Revista Española de Innovación, Calidad e Ingeniería del Software

REICIS

Volumen 6, Número 3 (especial XI JICS), noviembre, 2010

Web de la editorial: www.ati.es
Web de la revista: www.ati.es/reicis
E-mail: calidadsoft@ati.es
ISSN: 1885-4486

Publicado por la Asociación de Técnicos de Informática (ATI), Via Laietana, 46, 08003 Barcelona.

Secretaría de dirección: ATI Madrid, C/Padilla 66, 3º dcha., 28006 Madrid

Contenidos

REICIS

Comparando UML y OWL en la representación del conocimiento: correspondencia sintáctica

Susana M. Ramírez, Yisel Alonso, Violena Hernández, Arturo Cesar Arias y Dayana La Rosa
Universidad de las Ciencias Informáticas
{smramirez, yisel, violena, arturo, dlarosa}@uci.cu

Resumen

UML y OWL son lenguajes insignias de dos de los paradigmas más importantes que han emergido en los últimos tiempos para dar soporte al desarrollo de software. En la revisión de la literatura afín no se encuentra ampliamente documentada la relación entre ambos, a pesar del creciente interés en la utilización conjunta de UML y OWL. El propósito de este trabajo es proporcionar una comparación objetiva, con ejemplos concretos de la sintaxis de UML y OWL, que permita crear una base sólida para aprovechar las ventajas de cada uno y combinarlos en el proceso de desarrollo de software. Además se realiza una introducción al Ontology Definition Model para la utilización de la metodología, las herramientas y la tecnología UML como soporte para el desarrollo y el mantenimiento de ontologías.

Palabras clave: MOF, ODM, ontología, OWL, UML.

Comparing UML and OWL in knowledge representation: syntactic correspondence

Abstract

UML and OWL are insignia languages of two of the most important paradigms that have emerged in recent times to support software development. In the literature revision it is not widely studied the relation between UML and OWL, in spite of the growing interest on their combined use. The purpose of this paper is to provide an objective comparison with concrete examples of UML and OWL syntax, which would allow to create a solid base for making good use of their advantages, and to combine both languages in nowadays software development processes. Moreover, it makes an introduction to Ontology Definition Model for the use of UML's methodology, tools and technology, as a support for ontologies development and maintenance.

Key words: MOF, ODM, ontology, OWL, UML.

Ramírez,S.M,. Alonso,Y., Hernández,V., Arias, A.C.y La Rosa, D., "Comparando UML y OWL en la representación del conocimiento: correspondencia sintáctica", REICIS, vol. 6, no.3, 2010, pp.84-94. Recibido: 8-11-2010; revisado: 14-11-2010; aceptado: 19-11-2010

1. Introducción

En los últimos años dos importantes paradigmas han emergido para dar soporte al proceso de desarrollo de software. Por un lado se encuentra el paradigma que sitúa a los "modelos" *Model Driven Architecture* (MDA) en el centro del proceso de desarrollo con el *Unified Modeling Language* (UML) [1] del *Object Management Group* (OMG) como lenguaje insignia; y por el otro el paradigma de la ingeniería ontológica que ubica a las "ontologías" como la base del proceso. Esta ciencia, que evolucionó de la inteligencia artificial, cuenta con el *Web Ontology Language* (OWL) [2] de la *World Wide Web Consortium* (W3C) como su lenguaje insignia.

Dado que ambos paradigmas fueron desarrollados desde visiones diferentes, hasta hace muy poco se mantuvieron relativamente separados, cada comunidad evolucionando el lenguaje de acuerdo a sus propias necesidades. Sin embargo, desde que la representación del conocimiento en el desarrollo de software ha cobrado importancia y ambos lenguajes ofrecen la posibilidad de hacerlo en alguna medida, ha crecido el interés tanto de la industria como de la academia. Ambos intentan comprender cómo estos paradigmas se relacionan y cuál ofrece mejores capacidades para expresar el conocimiento bajo ciertas circunstancias, y cómo pueden ser usados de conjunto.

Para poder llevar esto a cabo, es importante establecer una comparación en cuanto a lo que ambas tecnologías ofrecen. No es posible tomar una buena decisión sobre cuál tecnología emplear, cuán efectivo sería usar los espacios de ambas tecnologías de conjunto y cómo integrarlas, sin conocer en qué se asemejan y difieren.

