Agregando Semántica a MongoDB

Nicolas, Padula
Departamento Ingeniería en Sistemas de Información, Universidad Tecnológica Nacional – Facultad Regional Santa Fe (Lavaise 610, Santa Fe)
INGAR – Instituto de Desarrollo y Diseño (CONICET-UTN) (Avellaneda 3657, Santa Fe)
nicolasvpadula@gmail.com

Resumen - *Bajo la premisa de satisfacer las necesidades de flexibilidad y diversidad en el almacenamiento de datos, MongoDB ha ganado popularidad en los últimos años, permitiendo almacenar datos sin la necesidad de definir un esquema previo. Diversas metodologías han sido desarrolladas para la implementación de tecnologías semánticas a partir de bases de datos relacionales. Por el contrario, el uso de tecnologías semánticas partiendo de bases de datos no relacionales es aún un área novedosa. Este trabajo presenta una comparación cualitativa de 3 herramientas (Morph-xR2RML, Tripod y MongoGraph) para la extensión de la capacidad de semántica de MongoDB.*

Palabras Clave: MongoDB, SPARQL, OBDA, Web Semántica, Razonamiento

Adding Semantics to MongoDB

Abstract – *Under the premise of satisfying the needs for flexibility and diversity in the data storage field, MongoDB has achieved popularity over the last years, allowing the storage of data without the need to previously define a schema. Several methodologies have been developed for semantic technologies implementation based on relational databases. On the contrary, the use of semantic technologies based upon non-relational databases is still a novel area.*
This work presents a qualitative comparison of 3 tools (Morph-xR2RML, Tripod, and MongoGraph) for the extension of MongoDB's semantic capabilities.

Keywords: MongoDB, SPARQL, OBDA, Semantic Web, Reasoning

INTRODUCCIÓN

La ubicuidad de las tecnologías de la información ha dado lugar a una mayor complejidad y diversidad en los datos que se producen y se procesan. La Web Semántica (Berners-Lee et al., 2010) y sus tecnologías asociadas surgen como alternativa para atender el aumento de complejidad, en conjunto con la necesidad de operar sobre datos provenientes de fuentes heterogéneas. Lenguajes como Resource Description Framework (Manola et al., 2014) (RDF) y Ontology Web Language (Hitzler et al., 2009) (OWL) son los fundamentos básicos sobre los que se constituyen las tecnologías semánticas, las cuales permiten representar, integrar y razonar sobre datos de diversos tipos. Un documento RDF es un grafo, representado mediante estructuras llamadas tripletas, cuyo formato es: <sujeto> <predicado> <objeto>, siendo el sujeto y objeto nodos del grafo, mientras que el predicado juega el papel de arco, uniendo dichos nodos. Los grafos RDF pueden almacenarse ya sea en archivos de texto plano, o en sistemas denominados *triplestores*, o bases de datos de tripletas, las cuales proveen funcionalidades similares a los motores de bases de datos tradicionales. Tantos las *triplestores* como los documentos RDF pueden consultarse mediante el lenguaje SPARQL (W3C SPARQL Working Group, 2013), lenguaje de consultas nativo para este formato.

Sin embargo, en la práctica, la gran mayoría de los datos no se almacenan usando representaciones directamente accesibles por tecnologías semánticas. El proceso de hacer uso de datos que no poseen una estructura

directamente accesible por tecnologías semánticas se conoce como *Ontology Based Data Access* (Kontchakov et al., 2013) (OBDA), y se lleva a cabo mediante un sistema intermedio que traduce los datos a una estructura compatible con la Web Semántica, generalmente, RDF. Esta traducción puede llevarse a cabo bajo demanda, reescribiendo las consultas SPARQL a el lenguaje de consulta nativo de la fuente de datos original, esta metodología se denomina *query rewriting* o reescritura de consultas. Como alternativa, el proceso de OBDA puede llevarse a cabo mediante *materialización,* esta alternativa consiste en generar el grafo RDF completo correspondiente a todos los datos almacenados en la fuente original. Para ambos casos se utilizan reglas de mapeo. La Figura 1 ilustra los dos tipos de OBDA.

