# Taller de Lenguajes y Tecnologías de la Web Semántica.

21 de septiembre de 2015

**Resumen**

La idea es que en los trabajos de taller se logre comprender la generalidad de los aspectos involucrados en el área, y alguno de esos aspectos sea el centro del trabajo.

## 1. Arquitectura General

La fig. 1 muestra una arquitectura general de una aplicación de sobre la Web Semántica. Este diagrama sólo representa los aspectos funcionales de una forma muy general.

Los rectángulos representan módulos del sistema. Los "discos" representan repositorios razonablemente estructurados (relacionales, XML,RDF,etc). La "nube", representa una fuente no demasiado estructurada. El caso típico es la web, en donde para recuperar información se debe hacer scrapping o aplicar técnicas de lenguaje natural. En el interés de representar una arquitectura general, se asume además que los módulos pueden ser ser muy simples (llegando a incluso a no existir), o muy complejos. Por ejemplo, el módulo de *Transformación* podría no existir si desde la Web se accede siempre a datos en RDF o OWL. El módulo de *Curado de Datos*, puede no existir o puede ser tan complejo como un juego orientado a confirmar los datos o la estructura de los mismos.

### 1.1. La Entrada

El módulo de **Entrada** debe permitir que el usuario provea datos y/o metadatos de la realidad en cuestión de una forma amigable. Para esto, se debe encargar de un conjunto de tareas que puede incluir (entre otras):

- Entrar datos (instancias).
- Entrar metadatos de datos conocidos.
- Recuperar los metadatos de datos conocidos de forma indirecta, por acciones que el usuario hace sobre los datos.

Al construir el módulo de entrada, se deben tener en cuenta la carga de trabajo que puede llevar construir los datos y metadatos. Si los volúmenes son grandes o la complejidad es muy alta o lleva demasiado tiempo, es razonable que el usuario pueda interrumpir su tarea y retomarla en otro momento.