2. Lenguajes UML y OWL. Comparación.

UML se ha convertido en el estándar de facto de la industria. Por su parte OWL es un lenguaje concebido específicamente para la Web Semántica, que sobresale por encima de sus similares. [3]. Sin embargo, hasta el momento no se han explotado las potencialidades de su uso combinado. En esta sección se presentan las principales características de cada uno, y una comparación técnica, fundamentalmente de la sintaxis de ambos lenguajes.

2.1. Unified Modeling Language

UML es definido por sus creadores en [4] como un lenguaje de modelado visual que se usa para especificar, visualizar, construir y documentar artefactos de un sistema de software.

Las versiones iniciales de UML (UML 1) se originaron con los tres principales métodos orientado a objetos (Booch, OMT[3] , y OOSE[4]) del momento. La última revisión de UML (UML 2.0) [5] [6], ha sido realzada con definiciones significativamente más precisas, una estructura del lenguaje más modular, y una altamente mejorada capacidad para el modelado de sistemas de gran escala.

UML capta la información sobre la estructura estática y el comportamiento dinámico de un sistema, no obstante carece de una semántica rigurosa, que permita utilizar razonadores automáticos sobre los modelos. Para realizar la comparación, este trabajo se enfocará en la representación de la información de estructura estática a través de los diagramas de clases.

UML se encuentra conceptualmente embebido dentro de la arquitectura de cuatro niveles de OMG, y es considerado como una instancia del *Meta Object Facility* (MOF) [7]. El estándar MOF define diversos metamodelos, abstrayendo la forma y la estructura que describen. Define los elementos esenciales: sintaxis y estructuras de metamodelos que se utilizan para construir modelos de sistemas. Varias son las tecnologías estandarizadas por OMG que usan MOF y sus tecnologías derivadas para el intercambio y manipulación de metadatos. Dentro de las más recientes iniciativas se encuentra el estándar *Ontology Definition Metamodel* (ODM) para la transformación entre metamodelos MOF. ODM será abordado con posterioridad en este trabajo.

2.2. Web Ontology Language

OWL es un lenguaje de ontologías para la Web Semántica con significado formalmente definido, desarrollado por el *W3C Web Ontology Working Group*. Es desarrollado como un lenguaje extensión del *Resource Description Framework* (RDF) [8] y es derivado del lenguaje *DAML+OIL* [9]. Está diseñado para facilitar el desarrollo de ontologías y el uso compartido por la Web, con el objetivo de hacer más accesibles los contenidos de la Web para las máquinas.

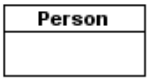
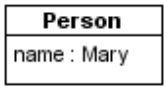
Las ontologías en OWL proveen clases, propiedades, individuos y valores de datos, y son almacenados como documentos Semánticos de la Web. La versión actual de OWL define tres sublenguajes que son presentados según su capacidad expresiva. [10]: OWL –

[3] Object Modeling Technique
[4] Object Oriented Software Engineering

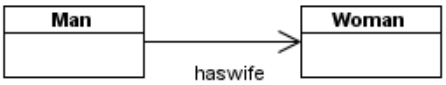
Lite, OWL – DL y OWL-FULL. Para la comparación se utilizará OWL – FULL por ser la versión más general del lenguaje con la que se logra mayor expresividad. Será representado usando *Functional Syntax*, debido a que es la sintaxis más simple para presentar la estructura formal de una ontología.

2.3. Comparación entre UML y OWL

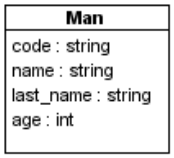
Es necesario destacar que ambos lenguajes fueron construidos bajo propósitos diferentes: OWL para la representación del conocimiento y UML para soportar el desarrollo de software. Sin embargo, ambos permiten la representación del conocimiento y emplean para esto objetos y relaciones entre ellos. A pesar de la similitud en cuanto a su fin, estos difieren en cuanto a cómo el conocimiento es entendido y expresado. Con UML se modelan las restricciones que satisfacen al conjunto de estados permitidos de un sistema. Con OWL se representa el conocimiento y este se emplea para inferir nuevo conocimiento. [11]. A continuación se muestra una comparación (no completa pero si representativa) de las características comunes de ambos lenguajes.