En la práctica, las bases de datos relacionales se constituyen como una de las alternativas más comunes usadas como medio de almacenamiento. La problemática de realizar OBDA con información almacenada en bases de datos relacionales ha sido abordado en numerosas ocasiones (Spanos et al., 2010). El World Wide Web Consortium (W3C) en la recomendación en (Das et al., 2012), especifica el lenguaje R2RML, el cual define reglas y métodos de mapeo entre tablas de bases de datos relacionales y grafos RDF. Estas reglas se definen a su vez en un documento RDF, que puede ser procesado por aquellas herramientas que implementen R2RML. El W3C también sugiere el enfoque de *Direct Mapping* (Arenas et al., 2011) como método por defecto de transformación.

Varias herramientas implementan y extienden las funcionalidades especificadas por el W3C para realizar el proceso de OBDA. Morph-RDB (Priyatna et al., 2014) implementa materialización y reescritura de consultas basándose en la especificación R2RML. D2RQ (Bizer y Seaborne, 2006) implementa su propio lenguaje de mapeo, y soporta el Direct Mapping especificado por el W3C, extendiéndolo con heurísticas por fuera del estándar. Ontop (Calvanese, et al., 2016) implementa OBDA sobre bases de datos relacionales mediante la reescritura de consultas permitiendo realizar inferencias RDFS y OWL-QL (Kontchakov et al., 2014).

En los últimos años, debido a la creciente complejidad de los datos, se han hecho esfuerzos por desarrollar nuevas formas de almacenamiento que permitan satisfacer las necesidades de flexibilidad y versatilidad en el almacenamiento de datos. Las bases de datos NoSQL, o no relacionales, se han afianzado como alternativa popular debido a su capacidad de acomodar datos de diversa naturaleza. En particular, MongoDB se ha posicionado como líder dentro del segmento de bases de datos NoSQL, siendo parte de la infraestructura de importantes empresas de base tecnológica. Los registros almacenados en MongoDB son documentos JSON (JavaScript Object Notation) los cuales consisten en una serie de pares (atributo, valor), los cuales a su vez pueden tratarse de documentos anidados. Esto, en conjunto con la flexibilidad de no tener que definir un esquema al que los datos deban adaptarse, y la capacidad de manejar grandes volúmenes

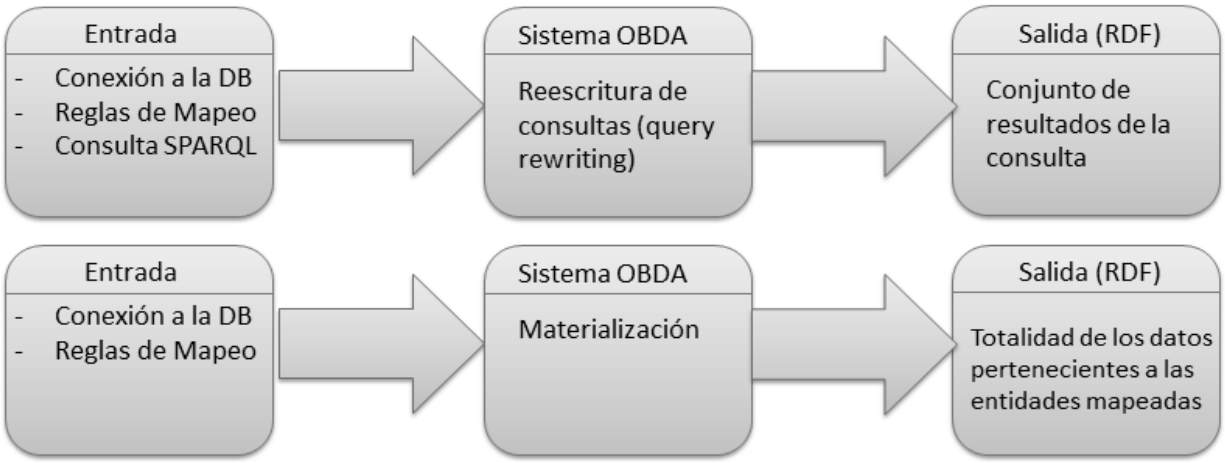

Figura 1. Distintas metodologías de OBDA

```
{
  "address": {"building": "1007", "coord": [-73.856077, 40.848447], "street": "Morris Park Ave", "zipcode": "10462"},
  "borough": "Bronx",
  "cuisine": "Bakery",
   "grades": [
      {"date": {"$date": 1393804800000}, "grade": "A", "score": 2},
      {"date": {"$date": 1378857600000}, "grade": "A", "score": 6},
      {"date": {"$date": 1358985600000}, "grade": "A", "score": 10},
      {"date": {"$date": 1322006400000}, "grade": "A", "score": 9},
      {"date": {"$date": 1299715200000}, "grade": "B", "score": 14}
      ],
  "name": "Morris Park Bake Shop",
  "_id": "30075445"
  }
```


