UNIVERSIDAD Rafael Belloso Chacín

Revista Electrónica de Estudios Telemáticos

TÉLÉMATIQUE

ONTOACTION: HERRAMIENTA GRÁFICA Y TEXTUAL PARA LA ESPECIFICACIÓN DE ONTOLOGÍAS ACTIVAS

(OntoACTION: Graphics and textual tool to specify active ontologies)

Recibido: 24/10/2012 **Aprobado:** 06/11/2013

Tovar, Elsa Liliana
Universidad de Carabobo, Venezuela
elsa.tovar@gmail.com

Alvarado, Norimi Coromoto
Universidad de Carabobo, Venezuela.
norimia@gmail.com

RESUMEN

Actualmente las ontologías son consideradas uno de los pilares de la web semántica por todas las ventajas que ellas pueden ofrecer en la representación formal del conocimiento. Paralelamente a esto, a medida que el campo de las ontologías se va desarrollando, la cantidad de usuarios no expertos en lenguajes de etiquetado va creciendo. Surge entonces la necesidad de contar con herramientas automatizadas que permitan una sencilla interacción con el usuario a fin de facilitar la creación y diseño de ontologías de propósito específico. En este trabajo se presenta OntoACTION, una herramienta de software que permita al usuario definir ontologías activas mediante grafos o mediante lenguajes de etiquetado, traduciéndolas al conjunto de axiomas procesables por una máquina razonadora de ontologías ACTION. Para probar la herramienta se especificó de forma gráfica y textual el dominio de los fenómenos climatológicos dado por el Servicio de Meteorología de la Aviación Militar Bolivariana de Venezuela.

Palabras clave: Traductor de ontologías, Editor de ontologías, Ontologías RDF/RDFS, Ontologías activas, Ontologías ACTION.

ABSTRACT

Ontologies are presently considered one of the semantic web columns due to the benefits they provide in representing formal knowledge. At the same time, while the ontologies field develops so does the amount of non-experts users in tag languages. Thus, emerges the need of tools to allow simple interaction with the user, in order to facilitate the creation and design of specified domain ontologies. OntoACTION is presenting in this work which is a software that allows the users to define active ontologies by means of graph and tag languages, translating them into axioms that a reazoner engine of ACTION ontologies can process. To test OntoACTION the climate phenomenal ontologies of the Servicio de Meteorología de la Aviación Militar Bolivariana de Venezuela was specify

Keywords: ontologies translator, ontologies editor, RDF/RDFS ontologies, active ontologies, ACTION ontologies.

Depósito Legal: PPX200002ZU2142 / ISSN:1856-4194. Volumen 13 Edición No 1 – Enero – Junio 2014

INTRODUCCIÓN

El desarrollo de las ontologías busca mejorar la descripción de los conceptos que componen la formalización de un dominio de conocimiento, añadiendo descriptores o metadatos. Los metadatos deben ser expresados adecuadamente para poder ser consultados y evaluados automáticamente por los sistemas que hagan uso de ellos.

Las ontologías pueden escribirse mediante lenguajes etiquetados como el XML (W3C, 2011), que ofrecen las ventajas propias del manejo estándar de información entre sistemas heterogéneos. Por esta razón, existen dialectos de XML con etiquetas con semántica específica como aquellas que representan a los constructores de RDF/RDFS, según W3C Semantic web (2004a) y W3C Semantic web (2004b).

Cualquier documento en XML, incluso si se trata de una ontología, es un archivo de texto que debe ser procesado por algún tipo de máquina de razonamiento (Stanford Center for Biomedical Informatics Research; Haarslev y otros, 1997 y W3C Semantic Web, 2004a) para extraer su información.

Sin embargo, para implementar una ontología se han propuesto tripletas, bases de datos relacionales, bases de datos deductivas, archivos planos y otras implementaciones. Toda implementación de ontologías, a partir de un documento escrito en XML, implica algún tipo de traducción para ser evaluada por la máquina de razonamiento que se va a utilizar.

Este trabajo se ubica dentro del área de los traductores de ontologías, cuya función consiste en partir de una ontología activa escrita en un dialecto de XML y generar a un conjunto de axiomas ontológicos que una máquina de inferencia pueda entender. Adicionalmente, este trabajo propone también un ambiente de edición y modelado de ontologías tanto gráfico como textual para facilitarle al usuario no especializado el diseño, especificación y manipulación de las ontologías.

TRABAJOS RELACIONADOS

Dou, McDermott, y Qi (2003) desarrollaron un servicio en línea para traducción de ontologías llamado Ontomerge. Su idea al desarrollar Ontomerge consistió en traducir ontologías en términos de fusiones de ontologías (ontology merging). Se define la ontología como una especificación formal de un vocabulario de conceptos, incluidos los axiomas sobre estos conceptos.

La fusión de dos ontologías se obtiene tomando la unión de la definición de axiomas y utilizando espacios de nombres XML (namespaces) para evitar conflictos de nombres. Luego, lo complementaron añadiendo puentes de axiomas (bridging axioms) que llevan los términos de una ontología a los términos de la otra.

Sin embargo, Ontomerge carece de una herramienta de edición y creación de nuevas ontologías para usuarios no expertos. Como resultado, Ontomerge está dirigido a una comunidad más especializada y, por lo tanto, más reducida. Otra limitante que debe

Depósito Legal: PPX200002ZU2142 / ISSN:1856-4194. Volumen 13 Edición No 1 – Enero – Junio 2014

enfrentar quien trabaje con Ontomerge es que está solo disponible como servicio en línea y no está disponible para usuarios particulares.

