Evolución y mejora de un índice espacial sobre JASPA para H2

S. González Prieto[(1)]*.; J.M. De Diego Alarcón*[(1)] *y Antoni Pérez-Navarro*[(1)]

(1) Estudios de Informática, Multimedia y Telecomunicación, Universitat Oberta de Catalunya, Rambla del Poblenou, 156. 08018. {sgonzalezpr,jde_diegoa,aperezn}@uoc.edu
.

RESUMEN

El presente trabajo introduce los trabajos realizados para la creación de un nuevo índice espacial para la extensión JASPA sobre H2. El punto de partida es un primer trabajo presentado en las V jornadas de SIG libre en el que se introdujo el índice espacial a la extensión JASPA. En el presente trabajo, el índice se adapta a la nueva versión de JASPA e incluye mejoras sustanciales, entre ellas, la más destacable es la integración del índice con los operadores y funciones espaciales de JASPA. Se ha mejorado, además, el acceso al índice espacial desde consultas SQL y se ha profundizado en la explotación de las tablas de metadatos definidas en el estándar SQL *simple features access* del OGC y en las sentencias de creación y modificación del índice.

El algoritmo de indexación elegido para la implementación del índice espacial ha sido el RTree. La implementación se ha realizado en el lenguaje de programación JAVA, lo que ha facilitado su integración con la extensión JASPA y la base de datos H2. Los desarrollos se han adaptado a la nueva versión de JASPA.

Con este desarrollo se ofrece una alternativa al almacenamiento de información geográfica en el entorno de un Sistema Gestor de Bases de Datos Relacionales (SGBDR) monousuario. El índice espacial desarrollado permitirá reducir los accesos a tablas mejorando sensiblemente el rendimiento de las funciones y operadores espaciales.

Palabras clave: *Índice espacial, RTree, JASPA, H2*

ABSTRACT

This paper introduces the work done to create a new spatial index for the extension JASPA on H2. The starting point is a first paper presented at the V Jornadas de SIG libre in which a the spatial index to the JASPA extension was introduced. In this new work, the index is suited for the new version of JASPA and includes substantial improvements, the most important being the integration of the index with the operators and spatial functions of JASPA. It has been also improved the access to the spatial index from SQL queries and work has been done on the exploitation of metadata tables defined in the SQL standard simple features access *of OGC and in the sentences of creation and modification of the index.*
The indexing algorithm chosen to implement the spatial index has been RTree. The implementation was done in the Java programming language, which facilitated its integration with the extension JASPA and H2 database. The developments have been adapted to the new version of JASPA.
With this development an alternative is provided to storing geographic information in the environment of a single-user RDBMS. The spatial index developed will reduce access to tables and significantly improve the performance of the functions and spatial operators.

Max: 300 words

Key words: *Spatial index, Rtree, JASPA, H2*

INTRODUCCIÓN

El uso de Bases de Datos (BBDD) ligeras para el almacenamiento de información espacial y su posterior explotación en Sistemas de Información Geográfica (SIG) constituye una tendencia en los últimos años en algunas de las aplicaciones SIG propietarias más extendidas.

Este tipo de Bases de Datos resuelven varios problemas en la gestión de información geográfica:

- Permiten el modelado relacional de información geográfica, no soportado por los formatos tradicionales basados en archivos como shapefile (shp).
- Facilitan el intercambio de información geográfica entre usuarios.
- Reducen al máximo las necesidades de administración y consumo de recursos, en comparación con las BBDD corporativas.

En 2009 se presentó la extensión JASPA (Java Spatial), soportada sobre PostgreSQL y H2. JASPA recoge funcionalidad espacial sobre PostgreSQL y H2 e implementa el estándar del Open Geospatial Consortium SQL Implementation Specification for Geographic information - Simple feature access - Part 2: SQL option.