Figura 2. Documento JSON describiendo un restaurant

de datos, son las características centrales de MongoDB, las cuales han dado lugar a su alto grado de adopción dentro de las bases de datos no relacionales. La Figura 2 muestra un ejemplo de un documento JSON en MongoDB.

Estas características están en concordancia con algunas de las ideas pertenecientes al contexto de la Web Semántica, dentro del cual se plantea que no se puede anticipar un esquema que pueda anticipar la diversidad de los datos que son producidos ya que los mismos cambian constantemente. Debido a estas similitudes y a su creciente adopción, resulta relevante analizar las posibilidades existentes de realizar OBDA sobre bases de datos MongoDB. Este trabajo tiene como objetivo presentar un análisis comparativo sobre las distintas opciones para extender las capacidades semánticas de MongoDB, disponibles al momento de la realización de este trabajo. En particular, las herramientas analizadas son: Morph-xR2RML, una extensión del ya mencionado Morph-RDB, MongoGraph, un componente dentro de la *triplestore* AllegroGraph, que permite la interoperabilidad con MongoDB y una mención de Tripod, un ORM que permite la utilización de MongoDB como capa de almacenamiento para grafos RDF.

METODOLOGÍA

Se analizaron las tres soluciones mencionadas anteriormente. Dentro de este análisis se consideró: forma de acceso a los datos originales, requerimientos estructurales en los datos originales, requisitos tecnológicos para el funcionamiento de la solución, mecanismos de OBDA (reescritura de consultas, materialización, otros), expresividad semántica (capacidad semántica de las consultas, existencia de reglas o razonamiento, etc), y funcionalidades adicionales no necesariamente relacionadas con MongoDB.

Debido a que tanto los objetivos como los mecanismos de ejecución de las distintas herramientas son diferentes, se decidió no hacer una evaluación de rendimiento. No obstante, se incluyen algunas consideraciones generales respecto de los distintos factores que pueden influir en el rendimiento de los sistemas analizados, y en carácter ilustrativo y no concluyente, mediciones de rendimiento obtenidas de una consulta sencilla. La consulta elegida fue *“seleccionar todos los restaurants cuya cocina es de hamburguesas”*.

En el caso de Morph-xR2RML esto se implementa mediante una consulta SPARQL pura, mientras que en MongoGraph se hace uso del predicado especial *find*. La Figura 3 muestra las consultas resultantes.

A

```
PREFIX ex:  <http://example.com/>
PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>
PREFIX rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
PREFIX xsd: <http://www.w3.org/2001/XMLSchema#>

SELECT ?r
WHERE {
  ?r ex:cuisine "Hamburgers".
  }
```

B

```
PREFIX mongo: <http://franz.com/ns/allegrograph/4.7/mongo/>
PREFIX f: <http://www.franz.com/>


SELECT ?r
WHERE {
  ?r mongo:find '{ cuisine: "Hamburgers" }' .
}
```

Figura 3. A) Consulta para Morph-xR2RML. B) Consulta para MongoGraph

El conjunto de datos de prueba fue una colección de más de 25.000 documentos JSON almacenados en una colección de MongoDB. Ambas pruebas fueron realizadas en el mismo equipo (Intel i5 4440, 8GB RAM). Los resultados obtenidos se presentan en la sección siguiente.