Se desarrolla una herramienta para modelar ontología basada en ORM (Halpin, 2005) bautizada como DogmaModeler (Mustafa, 2011), que busca facilitarle el proceso de modelar una ontología a usuarios no expertos o con muy poco conocimiento de esta tecnología. La base teórica de DogmaModeler considera los principios metodológicos bien definidos de doble articulación y los principios de la modularización (Mustafa, 2005) que facilitan la interacción con el usuario.

El principio de doble articulación sugiere que una aplicación de axiomatización debe ser construida en términos de cada dominio en específico. Mientras que el principio de modularización ontológica plantea que las solicitudes de axiomas sean trabajadas de forma modular. Implementando entonces un conjunto de módulos más sencillos para luego conformar con estos un solo módulo de axiomatización.

A diferencia de lo que se propone la presente investigación, basada en traducción de documentos XML a axiomas de una maquina razonadora tanto para ontologías estáticas como ontologías activas (Tovar y Vidal, 2009), DogmaModeler realiza la correspondencia automática entre los diagramas ORM y la interfaz de lógicas descriptivas (Dou, McDermott, y Qi, 2003) utilizando RACER como máquina de razonamiento.

Ruckhaus, Vidal y Ruiz (2006) presentan una arquitectura para responder eficientemente consultas conjuntivas sobre ontologías estáticas. En esta arquitectura se describe un traductor de ontologías OWL (escritas en documentos XML) donde el traductor toma los axiomas de OWL Lite y los lleva a predicados extensionales e intencionales de una base de datos deductivas. Igualmente (Tovar y Vidal, 2009) desarrollaron un traductor para ontologías activas ACTION mediante reglas DCG (Definite Clauses Grammars) en SWI-Prolog.

Ni Ruckhaus, Vidal y Ruiz (2006) ofrecen un ambiente de edición disponible a usuarios. Dichos trabajos hacen transparente la traducción debido a que ésta tarea forma parte de un marco de trabajo cuyo propósito está enfocado en responder consultas conjuntivas y en procesar razonamiento reactivo, respectivamente, y no en ofrecer un ambiente de edición y traducción a usuarios generales.

METODOLOGÍA

La investigación está guiada por la estrategia de investigación empírica denominada “Investigación Acción” (Baskerville, 1999), que utiliza una combinación de metodologías cuantitativas y cualitativas, que se detallan a continuación:

a) Fase de diagnóstico: está relacionada con la identificación y descripción de la situación actual.

b) Fase de planificación de la acción: define las acciones que deben ser ejecutadas para mejorar el problema.

Depósito Legal: PPX200002ZU2142 / ISSN:1856-4194. Volumen 13 Edición No 1 – Enero – Junio 2014

c) Fase de implementación de la acción: se lleva a cabo la acción planificada.

d) Fase de evaluación: una vez culminadas las acciones, se evalúan las salidas obtenidas.

e) Fase de especificación del aprendizaje: corresponde a la reflexión sobre los resultados de la fase de evaluación, lo cual podría dar inicio a una nueva iteración.

Por otro lado, ya que la investigación comprende la fase de desarrollo de software en el área de grandes repositorios de datos, se siguió la propuesta metodológica desarrollada en la Facultad de Ciencias y Tecnología de la Universidad de Carabobo, entendiéndose por MeDPE las siglas que la identifican: Metodología para el Desarrollo de Portales Educativos.

Sus autores la definen como "una propuesta metodológica ágil, centrada en el usuario la cual basa su proceso en una constante aplicación iterativa de las actividades de evaluación, considerando la usabilidad desde un principio, a la vez que integra actividades propias de la Ingeniería del Software". (Giugni, 2008).

RESULTADOS

FASE DE DIAGNÓSTICO

Esta fase de la investigación tuvo como objetivo estudiar cuales son las necesidades que tienen los usuarios no expertos en lenguajes de la web semántica, para definir ontologías que le permitan describir sus datos y la definición de un conjunto de pasos para encontrar soluciones a esos problemas.

Al culminar esta primera etapa los principales en el diagnóstico fueron:

- Se determinó que una manera de facilitar la generación de ontologías a los usuarios no expertos en lenguajes de etiquetado, puede ser la representación gráfica de las ontologías y puesto que es toda ontología puede representarse en forma de grafo. Se parte de la hipótesis que será más sencillo la definición de los conceptos ontológicos en modo gráfico.

- A partir del punto anterior se determinó la necesidad de contar con una representación gráfica formal para representar ontologías que sea usada por el software a desarrollar.

- Al realizar la investigación bibliográfica se observó también que para usuarios más avanzados sería de utilidad contar con un ambiente de edición de documentos XML, que ofrezca "fragmentos vacíos" de etiquetas que puedan ser llenados por el usuario para generar los documentos XML.

- En el diagnóstico también se observó la necesidad de hacer la correspondencia entre los símbolos gráficos y las etiquetas en XML con los axiomas de las ontologías.