<table>
<tr><th>Elementos UML</th><th>Elementos OWL</th></tr>
<tr><td>class</td><td>class</td></tr>
<tr><td colspan="2">Ambos lenguajes se basan en clases. Una clase en OWL es un set de cero o más instancias. Una clase en UML es una construcción más general, uno de sus usos es como un set de instancias. En OWL hay una clase universal Thing cuya extensión es todos los individuos en un modelo dado, y todas las clases son subclases de Thing.</td></tr>
<tr><td> 
</td><td>Class(: Person)</td></tr>
<tr><td>instance</td><td>individual</td></tr>
<tr><td colspan="2">En OWL la construcción individual es similar a la construcción instance en UML. Sin embargo OWL FULL permite diferentes formas de definición de una clase (como clase y como instancia). En el primer ejemplo se muestra como la clase Person es usado como clase mientras que en el segundo se utiliza como individuo actuando SocialRole como metaclase de Person.</td></tr>
<tr><td> 
</td><td>Individual(:Person :Mary)
Individual(:SocialRole :Person)</td></tr>
<tr><td>binary association, ownedAttribute</td><td>property</td></tr>
<tr><td colspan="2">Las relaciones entre las clases en OWL son llamadas property y representadas por ObjectProperty y DataProperty.
En el ejemplo la clase Man tiene una relación con la clase Woman llamada haswife, que es representada en el modelo UML como la asociación haswife y en OWL como una ObjectProperty del mismo nombre.</td></tr>
</table>

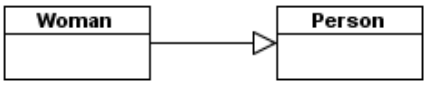
UML tiene la opción para las asociaciones de distinguir la navegabilidad (el final de relación), esta puede ser *navigable* o *non-navigable*. En contraparte a esto las propiedades OWL han distinguido finales designados como: dominio (*domain*) y rango (*range*). En el ejemplo se muestran el dominio y rango de la *ObjectProperty haswife*

	*ObjectProperty(: haswife* domain *(: Man)* *range (: Woman))*

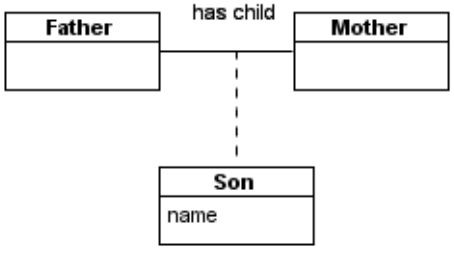
Los *DataProperty* surgen de la asociación llamada *ownedAttribute* entre *Class* y *Property* y son traducidos como *properties* cuyo dominio es una Clase y el rango es el tipo de la *Property*. En el ejemplo se presenta la definición en OWL para el *ownedAttribute* de la clase *Man* : *name*

	*DatatypeProperty (: name* *domain (:name :Person)* *range (:name xsd:string))*
generalization	**subclass subproperty**

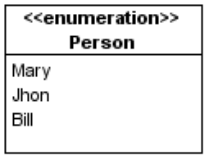
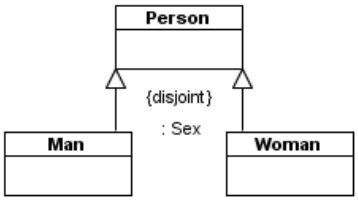
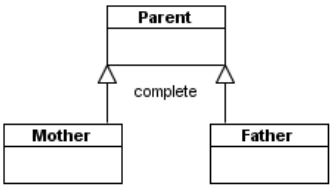
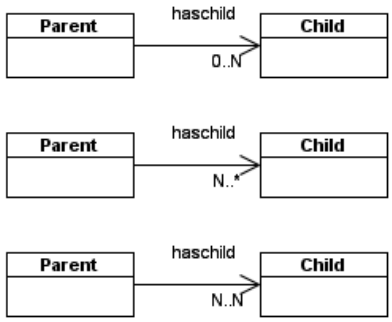
Ambos lenguajes soportan la relación *subclass*. En OWL con *subClassOf* y en UML se presenta como la relación de *generalization* para expresar la inclusión de una clase.