RESULTADOS

Morph-xR2RML

El proyecto Morph-xR2RML (Michel et al., 2016) es un *fork* del ya mencionado Morph-RDB. Morph-xR2RML implementa una extensión del lenguaje R2RML, denominado xR2RML (Michel et al., 2015).
xR2RML extiende el concepto de *logicalTable*, previamente especificado por R2RML y lo generaliza denominándolo *logicalSource*. El mismo explicita una fuente lógica de los datos que serán transformados en una regla de mapeo en particular.
Mientras que las fuentes lógicas de R2RML podían ser solo tablas de bases de datos relacionales, xR2RML generaliza este concepto a cualquier conjunto de resultados retornados de una consulta en particular, no solo de bases de datos relacionales, sino de diversos tipos de estructuras. Una fuente lógica de xR2RML puede ser una consulta en XPath/XQuery si los datos se encuentran en un archivo XML, una consulta de la consola de comandos de MongoDB, o JSONPath si se trata de un archivo JSON, así como también existe soporte para mapear resultados provenientes de servicios web implementados sobre HTTP o realizar mapeos RDF a RDF, mediante consultas SPARQL.
Es de destacar que, si bien el estándar de xR2RML permite el mapeo de diversas fuentes de datos de manera simultánea, esto no está soportado por la implementación actual de Morph-xR2RML.
La Figura 4 presenta ejemplos de reglas de mapeo sencillas en xR2RML para el ejemplo JSON mostrado en la Figura 2.

```
<#Restaurants>
a rr:TriplesMap;
xrr:logicalSource [
 xrr:query """db.restaurants.find( {'_id' : {$exists: 1} } )""";
];

rr:subjectMap [
 rr:template "http://example.org/restaurant/{$._id.*}";
     rr:class ex:Restaurant;
];

rr:predicateObjectMap [
 rr:predicate ex:cuisine;
 rr:objectMap [
   xrr:reference "$.cuisine";
 ];
];
```

Figura 4. Reglas xR2RML básicas

Morph-xR2RML soporta tanto reescritura de consultas como materialización para realizar el proceso de OBDA. Ambos mecanismos retornarán resultados acordes a las reglas de mapeo especificadas. En el caso de reescritura de consultas, Morph-xR2RML permite la reescritura y ejecución de consultas SPARQL individuales especificadas en archivos de texto, o la configuración de un *endpoint* SPARQL, es decir, un servicio web que permite la ejecución remota de consultas mediante protocolo HTTP.
Morph-xR2RML implementa la reescritura de consultas reescribiendo la consulta SPARQL primero en lo que sus autores denominan *abstract query*, que luego es traducida al lenguaje de consulta de la fuente de destino. Esto supone un alto grado de extensibilidad, con capacidad para añadir nuevas fuentes de mapeo. No obstante, este método de reescritura resulta en un aumento de la complejidad algorítmica del sistema, lo cual resulta en un rendimiento disminuido. Para el conjunto de datos de prueba definido, especificando solo cuatro reglas de mapeo, mapeando atributos simples (sin anidamiento), la reescritura de la consulta SPARQL mostrada en la Figura 3 (A) tomó 537ms.
Debe remarcarse que el rendimiento obtenido dependerá también en gran medida de los mapeos realizados. Las reglas de mapeo que involucren *joins* de distintas colecciones o entidades, o una mayor *reificación* (generar entidades en lugar de mapear elementos como literales) aumentarán la complejidad de los mapeos, e incrementarán la cantidad de tripletas resultantes del modelo generado, impactando así en el rendimiento del sistema.
Existen algunas limitaciones especificadas por los autores respecto del funcionamiento

de Morph-xR2RML, entre las cuales, las más significativas son: la imposibilidad de generar colecciones o contenedores RDF a partir de resultados de consultas SQL JOIN; la restricción de que los mapeos anidados sólo pueden contener un anidamiento simple, es decir, no pueden realizarse mapeos recursivos; bajo el modo de reescritura de consultas, no pueden realizarse mapeos que involucren distintas fuentes de datos en una sola regla.
Finalmente, dado que el mapeo realizado es de MongoDB a RDF, puede afirmarse que la expresividad semántica de esta solución es, precisamente, RDF.

MongoGraph

MongoGraph es un componente de la *triplestore* AllegroGraph (Franz Inc., 2006), el cual permite la interacción de un repositorio de AllegroGraph con una base de datos MongoDB ya existente. Esta interacción se realiza a través de la inclusión de tripletas especiales, denominadas *linking triples*, o tripletas de enlace, dentro del repositorio, las cuales referencian a un elemento en particular de una colección de MongoDB a través de su ID única. Una vez establecidas estas referencias, puede consultarse la base de datos MongoDB mediante SPARQL usando el predicado especial *find*.
La Figura 5 muestra ejemplos de algunas tripletas de enlace.

```
@prefix f: <http://www.franz.com/> .
@prefix id: <http://www.example.com/id#> .
@prefix xsd: <http://www.w3.org/2001/XMLSchema#> .

id:Restaurant1 f:hasMongoId "30075445".
id:Restaurant2 f:hasMongoId "30112340".
id:Restaurant3 f:hasMongoId "30191841".
id:Restaurant4 f:hasMongoId "40356018".
```

Figura 5. Tripletas de enlace de MongoGraph.

