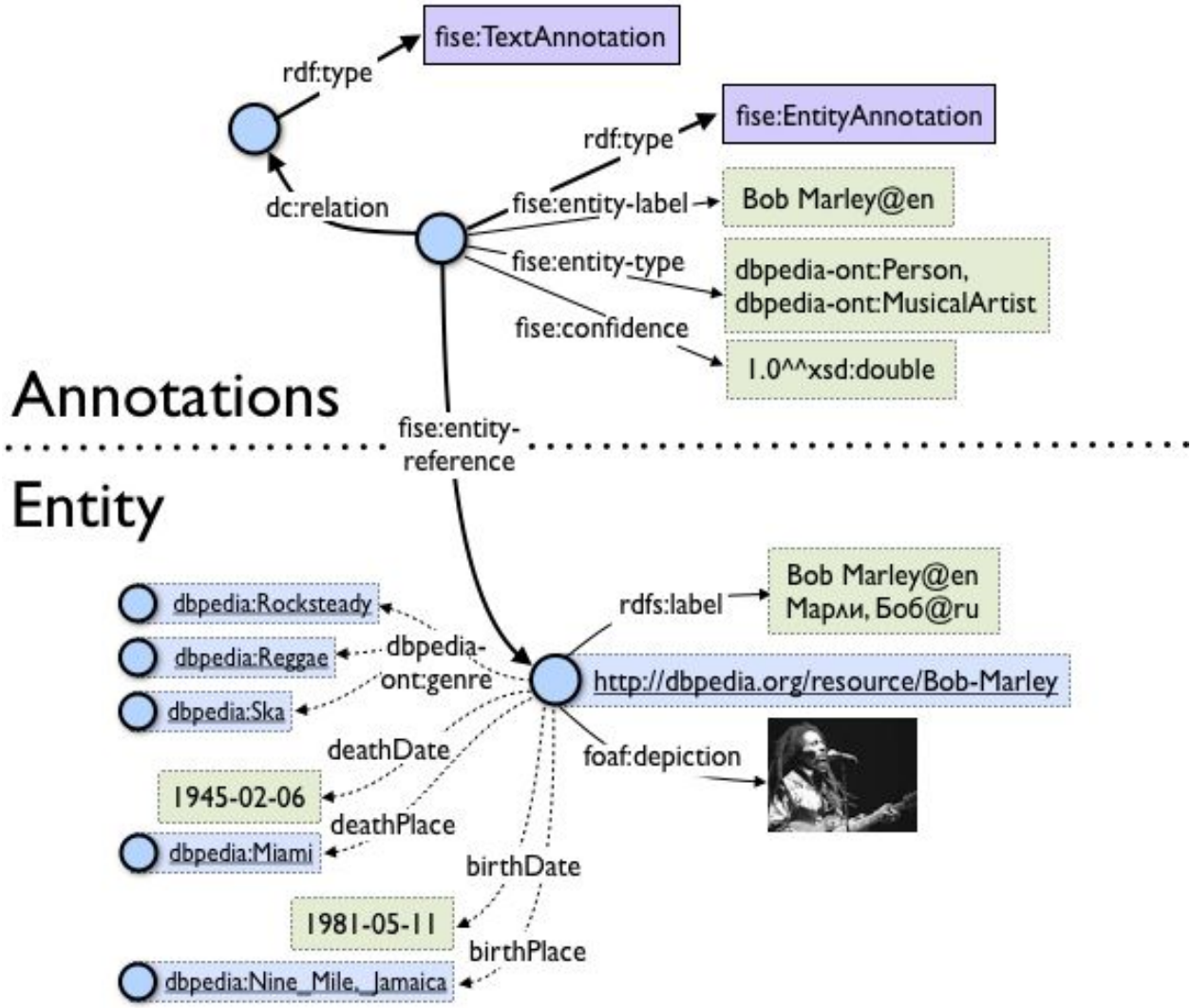
- “Entity tagging”: reemplaza los tags de texto como “Bob Marley” con entidades (dbpedia:Bob_Marley) para mejorar la búsqueda de contenido y categorización.
- “Entity disambiguation”: mejora la entidad etiquetada gracias a un apoyo que ayuda a diferenciar entre ambigüedades de las entidades propuestas, lo que permite a los usuarios crear enlaces hacia París de Texas, o el Bob Marley cómico, o entre otras entidades que tienen etiquetas similares.
- “Entity checker”: interacciona con las entidades similiares que han sido extraídas de la misma forma que los correctores de hoy en día. Muestra etiquetas sugeridas a partir del contenido, y permite al usuario interactuar entre los resultados si fuera necesario.

El etiquetado de entidades trata sobre sugerir entidades definidas por el usuario para etiquetar sus documentos, en lugar de cadenas de texto o “strings”. La diferencia es fácil de explicar: asumamos que un bloggero usa la etiqueta “Bob Marley” para etiquetar la entrada de un blog. El etiquetado trata simplemente sobre organizar contenidos. Si etiqueta simplemente como “Bob Marley” puede encontrar de forma fácil todos los documentos que usan esa etiqueta. Sin embargo, es muy posible que él también quiera crear una categoría de documentos sobre la música reggae y que esos documentos etiquetados como “Bob Marley” sean parte de ese grupo.

Mientras que el conocimiento de que “Bob Marley” está relacionado con la música reggae puede ser muy obvio para las personas, no lo es para la herramienta que use. Así que, de forma típica, la única forma que tiene para solventar esto es que etiquete el documento de las dos formas.

El hecho de que Bob Marley está relacionado con la música reggae no es nada nuevo. DBpedia, la base de datos de Wikipedia, lo sabe, y mucho más sobre la entidad dbpedia:Bob_Marley. Si el bloggero etiqueta su documento con “dbpedia:Bob_Marley”, no solo lo está etiquetando con Bob Marley, sino también con toda la información contextual provista por DBPedia, incluyendo el hecho de que Bob Marley era un intérprete de reggae.

La siguiente figura muestra cómo las entidades extraídas son descritas en los resultados de la mejora.