En 2011, se desarrolló un índice espacial para esta extensión para acelerar las operaciones sobre campos con información espacial en la base de datos H2. El presente trabajo amplía aquél con: la integración del índice con los operadores y funciones espaciales de JASPA; la mejora del acceso al índice espacial desde consultas SQL; y una mejora en la explotación de las tablas de metadatos definidas en el estándar SQL simple features access del OGC, en las sentencias de creación y modificación del índice.

El presente documento se estructura de la siguiente forma: se empieza mostrando las herramientas utilizadas. A continuación se recuerda cuál es la situación de partida para, seguidamente, entrar ya en el proyecto en sí. En este sentido se definen primero los requisitos de diseño y se describe a continuación como se ha implementado la solución y cómo llevar a cabo una prueba. Finalmente se exponen las conclusiones.

HERRAMIENTAS UTILIZADAS

Antes de pasar a la descripción técnica del trabajo, vamos en una primera instancia a revisar los instrumentos que soportarán el desarrollo realizado: H2 como el sistema de gestión de bases de datos donde se almacena la información que gestionamos y sobre el que se debe construir el índice espacial (elemento básico dentro de este desarrollo); JASPA como una extensión de java, lenguaje de programación sobre el que se crea la solución, para poder explotar los recursos de la base de datos h2 creados para el sistema espacial; y JSI como una extensión más de java que nos va a permitir trabajar con índices rtree.

H2

H2 es un sistema programado en su totalidad en Java. Fue publicado por primera vez en 2005 por Thomas Mueller, y cuya principal característica es que puede ser integrada completamente en los aplicativos desarrollados en este lenguaje. Así, se puede conectar a través de sockets como ocurre con otras bases de datos como MySQL o PostgreSQL, pero también se puede llamar de forma directa en este lenguaje, lo que permite na mayor velocidad e integración entre ambos elementos.

JASPA

JASPA (Java SPAtial) es una extensión en lenguaje JAVA que permite operar en bases de datos relacionales dentro del entorno de datos de Sistemas de Información Geográfica (SIG). Ofrece cerca de 200 procedimientos almacenados en SQL, lo que le permite tener una funcionalidad muy similar a la que cuenta PostGIS 1.4. Sigue los estándares de OGC respecto Simple Features for SQL, y está soportado para los SGBD PostgreSQL y H2. El 19 de Julio de 2011 se liberó la versión 0.2 de JASPA, que cuenta con nuevas características.

Sin embargo, en relación al presente trabajo, el aspecto más importante es que JASPA incorporaba un índice espacial para PostgreSQL basado en GIST, pero no incorporaba índice espacial para H2. Este es el aspedcto tratado por el trabajo de Calvillo et al. y que se continua en el presente proyecto.

JASPA proporciona también una serie de herramientas para poder trabajar con datos espaciales en las bases de datos, aún así tenemos un elemento importante relatado en puntos previos que no se engloba dentro de esta extensión, y es el trabajo con un índice espacial, en nuestro caso RTree. Para ello, nos vamos a apoyar en otra extensión Java llamada JSI.

JSI (Java Spatial Index)

JSI (Java Spatial Index) es un proyecto de código abierto, desarrollado por Aled Morris dentro del grupo Sourceforge, para implementar en lenguaje java el algoritmo de indexación espacial RTree.

Aunque aporta pocas funciones por defecto, ofrece un alto rendimiento y cuenta al menos con las operaciones básicas indicadas en la sección de RTree de este documento (consulta, adición y eliminación de objetos).

Las funcionalidades que ofrece se distribuyen en dos grupos principales:

- En un primer grupo encontramos las clases java que implementan los elementos geométricos simples: el punto (Point), el rectángulo (Rectangle), listas de prioridad (PriorityQueue) y la interfaz del índice espacial (SpatialIndex). Las clases Point y Rectangle definen básicamente ambos objetos espaciales con sus parámetros correspondientes. La clase PriorityQueue permite gestionar colas con prioridad para acelerar la gestión

de los elementos incluidos en los nodos del índice espacial a la hora de realizar operaciones en él y, finalmente, la clase SpatialIndex define los atributos y funciones que debe tener el índice espacial al implementarse.