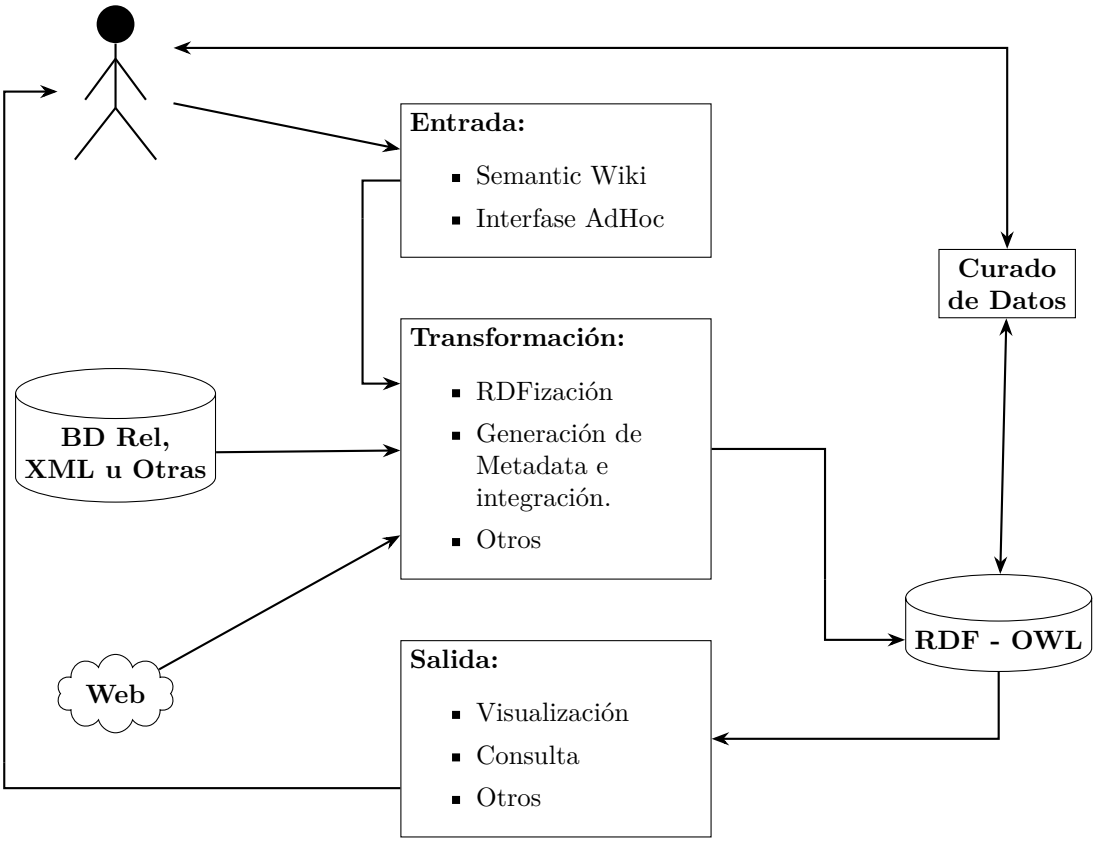


Fig. 1: Arquitectura de un Sistema de Información Basado en Web Semántica

La motivación del usuario también es importante. Si el usuario puede buscar una forma alternativa más simple o más conocida, seguramente hará el trabajo de esa otra otra forma o evitará hacerlo.

## 1.2. Transformación

El módulo de **Transformación** se tiene que encargar de:

- Hacer el el cambio de representación cuando corresponda. (Ej: Relacional - RDF)
- Obtener metadatos a partir de los datos y/o las entradas del usuario.
- Resolver los problemas de integración. (Ej: identificación o diferenciación de entidades, clasificación de entidades y propiedades, etc)
- En general, cualquier problema que incluya transformación de datos y/o generación de los metadatos adecuados para las funciones del sistema.

Entre los problemas típicos a resolver aquí se encuentran:

- El acceso a las fuentes. El acceso será dinámico o sobre copias directas de las fuentes?
- El mecanismo de transformación en si mismo. La transformación se realizará de forma virtual o consolidando los resultados?
- La asignación de las URIs de identificación de los objetos. Serán mnemotécnicas o simplemente un hash de los datos? Qué partes del objeto deben estar involucradas en la construcción de la URI?
- Cómo resolver los problemas de inferencia (clasificación de objetos, clases y propiedades, consistencia, etc). Conviene usar un razonador o simplemente implementar estrategias adhoc ? (Ej: recorridas en el grafo, de generación adecuada de los metadatos usando diferentes técnicas - Ej: PLN, etc).

## 1.3. Salida

El módulo de **Salida** debe encargarse de todo lo relativo a la comunicación de los resultados al usuario y el acceso al repositorio RDF.

Dentro de los problemas típicos a resolver, se encuentran:

- La visualización. Se usarán mecanismos gráficos, tabulares, textuales u otros?
- Selección de los datos. Se usarán consultas directas en el lenguaje de base (Ej: SPARQL, DL-Query, etc) o se deberán implementar mecanismos de consultas gráficas? Se usarán mecanismos de Browsing directo o facetado?
- Independencia de los mecanismos de visualización y selección. Para seleccionar qué datos visualizar, se puede utilizar cualquier mecanismo de selección o cada mecanismo de selección tiene un mecanismo de visualización específico?

Este módulo, dependiendo de cómo está implementado el repositorio semántico, puede incluir mecanismos de inferencia.

Obviamente, las interfases de entrada y salida pueden estar integradas de diferentes formas.

### 1.4. Curado de Datos

El módulo de **Curado de Datos** no siempre es implementado, pero es considerado en la arquitectura ya que puede ser de interés para la realización de proyectos de estudiantes.