	*subClassOf(:Woman :Person)*
N-ary association, association class	**class, property**

Una asociación en UML puede ser *N-ary*. Una asociación también puede ser una clase (*association class*) que puede participar en más relaciones. En el ejemplo la traducción descrita a OWL no es normativa debido a que no existe una construcción específica, por definición una relación se define entre pares de conceptos. La solución que se presenta es la utilización del patrón de diseño llamado *reification* que consiste en la creación de una nueva clase que conceptualiza la relación e involucra los conceptos que esta envuelve. Se crean n nuevas relaciones funcionales, una por cada participante en la asociación.

	*Class(: Son)* *ObjectProperty(: Father_Son* domain *(: Son)* r*ange (: Father))* *ObjectProperty(: Mother_Son* domain *(: Son)* r*ange (: Mother))*
enumeration	**oneOf**

Ambos soportan una extensión fija definida para una clase, aunque en UML un enumerador es considerado un tipo de datos antes que una clase.

<table>
<tr><td> 
</td><td>Class(: Person
OneOf(:Mary :Jhon :Bill))</td></tr>
<tr><td>disjoint, cover</td><td>disjointWith, unionOf</td></tr>
<tr><td colspan="2">Ambos lenguajes permiten subclases de clases ser declaradas disjuntas. En OWL se utiliza la construcción disjointWith. Además es posible que para una colección de subclases sea declarado que el cubrimiento es completo, es decir cada instancia de la superclase es una instancia de al menos una de las subclases. La construcción correspondiente en OWL para declarar que una superclase es la unión de sus subclases es unionOf. En UML se representa a partir de la definición de las restricciones correspondientes.</td></tr>
<tr><td> 
</td><td>disjointWith (:Woman :Man)</td></tr>
<tr><td> 
</td><td>Class(:Parent
unionOf(:Mother :Father))</td></tr>
<tr><td>multiplicity</td><td>minCardinality, maxCardinality, cardinality</td></tr>
<tr><td colspan="2">En OWL, una restricción de una propiedad aplicada a una clase puede imponer una restricción de cardinalidad dando el número mínimo (minCardinality), máximo (maxCardinality), o exacto (cardinality) de instancias que pueden participar de una relación. En UML una asociación puede tener cardinalidades mínimas y máximas (multiplicity). Se muestran tres ejemplos de las tres restricciones de cardinalidad.</td></tr>
<tr><td> 
</td><td>maxCardinality(N :hasChild :Parent)

minCardinality(N :hasChild :Parent)

cardinality(N :hasChild :Parent)</td></tr>
<tr><td colspan="2">Además, una propiedad OWL puede ser globalmente declarada como funcional (functionalProperty) o inversa funcional (inverseFunctional). Una propiedad funcional tiene una cardinalidad máxima de 1 sobre su rango, mientras una propiedad funcional inversa tiene una cardinalidad máxima de 1 sobre su dominio. En el ejemplo la ObjectProperty haswife es de los tipos.</td></tr>
</table>

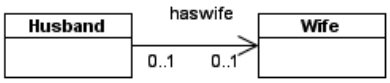
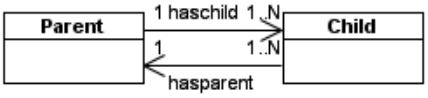
	*FunctionalProperty(:haswife)* *InverseFunctionalProperty(:haswife)*
Si una asociación binaria UML tiene una multiplicidad sobre ambos extremos, entonces la propiedad correspondiente OWL será un par inverso.	
	*inverseOf (:hasparent :haschild)* *maxCardinality(N :haschild :Parent)* *maxCardinality(1 :hasparent :Child)*
package	**ontology**
Ambos soportan el concepto de *namespace* como una construcción empaquetadora, definida como *Model* que es una subclase de la primitiva *Package* en UML y *Ontology* en OWL.	

Tabla 1. Comparación. Características comunes OWL – UML.

Para la comparación se han tenido en cuenta todas las características de UML que de alguna manera tienen algún equivalente en OWL. Sin embargo, existen también otras características sin equivalencia. Estas son las siguientes: *derived, abstract classifier, operations, interface classes, active classes, ports, connectors y complex objects.* Por su parte también existen características de OWL con ningún equivalente en UML: *allValuesFrom, someValuesFrom, SymmetricProperty, TransitiveProperty y complementOf.*

A pesar de las diferencias mostradas, la metodología, las herramientas y la tecnología UML parece ser un enfoque factible para soportar el desarrollo y el mantenimiento de ontologías. Recientemente, se ha comenzado a perfilar la idea del uso directo de UML como un lenguaje de ontologías, considerando la definición de varios estereotipos adicionales, que permitan un mapeo más detallado de UML a las primitivas brindadas por la lógica descriptiva [3]. Consecuentemente, OMG realizó la petición de propuestas para una definición de Ontology Definition Metamodel [12].