Depósito Legal: PPX200002ZU2142 / ISSN:1856-4194. Volumen 13 Edición No 1 – Enero – Junio 2014

FASE DE PLANIFICACIÓN E IMPLEMENTACIÓN DE LA ACCIÓN

Una vez realizado el diagnostico, se procesó a planificar el trabajo a realizar. El plan de trabajo se definió de la siguiente manera:

a) Revisión bibliográfica. Estudio de los trabajos relacionados y antecedentes, así como análisis de las propuestas existentes que más se aproximan a la solución del problema.

b) Se realizó una exhaustiva investigación con respecto a los trabajos que se han hecho para traducir y editar ontologías, para determinar cuáles son los axiomas que las componen y los datos y metadatos que requiere una máquina razonadora para procesar consultas de ontologías. También se revisaron trabajos relacionados con ontologías activas, su representación física y su implementación

c) Estudio de representaciones gráficas de ontologías.

d) Diseñar la interfaz de edición de ontologías tanto en modo gráfico como en etiquetas.

e) Implementar la herramienta que traduzca las ontologías ingresadas por el usuario a los axiomas que procesan una máquina razonadora.

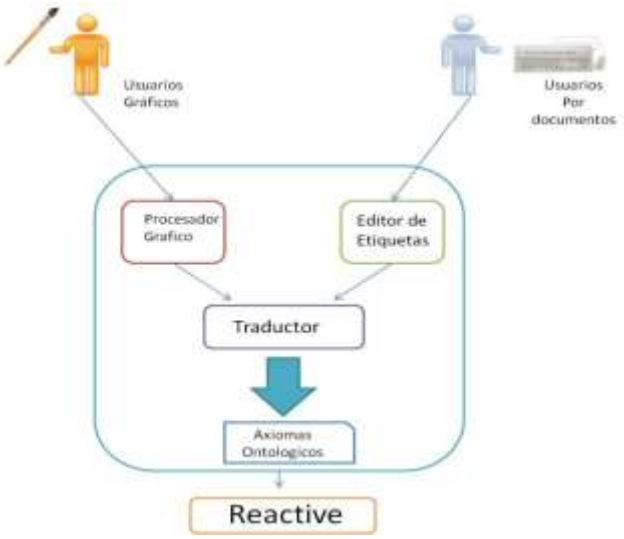
DISEÑO E IMPLEMENTACIÓN DE LA SOLUCIÓN

El trabajo de esta etapa de la investigación se centró en proponer el diseño de la solución al problema planteado y se establecieron todos los aspectos que estructuran la implementación de dicha solución. Para comprender el contexto en el cual se hizo la propuesta de solución, se presenta la Figura 1. Tal como se muestra, existen dos tipos de usuarios: aquellos que trabajan bajo el ambiente de la edición gráfica y los que trabajan con la edición de documentos en lenguaje etiquetado.

El usuario modo gráfico es gestionado bajo el módulo del Procesador Gráfico, el cual se encarga de proveer al usuario de símbolos gráficos para la creación de ontologías. Por otra parte, el usuario modo textual trabaja con el módulo del Editor de Etiquetas para generar documentos en XML donde se modela una ontología línea por línea.

Depósito Legal: PPX200002ZU2142 / ISSN:1856-4194. Volumen 13 Edición No 1 – Enero – Junio 2014

Figura 1. Arquitectura de software para la solución planteada



Fuente: elaboración propia

RESULTADOS

En principio se presenta la definición de representación gráfica definida en este proyecto para las ontologías ACTION. Luego se describirán los constructores que forman parte de la serialización de XML para ontologías ACTION seguidamente se describirán el conjunto de predicados con semántica predefinida que corresponde con los constructores de la serialización de XML para ACTION. Finalmente, se presenta una ontología activa especificada con la herramienta de software desarrolla mediante la metodología MeDPE.

REPRESENTACIÓN GRÁFICA DE LAS ONTOLOGÍAS ACTION Y PREDICADOS ADOB

La estructura asociada a una sentencia de las ontologías ACTION es un grafo dirigido etiquetado G, definido de la siguiente manera: G=(N,A,R) donde N es el conjunto de nodos que representan las clases, los eventos, las propiedades, los literales y los tipos de datos; A es el conjunto de arcos que conectan los nodos y que representan a las propiedades activas y estáticas de los datos y de los metadatos y R es el conjunto de arcos que representan las condiciones booleanas del comportamiento de las propiedades activas.

Los arcos de un grafo ACTION son etiquetados con URIs. Los nodos de un grafo ACTION pueden ser etiquetados con URIs o literales. Una sentencia o expresión de un grafo ACTION representa un predicado de ADOB que depende del tipo de etiqueta del arco. A continuación se describen los predicados por cada tipo de etiqueta para los arcos de A.

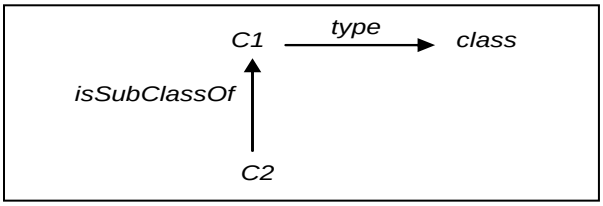
Depósito Legal: PPX200002ZU2142 / ISSN:1856-4194. Volumen 13 Edición No 1 – Enero – Junio 2014

CLASES Y SUBCLASES

Suponiendo que se tiene el grafo de la Figura 2, asumiendo que la ontología a la cual pertenecen las clases C1 y C2 se denomina llama O, entonces los predicados de ADOB correspondientes representan los metadatos asociados con C1 y C2 de la siguiente forma: isClass (C1), isSubClassOf (C2, C1).