Este módulo, se encarga de mejorar la calidad de los datos y metadatos. Esto se realiza mediante la intervención del usuario a través de dos tipos de mecanismos:

- Directos: Una encuesta sobre los datos que directamente le preguntan al usuario si los datos son correctos o cómo mejorarlos. Ante todo, el usuario es consciente del trabajo que está haciendo y porqué lo hace.
- Indirectos: El usuario no es necesariamente consciente de lo que hace. Algunas vecese se hace a través de juegos, u otras actividades que no se ven directamente como relacionadas a la tarea de la mejora de los datos [1].

## 2. Trabajos Posibles

### 2.1. Publicación en RDF de fuentes de Datos

Hay dos formas básicas:

- Transformando los datos y copiándolos.
- Definiendo (y usando) mecanismos mappings que hacen la transformación al vuelo.

Hay además varias formas para presentar los datos en la salida:

- Una URL que permite el acceso al grafo.
- Un SPARQL Endpoint.
- Una interfase web terminada.

#### 2.1.1. Puntos Fuertes

**Rdfización:** Estudiar productos y/o formas de transformar los datos en rdf y compararlas.

**Integración con otras Fuentes:** Si bien la publicación en sí misma es interesante, mostrar la integración con otras fuentes, aunque sea en casos particulares puede ser interesante.

[1] Un ejemplo, aunque usado en otra área, es el mecanismo de ReCaptcha: Los datos ingresados por lo usuarios como confirmación de que hay una persona, son usados luego para mejorar la calidad de mecanismos de reconocimiento de patrones.

#### 2.1.2. Posibles Fuentes de datos

**Compras Estatales:** El sitio `www.comprasestatales.gub.uy` tiene información de las compras de bienes y servicios que hace el estado. Ya hay una parte del trabajo hecho como un *cartridge* de Virtuoso, especialmente para las adjudicaciones. Este trabajo puede ser retomado.

**IMM:** La Intendencia de Montevideo posee varios datasets de los cuales ya tiene datos publicados pero siempre como CSV o Shapes para GIS. Hacer la transformación de los datos puede ser un trabajo interesante, sobre todo porque se puede analizar las ventajas y desventajas de hacer el trabajo de esta forma o con tecnologías tradicionales.

**INE:** Ej: datos del censo y mostrar su relación (por ejemplo geográfica)

**Catalogo de Datos:** El portal `https://catalogodatos.gub.uy/` tiene la información de una serie de datasets publicados. No hay ninguno de estos que estén publicados en RDF. En general, no tienen interfase. Se debería poder tomar un par de datasets de diferentes fuentes y buscar los mecanismos para relacionarlos.

### 2.2. Integración de Diferentes Fuentes

La idea es elegir dos fuentes de datos distintas, posiblemente publicadas por diferentes organizaciones y obtener algún subconjunto de datos interesante que los combine. Luego presentar ese conjunto de alguna forma razonable (gráfica, tabular, grafo navegable, html, etc.). En general, la estrategia debería pasar por definir una ontología capaz que de representar al conjunto combinado y las fuentes y establecer la correspondencia de alguna forma.

#### 2.2.1. Puntos Fuertes

**Diseño de las Ontologías y las correspondencias** Estudiar con un poco de detalle las ontologías que representan esos datos y ante todo, los mecanismos para establecer las correspondencias entre los conceptos.

**Acceso y Visualización:** Desarrollar una estrategia de acceso y mantenimiento de los datos integrados (se copian, se hace todo "al vuelo", etc) y decidir cuáles son los mecanismos de visualización adecuados. En particular, en caso de usar SPARQL Endpoints es interesante el desarrollo de consultas distribuidas o a través de mecanismos de caché.

#### 2.2.2. Posibles Fuentes de datos

La selección de la fuente de datos, se puede realizar:

- Estudiando los datos de `http://linkeddata.org` .
- Eligiendo fuentes de `http://datahub.io/`. Allí habría que elegir datasets que estuvieran en RDF o que tuvieran SPARQL Endpoints.
- Esta es la lista de disponibilidad de sparql endpoints `http://sparqles.ai.wu.ac.at/availability`.

