# Diseño de un repositorio RDF basado en tecnologías NOSQL

Ana Isabel Torre [1], Marta González [2], Arantza Illarramendi [3], Jesús Bermúdez [4]

[1,2] Tecnalia Research & Innovation
Parque Tecnológico Edif 202
Zamudio, 48170 - Vizcaya
+34 902760000
[3,4] Departamento de Lenguajes y Sistemas Informáticos. UPV-EHU.
Paseo Manuel de Lardizabal, 1
Donostia, 20018 - San Sebastián

[1] isabel.torre@tecnalia.com, [2] marta.gonzalez@tecnalia.com, [3] a.illarramendi@ehu.es, [4]jesus.bermudez@ehu.es

**Abstract.** Actualmente existen en la Web una cantidad ingente de datos en formato RDF, que pueden ser accedidos mediante el protocolo HTTP. Estos datos se encuentran publicados y vinculados los unos con los otros bajo el paradigma LinkedData. Un almacenamiento eficiente, requiere tener en cuenta, entre otros, los siguientes aspectos; distribución, escalabilidad y orientación a la consulta. Nuestra propuesta para lograrlo consiste en almacenarlos en bases de datos NOSQL, debido a sus propiedades de escalabilidad y rendimiento y a su naturaleza distribuida. En concreto en las de tipo familia de columnas, como Cassandra DB, ya que en su modelo de datos se puede mapear de forma simple el concepto de tripleta RDF (sujeto-predicado-objeto). En lo relativo a la consulta de los datos, planteamos en un primer momento la utilización del lenguaje estándar de consulta en RDF SPARQL, mediante un módulo intermedio que permita realizar la traducción desde este lenguaje a la consulta en bases de datos NOSQL, así como un mecanismo de indexación distribuido basado en MapReduce, otra tecnología proveniente del mundo NOSQL, que nos permitirá aumentar el rendimiento en el procesamiento de consultas. En resumen, esperamos obtener como resultado un repositorio RDF que mejore los tiempos de consulta y razonamiento de los datos y que ofrezca facilidades para llevar a cabo la escalabilidad y distribución de estos.

## 1 Introducción

Hoy en día se ha producido un aumento de la información disponible en la Web, propiciado por el crecimiento de Internet y la digitalización de contenidos [1]. En este contexto aparecen los datos vinculados que permiten la visualización, intercambio y asociación de la información, gracias a la utilización del lenguaje RDF para su formato, URIs referenciables para su identificación y el protocolo HTTP para su acceso. Con el objetivo de que estos datos sean procesables por las máquinas entran en juego las ontologías, que nos permiten dar a los datos sentido semántico y representar el co-

nocimiento del dominio al que pertenecen. En este contexto nos encontramos con un tipo de ontología, cuyo esquema o terminología del dominio (TBox), es pequeño en comparación con el número de instancias/individuos (Abox) que pueden llegar a existir para dicho dominio.

Sin embargo los repositorios RDF almacenan muchas tripletas y ello nos hace plantearnos la necesidad de un nuevo sistema de almacenamiento y consulta, ya que a pesar de existir un gran número de herramientas, seguimos detectando dos puntos conflictivos en los repositorios actuales: (1) la falta de facilidad en la distribución y escalabilidad [2] y (2) el bajo rendimiento en los tiempos de carga y consulta [2].

En el presente artículo se discute sobre una nueva alternativa para el diseño de la arquitectura de un RDFStore. Esta alternativa pretende contribuir con una posible solución apostando por el *mundo NOSQL* [11], que se aleja del concepto de base de datos relacional y está caracterizado principalmente por sus propiedades de distribución de datos y su alto rendimiento en grandes cantidades de datos, siendo precisamente sus puntos fuertes las carencias de las que antes hablábamos en los actuales TripleStore. Mediante el uso de sus dos tecnologías más conocidas, como son: las bases de datos NOSQL de tipo "familia de columnas", como Cassandra DB, y los modelos de programación MapReduce[12], como Hadoop, se pretende construir las capas de persistencia, indexación y razonamiento.

El artículo se va a dividir en las siguientes secciones: antecedentes de los repositorios RDF, un apartado explicativo de las tecnologías del mundo NOSQL, y las principales innovaciones que se aprecian en este nuevo enfoque de diseño. Por último, constarán las conclusiones y la línea de trabajo a futuro.