- En un segundo grupo se encuentran las clases que se encargan de gestionar los elementos básicos del árbol RTree: los nodos (Node), la implementación de la interfaz de índice espacial (RTree) definida en la interfaz SpatialIndex y las listas ordenadas de elementos (SortedList).

Es dentro de la clase RTree donde se localizan los método que van a permitir ejecutar las operaciones geométricas con los datos indexados: localizar los elementos más cercanos, buscar los elementos que se incluyen dentro de un rectángulo dado, encontrar elementos que intersectan con otra geometría, etc.

SITUACIÓN DE PARTIDA

En esta sección, aportaremos una visión sobre cómo estaba diseñado el sistema en la fase previa al estudio de las soluciones a los requerimientos indicados a partir de los documentos y el software proporcionados en trabajos previos.

A nivel de elementos base se utilizan principalmente tres:

- Base de Datos H2
- Biblioteca JASPA para el manejo de información espacial
- Biblioteca JSI para el manejo de árboles R.

El índice espacial creado sobre JASPA para la base de datos H2 se implementó sobre una tabla auxiliar.. Esta tabla presenta una entrada para cada uno de los objetos geométricos indexados, en relación 1 a 1.

El índice espacial se genera en memoria a partir de esta tabla auxiliar lo que permite que la creación del índice sea muy rápida, y que la posterior ejecución de las operaciones sobre él sean muy eficientes, ya que no tiene que ir a la tabla de la base de datos cada vez que se necesita recorrerlo.

La tabla auxiliar tiene la estructura que vemos en la tabla S'ha produït un error: No s'ha trobat la font de referència. Los principales elementos de la tabla son:

- Un identificador definido como clave primaria que, como hemos indicado anteriormente, permite relacionar la tabla con los datos geométricos.
- Cuatro campos en los que se definirán las dimensiones del rectángulo mínimo (MBR) que contiene el objeto espacial.

Esta tabla auxiliar se crea de forma automática al crear el índice espacial, y está sincronizada automáticamente con la tabla que contiene los objetos espaciales mediante *triggers* en la base de datos H2. De esta forma cualquier actualización quede reflejada tanto en la tabla como en el índice generado en memoria con los posteriores cambios que se produzcan por eliminación, adición o cambios en los registros.

Tabla 1 Descripción de la estructura de datos tabla índice auxiliar

SPATIALDATAMBRTABLE_<TABLA>_<COLUMNA GEO>		
Definición: Es la estructura de datos que almacenará los MBR[1] de cada uno de los registros *de la tabla que contiene los datos geométricos que se desea indexar.* *Organización: Indexada por el campo GID y está relacionada con la tabla que contiene los datos geométricos a indexar por este mismo campo.*		
Datos elementales que la componen:		
Nombre	*Descripción*	*Tipo*
GUID	*Corresponde al identificador de cada registro. Este campo será el campo índice de esta tabla para relacionarlo con la tabla base que contiene los datos geométricos.*	*INTEGER*
MINX	*Corresponde al valor menor de X del vértice del MBR*	*FLOAT*
MINY	*Corresponde al valor menor de Y del vértice del MBR*	*FLOAT*
MAXX	*Corresponde al valor mayor de X del vértice del MBR*	*FLOAT*
MAXY	*Corresponde al valor mayor de Y del vértice del MBR*	*FLOAT*

Esta capa de ejecución permite que la base de datos H2 se mantenga siempre indexada cuando se añaden, modifican o eliminan elementos,

Por otro lado, también existe una capa de programación en la cual el usuario puede, de forma activa, solicitar la creación de índices, así como su activación, eliminación, inserciones, etc. a través de línea de comandos.