No obstante, MongoGraph no permite una abstracción total de MongoDB, dado que, en primer lugar, el predicado *find* toma como argumento una consulta MongoDB tradicional, y retorna el individuo que hace referencia al elemento existente en la colección de MongoDB, y, por otro lado, las dos bases de datos deben mantenerse paralelamente por separado, por ejemplo, si se incluye un nuevo elemento dentro de MongoDB, este elemento no puede ser accedido desde MongoGraph hasta que se haya incluido la tripleta de enlace al repositorio de AllegroGraph, o, en el caso inverso, si se elimina una tripleta de enlace del repositorio, el elemento que era referenciado por esta tripleta permanece en la colección de MongoDB hasta que se elimine manualmente de allí. Además de la limitación de abstracción de MongoDB, este mecanismo permite observar que el factor principal dentro del rendimiento de la solución residirá no en MongoGraph, ya que la interacción no hace más que valerse del predicado *find*, el cual ejecuta la consulta de MongoDB textualmente, sino en el rendimiento de MongoDB en sí, el cual ha demostrado un buen rendimiento siempre y cuando el esquema esté bien definido (Kanade et al., 2014). La ejecución de la consulta presentada en la Figura 3 (B) demoró 37.38ms.
En cuanto a la expresividad semántica de MongoGraph, si bien, como ya se mencionó, no es posible abstraerse de la sintaxis de MongoDB, los individuos de las tripletas de enlace pueden usarse como cualquier otra tripleta dentro del repositorio de AllegroGraph, y como tal, pueden formar parte de nuevas tripletas que extiendan la información ya presente en MongoDB. A su vez, AllegroGraph soporta razonamiento *RDFS++* (razonamiento RDFS con determinados axiomas de OWL), y la generación de reglas y *scripting* mediante diversos lenguajes, comunmente Common Lisp, Prolog o JavaScript.

Tripod

Tripod (Talis Group Ltd., 2014), desarrollado por la empresa Talis, es un sistema que cumple la función de mapeador objeto-grafo, con el objetivo de utilizar MongoDB como capa de almacenamiento para grafos RDF. Como tal, no constituye un sistema de OBDA en sí, sino que más bien es más cercano a los *triplestores* tradicionales. El almacenamiento de los datos RDF se hace, no mediante tripletas tradicionales, sino haciendo uso del concepto de Concise Bounded Description (Stickler, 2005) (CBD). Si un documento RDF puede entenderse como un grafo, el CBD de un elemento *x* perteneciente a dicho grafo, será un subgrafo en el cual estarán todas las tripletas cuyo sujeto sea *x*. De esta manera, puede

proveerse una descripción sencilla de un individuo en cuestión dentro del grafo, minimizando la necesidad de recorrerlo. Tripod almacena sus datos manteniendo colecciones de CBD en MongoDB. Considerando esto, Tripod no soporta SPARQL, ni recorrer arbitrariamente el grafo RDF que aloja, sino que la capacidad de consulta y búsqueda se provee a través de tablas lógicas (como reemplazo de SPARQL SELECT) y vistas (como reemplazo de SPARQL DESCRIBE). Éstas tablas y vistas se definen a través de un archivo de configuración, el cual especifica las colecciones de CBD, los atributos a considerar, sobre qué atributos se generan índices, y si existen *joins*, bajo qué campos se realizan.

Teniendo en cuenta lo anterior, puede afirmarse que Tripod satisface una necesidad en particular que poco tiene que ver con OBDA o la Web Semántica, e incluso, en cierta manera, atenta contra los casos de uso para los que MongoDB fue diseñado. La principal limitación de Tripod es que requiere, no sólo que los datos estén previamente en formato RDF, sino también conocer su estructura de manera *a priori*. En base a esto, podemos afirmar que en el caso de Tripod, no existe una ganancia significativa en términos de expresividad semántica.