No es necesario indicar que C2 es una clase, pues al definirse de forma explícita la relación isSubClassOf con C1 y de forma explícita la relación type de C1 con class, una máquina razonadora podrá inferir el predicado isClass (C2,O).

Figura 2. Ejemplo de grafo clases y subclases

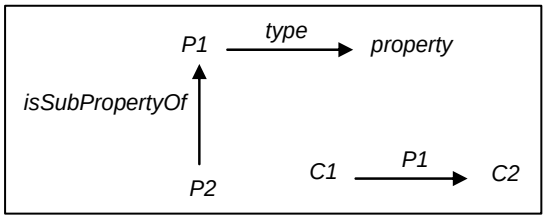


Fuente: elaboración propia.

PROPIEDADES Y SUBPROPIEDADES

Suponiendo que se tiene el grafo de la Figura 3, los predicados de ADOB que corresponden con el grafo son aquellos que se expresan explícitamente isProperty (P1, C1, C2), isSubPropertyOf (P2, P1). Una máquina razonadora puede inferir el predicado isProperty (P2,C1,C2).

Figura 3. Ejemplo de propiedades y subpropiedades



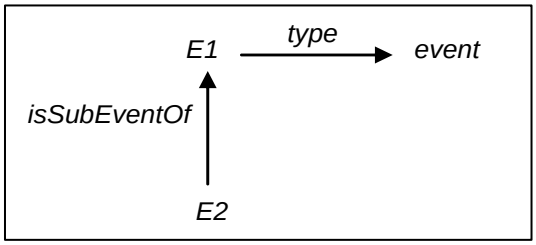
Fuente: elaboración propia

EVENTOS Y SUBEVENTOS

Suponiendo que se tiene el grafo de la Figura 4, los predicados de ADOB correspondientes representan los metadatos asociados con P1 y P2 de la siguiente forma: isEvent (E1), isSubEventOf (E2, E1). No es necesario indicar que E2 es un evento pues al definirse de forma explícita la relación isSubEventOf con E1 y de forma explícita la relación type de E1 con event, una máquina razonadora podrá inferir el predicado isEvent(E2).

Depósito Legal: PPX200002ZU2142 / ISSN:1856-4194. Volumen 13 Edición No 1 – Enero – Junio 2014

Figura 4. Ejemplo de eventos y subeventos

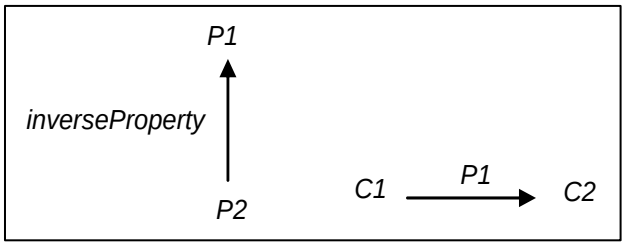


Fuente: elaboración propia.

PROPIEDADES INVERSAS

Suponiendo que se tiene el grafo de la Figura 5, los predicados de ADOB correspondientes representan los metadatos asociados con P1, P2, C1 y C2 de la siguiente forma inverseProperty (P2, P1), isProperty (P1, C1, C2). Una máquina razonadora puede inferir el predicado isProperty (P2, C2, C1).

Figura 5. Ejemplo de las propiedades inversas

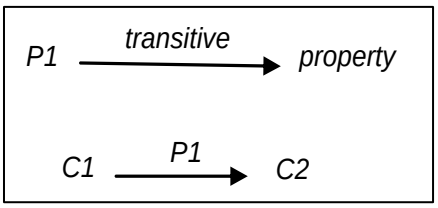


Fuente: elaboración propia.

PROPIEDADES TRANSITIVAS

Suponiendo que se tiene el siguiente grafo de la Figura 6, el predicado de ADOB que corresponde con este grafo es transitiveProperty(P1,C1,C2)

Figura 6. Ejemplo de las propiedades transitivas



Fuente: elaboración propia.

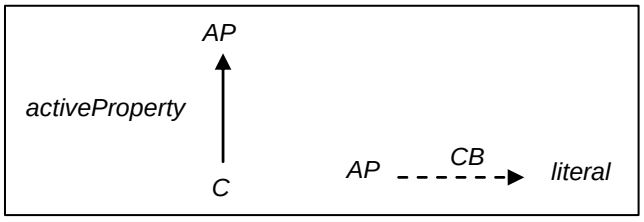
Depósito Legal: PPX200002ZU2142 / ISSN:1856-4194. Volumen 13 Edición No 1 – Enero – Junio 2014

PROPIEDADES ACTIVAS

Suponiendo que se tiene el siguiente grafo de la Figura 7, C es una clase que tiene una propiedad activa AP que es definida en el dominio de conocimiento. La propiedad activa tomará el valor de un literal RV si ocurre un evento E y se cumple la condición booleana CB.

La CB es de la forma: (P **symbol** V), donde P es una propiedad (que puede ser estática o activa), **symbol** es cualquier operador matemático válido para comparar y V es el valor que toma P en CB la hace verdad. Entonces, los predicados ADOB correspondientes son activeProperty (AP, C, literal) y reactiveBehavior (AP, E, P, S, V, RV).