A pesar de todas estas ventajas, crear los índices en memoria presenta algunos inconvenientes cuando las bases de datos contienen muchos datos, o cuando se quiere trabajar con múltiples índices en paralelo. Para resolver estos inconvenientes es necesario:

- Es necesario implementar una persistencia para el algoritmo del Rtree.
- Se realizarán algunos filtrados previos a la generación del índice, mejoras en las consultas SQL y en la integración con las tablas de metadatos del OGC que maneja JASPA en H2.

Además, dado que la versión actual de JASPA es la 0.2, se han tenido que adaptar a esta versión los desarrollos previos.

[1] Al rectángulo mínimo que engloba a un objeto se le denomina REM (Rectángulo Espacial Mínimo), en inglés, MBR (Minimum bounding rectangle)

Otro de los requerimientos ha sido llevar a cabo el desarrollo sin modificar el código original de los elementos base (H2 y JASPA), por lo que sólo se han añadiedo nuevas clases java o extendiendo aquellas relativas al uso del índice RTree, para cuya base se tomaron las librerías JSI ya existentes.

En el siguiente apartado describiremos cuáles son los requisitos de diseño y, a continuación, describiremos en detalle como se ha implementado la solución.

REQUISITOS DE DISEÑO

Las decisiones de diseño son:

- La utilización de la biblioteca JSI para la implementación de los árboles R que sirven de base a los índices espaciales a desarrollar.

- La modificación de dos de las funciones implementadas en JSI (contains e intersect) para eliminar uno de los parámetros necesarios en el desarrollo original y que en nuestro caso no es necesario. Este parámetro que aparece en el código original, permite ejecutar un procedimiento predefinido sobre cada uno de los elementos que se obtienen en las funciones de getIntersectFunction (método que devuelve los elementos del índice que están total o parcialmente incluidos en un rectángulo dado) y getContainsFunction (método que devuelve los elementos del índice que están totalmente incluidos en un rectángulo dado). También es necesario modificar la función findLeaf .

Por otro lado, se busca crear una solución basada en multiíndices. Para ello:

- Se modifica el número de hijos por nodo de 10 a 6 para mejorar el rendimiento cuando se utilizan una cantidad de datos no demasiado extensa (<50000 elementos).
- Se realizan los trigger por implementaciones de la clase interfaz de H2

Finalmente, es importante destacar que los índices espaciales también van permitir realizar filtros previos sobre los datos que utilizan las funciones JASPA y, por tanto, los propios índices van a redundar en una mejora del rendimiento de estos métodos.

Pero, ¿cómo se concretarán estas decisiones?

DESCRIPCIÓN DE LA SOLUCIÓN

Tras analizar el estado de la versión anterior del índice, y teniendo en cuenta las decisiones de diseño que se han tomado, se concreta en este punto cómo se ha llevado a cabo la implementación del índice.

El análisis realizado en el punto anterior, y teniendo en cuenta los requerimientos solicitados para el proyecto, hay varios aspectos que es necesario solventar:

- El desarrollo anterior estaba realizado sobre la versión 0.1 de JASPA. La versión 0.2 actual tiene cambios en el modelo de datos que influyen en los desarrollos realizados.
- La solución actual, sólo gestiona el uso de un índice en memoria a la vez, por lo que su uso para gestionar varias columnas geométricas no es posible.
- El usuario debe de cargar en memoria el índice cada vez que entra al sistema, lo que conlleva más carga de trabajo para éste.
- Los *triggers* o disparadores de actualización de los datos en las tablas origen no actualizan el árbol en memoria, sino solamente la representación

de este árbol en la base de datos, lo que conlleva tener que recargar este árbol manualmente cada vez que se produce un cambio, o bien que el índice en memoria no corresponda con los datos reales a los que representa.

