Domain Specific Language for Handling Modular Ontologies

Christian Humberto Cabrera Jojoa, Olga Mariño Drews *

***Abstract*—A Knowledge Object may be characterized in different forms, corresponding to different perspectives of the object. In the same sense, knowledge of a particular domain can be organized in different interconnected ontologies. The research on ontology modularization has advanced in proposing formalisms and tools to extend ontologies with inter-ontology connectors. Tools are needed to manipulate these new connectors. Existing tools are platform dependent, which presents problems of adaptability, portability and reusability. This paper presents the design and implementation of a Specific Domain Language for Handling Modular Ontologies, based on a model driven architecture (MDA).**

***Palabras Claves*— Knowledge management, MDA architecture, ε-connections, metamodels, modular ontologies.**

I. INTRODUCCIÓN

La representación de conocimiento es un proceso que modela una realidad compleja, abstrayendo de ella un conjunto de conceptos, propiedades y relaciones. El conjunto abstraído depende de la intención o perspectiva desde donde se observe el mundo representado. Por ejemplo, en el campo de la informática educativa un simulador de movimiento oscilatorio en física, puede verse desde su perspectiva pedagógica como un objeto de aprendizaje activo basado en un modelo constructivista, desde la perspectiva tecnológica como una simulación ejecutable en determinado ambiente y desde el punto de vista del dominio como un objeto de aprendizaje que refuerza los conocimientos de cinemática en física. Cada una de estas perspectivas describe un conocimiento que puede modelarse en una ontología. Así, por ejemplo la ontología pedagógica incluirá una primera partición en estrategias expositivas, activas e interactivas y dentro de las estrategias activas estarán las de construcción, simulación, análisis de casos, etc. Más aún, existen relaciones semánticas entre estas perspectivas. Se puede por ejemplo afirmar que el aprendizaje de ciertos temas de la física (ontología de dominio) sólo se logra mediante pedagogías activas (ontología pedagógica).

978-1-4799-6717-9/14/$31.00 ©2014 IEEE

* Grupo de investigación TICSw, Departamento de Ingeniería de Sistemas y Computación, Universidad de los Andes.

La investigación en representaciones de conocimiento multiperspectivas no es nueva [1], [2]. Sin embargo, avances en la investigación en ontologías modulares permiten darle una nueva mirada al problema y a su solución. En efecto, los objetivos de la modularización de conocimiento son dobles. Por una parte, se busca reducir la complejidad de un modelo de conocimiento partiéndolo en modelos menores y por otra parte, se busca poder integrar diferentes modelos que describen una misma realidad desde diferentes puntos de vista, para crear una representación completa, se encuentra la creación de modelos ontológicos que describan la información teniendo en cuenta los diferentes aspectos que conforman su contexto. Lo anterior ha impulsado el desarrollo de técnicas que permiten enlazar ontologías que describan un mismo dominio, desde su perspectiva. Basándose en el principio conocido como divide y vencerás la modularización propone crear modelos de conocimiento complejos, a partir de un conjunto de módulos de menor tamaño[3]. Es esta segunda acepción la que interesa explorar en este trabajo: ¿Cómo modelar e interconectar ontologías de puntos de vista diferentes de una realidad?

Por otra parte, se busca que el modelo y sus mecanismos de manipulación sean independientes de la plataforma de implementación, de forma que se asegure adaptabilidad, portabilidad y reusabilidad. Así la pregunta completa que se busca trabajar es: ¿Cómo modelar y manipular múltiples ontologías y sus interconexiones en un modelo independiente de la plataforma?

Desde el área de ontologías modulares se han planteado diferentes formalismos para establecer enlaces semánticos entre ontologías y algunos grupos de investigación han propuesto herramientas para modelar estas conexiones, basadas en dichos formalismos. Las herramientas analizadas son dependientes de la plataforma y no permiten manipular el modelo ontológico modular como un todo ya que ofrecen espacios descentralizados e independientes para la edición y procesamiento de los módulos.