3. Ontology Definition Metamodel

ODM ha sido concebido dentro del MOF como cualquier otro metamodelo de OMG. La especificación define una familia de metamodelos independientes, perfiles relacionados, y mapeos entre los metamodelos, correspondiendo a varios estándares internacionales para ontologías, así como la capacidad de soportar paradigmas convencionales de modelado para captar conocimiento conceptual, como UML y *Entity-Relationship Diagram*.

Los metamodelos en ODM son tratados de igual manera, aun cuando son independientes entre sí. No es necesario entender o estar al tanto de los demás para alcanzar el entendimiento total de uno específico. La única excepción es el metamodelo para OWL, que extiende del metamodelo para RDF, tanto como el lenguaje OWL extiende del RDF.

Sin embargo, en un proyecto de desarrollo de una ontología debería ser necesario el uso de varios metamodelos. Teniendo en cuenta el amplio uso de UML, comúnmente el desarrollador se encontrará ante la necesidad de reutilizar artefactos ya existentes, o de aprovechar la experiencia ya acumulada con herramientas asociadas a estas, para realizar al menos una primera aproximación de alineamiento con el modelo OWL. ODM, por tanto, necesita proveer facilidades para establecer relaciones entre instancias de sus metamodelos, incluyendo UML. Existen dos formas para lograr esto: perfiles UML y mapeos.

3.1 Perfiles UML y Mapeo

Un perfil UML de la perspectiva ODM, tiene como objetivo proveer un puente entre UML y los lenguajes de representación del conocimiento sobre una bien fundada, base semántica. Los perfiles facilitan la implementación usando la notación común en herramientas existentes UML. Igualmente proveen a los usuarios capacidades para utilizar UML como la base para el desarrollo de ontologías para un lenguaje especifico de representación del conocimiento como OWL. Por lo tanto se necesita especificar los mapeos de un metamodelo a otro. Para el trabajo con múltiples metamodelos se requiere un elemento de traducción del modelo por cada elemento de los modelos a mapear.

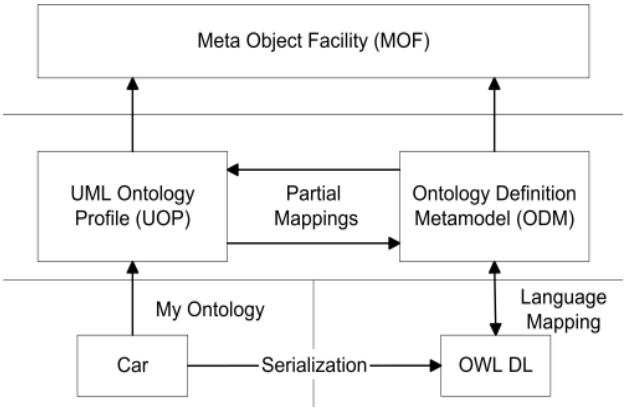


Figura 2. Mapeo a OWL DL [13]

ODM, además de definir una notación visual a través de los perfiles, debe proveer los mapeos parciales en ambas direcciones entre el metamodelo y el perfil definido. Del ODM

se debe poder generar una representación de una ontología en un lenguaje como OWL DL, una serialización usando el XML Metadata Interchange (XMI) y una sintaxis de cambio. Este formato XMI permite intercambiar un metamodelo ODM entre herramientas. La figura anterior muestra este proceso.

Tres perfiles UML han sido desarrollados para el uso con ODM: para RDF, OWL, y Topic Maps (TM). Estos permiten el uso de notación UML (y las herramientas) para el modelado de ontologías y facilitan la generación de las correspondientes descripciones de ontologías en RDF, OWL, y TM, respectivamente. Con estos perfiles se reutilizan construcciones UML que tienen la misma semántica que dichos lenguajes, y, cuando no es posible encontrar construcciones exactamente equivalentes desde el punto de vista semántico se utilizan entonces construcciones que son consistentes y cercanas. Una descripción detallada de cada perfil se encuentra en [12].