## 2 Antecedentes

Actualmente, existe un gran número de herramientas dedicadas al almacenamiento y consulta de sentencias RDF, por lo que para nuestro estudio hemos realizado una selección en base a la lista del W3C de los repositorios RDF más escalables [9]. Sobre esta lista hemos seleccionado los más conocidos, y con mayor bibliografía y documentación. También hemos seleccionado Oracle 11g, por su experiencia con sistemas de almacenamiento y distribución de datos, y por sus características en torno a escalabilidad, seguridad, fiabilidad y rendimiento [6]. Quedándonos finalmente con los siguientes: OpenLink Virtuoso [5], JENA [8], Sesame [7] y Oracle 11g.

Sobre estos, fijaremos las características que constituyen el marco de comparación a observar: a) tipo almacenamiento, b) número máximo de tripletas a almacenar, c) tiempos de consulta sin razonamiento incluido, extraídos de pruebas de rendimiento sobre el dataset DBPedia[1] [10], que es representativo de los conjuntos de datos que presentábamos en la introducción, d) tipo de razonamiento y e) existencia de materialización, valores que se muestran en la tabla 1.

Como conclusión del análisis de las distintas opciones mostradas en la tabla 1, se puede observar que OpenLink Virtuoso ofrece la mejor opción. Lo consigue basándose en Virtuoso Database, que consiste en un híbrido entre RDBMS, ORDBMS y ser-

[1] Ontología de múltiples dominios derivada de la información contenida en Wikipedia

vidores de ficheros. El resto de opciones basadas en bases de datos relacionales no ofrecen resultados destacables, lo que nos afianza en nuestra idea de que para repositorios del tipo presentado, los modelos y bases de datos relacionales tienen que quedar aparcados en beneficio de nuevas tecnologías, con características escalables y modelos de datos más acordes, como es el caso, en nuestra opinión, de las bases de datos NOSQL, como Cassandra DB. La búsqueda de alternativas que puedan aportar mayores beneficios que las bases de datos relacionales y los sistemas actuales, es la idea de artículos como el [4], en el que se estudia las posibilidades del almacenamiento mediante un particionamiento vertical similar a la forma de almacenamiento de las BDD NOSQL pero más primitivo.

| | Tip.Almac. | Nº Triples | T. Consulta | Tip.Razonam. | Material. |
|---|---|---|---|---|---|
| Open-Link Virtuoso | Virtuoso Database | 15.4 billones [9] | 26.69 seg. [10] | Hibrido/ Backward | SI |
| JENA | PostgreSQL | 200 millones[9] | 107.88 seg. [10] | Hibrido / For-Backward | SI |
| Sesame | PostgreSQL / MySQL | 70 millones [9] | 1 min. 30 seg.[10] | Sin inferencia/ Forward | SI |
| Oracle 11g | Oracle DB 11g | 1 billón [6] | Respuestas incompletas | Forward | SI |

**Table 1.** Tabla comparativa de las herramientas para la gestión de ontologias mas relevantes

## 3 Tecnologías del mundo NOSQL

Este apartado trata del concepto NOSQL, entendiendo éste como solución posible para el almacenamiento y manejo de una gran cantidad de información, fuera de todo lo que sea el mundo relacional. En este punto aparecen dos tecnologías a tener en cuenta, las bases de datos NOSQL y los modelos de programación para el desarrollo de sistemas altamente escalables y de alto rendimiento, MapReduce [13].

La primera la podemos definir como una nueva generación de almacenes de datos, que cumplen alguno de estos puntos: ser no-relacionales, distribuidos, escalables y de código abierto. Por otra parte se les pueden imputar una serie de características comunes, como un esquema libre, facilidad de replicación, API de acceso sencilla, incumplimiento de las propiedades ACID (Atomicidad, Consistencia, Aislamiento y Durabilidad) y capacidad de tratamiento de una enorme cantidad de datos. Existen diferentes tipos, pero nosotros nos centraremos en aquellas del tipo “familia de columnas”, cuyo principal exponente es Cassandra DB [12]. Este tipo ha sido elegido por su modelo de datos, que introduciremos a continuación, definiendo sus componentes:

- *Cluster*: Es el conjunto de nodos en los que está instalada la misma instancia lógica de Cassandra DB. Por cada cluster puede haber varios espacios de clave.
- *EspacioClaves*: Agrupación lógica de las distintas familias de columnas, normalmente se asocia a un concepto clave, la aplicación, en nuestro caso la ontología.

- *FamiliaColumnas*: Agrupa varias columnas relacionadas.
- *Row*: Es una agregación de columnas o supercolumnas asociadas por clave.
- *SuperColumna*: Representa el concepto de una columna que contiene subcolumnas.
- *Columna*: Es la unidad básica, y está formada por nombre, valor y timestamp.

