*Serie Científica de la Universidad de las Ciencias Informáticas*
Vol. 9, No. 2, Febrero, 2016
ISSN: 2306-2495 | RNPS: 2343
http://publicaciones.uci.cu
Pág. 41-55

Tipo de artículo: Artículo original
Temática: Software libre
Recibido: 29/01/2016 | Aceptado: 20/02/2016

# Componente de conexión a MongoDB como sistema de base de datos no relacional para la generación de reportes en el Generador Dinámico de Reportes 2.0

# *Connection component database system MongoDB as non-relational data to generate reports Dynamic Report Generator 2.0*

**Ibis Brito Amaya [1*], Gustavo Crespo Sánchez [2]**

[1] Dpto. de Soluciones Informáticas para Internet, Facultad 1, Universidad de las Ciencias Informáticas, Cuba. ibrito@uci.cu
[2] Dpto. de Desarrollo Componentes, Facultad 4, Universidad de las Ciencias Informáticas, Cuba. gcrespo@uci.cu

* Autor para correspondencia: ibrito@uci.cu

**Resumen**

En el Centro de Tecnologías de Gestión de Datos (DATEC) se encuentra el proyecto Generador Dinámico de Reportes 2.0, que permite generar reportes de forma dinámica partiendo de datos persistentes en los orígenes de datos soportados por el sistema. Cuenta con un conjunto de módulos que brindan las funcionalidades para dar soporte al ciclo de vida de los reportes. El módulo Diseñador de Modelos es el encargado de realizar las conexiones con los sistemas gestores de base de datos relacionales PostgreSQL, MySQL y SQLServer. La presente investigación está enmarcada en el desarrollo de una nueva conexión del GDR con MongoDB como sistema gestor de bases de datos no relacional, sustentado en los beneficios que reportaría contar con este gestor de base de datos a la aplicación y en la gran aceptación que han adquirido a nivel mundial. Para el desarrollo de la presente investigación se utilizaron un conjunto de herramientas y tecnologías que permitieron diseñar, implementar y validar la solución siguiendo las pautas que propone la metodología OpenUP para construir un software. Como producto final se obtuvo un componente de conexión a MongoDB para el GDR.

**Palabras clave:** Generador Dinámico de Reportes, orígenes de datos, Sistemas Gestores de Bases de Datos, MongoDB.

*Serie Científica de la Universidad de las Ciencias Informáticas*
Vol. 9, No. 2, Febrero, 2016
ISSN: 2306-2495 | RNPS: 2343
http://publicaciones.uci.cu
Pág. 41-55

***Abstract***

*In the center of data management technologies (DATEC) is the GDR 2.0 project, which allows to generate reports dynamically based on persistent data on the origins of data supported by the system. It has a set of modules that provide the capabilities to support the life cycle of reports. Designer models module is what allows to make connections with the relational database managers, PostgreSQL, MySQL and SQLServer. This research is framed in the development of a new connection of the GDR with MongoDB as non-relational database system, based on the benefits of these database managers and the great acceptance that have acquired around the world. A set of tools and technologies which allowed design and implement the component classes and followed the guidelines proposed by the OpenUP methodology for building a software were used for the development of this research. As a final product was a component of MongoDB to the DDR connection. As final product a connection component was obtained MongoDB for GDR, offering the possibility to negotiate origins of data doesn't relate them.*

***Keywords:*** *Generator Dynamic Reporting, data sources, Management Systems databases, MongoDB.*

## Introducción

El desarrollo actual ha aumentado considerablemente a partir de la constante evolución y utilización de las Tecnologías de la Información y las Comunicaciones (TICs) a nivel mundial. Desde este punto de vista la creación de la Universidad de las Ciencias Informáticas en el año 2002 fue un paso de avance para la nación cubana, convirtiéndose esta en una sólida base de apoyo a la informatización de la sociedad con su modelo de formación estudio-trabajo.

La universidad cuenta con varios centros productivos, entre los que se encuentra el Centro de Tecnologías de Gestión de Datos (DATEC). Como componente fundamental de este centro, se tiene el Generador Dinámico de Reportes (GDR), el cual es una aplicación web compuesta por seis módulos que garantizan el ciclo de desarrollo de los reportes que genera en distintos formatos; con gran variedad de opciones en su diseño y de forma rápida. Uno de ellos es el diseñador de modelos que gestiona orígenes de datos a través de los sistemas gestores de bases de datos relacionales PostgreSQL, SQL Server y MySQL a partir de consultas que se les realizan para diseñar los modelos que serán utilizados posteriormente en la confección de los reportes.