A partir de lo anterior, en este artículo se propone la creación de un lenguaje de dominio específico para la manipulación de ontologías modulares, independiente de la plataforma, llamado MOWL (Modular Web Ontology Language). El desarrollo de este lenguaje se basa en el enfoque de Ingeniería Dirigida por Modelos (MDE) y básicamente propone la extensión del

lenguaje OWL[4], [5], a nivel de metamodelo, con elementos que permitan modelar ontologías modulares a partir del modelo formal presentado en la teoría de ε-connections[6]. Para permitir una manipulación completa de los modelos de conocimiento, se implementa un módulo de inferencia basado en Jess.

Este artículo empieza por hacer una presentación del contexto de web semántica, ontologías, modularización y MDE en la sección 2. Luego presenta la visión general de la solución, en la sección 3. Posteriormente, define la extensión y los metamodelos utilizados, para dar paso, a la descripción de la cadena de transformación y la herramienta desarrollada, en las secciones 4 y 5. Finalmente, se presenta brevemente la evaluación realizada en el campo de la informática educativa, así como las conclusiones y trabajos futuros que surgen a partir de esta investigación, en las secciones 6 y 7.

II. CONTEXTO

A. *Web Semántica y Ontologías*

La web semántica aparece como una extensión de la web actual[7], como respuesta a la necesidad de automatizar procesos y organizar, a nivel semántico, la información disponible en internet facilitando y mejorando los procesos de recuperación, manipulación y gestión de la misma. En las ciencias de la computación, las ontologías, representaciones formales y explicitas de una conceptualización compartida[8], fueron adoptadas en inteligencia artificial para facilitar la reutilización y el intercambio de información, a través del modelado del conocimiento de dominios específicos que involucra el vocabulario o los conceptos, teniendo en cuenta las interconexiones semánticas, reglas de inferencia y restricciones. Dicho modelo debe hacerse de manera formal, no ambigua, consistente e independiente de los detalles de implementación, para ello se utilizan diferentes lenguajes de descripción, especialmente diseñados para definir ontologías. Entre estos lenguajes se encuentra Ontology Web Language (OWL). Este lenguaje describe clases, propiedades y relaciones entre objetos conceptuales de tal forma que facilita la interoperabilidad entre máquinas y el procesamiento del contenido web. OWL es organizado de forma incremental en tres sublenguajes: OWL Lite, OWL-DL y OWL-Full[9]. A nivel comercial, existen diferentes herramientas que permiten crear y manipular ontologías de forma individual, entre ellas están: Protégé[2] y SWOOP[3], basadas en el lenguaje OWL.

B. *Ontologías Modulares y Multiontologías*

La modularización de ontologías busca, por una parte, reducir la complejidad de una ontología partiéndola en varias ontologías más sencillas y por otra parte enriquecer los modelos ontológicos que describen la información, permitiendo que varias ontologías anoten, desde distintos puntos de vista, un mismo objeto. Estas ontologías son creadas de forma individual y posteriormente se enlazan de acuerdo a las relaciones que puedan tener alrededor de un dominio. Esto con el fin, de describir un todo a través de un modelo multiontológico[3].

Existen diferentes formalismos que buscan utilizar elementos semánticos para establecer enlaces entre ontologías, entre ellos: DDL (Distribution Description Logics), P-DL (Package-based Description Logics) y ε-connections[6], [10]. La idea principal de este último formalismo es conectar ontologías que describen diferentes aspectos del dominio, a través de relaciones que establecen conexiones entre estos aspectos. Estos enlaces o ε-connections pueden ser múltiples y cuentan con un significado propio, lo que hace posible tener mayor expresividad y hacer inferencias más complejas sobre el modelo de conocimiento. Esta técnica establece que los módulos relacionados deben ser disyuntos, esto quiere decir que la técnica garantiza una mayor consistencia en los modelos desarrollados[3],[11],[10].