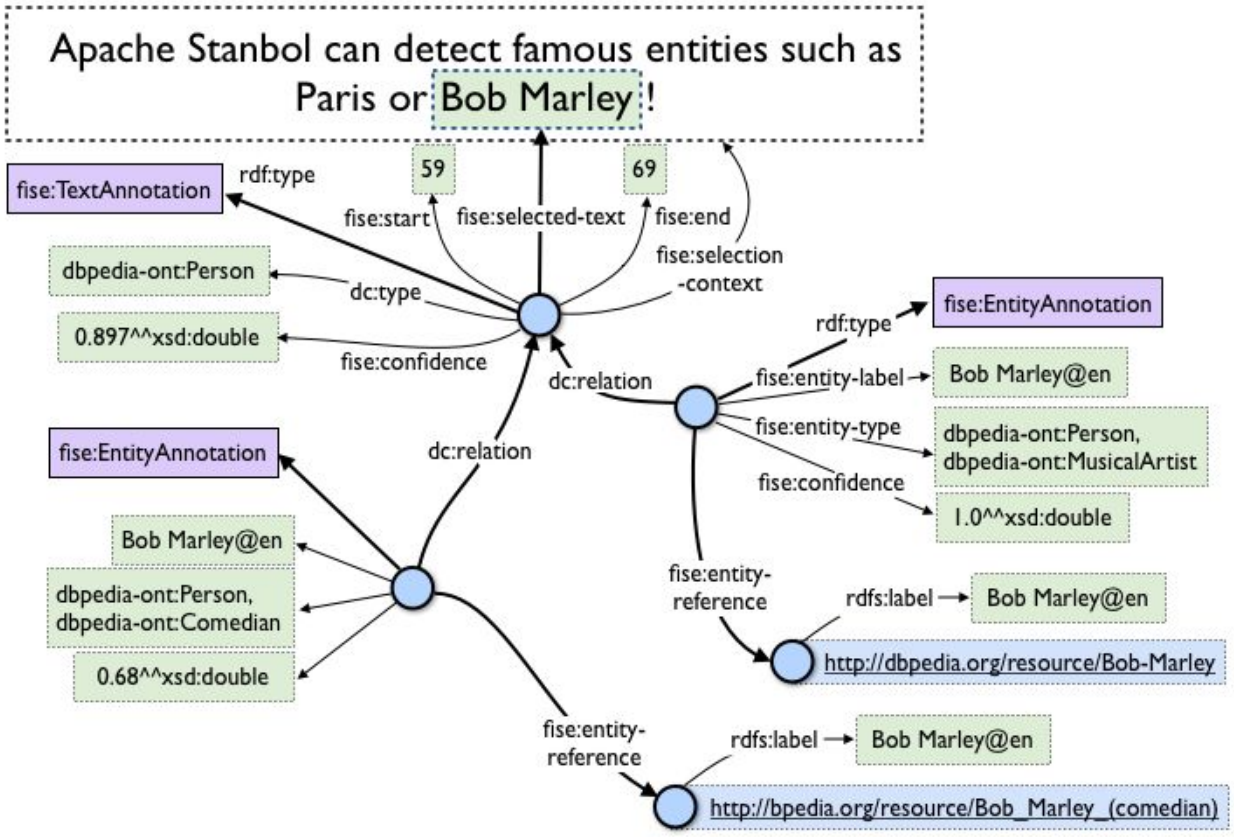


En principio, hay dos recursos que son de interés para el caso de uso de etiquetar:

1. EntityAnnotations: los recursos con el 'rdf:type' 'fise:EntityAnnotation' representan la sugerencia de entidad de Apache Stanbol. Esto proporciona el tipo de la etiqueta y, lo más importante, la URI de la entidad extraída. Además, el valor del 'fise:confidence' [0..1] puede ser usado como un indicador de cómo de seguro está Apache Stanbol Enhancer sobre esta entidad.
2. Entities: se refiere a todos los recursos con una relación con la entidad dada. Los motores de mejora pueden ser configurados para "deferenciar" las entidades sugeridas, lo que significa que se debe usar la URI de la entidad para recuperar información adicional. En este caso, la información adicional

sugerida estará disponible en los resultados de la mejora. Si no fuera el caso, los usuarios deberían "deferenciar" las entidades sugeridas por ellos mismos.

En caso de que la entidad detectada en el texto analizado se pueda referir a entidades diferentes, es necesario diferenciar cuál estamos buscando. La siguiente figura muestra un ejemplo donde Bob Marley es detectado como una persona en el texto, pero hay dos resultados distintos según el vocabulario controlado:

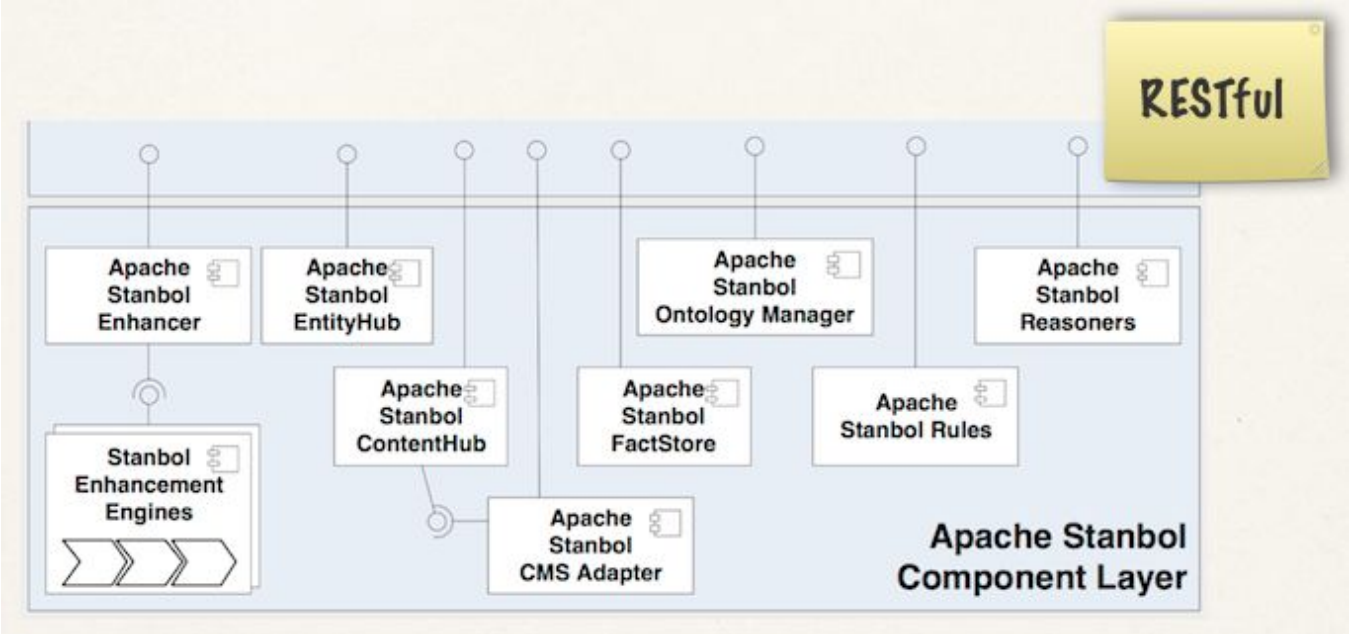


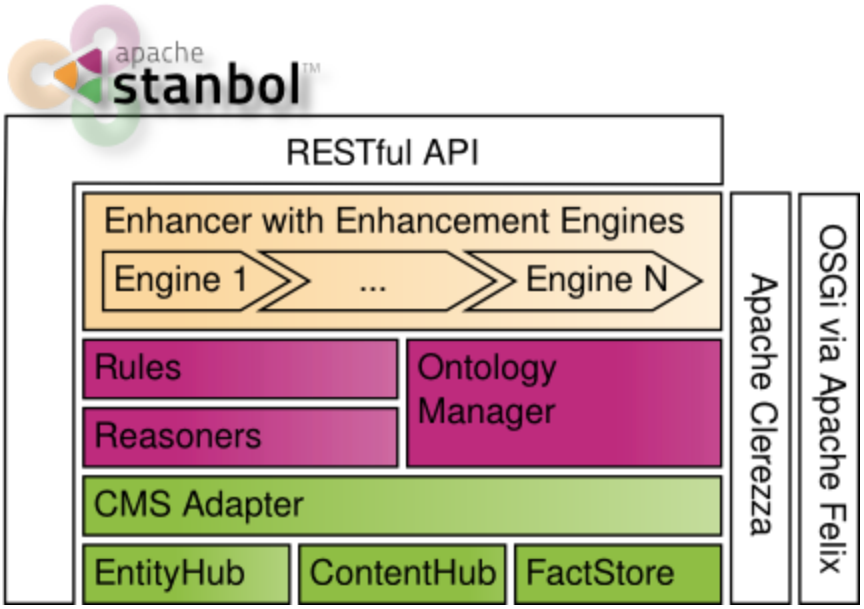
El hecho de que una entidad detectada en el texto (representada por 'dise:TextAnnotation') pueda tener múltiples entidades sugeridas (representadas por los dos 'dise:EntityAnnotation's') tiene un impacto negativo en la interfaz de etiquetado de entidades que sugiere etiquetas basadas en 'fise:entityAnnotation's. Esto es porque esta interfaz mostraría en el caso de arriba dos sugerencias: 1 para 'dbpedia:Bob_Marley' y 2 dbpedia:Bob_Marley_(comedian). Así que si el usuario quiere etiquetar contenido con Bob Marley, necesitará al menos rechazar una de las 2 sugerencias.

Componentes

Apache Stanbol está construido como un conjunto de componentes. Cada componente es accesible a través de su propia interfaz web RESTful. Desde este punto de vista, todas las características de Apache Stanbol pueden ser usadas via llamadas RESTful.

Ninguno de los componentes depende de otro, pero pueden ser combinados muy fácilmente en caso de que fuera necesario. Esto hace que la lista de los componentes usados dependa del escenario de uso específico y no de la arquitectura de Apache Stanbol.





- El componente Mejora (‘Enhancer’), junto con los motores de mejora, provee la habilidad de postear contenido a Apache Stanbol y obtener sugerencias de posibles entidades. Las mejoras se proveen a través del procesamiento de texto en lenguaje natural, extracción de metadatos y relaciones de las entidades a repositorios públicos o privados. Además, Apache Stanbol provee una maquinaria para procesar estos datos y añadir conocimiento adicional y enlaces a través de reglas y razonamientos. Técnicamente, las mejoras son guardadas en un triple-grafo mantenido por Apache Clerezza.
- La vista de Sparql da acceso a grafos RDF de Apache Stanbol. Especialmente incluye el grafo con todas los resultados de las mejoras gestionado por Apache Stanbol Contenthub.
- El ‘EnhancerVIE’ es una interfaz para enviar contenido para ser analizado y guardar los resultados en el servidor. Entonces es posible navegar por los resultados de las mejoras.
- Las reglas (‘Rules’) es un componente que provee con la posibiliad de refactorizar los grafos de conocimiento, por ejemplo para soportar el vocabulario de schema.org para la optimización de los motores de búsqueda.
- El razonador (‘Reasoner’) puede ser usado para automáticamente inferir conocimiento adicional. Es usado para obtener nuevos hechos en la base de conocimiento, por ejemplo si el contenido mejorado nos dice una tienda

localizada en París, podemos inferir a través de la relación "located-in" que la misma tienda está localizada en París, en "Île-de-France" y en Francia.

- El gerente de ontologías (“Ontology Manager”) es el componente que gestiona nuestras ontologías. Las ontologías son usadas para definir modelos de conocimiento que describen los metadatos del contenido. Adicionalmente, la semántica de nuestros metadatos puede ser definido a través de una ontología.
- El componente de adaptador CMS actúa como un puente entre gestores de contenido JCR/CMIS y Apache Stanbol. Puede ser usado para mapear un repositorio de contenido a un modelo RDF o vice-versa. También provee servicios para la gestión de los contentItems en el Contenthub.
- El componente “Entityhub” nos permite cachear y gestionar índices locales de repositorios como DBPedia, pero también datos personalizados, como descripciones de productos, datos de contacto, etc.
- El componente Contenthub nos provee almacenamiento de documentos de persistencia, cuyo back-end es Apache Solr. Además del almacenamiento, provee ayuda del indexado semántico durante el envío del texto, y búsqueda semántica.
- El “FactStore” es un componente que nos permite usar relaciones entre entidades identificadas mediante sus URIs. Esta relación entre dos entidades es llamada “fact” (hecho).