Figura 7. Ejemplo de las propiedades activas

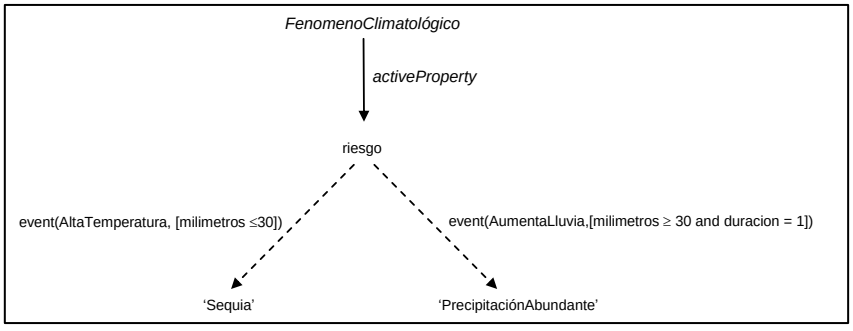


Fuente: elaboración propia.

No siempre una sentencia de un grafo ACTION corresponde con un solo predicado en ADOB, debido a que las propiedades activas toman su valor dependiendo de si se cumple CB y CB puede estar compuesta por varias subcondiciones, expresadas en las etiquetas de los arcos del conjunto R.

Estas condiciones pueden ser complejas de acuerdo a los diferentes eventos que pueden afectar a la propiedad activa y/o a las condiciones que se deben cumplir. Por ejemplo, suponga que se toma el dominio de Climatología y se tiene el grafo ACTION de la Figura 8.

Figura 8. Ejemplo de Grafo ACTION



Fuente: elaboración propia.

Depósito Legal: PPX200002ZU2142 / ISSN:1856-4194. Volumen 13 Edición No 1 – Enero – Junio 2014

Si bien, el primer arco entre el nodo 'riesgo' y el nodo 'Sequía' corresponde con un predicado reactiveBehavior de la forma reactiveBehavior (riesgo, AltaTemperatura, milímetros, >=, 30, 'Sequia'), en el caso del arco que conecta el nodo 'riesgo' y el nodo 'PrecipitacióAbundante' corresponde con un conjunto de predicados reactiveBehavior.

Para ese arco habrá tantos predicados reactiveBehavior como términos haya en la expresión booleana:
reactiveBehavior(riesgo,AumentaLluvia,milímetros,>=,30,'PrecipitacionAbundante')
reactiveBehavior(riesgo,AumentaLluvia,duracion,=,1,'PrecipitacionAbundante').

CONSTRUCTORES ACTION EN XML Y PREDICADOS ADOB

Los constructores XML definidos en este trabajo de investigación fueron reformulados a partir de Tovar y Vidal (2010) y se describen con sus correspondientes predicados ADOB a continuación. En los fragmentos presentados a continuación se usarán los prefijos rdf, rdfs, owl y aowl para indicar el espacio de nombres desde donde se toman las definiciones de los constructores usados en los distintos fragmentos XML.

LA ONTOLOGÍA Y SUS CLASES DIRECTAS

Dado el fragmento en XML, mostrado en la Figura 9, de la definición de una ontología y sus clases se asumen las definiciones del lenguaje OWL y se especifican los correspondientes predicados ADOB usados en la máquina de inferencia REACTIVE. Los predicados ADOB asociados a estos fragmentos de XML de la Figura 9, son isClass (C1,O) y isSubClassOf (C1, C2).

Figura 9. Ejemplo de encabezado y definición de clase directas de un archivo XML ACTION

```
<owl:Ontology rdf:about="">
 <rdfs:comment> </rdfs:comment>
 <rdfs:label>O</rdfs:label>
 <owl:versionInfo> </owl:versionInfo>
</owl:Ontology>

<classCLS>
<owl:Class rdf:ID="C1">
 <rdfs:label> </rdfs:label>
 <rdfs:subClassOf rdf:resource="#C2" />
</owl:Class>
…
</classCLS>
```

Fuente: elaboración propia.

Depósito Legal: PPX200002ZU2142 / ISSN:1856-4194. Volumen 13 Edición No 1 – Enero – Junio 2014

LA ONTOLOGÍA Y LAS CLASES DEFINIDAS CON INTERSECCIÓN Y SUBCLASES

Los predicados ADOB asociados a los fragmentos de XML de las Figuras 10 y 11 son isClass (C1, O) y isSubClassOf (C1, C2). Por otra parte, el único predicado de los fragmentos de las Figuras 12 y 13 es isClass (C1, O).

Figura 10. Ejemplo definición de clases con intersecciones y subclases de un archivo XML ACTION

```
<classCLIS>
<owl:Class rdf:ID="C1">
 <rdfs:label> </rdfs:label>
 <owl:intersectionOf
rdf:parseType="Collection">
 <owl:Class rdf:about=" " />
 <owl:Restriction>
 <owl:onProperty rdf:resource=" " />
 <owl:someValuesFrom>
 <owl:Class rdf:about=" " />
 </owl:someValuesFrom>
 </owl:Restriction>
 </owl:intersectionOf>
 <rdfs:subClassOf rdf:resource="#C2" />
</owl:Class>
</classCLIS>
```

Fuente: elaboración propia.