Un ε-connection se define como un conjunto de ontologías conectadas y una ontología *ε-connected* contiene información sobre sus clases, propiedades e individuales, además de la información referente a un nuevo elemento llamado *link property*, que en OWL es similar al elemento datatype property. Este link property permite establecer los enlaces entre las ontologías y debe ser interpretado como una relación binaria cuyo dominio se encuentra en la ontología fuente y el rango en la ontología destino. Debido a las ventajas que ofrece la teoría de *ε-Connections,* en términos de consistencia y razonamiento[6], [11], esta técnica es la que se tiene en cuenta, inicialmente, para desarrollar el lenguaje de dominio específico en esta investigación.

En esta línea, el trabajo que más se acerca a nuestros intereses es el desarrollado en la Universidad de Maryland[11] que plantea una extensión sobre el lenguaje OWL que permite expresar *ε-connections.* La extensión se realiza a nivel de gramática incluyendo un nuevo espacio de nombres para el elemento *LinkProperty,* el cual establece una relación con una ontología foránea. En esta propuesta también se extiende SWOOP, un editor diseñado con base en el lenguaje estándar OWL en el cual se habilita la funcionalidad de guardar y cargar ontologías *ε-connected.* Cada ontología se desarrolla en archivos separados, en los cuales se debe especificar, si se quiere, un *LinkProperty*, por esta razón no existe una visión de la ontología modular completa. Finalmente, el razonamiento sobre este nuevo tipo de ontologías exige también extender la herramienta de inferencias, en este caso el motor de inferencia Pellet. Como se puede ver, en dicha propuesta la extensión de un elemento en OWL implica la extensión de al menos dos herramientas más: el editor y el motor

[2] Protégé - http://protege.stanford.edu/
[3] SWOOP - https://code.google.com/p/swoop/

de inferencias. Por esta razón, se puede decir que la extensibilidad a nivel de gramática es compleja. Por otro lado, en el momento de procesar y validar un modelo ontológico modular el trabajo se hace de forma descentralizada, dado que cada ontología se describe en un archivo independiente.

Parece entonces más prometedor buscar una solución en la que los modelos sean directamente manipulables, tal como lo propone la ingeniería dirigida por modelos.

C. *Ingeniería Dirigida por Modelos*

La ingeniería dirigida por modelos, MDE (Model Driven Engineering), propone usar modelos como ciudadanos de primera clase. La idea principal de este enfoque es considerar que "todo es un modelo" lo cual permite separar las preocupaciones en el desarrollo de software por medio de varios niveles de abstracción[12], [13], entendiendo los modelos como un conjunto formal de elementos que representan una abstracción de la realidad y que se construyen con un fin[14].

OMG (Object Management Group) propone MDA (Model Driven Architecture) como un conjunto de estándares que crea el marco de trabajo para implementar los principios de MDE. OMG propone una arquitectura de 4 niveles, llamada arquitectura 3 + 1. La capa M0 corresponde al sistema real. Luego, un modelo representa dicho sistema en la capa M1. Este modelo es conforme a un metamodelo definido en M2; finalmente este metamodelo es conforme a un metametamodelo definido en la capa M3, el cual es conforme a sí mismo, en este proyecto en este nivel se utiliza el metamodelo definido por EMF[4]. Cuando se dice que un sistema es representado por un modelo, se entiende que el sistema puede ser descrito por el modelo y cuando se dice que un modelo es conforme a un metamodelo, se entiende que el modelo está escrito en el lenguaje del metamodelo[12].

A partir de la arquitectura MDA se desarrolla MDD (Model Driven Development) el cual se basa en la construcción y transformaciones de modelos[14]. Estas transformaciones pueden ser de modelo a modelo, de modelo a texto o de texto a modelo. Actualmente, existen diferentes tecnologías que permiten realizar dichas transformaciones. En el caso de las transformaciones de modelo a modelo se tiene, por ejemplo, ATL[5], ETL[6] y QVTO[7] y en el caso de las transformaciones de modelo a texto se encuentran ACCELEO[8], EGL[9], XPand[10] entre otros. La mayoría de ellas se agrupan en la herramienta MTC Flow[11] , desarrollada en la Universidad de los Andes, la cual permite diseñar, desarrollar, probar y ejecutar cadenas de transformación de modelos[15].