Configuración

A continuación se mostrarán los pasos para completar la instalación y la configuración de Apache Stanbol en un entorno Linux. Para sistemas operativos basados en Windows u OSX se realizará de forma análoga, e incluso más sencilla.

Para la instalación y configuración de Apache Stanbol necesitaremos previamente lo siguiente:

- Java 6 instalado y configurado para que se use por defecto.
- Maven 3.0.3 o superior.

Para la instalación de Java 6 debemos descargar el archivo binario correspondiente desde la web de Oracle. En concreto: ***jdk-6u45-linux-x64.bin.***

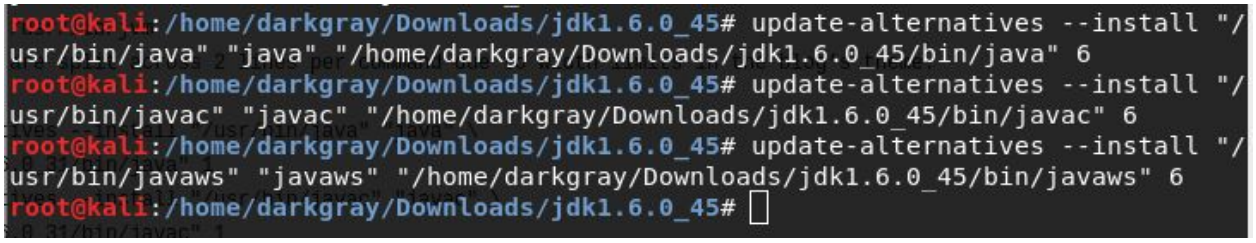
Una vez descargado procederemos a otorgar permisos de ejecución al archivo .bin con el siguiente comando:

Como se puede apreciar en la imagen, el paquete *jdk-6u45-linux-x64.bin* aparece en color verde lo que indica que se puede ejecutar. Para ello simplemente ejecutaremos:

./jdk-6u45-linux-x64.bin

El resultado de este comando será una carpeta con el nombre ***jdk1.6.0_45*** que ya se puede apreciar en la imagen puesto que se ha hecho anteriormente.

A continuación, desde el terminal accederemos a la carpeta que se ha generado y ejecutaremos los siguientes comandos:



Con esto quedará instalado en nuestro sistema Java 6, sin embargo, no está asignado como el principal por lo que debemos usar el siguiente comando para hacerlo:

```
root@kali:/home/darkgray/Downloads/jdk1.6.0_45# update-alternatives --config java
There are 5 choices for the alternative java (providing /usr/bin/java).

  Selection    Path                                            Priority   Status
------------------------------------------------------------
  0            /usr/lib/jvm/java-10-openjdk-amd64/bin/java      1101      auto mode
* 1            /home/darkgray/Downloads/jdk1.6.0_45/bin/java    6         manual mode
  2            /usr/lib/jvm/java-10-openjdk-amd64/bin/java      1101      manual mode
  3            /usr/lib/jvm/java-8-openjdk-amd64/jre/bin/java   1081      manual mode
  4            /usr/lib/jvm/java-8-oracle/jre/bin/java          1081      manual mode
  5            /usr/lib/jvm/java-9-openjdk-amd64/bin/java       1091      manual mode

Press <enter> to keep the current choice[*], or type selection number: 1
root@kali:/home/darkgray/Downloads/jdk1.6.0_45#
```

Con esto nuestro jdk por defecto será el correspondiente a Java 6. Se puede comprobar fácilmente a través del terminal:

```
darkgray@kali:~/Pictures$ java -version
java version "1.6.0_45"
Java(TM) SE Runtime Environment (build 1.6.0_45-b06)
Java HotSpot(TM) 64-Bit Server VM (build 20.45-b01, mixed mode)
darkgray@kali:~/Pictures$
```

Lo siguiente será la instalación de Maven en nuestro sistema. Para ello simplemente ejecutaremos el comando correspondiente para su instalación:

```
root@kali:~# apt-get install maven
Reading package lists... Done
Building dependency tree
Reading state information... Done
maven is already the newest version (3.5.2-2).
0 upgraded, 0 newly installed, 0 to remove and 1 not upgraded.
root@kali:~#
```

Dado que ya está instalado en nuestro entorno aparece ese mensaje. En caso de no estar instalado ejecutando el mismo comando se procederá a la instalación de forma sencilla.

Una vez completados los pasos anteriores ya tendremos todo listo para proceder con la instalación y configuración de Apache Stanbol.

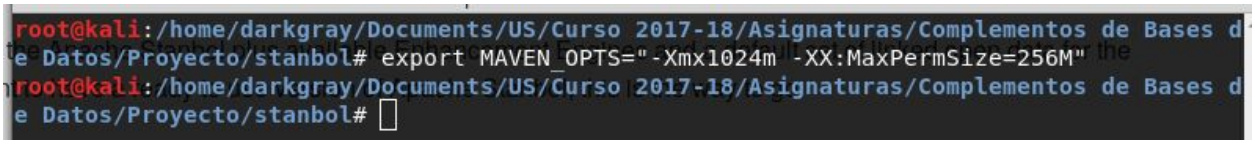
Todos los archivos necesarios para la instalación de Stanbol se encuentran en su repositorio de Github, recordemos que es *Open Source.* El proyecto se encuentra concretamente en la siguiente dirección: https://github.com/apache/stanbol. Una

vez ahí podemos descargarnos todo el proyecto con la opción de 'Download Zip' o podemos clonarnos el proyecto con el comando:

git clone https://github.com/apache/stanbol

Ya lo tenemos todo listo para proceder con la instalación, finalmente. Lo primero que debemos hacer es acceder a la raíz de la carpeta de stanbol (en caso de haber descargado el .zip, accedemos tras haber realizado la descompresión de la misma) desde el terminal.