ODM encierra los fundamentos de un enfoque para el desarrollo y despliegue de sistemas que ha dado en llamarse *Ontology Driven Architecure* (ODA) [14].

4. Conclusiones

En este trabajo, se ha presentado una comparación detallada de la sintaxis de UML y OWL, que permitan valorar las capacidades de expresión del conocimiento en determinadas circunstancias de ambos lenguajes. Esta comparación puede ser interpretada desde dos perspectivas dependiendo de la intención del lector, resaltando las marcadas diferencias existentes o las innegables similitudes entre ellos.

Las similitudes se evidencian a partir de que ambos lenguajes permiten la representación del conocimiento y emplean para esto objetos y relaciones entre ellos. Adicionalmente, la sintaxis abstracta de ambos presenta un alto grado de similaridad, manifestada en la correspondencia existente en la mayor parte de sus constructores.

Por otra parte, se diferencian en que fueron concebidos bajo diferentes propósitos: OWL para la representación del conocimiento y UML para soportar el desarrollo de software, lo que se constata en diferentes formas. Con UML se pretende modelar las restricciones que satisfacen al conjunto de estados permitidos de un sistema, mientras que con OWL se representa el conocimiento y este se emplea para inferir nuevo conocimiento.

Como resultado, es imposible la traducción de ontologías OWL a modelos UML y viceversa, sin la pérdida o corrupción de la información.

Parte de la solución a dicha problemática se ofrece en la propuesta de integración que ha realizado el OMG con una de sus más recientes iniciativas el *Ontology Definition Metamodel*, dado que ambos lenguajes y las tecnologías relacionadas ocupan un puesto central en la especificación del mismo. Con este estándar se trata de utilizar la notación UML y las tecnologías asociadas a este para el modelado de ontologías, de manera que se aprovechen las ventajas de ambas tecnologías y combinen durante el proceso de desarrollo. Una introducción de las características y tecnologías envueltas en ODM es provista en este trabajo.

Referencias

[1] Object Management Group, *Unified Modeling Language (OMG UML) v1.3*, OMG, 2000.

[2] World Wide Web Consortium, *OWL Web Ontology Language Semantics and Abstract Syntax*, W3C, 2004.

[3] García Noguera, M., *Modelado y Análisis de Sistemas CSCW siguiendo un enfoque de Ingeniería dirigida por Ontologías.* Tesis Doctoral. Universidad de Granada, España, 2009.

[4] Booch, G., Jacobson, I. y Rumbaugh, J., *El Lenguaje Unificado de Modelado. Manual de Referencia,* Addison Wesley, 2000.

[5] Object Management Group, *Unified Modeling Language (OMG UML), Infrastructure Specification, v2.3*, OMG, 2010.

[6] Object Management Group, *Unified Modeling Language (OMG UML), Superstructure Specification v2.3*, OMG, 2010.

[7] Object Management Group, *Meta Object Facility Core Specification v 2.0*, OMG, 2006.

[8] World Wide Web Consortium, *Recommendation. Resource Description Framework (RDF): Concepts and Abstract Syntax*, W3C, 2004.

[9] World Wide Web Consortium, *Note. DAML+OIL Reference Description*, W3C, 2001.

[10] Horrocks, I., Patel-Schneider, P. y van Harmelen, F., "From SHIQ and RDF to OWL: The Making of a Web Ontology Language", *Journal of Web Semantics*, vol. 1, nº 1, pp. 7-26, 2003.

[11] Kiko, K. y Atkinson, C., "A Detailed Comparison of UML and OWL", *University of Mannheim*, Technical Report TR-2008-004, Department for Mathematics and Computer Science, University of Mannheim, 2008.2008.

[12] Object Management Group, *Ontology Definition Metamodel v1.0*, OMG, 2009.

[13] Brockmans, S, Volz, A., Eberhart, A. y Loeffler, P., "Visual Modeling of OWL DL Ontologies Using UML", En: van Harmelen, F., Mcllraith, S. A. y Plexousakis, D. (eds.) *The Semantic Web- ISWC. Hiroshima (Japón), Noviembre*, pp. 198-213, 2004.

[14] World Wide Web Consortium, *Working Draft. Ontology Driven Architectures and Potential Uses of the Semantic Web in Systems and Software Engineering. Editors' Draft,* W3C, 2006.