III. VISIÓN GENERAL DE LA SOLUCIÓN

De acuerdo a lo anterior, se plantea la creación de un lenguaje de dominio específico para la manipulación de ontologías modulares, basado en MDE y la teoría de *ε-connections*. Este lenguaje, independiente de la plataforma, habilita un ambiente centralizado donde la manipulación de ontologías modulares es posible en términos de creación, edición y razonamiento, gracias a la tecnología SIRIUS[12].

La figura 1, presenta la visión general de la solución propuesta. Este proceso tiene 2 niveles: Arquitectura MDA y Cadena de Transformación. En el primer nivel se empieza por definir el metamodelo MOWL y desarrollar el lenguaje de dominio específico. Luego, se define el metamodelo de Jess a partir del cual, en la segunda fase, se diseña e implementa la cadena de transformación que permite obtener una base de conocimiento, compuesta de hechos y reglas que representan el modelo de conocimiento inicial en sintaxis de Jess, sobre el cual se pueden realizar inferencias.

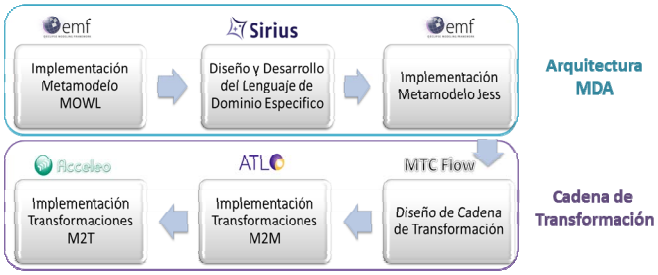

Fig. 1 Visión General de la Solución

IV. ARQUITECTURA MDA

Teniendo en cuenta la propuesta de arquitectura 3+1 formulada por el OMG[12], los niveles de abstracción en el diseño, desarrollo e implementación del lenguaje de dominio específico para la manipulación de ontologías modulares, se definen como se muestra en la figura 2.

[4] EMF - http://www.eclipse.org/modeling/emf/
[5] ATL - http://www.eclipse.org/atl/
[6] ETL - http://www.eclipse.org/epsilon/doc/etl/
[7] QVTO - http://projects.eclipse.org/projects/modeling.mmt.qvt-oml
[8] ACCELEO - http://www.eclipse.org/acceleo/
[9] EGL - http://www.eclipse.org/epsilon/doc/egl/
[10] XPAND - http://www.eclipse.org/modeling/m2t/?project=xpand
[11] MTCFlow - http://www.eclipse.org/modeling/m2t/?project=xpand
[12] SIRIUS - http://www.eclipse.org/sirius/

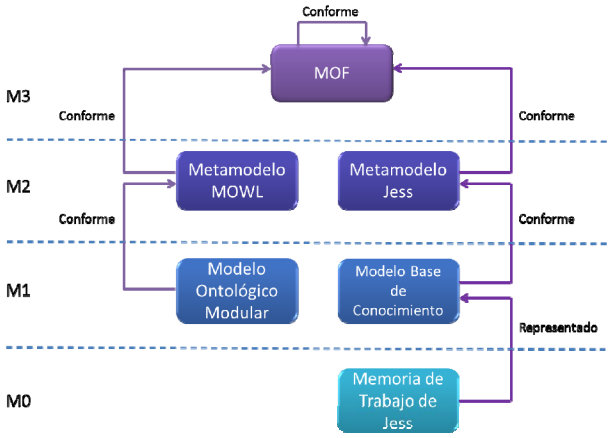

Fig. 2 Arquitectura MDA 3+1

En la capa M3 de la arquitectura, se encuentra el metamodelo MOF, el cual es conforme a sí mismo. En la capa M2 se encuentra el metamodelo MOWL y el metamodelo de Jess los cuales son conformes al metamodelo MOF. En M1, se encuentra el modelo ontológico modular, conforme al metamodelo MOWL y el modelo de la base de conocimiento, conforme al metamodelo de Jess. Finalmente, en M0 se encuentra la memoria de trabajo de Jess que es el lugar donde, en tiempo de ejecución, se almacenan los hechos que pueden ser manipulados por las aplicaciones basadas en conocimiento, esta memoria de trabajo es representada por el modelo de la base de conocimiento en M1.