- El índice creado no se utiliza para mejorar el rendimiento de las consultas propias de la extensión JASPA ya que no crea o modifica sus funciones originales para hacer uso de él, sino que solamente lo hacen las propias funciones de la librería JSI a la que se hicieron ciertas modificaciones.
- Para solventar los puntos anteriores, vamos a ir desgranando a alto nivel las distintas soluciones aportadas para, posteriormente, indicar cuáles han sido las mejoras que se han introducido.
- Respecto al cambio de versión de la librería JASPA, además de la inclusión de nuevas funciones espaciales y de corregir algunos fallos en el sistema, se elimina del modelo de datos la tabla GEOMETRY_COLUMNS_EXTEND. Esta tabla era utilizada por el desarrollo inicial para indicar si un índice está activo o no en el sistema, y de forma gráfica seguía el esquema de la Figura 1.

En nuestro caso, como no vamos permitir que el usuario active o desactive los índices, no se tendrá en cuenta este uso por lo que simplemente eliminaremos esta funcionalidad, consiguiendo también acabar con la incompatibilidad del aplicativo con la versión JASPA 0.2.

En cuanto a la limitación de contar con un solo índice en memoria a la vez, se solventará con la creación de un array de índices RTree, apoyándonos en una nueva tabla ACT_IDX (descrita con más detalle en el apartado 5.1.4) dentro del modelo de datos, que nos ayudará a saber qué índice es el que pertenece a cada campo geométrico. Esta tabla contendrá los datos básicos para identificar el elemento geométrico (tabla y campo) y la posición que su índice ocupa dentro del array de índices espaciales. Con ello, podremos identificar en cada momento necesario en cuál de los árboles R debemos realizar la búsqueda de los elementos solicitados.

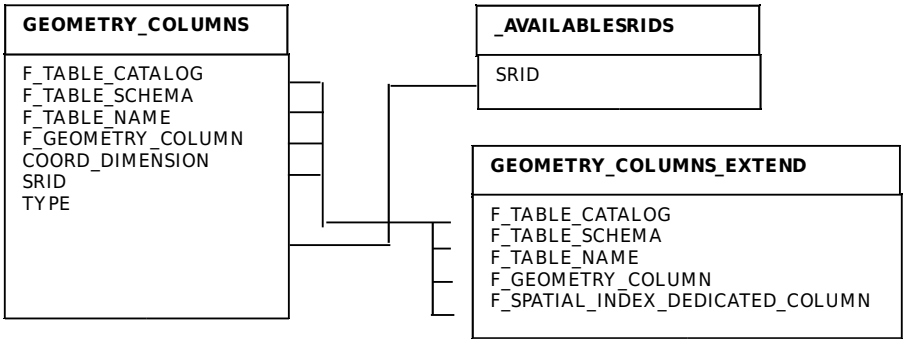

Figura 1: Modelo de datos JASPA 0.1

Todos estos índices se cargarán en memoria de forma automática en el momento en que el usuario entra en el sistema de H2. Para ello, se añadirá un nuevo parámetro dentro de la url de conexión a la base de datos, que permitirá la ejecución de un script que realizara la carga de los índices tanto persistente como en memoria. La información que para cada elemento geográfico se está almacenando en estos índices es muy pequeña en cuanto a su espacio reservado, por lo que podría utilizarse para capas geográficas cuya cantidad de registros fuera alta.

El script se apoya, entre otras, de la tabla de metadatos de JASPA GEOMETRY_COLUMNS, que es la tabla que se utiliza para mantener el estándar del OGC a nivel de modelo de datos, para saber los índices que deben ser creados.

Además, estos índices serán actualizados tanto de forma persistente como en memoria a través de los *triggers* de cada tabla de datos, sin que el usuario tenga que realizar ninguna acción para que estos se mantengan al día.