A través de estos mecanismos, Tripod promueve una forma de uso más similar a una base de datos relacional, que a una *triplestore*, o MongoDB. Como tal, también presenta algunas ventajas propias de las bases de datos relacionales, como su eficiencia para la lectura, y su soporte de atomicidad y transacciones.

Se concluye esta sección con una síntesis de

Tabla I. Síntesis de los resultados.

	Morph-xR2RML	**MongoGraph**	**Tripod**
Objetivo	Mapeo de MongoDB a RDF.	Interfaz de AllegroGraph con MongoDB.	Uso de MongoDB como capa de almacenamiento.
Requisitos técnicos	-	Se debe mantener de forma paralela tanto AllegroGraph como MongoDB.	Los datos deben estar en RDF. Se requiere un fuerte conocimiento previo de la estructura de los datos para diseñar tablas y vistas.
Mecanismo de consulta	SPARQL.	SPARQL. Las consultas relacionadas a datos almacenados en MongoDB se hacen mediante la sintaxis de Mongo Shell.	Tablas y vistas definidas con anterioridad.
Capacidad de abstracción	Total.	Parcial. Debe usarse la sintaxis de Mongo Shell para consultas	Nula.
Mecanismos de OBDA	Reescritura de consultas y materialización.	Enlace a través de tripletas de enlace.	-
Expresividad semántica	RDF.	RDFS++, con la posibilidad de definición de reglas deductivas y *scripts.*	-
Licencia	Open Source.	Propietario. Gratuito hasta 5 millones de tripletas.	Open Source.

las diferencias y similitudes ya mencionadas sobre las tres herramientas, con el agregado del esquema de licenciamiento. La misma se presenta en la Tabla I.

DISCUSIÓN

Analizando la capacidad expresiva de las distintas soluciones, se ha observado que MongoGraph provee una gran flexibilidad para extender la capacidad semántica de RDF haciendo uso de distintos lenguajes de *scripting* y definición de reglas. Para el caso de Morph-xR2RML, la opción tradicional para obtener una semántica más expresiva que RDF sería llevar a cabo un proceso de materialización, con el cual se obtendría el conocimiento asercional, y extender este conocimiento asercional mediante el uso de conocimiento terminológico (axiomas OWL) o la definición de reglas deductivas.

Sin embargo, la materialización no siempre es aplicable, por ejemplo, en un dominio en el que los datos cambian con mucha frecuencia, no podría aplicarse materialización. A su vez, la materialización implica la manutención de un conjunto de datos paralelo al original (el almacenado en MongoDB), con los costos técnicos que esto conlleva.

Para salvar estas dificultades, diversos trabajos están siendo realizados sobre el campo de reescritura de consultas con foco en OBDA (Pérez-Urbina et al., 2009, Eiter et al., 2012, Imprialou et al., 2012 y Nakkerud, 2014). Estos sistemas y algoritmos toman como entrada un conjunto de axiomas de conocimiento terminológico, y otro de conocimiento asercional, y realizan la transformación del primero, generando una semántica equivalente mediante la reescritura del conocimiento terminológico en consultas, las cuales pueden ser respondidas usando exclusivamente el conocimiento asercional. De esta manera, es posible extender la semántica de un conjunto de datos sobre el que no se posee control total, por ejemplo, un *endpoint* SPARQL. Esta posibilidad resulta de suma importancia no sólo para el caso de OBDA sobre MongoDB, sino que abre camino sobre el razonamiento e integración sobre bases de conocimiento distribuídas (en conocimiento asercional y terminológico), un eje central en el contexto de la Web Semántica, y con lo cual, constituye una línea de investigación relevante dentro del campo de las tecnologías semánticas.

CONCLUSIONES

Se presentó un análisis de las herramientas actuales que permiten la manipulación de conjuntos de datos RDF usando MongoDB. MongoGraph y Morph-xR2RML permiten resolver problemas similares de maneras distintas, con lo cual la elección de alguna de estas herramientas debe realizarse evaluando y priorizando los requisitos puntuales del dominio de aplicación. Tripod limita esencialmente la expresividad de los datos, y tal vez pueda obtenerse un mejor balance de rendimiento y flexibilidad utilizando *triplestores* tradicionales.