A continuación se explican los metamodelos desarrollados a nivel de M2. En la sección V se describe la cadena de transformación que permite traducir un modelo conforme al metamodelo multiontológico MOWL en un conjunto de reglas y hechos manipulable por Jess. La sección VI describe la herramienta de edición desarrollada para crear modelos multiontológicos conformes a MOWL.

A. Metamodelo MOWL

El metamodelo MOWL es una extensión del metamodelo de OWL. En esta etapa, se tuvieron en cuenta los trabajos del ODM (Ontology Definition Metamodel)[16] y la sintaxis OWL definida por la W3C[17] quienes proponen un metamodelo conforme a MOF para este lenguaje[18]. Este metamodelo fue extendido, por una parte, con elementos que permiten expresar conectores de ontologías de tipo *ε-Connections* y por otra parte, con axiomas descritos en un lenguaje para restricciones para garantizar la consistencia de la multiontología. La primera extensión que se propone, como se muestra en la figura 3, es la creación del elemento *Multiontology* el cual agrupa ontologías. Así mismo este elemento puede tener conectores de ontología y axiomas de multiontología que describirán las propiedades de estos conectores.

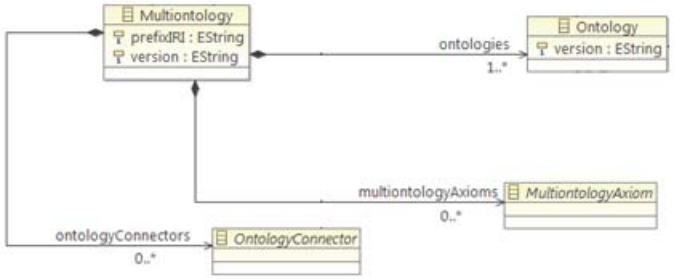

Fig. 3 Multiontology

Los elementos que permiten conectar ontologías se estructuran a partir de la clase *OntologyConnector,* superclase de la clase *Econnector*, creada para permitir la extensión del metamodelo, más allá de los conectores propuestos por la teoría de *ε-Connections*. Como la teoría de *ε-Connections* utiliza el elemento *link property* para establecer los enlaces entre ontologías, se define la clase *LinkProperty,* la cual se relaciona con una ontología origen y una ontología destino. Además del dominio y el rango de la propiedad que son clases de las respectivas ontologías, una *LinkProperty* puede tener asociados axiomas que permitan describir la semántica de la conexión en relación con otras conexiones, tales como función equivalencia, disyunción, *sublink*, etc. La Figura 4 muestra la estructura de los conectores.

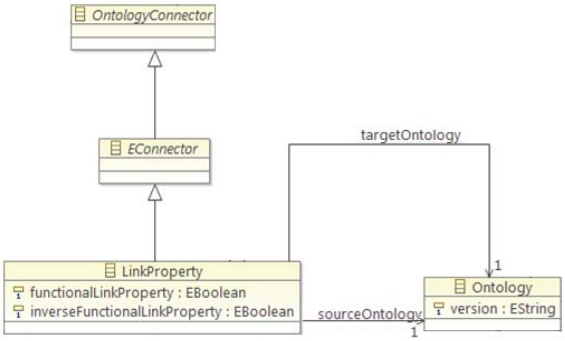

Fig. 4 Ontology Connector

B. Metamodelo Jess

Uno de los objetivos principales de crear conexiones entre ontologías es poder realizar inferencias con base en la información de las diferentes ontologías y la semántica de estas conexiones. Se busca, entonces, poder traducir modelos conformes al metamodelo de MOWL a modelos de hechos y reglas sobre las que se puedan realizar razonamientos. Como motor de inferencia se propone el uso del sistema Jess. Dentro del enfoque MDA propuesto, se utilizó el metamodelo de Jess, desarrollado por el grupo AtlanMode en [19], teniendo en cuenta el tutorial oficial del lenguaje[20]. El metamodelo original de Jess, cuenta con las estructuras que describen todos los elementos que pueden componer un archivo en sintaxis de Jess. Las expresiones utilizadas son las que permiten modelar hechos

y reglas. La única modificación que se realizó sobre este metamodelo, fue la creación de un elemento llamado *JessFile* que contiene grupos de expresiones, con el fin de obtener un modelo donde los hechos y las reglas aparezcan ordenados por ontología.