Una vez dentro usaremos el siguiente comando:

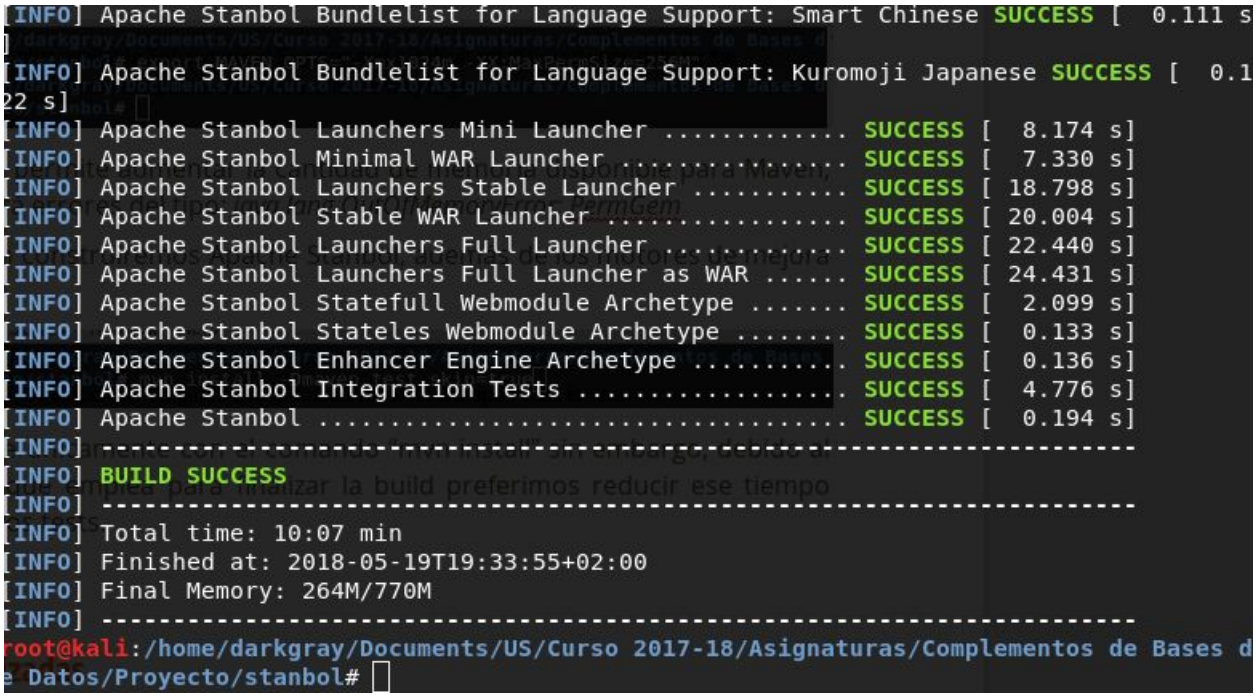


Este comando permite aumentar la cantidad de memoria disponible para Maven, algo que evitará errores del tipo: *java.lang.OutOfMemoryError: PermGem*

A continuación construiremos Apache Stanbol, además de los motores de mejora disponibles.

Podría hacerse únicamente con el comando "mvn install" sin embargo, debido al largo tiempo que emplea para finalizar la build preferimos reducir ese tiempo desactivando los tests.

Esto tardará entre 8-10 minutos y, si hemos seguido los pasos, se mostrará un mensaje como el siguiente:



BUILD SUCCESS: indicativo de que todo ha ido bien y la construcción se ha realizado de forma correcta.

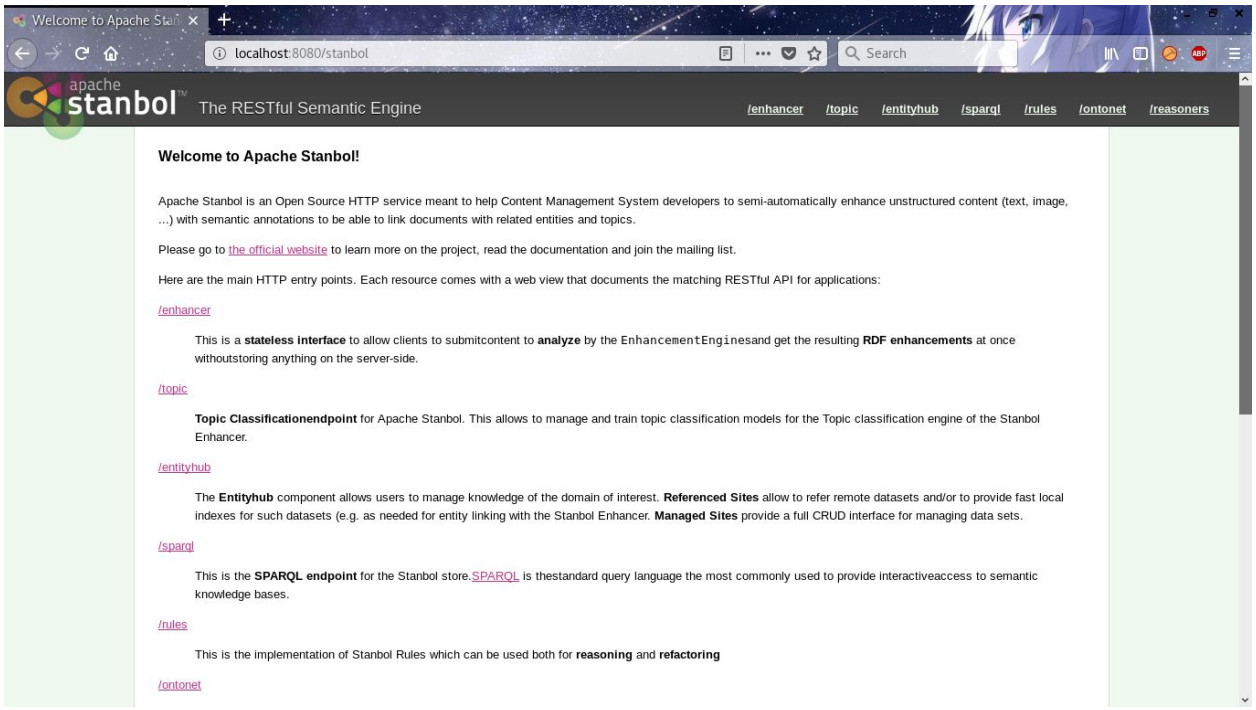
Lo único que quedará ahora será desplegarlo.