Con todo ello, se resuelven algunos de los inconvenientes encontrados en el sistema inicial, pero tenemos que tener en cuenta también que se han añadido las siguientes mejoras:

- Adaptación de los scripts ya existentes que utilizan el índice espacial en memoria para que puedan tener en cuenta la situación de multiíndices actual. Tendremos en cuenta todas las funciones a las que afecta esta situación que no son sólo aquellas que diseñemos dentro del ámbito JASPA como se indica en el siguiente punto, sino también las clases de JSI que fueron modificadas en el proyecto anterior para eliminar acciones que no eran necesarias dentro de su ámbito (getIntersectFunction y getContainsFunction).
- Creación de funcionalidades nativas de JASPA para que utilicen los índices espaciales existentes actualmente, una opción que la librería no incluye de forma nativa para la base de datos H2.

A continuación veremos cuál es el modelo de datos que se ha utilizado en esta nueva implementación.

Modelo de datos

El modelo de datos básico del sistema está basado en el de la librería JASPA en su versión 0.2, esquema diseñado para cumplir la normativa del estándar OGC respecto al almacenamiento de los datos espaciales, y que podemos ver en la figura 2.

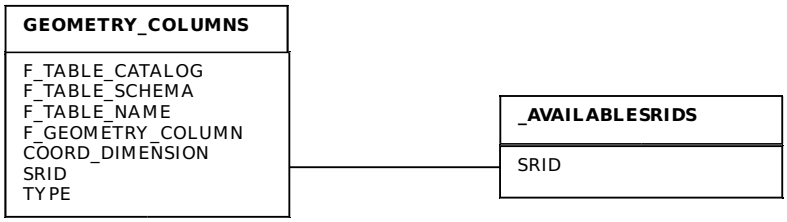

Figura 2. Estructura de tablas de la extensión JASPA 0.2

Los datos de la tabla GEOMETRY_COLUMNS se incorporan de forma automática cuando desde una sentencia SQL en H2 se incorpora un campo geométrico al sistema utilizando el método addgeometrycolumn de JASPA. Esta operación, además de incluir el campo en la tabla de datos concreta que estamos modificando, también redirige esta información para ser incorporada en el esquema de JASPA.

En nuestro caso, vamos a apoyarnos en la tabla GEOMETRY_COLUMNS y en los datos que en ella se incorporan para utilizarlos en el momento de la creación automática de los índices espaciales, tanto en memoria como en la propia Base de Datos, ya que nos va a permitir conocer cuántos y cuáles son los campos geométricos a indexar.

Tabla 2. Descripción de la estructura de datos que almacena el gestor de índices

ACT_IDX
Definición: Es la estructura de datos que almacenará los campos relativos a la tabla de datos y el campo geométrico que se indexa, así como la posición del índice relativo a ella dentro del array general de índices espaciales. **Organización:** Indexada por los campos TABLENAME y FIELD y está relacionada con la tabla que contiene los datos geométricos a indexar por este mismo campo.

Datos elementales que la componen:		
Nombre	Descripción	Tipo
TABLENAME	Corresponde al nombre de la tabla de datos que contiene el campo geométrico	VARCHAR
FIELD	Corresponde al campo geométrico a indexar perteneciente a la tabla TABLE NAME	VARCHAR
POSITION	Corresponde a la posición dentro del array de índices espaciales que ocupa	INT

Además del modelo de datos que aporta la librería JASPA 0.2, y de la extensión realizada por el proyecto base con las tablas que realizan la persistencia de los datos del índice espacial, se ha visto necesaria la creación de un nuevo elemento para poder gestionar el uso de múltiples índices espaciales relacionados con distintos campos geométricos.

Esta nueva tabla (cuya descripción podemos ver en la Tabla 2), contiene los datos básicos para localizar el campo geométrico así como la posición de su índice en memoria dentro de la lista de índices activos.

La tabla se relaciona mediante su clave primaria con la tabla GEOMETRY_COLUMNS de metadatos de JASPA de forma 1:1, de forma que no pueda crearse dos índices distintos para el mismo campo geométrico.