V. CADENA DE TRANSFORMACIÓN

La cadena de transformación definida en la figura 5, permite obtener modelos de base de conocimiento conformes al metamodelo de Jess. Para su diseño, se utilizó la herramienta desarrollada en la Universidad de los Andes MTCFlow[15].

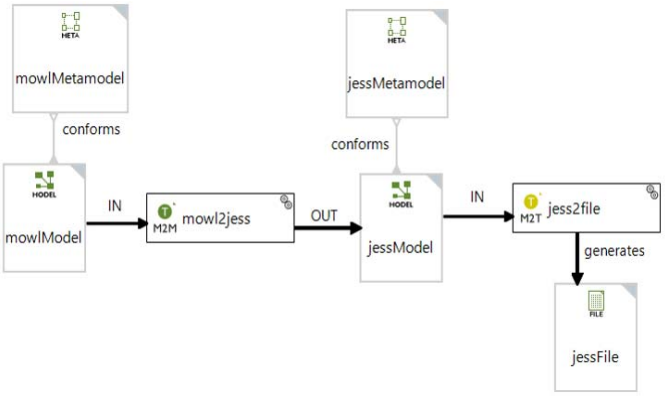

Fig. 5 Cadena de Transformación

Como se puede ver en la figura 5, esta cadena de transformación toma un modelo conforme al metamodelo MOWL y, a través de una transformación de modelo a modelo, lo traduce a un modelo conforme al metamodelo de Jess, el cual es transformado a un archivo en sintaxis de Jess, a través de una transformación de modelo a texto.

A. Transformaciones de Modelo a Modelo

En la primera fase, un modelo multiontológico es traducido a un modelo de hechos y reglas, utilizando la tecnología ATL. Esta transformación es hecha con reglas que toman un elemento conforme al metamodelo MOWL y crea uno o más elementos conformes al metamodelo de Jess. Al final de esta transformación se obtiene una estructura jerárquica como entrada para la siguiente fase.

B. Transformaciones de Modelo a Texto

Utilizando la tecnología Acceleo, se define un template para escribir un archivo de texto en la sintaxis de Jess. Este template toma como entrada la estructura jerárquica generada en la transformación anterior y de acuerdo a los elementos que en ésta se encuentren, construye el archivo que puede ser ejecutado por el motor de inferencia para construir la memoria de trabajo sobre la cual se pueden desarrollar aplicaciones basadas en conocimiento.

VI. LENGUAJE GRÁFICO DE DOMINIO ESPECÍFICO MOWL

En esta sección, se presenta el editor gráfico desarrollado para crear modelos ontológicos modulares conformes a MOWL. Este editor fue creado con la herramienta SIRIUS, la cual permite desarrollar lenguajes gráficos de dominio específico definiendo los diferentes elementos y relaciones del lenguaje a partir de un metamodelo conforme a MOF.

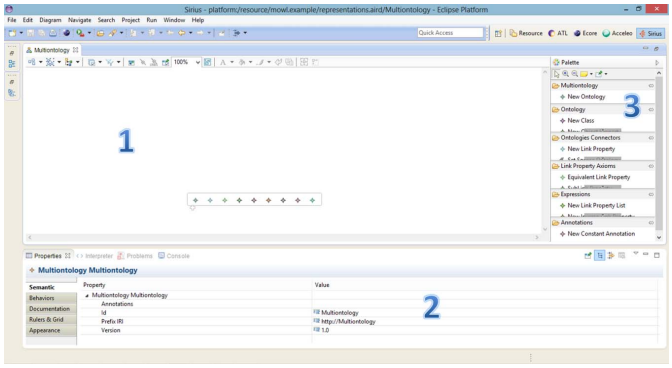

Fig. 6 Herramienta MOWL

Como se mira en la figura 6, el entorno gráfico tiene 3 secciones. La primera, es el espacio principal donde se visualizan los elementos que conforman el modelo y sus relaciones. La sección 2 es donde se presentan las propiedades de los elementos que conforman el modelo. Finalmente, la sección 3 tiene la paleta de herramientas que permite crear los elementos del modelo, para ello dichos elementos se deben arrastrar al espacio 1.