Figura 11. Ejemplo definición de clases mediante etiquetas de subclases y restricciones de un archivo XML ACTION

```
<classCLSR>
<owl:Class rdf:ID="C1">
 <rdfs:label> </rdfs:label>
 <rdfs:subClassOf rdf:resource="#C2" />
 <rdfs:subClassOf>
 <owl:Restriction>
 <owl:onProperty rdf:resource=" " />
 <owl:someValuesFrom>
 <owl:Class rdf:about=" " />
 </owl:someValuesFrom>
 </owl:Restriction>
 </rdfs:subClassOf>
</owl:Class>
</classCLSR>
```

Fuente: elaboración propia.

Depósito Legal: PPX200002ZU2142 / ISSN:1856-4194. Volumen 13 Edición No 1 – Enero – Junio 2014

Figura 12. Ejemplo definición de clases solo con etiquetas de intersección de un archivo XML ACTION

```
<classCLI>
<owl:Class rdf:ID="Director">
 <rdfs:label>director</rdfs:label>
 <owl:intersectionOf rdf:parseType="Collection">
 <owl:Class rdf:about="#Person" />
 <owl:Restriction>
 <owl:onProperty rdf:resource="#headOf" />
 <owl:someValuesFrom>
 <owl:Class rdf:about="#Program" />
 </owl:someValuesFrom>
 </owl:Restriction>
 </owl:intersectionOf>
</owl:Class>
</classCLI>
```

Fuente: elaboración propia.

Figura 13. Ejemplo definición de clases solo con etiquetas label de un archivo XML ACTION

```
<classCL>
<owl:Class rdf:ID="Organization">
 <rdfs:label>organization</rdfs:label>
</owl:Class>
</classCL>
```

Fuente: elaboración propia.

PROPIEDADES-OBJETO DEFINIDAS CON DOMINIO Y RANGO

Los fragmentos de XML mostrados en la Figura 14, aun siendo diferentes en la serialización XML, se traducen a predicados ADOB similares de la forma sProperty (P, D, R), isProperty (P1, D, R), inverseProperty (P1, P2) y isProperty (P1, D, R), isSubPropertyOf (P1, P2) respectivamente.

DEFINICIÓN DE EVENTOS

Las etiquetas de XML para ontologías ACTION que expresan la reactividad se muestran en la Figura 15, donde aparecen las etiquetas para definir eventos, subeventos y propiedades activas. Los predicados ADOB correspondientes son isEvent(E1), isSubEventOf(E1,E2) y activeProperty(AP,_,D,_), reactiveBehavior(AP,E,P,S,V,RV), donde S={≥,≤,>,<,≠,=}

Depósito Legal: PPX200002ZU2142 / ISSN:1856-4194. Volumen 13 Edición No 1 – Enero – Junio 2014

Figura 14. Ejemplo de varios fragmentos de definición de propiedades estáticas de un archivo XML ACTION

```
 <opDR>
 <owl:ObjectProperty rdf:ID="P">
  <rdfs:label> </rdfs:label>
  <rdfs:domain rdf:resource="#D" />
  <rdfs:range rdf:resource="#R" />
 </owl:ObjectProperty>
 </opDR>

<opDRI>
<owl:ObjectProperty rdf:ID="P1">
 <rdfs:label> </rdfs:label>
 <rdfs:domain rdf:resource="#D" />
 <rdfs:range rdf:resource="#R" />
 <owl:inverseOf rdf:resource="#P2"/>
</owl:ObjectProperty>
</opDRI>

<opDRS>
<owl:ObjectProperty rdf:ID="P1">
 <rdfs:label> </rdfs:label>
 <rdfs:domain rdf:resource="#D" />
 <rdfs:range rdf:resource="#R" />
 <rdfs:subPropertyOf rdf:resource="#P2" />
</owl:ObjectProperty>
</opDRS>

<opLS>
<owl:ObjectProperty rdf:ID="P1">
 <rdfs:label> </rdfs:label>
 <rdfs:subPropertyOf rdf:resource="#P2"/>
</owl:ObjectProperty>
<opLS>
```

Fuente: elaboración propia.

Depósito Legal: PPX200002ZU2142 / ISSN:1856-4194. Volumen 13 Edición No 1 – Enero – Junio 2014

Figura 15. Ejemplo de fragmentos de definición de eventos y de propiedades activas de un archivo XML ACTION

```
<events>
<aowl:Event rdf:ID="E1">
 <aowl:subEventOf rdf:resource="#E2"/>
</aowl:Event>
</events>

<activeProperties>
<aowl:ActiveProperty rdf:ID="AP">
 <rdfs:domain rdf:resource="#D"/>
 <aowl:condition>
 <aowl:refEvent rdfs:about="#E"/>
 <aowl:complex>
 <aowl:refProperty rdfs:about="#P"/>
 <aowl:symbol rdfs:about="S"/>
 <aowl:BooleanCond aowl:bc="V"/>
 </aowl:complex>
 …
 <aowl:reactiveValue aowl:v="RV"/>
 </aowl:condition>
 …
</aowl:ActiveProperty >
</activeProperties>
```

Fuente: elaboración propia.

ESTUDIO EXPERIMENTAL

Para probar la herramienta definiendo una ontología activa de un dominio real se especificó, tanto gráfica como textualmente, la ontología climatológica del Servicio de Meteorología de la Aviación Militar Bolivariana. En la Figura 16, se muestra el menú principal de OntoACTION mostrando el encabezado del XML para ACTION y en la Figura 17 se muestra el grafo que corresponde con esa misma ontología.