Para ello accedemos desde la raíz de la carpeta de stanbol a "/launchers/full-war/" y dentro de esa carpeta ejecutamos el siguiente comando:

El terminal donde realicemos este comando no debe cerrarse ya que será el que mantenga ejecutado el tomcat para el uso de Stanbol. Ahora, simplemente tendremos que acceder a:

localhost:8080/stanbol

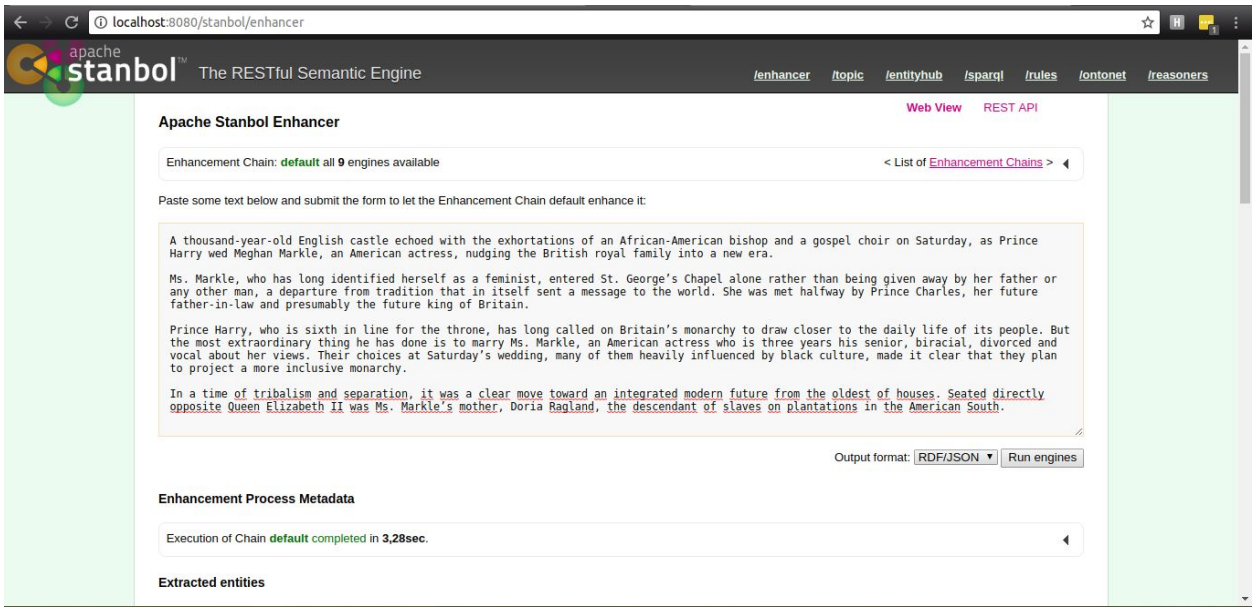
Y nos encontraremos en la ventana principal de Apache Stanbol:

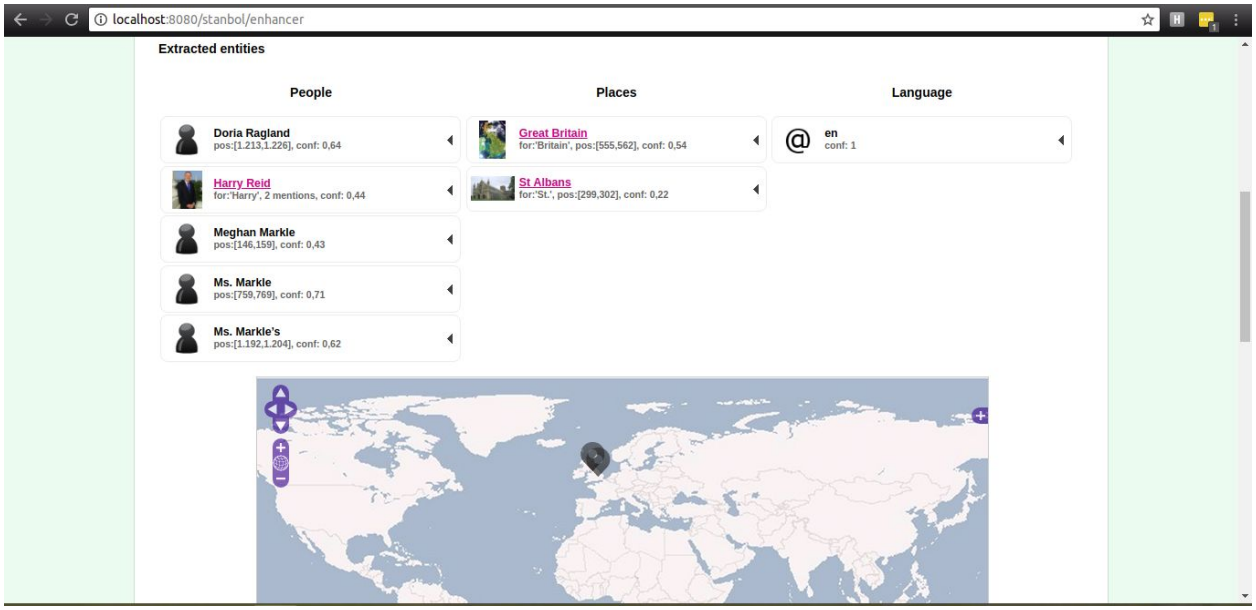


Con esto quedaría finalizada la parte de instalación y configuración.