A. Edición gráfica de una multiontología

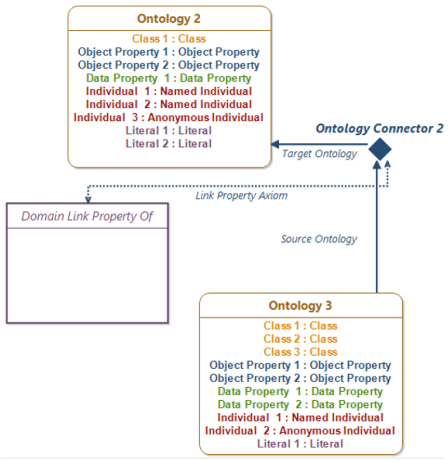

Fig. 7 Fragmento de Multiontología

La pantalla principal de la herramienta, permite modelar la multiontología, para ello se deben adicionar ontologías y elementos desde la paleta de herramientas. De igual forma se deben definir los conectores de ontologías, señalando la ontología fuente y destino. Para cada uno de ellos se pueden definir los axiomas respectivos, tal como se muestra en la figura 7, en el caso del elemento *Domain Link Property Of*.

En cuanto a cada ontología, la herramienta permite modelarlas individualmente en un espacio similar al anterior. Permitiendo toda la expresividad de OWL.

VII. Conclusiones y Trabajo Futuro

Este artículo presentó el diseño e implementación de MOWL (Modular Ontology Web Language), un lenguaje de dominio específico para la manipulación de ontologías modulares. La definición del lenguaje se realizó a nivel de metamodelo, utilizando MDE, lo que lo hace independiente de las plataformas y le brinda características de adaptabilidad, portabilidad y reusabilidad. La técnica de modularización implementada se base en la teoría de *ε-connections,* la cual garantiza la consistencia del modelo. MOWL se diseñó para permitir su extensión añadiendo elementos que las habiliten. Para una manipulación completa de los modelos de conocimiento, se desarrollaron transformaciones adicionales modelo a modelo y modelo a texto de la multiontología a un conjunto de hechos y reglas manipulable por el motor de inferencia de Jess. El trabajo realizado en este proyecto ofrece un ambiente completo para el modelaje, implementación y explotación de ontologías modulares, en particular ontologías que representen puntos de vista diferentes sobre una misma realidad, y esto en cualquier área del conocimiento. Dicho ambiente, así como ejemplos de aplicación se pueden encontrar en el sitio del proyecto[13].

Este trabajo fue evaluado en el campo de la informática educativa para modelar los objetos de aprendizaje que se están desarrollando en el centro ConectaTE de Innovación en tecnología y educación de la Universidad de los Andes[14]. Se definieron cuatro ontologías con varios conectores y se probó la generación automática de inferencias relacionadas con la semántica de los conectores. La evaluación permitió probar la efectividad del proceso. Actualmente se está trabajando en la integración de la herramienta con el repositorio de objetos de aprendizaje del centro. Como producto de esta evaluación se identificaron posibles mejoras sobre el editor con el fin de permitir una mejor expresión de los modelos gráficos. También se plantea la creación de módulos de exportación e importación de ontologías en formato OWL. Finalmente, resulta interesante evaluar la herramienta en diferentes dominios.

[13] MOWL - https://code.google.com/p/mowl-multiontology-web-language/
[14] Conecta-TE - http://conectate.uniandes.edu.co/

Referencias

[1] O. Mariño, F. Rechenmann, and P. Uvietta, "Multiple Perspectives and Classification mechanism in object- oriented representation," in *9th European Conference on Artificial Intelligence*, 1990.