CONCLUSIONES

Como una manera de atender la necesidad de facilitar a un usuario no experto el diseño y especificación de ontologías básicas RDF/RDFS y las ontologías ACTION, se desarrolló esta investigación produciendo el software OntoACTION, para la especificación de ontologías de forma gráfica y textual.

Para proveer a la herramienta de software de las facilidades gráficas de diseño se procedió a hacer la formalización de lo que sería el grafo de ontologías ACTION. Debido a que las ontologías ACTION representan tanto conocimiento estático como activo, el grafo resultante incluye dos tipos de arcos, uno para cada tipo de conocimiento.

Depósito Legal: PPX200002ZU2142 / ISSN:1856-4194. Volumen 13 Edición No 1 – Enero – Junio 2014

Con esto se tiene un arco para definir propiedades y otro tipo de arco para definir el comportamiento reactivo de las propiedades activas (a través de los arcos complex). Así mismo, y respetando la definición original de las ontologías ACTION, los eventos que el usuario pueda definir en su dominio, son tratados al mismo nivel de las clases en el grafo para ACTION, razón por lo cual solo se tiene un tipo de nodo que representa conceptos.

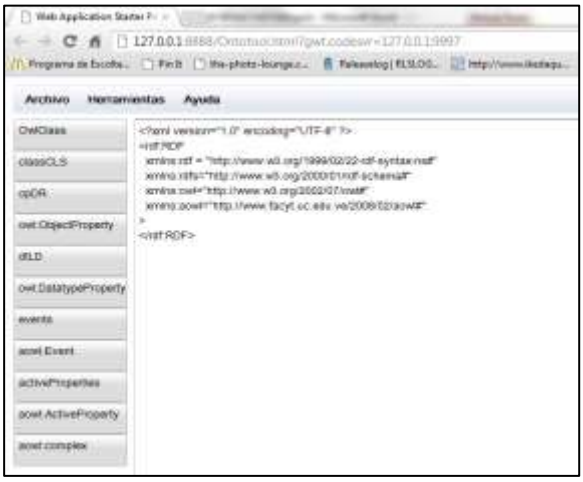
Una vez definido este tipo de grafo, OntoACTION contó con la representación gráfica a modelar. Cabe destacar que las facilidades gráficas están asistidas mediante cajas de diálogo que permiten delimitar solo las definiciones válidas. Por ejemplo, no se puede indicar en una condición booleanas de un arco complex, una propiedad que no existe.

Este control de restricciones es único en la elaboración de ontologías ACTION ya que en el modo textual se puede incurrir en errores de nombres de propiedades o de propiedades inexistentes, debido a la libertad que ofrece XML de aceptar todas las etiquetas definidas por el usuario con la única condición de que el documento esté bien formado. Cuando este tipo de errores ocurre una máquina razonadora no puede llegar a conclusiones válidas o no puede determinar contradicciones.

Con respecto al traductor del modo textual a los axiomas ontológicos de ADOB, fue necesario determinar dos conjuntos: el conjunto de los constructores que forman parte de la serialización de XML que corresponden con las ontologías ACTION y el conjunto de predicados con semántica predefinida que corresponden con los constructores XML de ACTION.

De la anterior tarea se obtuvo un tipo de etiqueta denominada <ActiveProperty> que contiene elementos anidados como <condition> y <complex> que permiten definir el comportamiento reactivo y se determinó que por cada condición booleanas se debe asociar un predicado con semántica predefinida reactiveBehavior de ADOB.