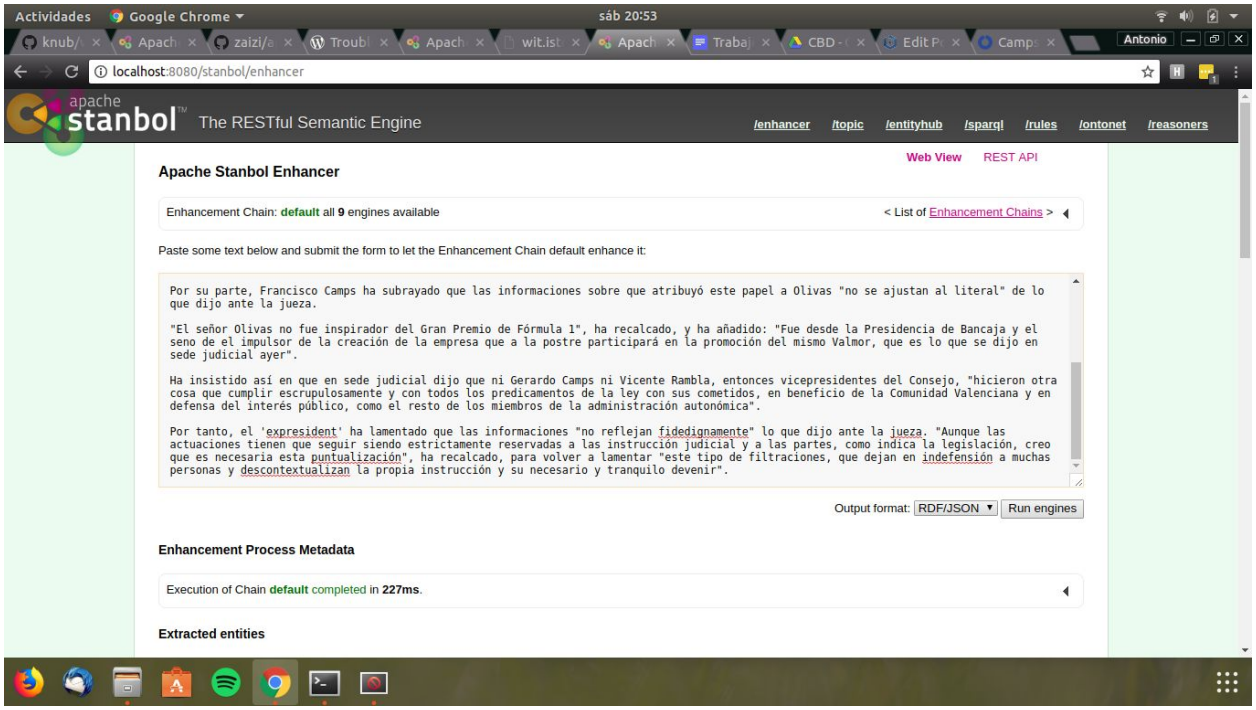
Pruebas realizadas

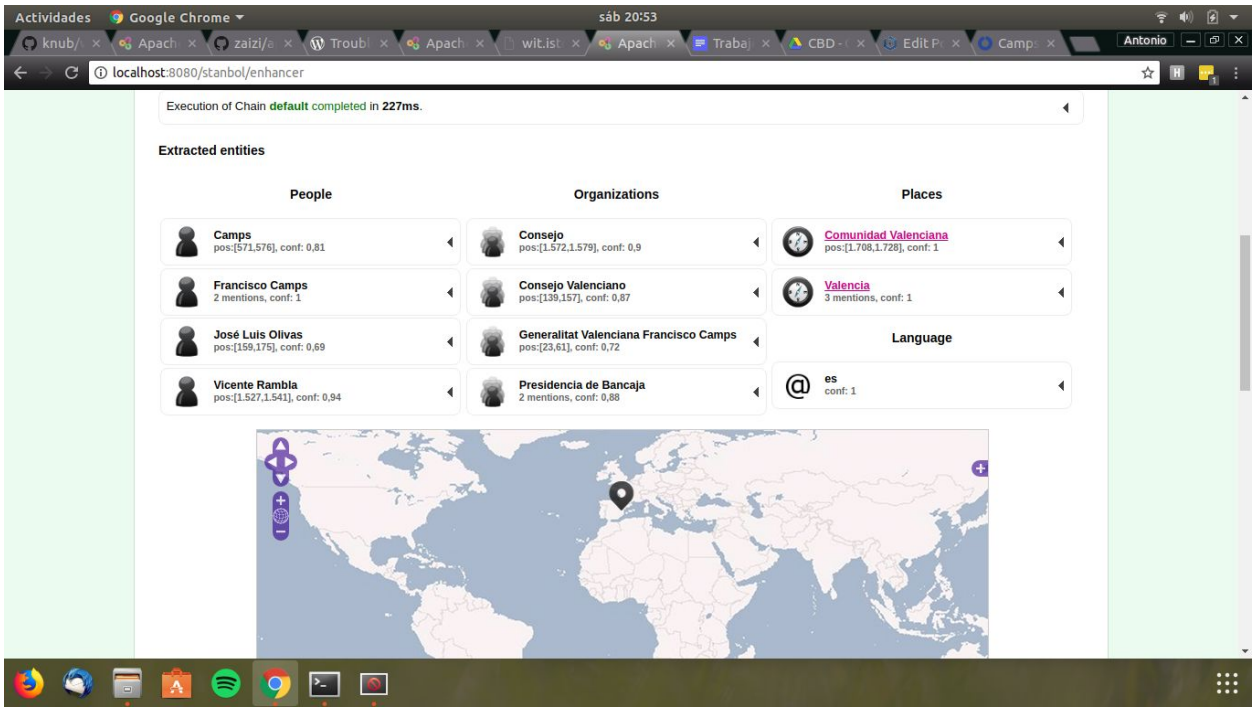
Como prueba del funcionamiento del componente Enhancer (mejora), hemos usado un extracto de un artículo del periódico The New York Times (https://www.nytimes.com/2018/05/19/world/europe/meghan-markle-prince-harry-wedding.html?hp&action=click&pgtype=Homepage&clickSource=story-heading&module=a-lede-package-region&region=top-news&WT.nav=top-news) sobre la Boda Real en Inglaterra. Como podemos ver, tras enviar el texto al componente para su análisis, obtenemos de resultado una lista con las entidades mencionadas en él, así como los lugares y el idioma del texto. También obtenemos un mapa generado en base a estos lugares y sus coordenadas.





Como otro ejemplo, hemos escogido esta vez un artículo en español de "El diario" (https://www.eldiario.es/politica/Camps-Olivas-F1-Presidencia-Bancaja_0_773073119.html) para comprobar si la búsqueda de entidades funcionaba correctamente, y como podemos comprobar, así es.