[2] M. Ribieri and R. Dieng-Kuntz, "A Viewpoint Model for Cooperative Building of an Ontology," in *10th International Conference on Conceptual Structures*, 2002, pp. 220 – 234.

[3] H. Stuckenschmidt, C. Parent, and S. Spaccapietra, Eds., *Modular Ontologies*, vol. 5445. Berlin, Heidelberg: Springer Berlin Heidelberg, 2009.

[4] W3C OWL Working Group, "OWL 2 Web Ontology Language Document Overview," *Second Edition*, 2012. [Online]. Available: http://www.w3.org/TR/owl2-overview/. [Accessed: 08-Jun-2014].

[5] S. Bechhofer, F. Van Harmelen, J. Hendler, I. Horrocks, D. L. McGuinness, P. F. Patel-Schneider, and L. A. Stein, "OWL Web Ontology Language Reference," 2004. [Online]. Available: http://www.w3.org/TR/owl-ref/. [Accessed: 15-May-2014].

[6] O. Kutz, F. Wolter, and M. Zakharyaschev, "E -connections of Description Logics," in *International Workshop on Description Logics*, 2004.

[7] T. Berners-Lee, J. Hendler, and O. Lassila, "The Semantic Web," *Sci. Am.*, vol. 284, no. 5, pp. 34–43, May 2001.

[8] T. R. Gruber, "A Translation Approach to Portable Ontology Specifications," *Knowl. Acquis.*, vol. 5, no. April, pp. 199–220, 1993.

[9] W3C OWL Working Group, "OWL Web Ontology Language Guide," 2004. [Online]. Available: http://www.w3.org/TR/owl-guide/. [Accessed: 15-May-2014].

[10] Y. Wang, P. Haase, and J. Bao, "A Survey of Formalisms for Modular Ontologies," in *Proceedings of the International Joint Conference on Artificial Intelligence 2007 (IJCAI'07) Workshop SWeCKa*, 2007.

[11] B. Cuenca, B. Parsia, and E. Sirin, "Combining OWL Ontologies Using E -Connections," *Web Semant. Sci. Serv. Agents World Wide Web*, vol. 4, no. 1, pp. 40 – 59, 2006.

[12] J. Bézivin, "In Search of a Basic Principle for Model Driven Engineering," *Novatica J.*, vol. 5, no. Special Issue, pp. 21–24, 2004.

[13] J. Bézivin, "Model Driven Engineering□: An Emerging Technical Space," in *Generative and Transformational Techniques in Software Engineering*, 2006, pp. 36–64.

[14] S. J. Mellor, A. N. Clark, and T. Futagami, "Model-Driven Development," *IEEE Softw.*, vol. 20, no. 5, pp. 14 – 18, 2003.

[15] C. Alvarez and R. Casallas, "MTC Flow□: A tool to design , develop and deploy model transformation chains," in *Proceedings of the workshop on ACadeMics Tooling with Eclipse*, 2013.

[16] C. Document, "Ontology Definition Metamodel," vol. 1, no. February, 2014.

[17] W3C, "Syntax - OWL." [Online]. Available: http://www.w3.org/2007/OWL/wiki/Syntax. [Accessed: 15-May-2014].

[18] S. Brockmans, P. Haase, and B. Motik, "OWL 2 Web Ontology Language MOF-Based Metamodel (Second Edition)," 2008. [Online]. Available: http://www.w3.org/2007/OWL/wiki/MOF-Based_Metamodel. [Accessed: 25-Apr-2014].

[19] G. Doux, "A metamodel for the Jess Rule Language.," *AtlanMode Group*, 2008. [Online]. Available: http://www.emn.fr/z-info/atlanmod/index.php/Ecore#Jess_1.0. [Accessed: 25-Apr-2014].

[20] S. N. Laboratories, "The Rule Engine for the Java ™ Platform Version 7.1p2 November 5, 2008 ©," 2008.