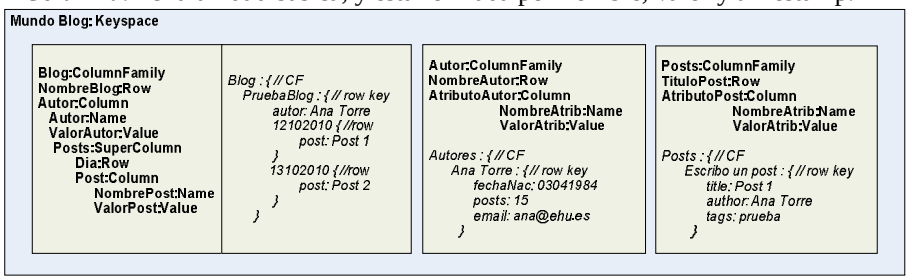


**Fig. 1.** Ejemplo del modelo de datos de Cassandra DB

En la figura 1 se muestra un ejemplo basado en los blogs, sus autores y sus posts, el espacio de claves MundoBlog. Las familias de columna serán autores y posts, que tendrán como row/clave el nombreAutor y tituloPost. Los blogs son otra familia de columnas, tendrán como row el título del blog, y contendrán a los autores (columna) y las entradas de estos (supercolumna) ordenadas por día de publicación (row/clave de la supercolumna).

En segundo lugar se encuentran los modelos de programación para la construcción de aplicaciones distribuidas, como Hadoop[2]. Estos modelos se basan en dos operaciones básicas map y reduce, en la primera se distribuyen los datos por los nodos del cluster en forma de pares clave-valor y en la segunda se recogen aquellos pares clave-valor que cumplan los criterios del resultado deseado. En el artículo [3], se habla de cómo crear una extensión de SPARQL, que mediante Hadoop permite la indexación de las consultas, proporcionándonos una referencia a tener en cuenta en la implementación de nuestro mecanismo de indexación.

## 4 Visión general de la arquitectura y principales innovaciones

Las funciones en las que se centra nuestro repositorio son el almacenamiento y la consulta de datos RDF, para ello en la figura 2 se plasma una posible arquitectura.

Las principales líneas de innovación se centran en los siguientes puntos: (1) la posibilidad de utilizar las bases de datos NOSQL como mecanismo de persistencia del repositorio, (2) la indexación y el razonamiento mediante mecanismos MapReduce y (3) los mecanismos de traducción entre el lenguaje SPARQL y los lenguajes de consulta NOSQL. Por esto los módulos más importantes de la arquitectura, sobresaltados en la figura 2 en un color más oscuro, son el *API de almacenamiento y seguridad de datos*, que nos abstrae del mecanismo de persistencia siendo una opción las bases de datos NOSQL, *el módulo de indexación*, que como ya hemos comentado se implementaría mediante Hadoop y el *módulo de traducción de consultas*.

[2] http://hadoop.apache.org/

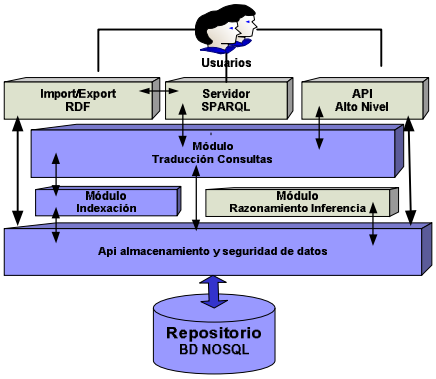


**Fig. 2.** Planteamiento de arquitectura de un repositorio RDF

Uno de los puntos más importantes es la escalabilidad, ya que como hemos visto en apartados anteriores los repositorios de este tipo de información trabajan con billones de triples y esta cifra sigue aumentando, por lo que para nosotros una opción lógica es la utilización de bases de datos NOSQL que precisamente están pensadas para proporcionar escalabilidad horizontal. Esta idea viene avalada por el artículo [14] en el que puede verse como los repositorios de columnas son un almacenamiento a tener en cuenta en los datos RDF.

Pero para que las bases de datos NOSQL pueden ser una opción a tener en cuenta, es necesario un correcto mapeo de la información de la ontología (tripletas RDF: sujeto-predicado-valor) al modelo de datos propio de la base de datos NOSQL (Cassandra DB). En la figura 3 mostramos un posible camino a seguir para ello. Se puede apreciar como la ontología forma un espacio de claves, formado a su vez por dos familias de columnas, que representan el esquema TBox y el esquema Abox. Dentro de cada uno de ellos habrá una supercolumna para almacenar los sujetos de las tripletas, que contendrá otra supercolumna, con los predicados de las sentencias RDF, en esta ltima supercolumna, habrá tres columnas con: el valor de la tripleta, información de la inferencia (en el caso de que exista materialización de esta) y la fecha de creación.