Actividades
Google Chrome
sáb 20:53
Antonio
localhost:8080/stanbol/enhancer
apache stanbol
The RESTful Semantic Engine
/enhancer
/topic
/entityhub
/sparql
/rules
/ontonet
/reasoners
Web View
REST API
Apache Stanbol Enhancer
Enhancement Chain: default all 9 engines available
< List of Enhancement Chains >
Paste some text below and submit the form to let the Enhancement Chain default enhance it:
Por su parte, Francisco Camps ha subrayado que las informaciones sobre que atribuyó este papel a Olivas "no se ajustan al literal" de lo que dijo ante la jueza.
"El señor Olivas no fue inspirador del Gran Premio de Fórmula 1", ha recalcado, y ha añadido: "Fue desde la Presidencia de Bancaja y el seno de el impulsor de la creación de la empresa que a la postre participará en la promoción del mismo Valmor, que es lo que se dijo en sede judicial ayer".
Ha insistido así en que en sede judicial dijo que ni Gerardo Camps ni Vicente Rambla, entonces vicepresidentes del Consejo, "hicieron otra cosa que cumplir escrupulosamente y con todos los predicamentos de la ley con sus cometidos, en beneficio de la Comunidad Valenciana y en defensa del interés público, como el resto de los miembros de la administración autonómica".
Por tanto, el 'expresident' ha lamentado que las informaciones "no reflejan fidedignamente" lo que dijo ante la jueza. "Aunque las actuaciones tienen que seguir siendo estrictamente reservadas a las instrucción judicial y a las partes, como indica la legislación, creo que es necesaria esta puntualización", ha recalcado, para volver a lamentar "este tipo de filtraciones, que dejan en indefensión a muchas personas y descontextualizan la propia instrucción y su necesario y tranquilo devenir".
Output format: RDF/JSON
Run engines
Enhancement Process Metadata
Execution of Chain default completed in 227ms.
Extracted entities


Actividades
Google Chrome
sáb 20:53
Antonio
localhost:8080/stanbol/enhancer
Execution of Chain default completed in 227ms.
Extracted entities
People
Camps
pos:[571,576], conf: 0,81
Francisco Camps
2 mentions, conf: 1
José Luis Olivas
pos:[159,175], conf: 0,69
Vicente Rambla
pos:[1.527,1.541], conf: 0,94
Organizations
Consejo
pos:[1.572,1.579], conf: 0,9
Consejo Valenciano
pos:[139,157], conf: 0,87
Generalitat Valenciana Francisco Camps
pos:[23,61], conf: 0,72
Presidencia de Bancaja
2 mentions, conf: 0,88
Places
Comunidad Valenciana
pos:[1.708,1.728], conf: 1
Valencia
3 mentions, conf: 1
Language
es
conf: 1

Problemas encontrados

Desde un principio intentamos la integración de Apache Stanbol con alguna herramienta que permitiese aprovechar sus funcionalidades y que el resultado general fuese más curioso. Sin embargo, nos hemos ido encontrando con diversos problemas que explicaremos a continuación.

Una de las integraciones que se intentó fue con Alfresco. Por si no es muy conocido, Alfresco es un software de gestión documental para documentos, imágenes, páginas webs y más. Lo que intentamos fue integrar Stanbol para aprovechar características como la de obtener enlaces, información enciclopédica, imágenes y demás de las personas, lugares, etcétera.

Para esta integración se utilizó el plugin ***semantics4alfresco*** que se encuentra en el repositorio: https://github.com/esplinr/semantics4alfresco. Se realizó la instalación y configuración siguiendo todos los pasos (recopilados de foros ya que el README.md estaba bastante incompleto). Tras tener todo listo, se desplegó Stanbol en el puerto 9080 (para que no hubiese conflicto con Alfresco) y Alfresco en el puerto 8080. Sin embargo, tras acceder a Alfresco, y tras haber instalado y configurado los plugins a través del terminal para integrar Stanbol, no ocurría nada ni se mostraba nada de lo indicado en las instrucciones. Debido a la falta de mantenimiento, ya que el último commit fue en 2012, decidimos buscar otras alternativas.

Como implementación de Apache Stanbol en un CMS normal, intentamos usar un plugin de Wordpress llamado Wordbol (https://github.com/knub/wordbol/) y realizar un ejemplo en el que, creando un post de Wordpress y usando de texto una noticia de algún periódico de habla inglesa (para que los resultados fuesen más exactos), nos mostrase las entidades encontradas, y poder vincularlas al post.

La instalación del plugin funcionó con normalidad, siguiendo paso a paso las indicaciones del creador. Pero, no sabemos por qué, a la hora de recuperar las entidades cuando se escribía el post, el plugin no funcionaba y mostraba de forma indefinida un marcador de que estaba cargando, como podemos ver en la figura.

No sabemos cuál es el motivo. Hemos intentado con varias versiones de Wordpress y buscando información en la web, pero no encontramos nada.

También intentamos usar un programa llamado Apache Stanbol Client (https://github.com/zaizi/apache-stanbol-client) pero, a la hora de compilarlo con Maven, obtuvimos errores en los tests que no supimos resolver, por lo que desechamos la idea pronto.

Pensamos que la mayor dificultad a la hora de realizar este proyecto ha sido la falta de documentación en la red (sólo la página oficial de Apache Stanbol nos ha ayudado en este sentido). El mantenimiento del proyecto es muy escaso, habiendo pasado mucho tiempo desde su última actualización, y la ayuda en la red es casi inexistente. Por lo que nos hemos visto obligados a adentrarnos de lleno en la documentación oficial para resolver los errores, y en muchas ocasiones estos no estaban contemplados.

En general, pensamos que la idea de este programa es muy buena, a la que se le puede dar muchísimo uso en bastantes CMS, pero la falta de apoyo por parte de los desarrolladores es preocupante, y la dificultad de su implementación lo hace poco atractivo para su uso.

Bibliografía

- https://stanbol.apache.org/index.html - Página principal de Apache Stanbol
- https://www.youtube.com/watch?v=l7n6aRFcn1U - Video de Apache Stanbol en funcionamiento
- https://github.com/knub/wordbol/ - GitHub del plugin Wordbol
- https://www.webempresa.com/blog/que-es-un-error-500-y-como-localizarlo-en-nuestra-web.html - Para depurar errores de Tomcat
- https://codex.wordpress.org/Troubleshooting - Para depurar errores de Wordpress
- https://github.com/zaizi/apache-stanbol-client - GitHub de Apache Stanbol Client