CAPITULO 4 SELECCIÓN DE PLATAFORMAS PARA IMPLEMENTAR LA ONTOLOGÍA

En el capítulo anterior, se describieron algunas plataformas tentativas para la implementación de la ontología, lo que sigue ahora es analizar y evaluar dichas tecnologías para seleccionar la más apropiada para el proyecto.

Las plataformas por analizar son:

- Protégé (con plugin Datamaster)
- ARC
- RAP
- Triplify

Para poder seleccionar la herramienta adecuada es necesario revisar las necesidades que tiene el proyecto. Antes que nada es necesario un editor de ontologías, además, como se desea alimentar la ontología por medio de bases de datos mysql es necesario una herramienta que ayude a realizar dicha tarea.

4.1 Análisis de las plataformas

A continuación se analizarán las plataformas de una forma detallada para poder seleccionar la o las que se utilizarán para realizar el proyecto.

4.1.1 Protégé

Protégé, es una herramienta que permite editar ontologías que es bastante cómodo y sencillo de utilizar ya que nos brinda las siguientes características.

- Interfaz gráfica de usuario amigable y fácil de adaptarse a ella

- Una gran capacidad de trabajar con frames
- Se puede hacer la herramienta más potente con la ayuda de plugins
- Se pueden cargar aplicaciones a una base del conocimiento

Además de ser un editor de ontologías muy práctico y sencillo de utilizar, se eligió esta plataforma por su facilidad de exportación y conversión entre varios formatos de archivos (Archivos protégé, base de datos protégé, XML, OWL/RDF data base, OWL/RDF, CLIPS, N-triple, N3, OWL, HTML, RDF Schema y Turtle).

DATAMASTER

Datamaster es un plug-in (extensión) de protégé, que sirve para importar la estructura y la información de las bases de datos, ya que muchas veces se busca la integración o la interoperación de la información que se encuentra almacenada en la misma.

Datamaster da la posibilidad de poder utilizar cuatro tipos diferentes de ontologías, 3 ontologías OWL y una de Frames.

Estructura para Protege Frames, en este caso, datamaster la función que realiza es la de importar la estructura de la base de datos hacia la ontología, de la siguiente forma:

- Tablas se convierten en Clases
- Campos se convierten en Slots
- Registros se convierten en Instances
- Llaves Foráneas, por cada llave foránea se crea una instancia de la clase ForeignKeys, la cual será utilizada para relacionar las clases correspondientes.

Estructura donde las tablas de la base de datos se toman como clases OWL, existen diferencias, con relación al trabajar con Frames:

- Las clases ya no serán instancias de una Meta Clase común, lo que conlleva a que todos slots de dicha Meta Clase pasan como propiedades de las clases importadas.

- Se utiliza una nomenclatura diferente, la cual no lleva espacios, ejemplo, Foreign Keys en frames, se importaría como Foreignkeys.

- Se mejoró la navegación, utilizando 4 propiedades adicionales a la clase Foreignkey, haciendo referencia la tabla y a la columna de la llave foránea.

Estructura utilizando las tablas de la base de datos como instancias OWL usando Relational.OWL, se definen 4 clases, DataBase, Table, Colum y PrimaryKey, las instancias de las clases y sus relaciones pueden representar la estructura de cualquier base de datos.

Estructura para tablas de bases de datos usando meta clases, en Relational.OWL las tablas son representadas como individuos de la Clase Table, esto resulta muy útil si lo que se quiere es importar solo la estructura de la base de datos sin la información contenida en ella, este tipo de importación es de mucha ayuda cuando se quiere tener una conexión viva a la base de datos.

4.1.2 Arc

Como ya se describió anteriormente ARC es una plataforma muy rica en varios aspectos, sin embargo, tiene un primer detalle que podría ser un problema, solo es ejecutable bajo plataformas Linux o Unix, lo cual para este caso representa un problema, ya que el diseño y la implementación del proyecto se está haciendo bajo plataforma Windows.

Haciendo a un lado el detalle de la plataforma se tienen que tomar en cuenta las ventajas con las que cuenta la herramienta:

- ARC utiliza programación orientada a objetos para sus componentes y métodos, sin embargo, el procesamiento de las estructuras de información es por medio de simples arreglos asociados, lo que permite un procesamiento más rápido y menos consumo de memoria.

- ARC tiene su estructura en 2 grandes núcleos los “Triplesets” y los “Resource Indexes”

    - Tripleset: Arreglo sencillo que contiene asociados tres arreglos, un triple set puede ser programado con un sencillo bucle.

    - Resource Indexes: Es un arreglo asociativo entre triplesets, los índices pueden ser de dos formas, una que utiliza objetos sin tomar detalles en específico, lo que es útil para operaciones sencillas, pero puede con llevar una pérdida de información, la segunda es aquella que respeta y guarda la estructura del objeto por lo que es menos rápida pero sin pérdida de información.

    - Tiene herramientas para convertir archivos a rdf así como herramientas para rdf storage (guardar la información en un rdf en una base de datos mysql)

    - Soporta el lenguaje SPARQL así como scripts del mismo

    - Soporta 4 tipos de socialización, N-triple, turtle, RDF/JSON, RDF/XML

Como se puede observar, es una herramienta muy completa, y como todas, tiene sus ventajas y desventajas. Procederemos con la siguiente herramienta de las 4 que evaluaremos.

4.1.3 Rap

RAP (RDF API for PHP) es una serie de herramientas para la web semántica destinada a programadores en PHP, dentro de la nueva generación de la web semántica se tiene una tendencia a representar la información de una forma más entendible para la máquina y que esta última puede inferir la semántica de la información, de ahí el nombre de web semántica (RAP, 2002).