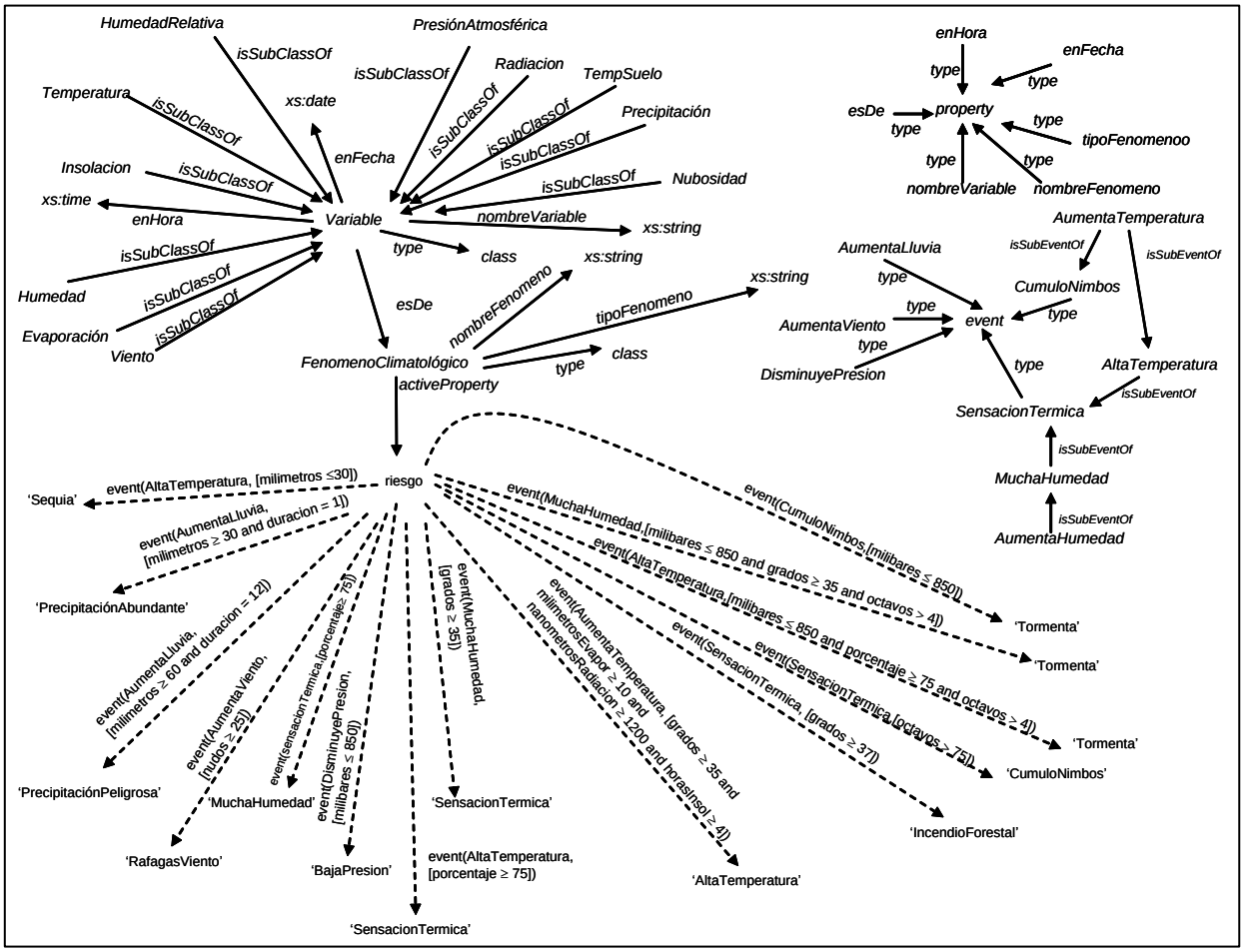
Figura 16. Menú principal de la herramienta OntoACTION



Fuente: elaboración propia.

Depósito Legal: PPX200002ZU2142 / ISSN:1856-4194. Volumen 13 Edición No 1 – Enero – Junio 2014

Figura 17. Grafo para ontologías ACTION que usa la herramienta OntoACTION. Caso ontología Climática de Venezuela



Fuente: elaboración propia.

REFERENCIAS BIBLIOGRÁFICAS

Baskerville, R. (1999). Investigating informatyon system with action research. Journal Comunnications of the AIS. Vol. 2, Noviembre, Art. 4.

Dou, D.; McDermott, D. y Qi, P. (2003). Ontology translation on the semantic web. On The Move to Meaningful Internet Systems 2003: CoopIS, DOA, and ODBASE - OTM Confederated International Conferences. Italia, Noviembre.

Giugni, L. (2008). Metodología para el desarrollo de portales centrada en el usuario: una evaluación empírica. Revista Telematique, Vol. 7, Núm. 3, Pp. 54-70.

Depósito Legal: PPX200002ZU2142 / ISSN:1856-4194. Volumen 13 Edición No 1 – Enero – Junio 2014

Haarslev, V.; Möller R.; Hidde, K. y Wessel, M. (1997). Renamed Abox and Concept Expression Reasoner. Documento en línea. Disponible en: http://www.sts.tu-harburg.de/~r.f.moeller/racer/. Consulta: 25/04/2011.

Halpin, T. (2005) Object Role Modeling ORM. Documento en línea. Disponible en: http://www.orm.net/. Consulta: XX/04/2011.

Mustafa, J. (2005). Towards methodological principles for ontology engineering. Tesis para optar por el título de PhD. Vrije Universiteit Brusse. Bélgica.

Mustafa, J. (2011). DogmaModeler. Documento en línea. Disponible en: http://www.jarrar.info/Dogmamodeler/. Consulta: XX/04/2011.

Ruckhaus, E.; Vidal, M. y Ruiz, E. (2006). Query Evaluation and Optimization in the Semantic Web. Best Paper Award. International Workshop on Applications of Logic Programming in the Semantic Web and Semantic Web Services.

Tovar E., Vidal, M. (2010). Expressing and Managing Reactivity in the Semantic Web. Ontologies and Databases and Applications of Semantics (ODBASE 2010).

Tovar, E.; Vidal, M. (2009). Reactive: A Rule_based Framework to Process Reactivity In Proceedings of The 3rd. International Workshop on Dynamics Ontology, in conjunction with 8th International Semantic Web Conference. Estados Unidos.

W3C (2011). Extensible Markup Language (XML). Documento en línea. Disponible en: www.w3.org/XML. Consulta: 25/04/2011.

W3C Semantic Web (2004a). Resource Description Framework RDF. Documento en línea. Disponible en: http://www.w3.org/RDF/. Consulta: XX/04/2011.

W3C Semantic Web (2004b). RDF Vocabulary Description Language 1.0: RDF Schema RDFS. Documento en línea. Disponible en: http://www.w3.org/TR/rdf-schema/. Consulta: 25/04/2011.

Depósito Legal: PPX200002ZU2142 / ISSN:1856-4194. Volumen 13 Edición No 1 – Enero – Junio 2014