PRUEBA DEL SISTEMA

En este punto se va a llevar a cabo una prueba del desarrollo llevado a cabo. Para ello:

- Inicialmente, hay que crear el siguiente ALIAS en H2 que nos permita que sea más cómoda la llamada a la función de arranque de sesión:

```
CREATE ALIAS IF NOT EXISTS JASPA.ST_INIT_SPIDX FOR "FunctionIdx.st_init_IDX"
```

- Se realiza la carga de los ficheros sql con los datos de ejemplo (en nuestro caso soilsh2.sql, riversh2.sql, usesh2.sql y países.sql).
- A continuación, se crea el script de carga del sistema (arranque.sql), en el cual hay que introducir la siguiente sentencia:

```
SELECT ST_INIT_SPIDX();
```

- Se modifica la cadena url de conexión según corresponda, que en nuestro caso puede ser algo por la siguiente:

jdbc:h2:tcp://localhost/~/myfirstjaspadb;INIT=RUNSCRIPT FROM '~/arranque.sql';SCHEMA_SEARCH_PATH=PUBLIC,JASPA

- Cuando aparezca la pantalla de H2, tendremos creadas las tablas persistentes de índices para los campos geométricos cargados, y que se encuentran también en las tablas de Metadatos de JASPA.

- A partir de este momento, ya podemos realizar consultas espaciales utilizando las funciones JASPA o las incorporadas en nuestro proyecto.

SELECT ST_CONTAIN_SPIDX(TablaGeometrica,CampoGeometrico,minX,MinY,MaxX,MaxY)

- La función anterior devuelve los elementos de TablaGeometrica con el campo geométrico CampoGeometrico incluidos totalmente dentro del rectángulo formado por los puntos minX, minY, maxX y maxY, utilizando el índice espacial

SELECT ST_INTERSECTS_SPIDX(TablaGeometrica,CampoGeometrico,minX,MinY,MaxX,MaxY)

- La función anterior devuelve los elementos de TablaGeometrica con el campo geométrico CampoGeometrico incluidos total o parcialmente dentro del rectángulo formado por los puntos minX, minY, maxX y maxY utilizando el índice espacial.

SELECT ST_INTERSECT_SPIDX('RIVERS','GEOM',5700,4600,6200,6000);

SELECT _st_intersects(geometriaA, TablaGeometricaA, CampoGeometricoA, geometriaB, TablaGeometricaB, CampoGeometricoB)

- Esta función devuelve el resultado de la operación de intersección entre la geometría A y la geometría B, tomando como base los índices espaciales de cada uno de los campos.

Para poder realizar la prueba, la mejor forma de mostrar su utilidad ha sido el combinarla como filtro previo a la ejecución de una función estándar JASPA. En este caso hemos usado la función ST_INTERSECTION(geometríaA, geometríaB) que obtiene las geometrías resultantes de hacer intersección entre los elementos de A y B. La instrucción utilizada es la siguiente:

SELECT ST_INTERSECTION (A.GEOM, B.GEOM) FROM USES A, SOILS B WHERE _ST_INTERSECTS(A.GEOM,'USES','GEOM',B.GEOM,'SOILS','GEOM');

Los resultados obtenidos sin el uso del filtro, devuelven una gran cantidad de resultados con valor *null*, es decir, la función JASPA se ejecuta por todas y cada una de las combinaciones de elementos de las capas que interviene, haya o no intersección.

Sin embargo, si se utiliza la función creada como filtro, no se recibe ningún resultado *null*, y la función JASPA no se ejecutará ninguna vez para casos en los que no hay intersección, con lo que reduce el tiempo empleado en encontrar esta situación, como se puede comprobar en los datos ofrecidos en la tabla 4.