## 2.3. Gestión de Anotaciones en Multimedia

La intención del proyecto es consolidar y completar trabajos ya realizados sobre **OpenFING**.

La idea general es que se puede anotar los fragmentos de video con comentarios, preguntas, materiales o temas (o cualquier otra cosa) permitiendo:

- Su despliege sincronizado con el video.
- Permitir la relación entre videos de diferentes cursos.
- La clasificación de fragmentos de video por temas.
- Búsquedas automáticas de materiales relacionados.
- Análisis de las relaciones establecidas por un grupo de estudiantes para estudiar el aprendizaje.
- (agregue su idea :-) )

### 2.3.1. Puntos Fuertes

**Diseño de las Ontologías y las correspondencias:** Si bien hay una definición de la ontología a utilizar puede necesitar ajustes y extensiones.

**Entrada de Datos:** Las anotaciones deben quedar asociadas a los fragmentos de video, por lo que, esa asociación debe realizarse durante la visualización del video.

**Salida:** El despliegue de la información puede ser realizado de diferentes formas con respecto a la sincronización con el video:

- Sincronizado: la información se despliega en diferentes paneles (preferentemente por tipo de información) en forma sincronizada con el video, similar a los subtítulos.
- Previo a la visualización: la información se despliega de forma previa a la visualización del video y mediante un mecanismo de selección se visualizan los fragmentos de video relacionados.

Obviamente, se debe disponer de diferentes mecanismos de selección de la información a visualizar.

**Análisis:** Desarrollar estrategias de análisis con diferentes objetivos La información recogida, puede ser analizada con diferentes objetivos.

## 2.4. Problemas de Matching

Los problemas de **Matching** son aquellos en los que se debe determinar la relación de instanciación entre una instancia y un esquema o de subesquema de un esquema con respecto a otro. En general se trata de problemas en donde hay que verificar que una determinado conjunto de objetos relacionados cumple con un cierto patrón. En general, este tipo de problemas se puede resolver con mecanismos de razonamiento. Algunos ejemplos de estos problemas, pueden ser:

- Resolución de Licitaciones: El pliego de la licitación se ve como el esquema, las ofertas como las instancias. Si la oferta cumple con las propiedades especificadas por el pliego entonces encontramos que la oferta es una instancia adecuada para el pliego, si no, no lo es.
- Quien es Quien: Es un juego en donde se dan una serie de condiciones que deben cumplir diferentes elementos en diferentes categorías. El esquema está formado por las condiciones y las categorías y la instancia por los elementos y las relaciones entre ellos. (caso analizado en el teórico)
- Determinación de condiciones de funcionamiento de una instalación informática: el esquema está formado por las condiciones y las instancias pueden ser codificaciones de los logs del sistema. Especificando las condiciones correctamente, se pueden clasificar los eventos registrados en el sistema determinando su funcionamiento correcto o incorrecto.

### 2.4.1. Puntos Fuertes

**Diseño de la Ontología:** En estos problemas, el diseño de la ontología es crítico. Si se desea resolver automáticamente, hay que tener algo de cuidado con el perfil OWL que uno termina eligiendo para la especificación.

**Entrada:** La entrada o codificación de los datos para poder resolver el problema de la forma más automática posible es también algo bastante crítico. Ej: en el caso de las licitaciones, es necesario tomar una estrategia de codificación adecuada tanto de los pliegos como de las ofertas. Se impone un sistema de entrada que ya genere una versión codificada? Se analiza el texto libre para obtener las versiones codificadas?

**Salida:** La presentación de los resultados, es al menos interesante. Se va a trabajar con un editor de ontologías ? Se va a disponer de una interfase adecuada de presentación de los resultados?