AGRADECIMIENTOS

Este trabajo se realizó en el marco del proyecto “Interoperabilidad de sistemas de información en empresas de manufactura utilizando tecnologías semánticas” – UTI3810TC. Se agradece el apoyo brindado por la Universidad Tecnológica Nacional.

REFERENCIAS

Berners-Lee, T., Hendler, J., Lassila, O. (2001). The Semantic Web: A new form of Web content that is meaningful to computers will unleash a revolution of new possibilities. *Scientific American*.

Bizer, C., & Seaborne, A. (2004, November). D2RQ-treating non-RDF databases as virtual RDF graphs. *Proceedings of the 3rd international semantic web conference (ISWC2004)* (Vol. 2004). Springer.

Stickler, P. (2005). CBD - Concise bounded description. *W3C Member Submission*, *3*, 29.

Franz Inc. (2006). Allegro graph: RDF triple database. Web: http://franz.com/agraph/

Hitzler, P., Krötzsch, M., Parsia, B., Patel-Schneider, P. F., & Rudolph, S. (2009). OWL 2 web ontology language primer. *W3C recommendation*, *27*(1), 123.

Pérez-Urbina, H., Motik, B., & Horrocks, I. (2009). A Comparison of Query Rewriting Techniques for DL-lite. *Description Logics*, *477*, 29.

Spanos, D., Stavrou, P., Mitrou, N. (2010). Bringing Relational Databases into the

Semantic Web: A Survey. *Semantic Web Journal*, 13.
Arenas, M., Prud'hommeaux, E., Sequeda, J. (2011). A Direct Mappping of Relational Data to RDF. Web: https://www.w3.org/TR/rdb-direct-mapping/
Das, S., Sundara, S., Cyganiak, R. (2012). R2RML: RDB to RDF Mapping Language.
Eiter, T., Ortiz, M., Simkus, M., Tran, T. K., & Xiao, G. (2012, July). Query Rewriting for Horn-SHIQ Plus Rules. *AAAI*.
Imprialou, M., Stoilos, G., & Grau, B. C. (2012, July). Benchmarking Ontology-Based Query Rewriting Systems. *AAAI*.
Zakharyaschev, M. (2013). Ontology-based data access with databases: A short course. *Reasoning Web. Semantic Technologies for Intelligent Data Access* (pp. 194-229). Springer Berlin Heidelberg.
W3C SPARQL Working Group. (2013). SPARQL 1.1 Overview. *W3C Recommendation. W3C*.
Kanade, A., Gopal, A., & Kanade, S. (2014, February). A study of normalization and embedding in MongoDB. *Advance Computing Conference (IACC), 2014 IEEE International* (pp. 416-421). IEEE.
Kontchakov, R., Rezk, M., Rodríguez-Muro, M., Xiao, G., & Zakharyaschev, M. (2014, October). Answering SPARQL queries over databases under OWL 2 QL entailment regime. *International Semantic Web Conference* (pp. 552-567). Springer International Publishing.
Kontchakov, R., Rodríguez-Muro, M., & Manola, F., Miller, E., & McBride, B. (2014). RDF primer. *W3C recommendation*, *10*(1-107), 6.
Nakkerud, A. (2014). The tree-witness ontology rewriting with perfect mappings.
Priyatna, F., Corcho, O., Sequeda, J. (2014). Formalisation and Experiences of R2RML-based SPARQL to SQL query translation using Morph. *World Wide Web Conference (WWW 2014)*.
Talis Group Ltd. (2014). Web: https://github.com/talis/tripod-php
Michel, F., Djimenou, L., Faron-Zucker, C., Montagnat, J. (2015). XR2RML: Relational and Non-Relational Databases to RDF Mapping Language. Research Report.
Calvanese, D., Cogrel, B., Komla-Ebri, S., Kontchakov, R., Lanti, D., Rezk, M., Rodriguez-Muro, M., Xiao, G. (2016). Ontop: Answering SPARQL Queries over Relational Databases. *Semantic Web Journal*.
Michel, F., Faron-Zucker, C., Montagnat, J. (2016). A Generic Mapping-Based Query Translation from SPARQL to Various Target Database Query Languages. *12Th Proceedings of International Conference on Web Information Systems and Technologies (WEBIST'16).*