Tabla 4: Comparativa de resultados en uso índices espaciales integrados con funciones JASPA y sin ellos

Capa 1	Capa 2	Nº filas devueltas/Tiempo	Nº filas devueltas /Tiempo

		sin Filtro	con Filtro
USES	SOILS	3268 / 1140 ms	477 / 438 ms
USES	RIVERS	8056 / 1516 ms	440 / 531 ms
SOILS	RIVERS	4558 / 1141 ms	383 / 484 ms
SOILS	PAISES	+ de 10.000 / + de 45000 ms	0 / 10282 ms
RIVERS	PAISES	+ de 10.000 / + de 42906 ms	0 / 24344 ms
USES	PAISES	+ de 10.000 / + de 45109 ms	0 /17703 ms

CONCLUSIONES

Las principales conclusiones del presente trabajo son:

- Se ha realizado la integración del índice espacial con la extensión JASPA. Se han incluido nuevas funciones y operadores en JASPA que hacen uso del índice espacial. Concretamente, se ha incorporado el operador _ST_INTERSECTS y las funciones ST_INTERSECT_SPIDX y ST_CONTAIN_SPIDX.

- Se ha implementado la persistencia para el algoritmo de ArbolR (RTree). Para ello se ha permitido que la representación del índice espacial en memoria RAM, que se realiza en tablas de la base de datos de forma persistente, se mantenga actualizada y a disposición de las funciones que necesiten acceder a ellas en cualquier momento.
- Se ha eliminado la necesidad de que el usuario tenga que actuar de forma directa para activar los índices espaciales, haciendo que sea el propio sistema H2 el que se encargue de crear y mantener dichos índices.

- Se ha mejorado la integración con las tablas de metadatos definidas en el estándar Simple Feature Access: SQL (OGC, 2006). La información almacenada en las tablas de metadatos de JASPA es utilizada durante la creación del índice.

- Se ha adaptado la versión anterior del índice a JASPA 0.2.

En cuanto a las adaptaciones necesarias, se podrían destacar las realizadas dentro del uso de los índices espaciales ya existentes en trabajos anteriores para la mejora de rendimiento de las funciones JASPA, la automatización de la creación y mantenimiento de los índices espaciales, de forma que el usuario no tenga que preocuparse de ese tipo de tareas o la ampliación para el soporte en el sistema de más de un índice espacial en el mismo momento.

Para ello, se ha tenido que modificar el uso de las tablas de metadatos de JASPA en su versión 0.2 para el apoyo en la gestión de los índices espaciales, además de la ampliación de las tablas auxiliares que permiten tener los índices espaciales no solo en memoria sino también en tablas de la base de datos.

También ha sido necesario modificar la cadena de conexión a la base de datos H2 para que permita el inicio de la gestión automática de los índices espaciales, sin intervención directa del usuario.

REFERENCIAS

[1] Martínez, J.C. y González, M. (2011) “JASPA – Java Spatial for PostgreSQL and H2” Universidad Politécnica de Valencia. Disponible en: (Última consulta: 11 de marzo de 2012

[2] Calvillo Ardila, J.A. (2011) “Implementación de una extensión espacial para la extensión JASPA sobre H2” Proyecto Final de Carrera. Universidad Oberta de Catalunya

[3] Calvillo J.A.; de Diego; Pérez-Navarro, A. (2011) “Desarrollo de un índice espacial para la extensión JASPA sobre la base de datos H2” V Jornadas SIG libre. Disponible en: http://www.sigte.udg.edu/jornadassiglibre2011/uploads/articulos/art3.pdf (Última consulta: 11 de marzo de 2012)

[4] VV.AA. (2011) H2 Database Engine. Disponible en: www.h2database.com. (Última consulta: 11 de marzo de 2012)

[5] Morris, Aled (2011) “JSI (Java Spatial Index) RTree Library ”Disponible en: http://jsi.sourceforge.net/ (Última consulta: 11 de marzo de 2012)

[6] Pérez-Navarro, A. (coord.) et al. (2011) “Introducción a los sistemas de información geográfica y geotelemática” Barcelona. Ediuoc

[7] OGC (2006) OpenGIS® Implementation Specification for Geographic information - Simple feature access - Part 2: SQL option. Open Geospatial Consortium Inc.