Rap es un proyecto que inicio en la Universidad de Berlin en 2002, Rap ofrece una serie de herramientas para, manipular, convertir, almacenar, utilizar sentencias, servir y serializar gráficas RDF.

Trabajar con Graficas RDF en RAP se lleva acabo de la siguiente manera. Las gráficas son representadas como instancias de la clase modelo, cada modelo tiene tres Statements, donde cada uno está compuesto por tres nodos, el sujeto, el predicado y el objeto. Además ofrece tres interfaces para programar la manipulación de las gráficas:

- Statement Centric Model: expone la gráfica RDF como una serie de statements RDF, similar a la estructura del almacenaje por statements.

- Resource-Centric Programming Interface: Representa la gráfica como una serie de recursos que tienen sus propias características, esto lleva a una navegación más amigable por la gráfica, ya que si necesitamos encontrar un recurso, lo buscamos en la colección de dicho recurso.

- Ontology centric Programming Inerface: Este es una API que sirve como extensión al anterior, da soporte a clases ontológicas primitivas (en la herencia de clases), propiedades (en la herencia de las mismas) e

individuos, además no solo soporta el lenguaje ontológico RDF-Schema, también es compatible con partes de OWL cargando un vocabulario.

- NETAPI: RAP cuenta con un servidor RDF para publicar modelos RDF sobre la web y hacerlos accesibles para clientes y aplicaciones remotas, la gran ventaja es que puede ser ejecutado en cual servidor que soporte PHP.

RAP ofrece una gran variedad de herramientas necesarias para el desarrollo del proyecto, especial la posibilidad de servir como un servidor web. Continuando con la evaluación de nuestras herramientas para el desarrollo e implementación del proyecto aún falta Triplify. Se tienen que analizar las herramientas para poder formar un criterio amplio y escoger la óptima para el proyecto.

4.1.4 Triplify

Triplify es una herramienta que permite de una forma simple publicar información ligada a una base de datos. La forma en que trabaja Triplify es la siguiente: toma peticiones HTTP-URI y las mapea en sentencias de bases de datos relacionales, transformando los resultados en statements RDF y publica la información en varias serializaciones RDF. Triplify tiene los siguientes objetivos como herramienta:

- Facilitar a los desarrolladores web el poder publicar RDF triples, información ligada, entre otros de sus aplicaciones web.

- Facilitar a los buscadores de información, el poder obtener información actualizada sin la necesidad de volver a buscar en la información vieja.

- Mostrar la flexibilidad y la escalabilidad que puede tener la información publicada de forma semántica.

Como se dijo previamente, se basa en tomar la estructura de una base de datos relacional y transformarla en gráficas RDF, para lo anterior, se debe

observar la configuración de triplify que es de la siguiente forma: es una tripleta donde (*s* es el nombre del esquema, *o* es el mapeo de prefijos a uris y *u* es el mapeo de urls a sentencias de mysql).

Para poder convertir una sentencia SQL a un modelo de información RDF, triplify sigue un modelo tablas- múltiples a clases para lo que necesita una cierta estructura para poder ser representado en RDF:

- Una columna para el identificador, este será utilizado para generar las uris de las instancias, ejemplo la llave primaria de la base de datos.

- Nombres adecuados para las columnas ya que se utilizan para generar uris de propiedad.

- Celdas individuales contienen la información que después será utilizada para crear los objetos que serán utilizados para crear los triples RDF.

Triplify da la posibilidad de inyectarlo a una aplicación web, y a partir de sus procesos convertir la información que se tiene en nuestra base de datos relacional, en información del tipo RDF para poder aprovechar la semantificación de la misma. Esto ayudaría en caso de querer obtener directamente la información de la aplicación que se tiene para el llenado de los datos y exponer las relaciones semánticas de esa aplicación, a lo que se quiere llegar es que no trabaja propiamente con ontologías, que es lo que se necesita para el proyecto.

4.2 Selección de la herramienta a utilizar

Después de analizar las diferentes herramientas que podían considerarse para desarrollar el proyecto, interesa que se cumpla con los criterios de: facilidad de uso, satisfacer las necesidades del proyecto y factibilidad económica, el ultimo

criterio lo aprobaron todas porque son de uso gratuito, sin embargo, la que destacó por encima de las otras en los otros tres criterios fue y con un alto margen, Protégé con su plugin Datamaster. A continuación las razones puntuales del porqué dicha elección:

1. Datamaster es un plugin de protégé, esto nos da la ventaja de que trabajamos en la misma herramienta donde se desarrolla la ontología.

2. Nos brinda 4 formas diferentes de atacar el problema con la misma herramienta, la primera forma de trabajar es con los frames de protégé, para poder manipularlo como proyecto del mismo protégé; La segunda, importar la base de datos como tablas de owl que es muy parecido a la primera pero con unas pocas de diferencias; la tercera forma es la de importarlo como owl relacional, y por último realizando la importación de tablas por medio de metaclases.

3. Resalta su amabilidad y facilidad de uso, ya que es muy intuitiva su interfaz y el usuario prácticamente solo teclea las direcciones del driver y los parámetros de la base de datos, lo demás se hace con solo presionar botones.

4. El resultado de los procesos de búsqueda es prácticamente instantáneo.

5. Permite además la exportación y conversión entre varios formatos de archivos (Archivos protégé, base de datos protégé, XML, OWL/RDF data base, OWL/RDF, CLIPS, N-triple, N3, OWL, HTML, RDF Schema y Turtle)

Por estas razones que son únicas de protégé se eligió como la herramienta para realizar el diseño de la ontología que será la que hará las inferencias sobre las relaciones de los conceptos que vamos a establecer para el proyecto.