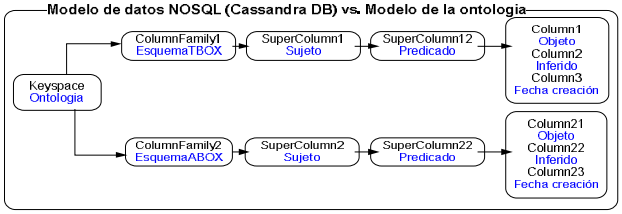


**Fig. 3.** Almacenamiento del modelo de una ontología mediante el modelo de datos NOSQL

Con este primer esbozo, se puede intuir cómo podría unirse el mundo de las bases de datos NOSQL con el de los repositorios RDF, con el principal propósito de una mayor escalabilidad y distribución de los datos, para así poder cumplir los requisitos de almacenamiento que requieren los datos vinculados (Linked Data).

## 5 Conclusiones y líneas de trabajo futuro

En este artículo, hemos intentado mostrar las posibilidades y ventajas que proporciona basar el diseño de un repositorio RDF sobre tecnologías NOSQL, en la búsqueda de una solución a los problemas de escalabilidad y rendimiento en el almacenamiento y la consulta de datos. Estos problemas son especialmente importantes en los datos vinculados (LinkedData), ya que las ontologías utilizadas para su representación tienen un número de instancias de dominio, su crecimiento es muy rápido y su utilización principal es la consulta de sus instancias. Como solución, intentamos ofrecer un nuevo enfoque integrando las ventajas de las bases de datos NOSQL y los modelos de programación MapReduce, a las características de los repositorios ya existentes.

En un futuro, la línea principal a seguir será la profundización en las características propias de los sistemas NOSQL y su adaptación al contexto considerado. Una vez se haya cumplido esta tarea el objetivo final será la implementación del repositorio y la realización de pruebas de evaluación entre ésta y los sistemas elegidos para el análisis. En resumen, esperamos obtener como resultado un repositorio que supere a los actuales en las características de escalabilidad, capacidad de almacenamiento y rendimiento en tiempo de carga, consulta y razonamiento.

## Referencias


[1] "Semantic framework for complex knowledge domains", Marta González, Stefano Bianchi, Gianni Vercelli, Proc. of the Poster and Demonstration Session at the 7th International Semantic Web Conference (ISWC2008)

[2] "Reasoning with large ontologies stored in relational databases: The OntoMind approach", Lina Al-Jadir, Christine Parent, Stefano Spaccapietra, Data - Knowledge Engineering (2010)

[3] "SPARQL Query Answering on a Shared-Nothing Architecture", Spyros Kotoulas, Jacopo Urbani, Proc.of the Workshop on Semantic Data Management (VLDB 2010)

[4] "Scalable Semantic Web Data Management Using Vertical Partitioning", D. J. Abadi, A. Marcus, S. R. Madden et al. Proceedings 33- Conference on Very Large Data Bases, (2007)

[5] "Implementing a SPARQL compliant RDF triple store using a SQL-ORDBMS. OpenLink Software Virtuoso", O Erling, Technical Report, OpenLink Software Virtuoso (2001)

[6] "Oracle Database Semantic Technologies/ Feature Overview" (http://www.oracle.com/technetwork/database/options/semantic-tech/index.html) - 28/02/2011

[7] "Sesame: A Generic Architecture for Storing and Querying RDF and RDF Schema", Jeen Broekstra , Arjohn Kampman , Frank Van Harmelen, Proc. of the First International Semantic Web Conference (ISWC 2002)

[8] "Jena: Implementing the RDF Model and Syntax Specification", Brian McBride, Hewlett Packard Laboratories, Semantic Web Workshop (2001)

[9] "Large Triple Store", (http://www.w3.org/wiki/LargeTripleStores) -28/02/2011

[10] "RDF Store Benchmark with DBPedia", C. Becker (2008)

[11] "NOSQL Databases", (http://nosql-database.org/)- 28/02/2011

[12] "Apache Cassandra", (http://wiki.apache.org/cassandra/) - 28/02/2011

[13] "MapReduce: Simplified Data Processing on Large Clusters", J. Dean, S. Ghemawat, Proc. of the 6th Symposium on Operating Systems Design and Implementation (2004).

[14] "Column-store support for rdf data management: not all swans are white", L. Sidirourgos, R. Goncalves, M. Kersten, N. Nes,, S. Manegold, *Proc. VLDB Endow.*, vol. 1, no. 2, pp. 1553–1563, 200, (2008)