En la actualidad existen herramientas para la generación de reportes que brindan soporte a sistemas gestores de bases de datos no relacionales con el fin común de optimizar sus funciones y aumentar su competitividad en el mercado. Por ello, el GDR necesita gestionar orígenes de datos mediante algún sistema de base de datos no relacional para incrementar la oportunidad de información de la aplicación y evitar la pérdida de oportunidades de negocio con

algunas de las instituciones que lo utilizan debido al gran auge y aceptación que han tenido los sistemas de base de datos no relacionales a nivel mundial.

## Materiales y métodos

Durante la primera fase del estudio se utilizaron los siguientes métodos de investigación. Los métodos teóricos utilizados fueron:

- Histórico-lógico: para investigar las características y funcionalidades de los sistemas gestores de bases de datos no relacionales, profundizando en los sistemas orientados a documentos.
- Analítico-sintético: para analizar las principales potencialidades que brindan los sistemas de base de datos no relacionales a los sistemas generadores de reportes que los utilizan.
- Modelación: para elaborar los diagramas necesarios con el fin de guiar el proceso de implementación del componente de conexión a realizar.

Los métodos empíricos utilizados en la investigación fueron:

- Análisis de documentos: para recopilar información referente al funcionamiento del Generador Dinámico de Reportes buscando definir los requerimientos del componente de conexión a desarrollar.
- Observación: para determinar la forma en que el Generador Dinámico de Reportes se conecta a los sistemas gestores de bases de datos que utiliza y para comprobar el correcto funcionamiento del sistema.

## Resultados y discusión

### Sistemas gestores de bases de datos no relacionales

Los sistemas gestores de bases de datos no relacionales son una amplia clase de sistemas de gestión de bases de datos que a diferencia del modelo relacional no usan SQL como principal lenguaje de consultas, los datos almacenados no requieren estructuras fijas como tablas y normalmente no soportan operaciones JOIN. Las bases de datos no relacionales están altamente optimizadas para las operaciones recuperar y agregar, y mayormente son utilizadas en el almacenamiento de registros de información. Existen varias aproximaciones para clasificar las bases de datos no relacionales tales como: orientadas a columnas, de clave-valor y orientadas a documentos. (Leavitt, Neal, 2011)

### Clasificaciones de los sistemas gestores de bases de datos no relacionales

#### De clave-valor

Estos son los sistemas gestores de bases de datos no relacionales más simples en cuanto a su uso (la implementación puede ser muy complicada), ya que simplemente almacenan valores identificados por una clave. Su limitante está en que su estructura no sigue un modelo de datos, todo lo que almacenan es un valor binario. Son extremadamente rápidos, pero no permiten consultas complejas más allá de buscar por su clave. (QuantumMode, 2014) No posee un lenguaje de consulta estándar o herramientas de consulta (Consultores, 2013), por lo que no garantiza la consistencia en los resultados de las consultas.

A continuación, se mencionan algunos de los principales sistemas de base de datos de clave-valor:

- DynamoDB
- Redis

**Orientadas a columnas:**

Los sistemas gestores de bases de datos orientados a columnas almacenan los datos por columnas en lugar de filas para realizar consultas y agregaciones sobre grandes cantidades de datos; es decir, funcionan de forma similar a los sistemas gestores de bases de datos relacionales. Su principal desventaja reside en que requieren que cada tabla que utilizan se almacene en archivo o memoria, y para ello necesita datos estructurados y esquemas relacionales, heredando muchos de los inconvenientes que estos poseen, incluyendo el esfuerzo y el tiempo de procesamiento del sistema necesario para mantener las tablas canónicas. (Consultores, 2013)

A continuación, se mencionan algunos de los principales sistemas gestores de bases de datos orientados a columnas:

- Cassandra
- HBase

**Orientados a documentos:**

Son aquellos sistemas gestores de bases de datos que gestionan datos semi-estructurados. Son en esencia un almacén clave-valor con la excepción de que el valor no se almacena exclusivamente como un campo binario, sino con un formato definido de forma tal que el servidor pueda entenderlo. Esto no significa que siempre sigan el mismo esquema, sino que sólo se tienen dos campos donde uno de ellos es binario y puede ser entendido por la base de datos. Dicho formato puede ser Notación de Objetos de JavaScript (JSON, por sus siglas en inglés), Binary JSON (BSON, por sus siglas en inglés), Lenguaje de Etiquetado eXtensible (XML, por sus siglas en inglés) o cualquier otro formato estándar. De hecho, varias de sus implementaciones permiten ejecutar consultas muy avanzadas sobre los datos, e incluso establecer relaciones entre ellos sin permitir operaciones JOIN. Por lo que ofrecen buen rendimiento y escalabilidad sin perder del todo los beneficios del modelo relacional. (Sitio Oficial del proyecto MongoDB, 2012)

A continuación, se mencionan algunos de los principales sistemas de base de datos orientados a documentos:

- CouchDB
- Google BigTable
- MongoDB

A partir del estudio realizado se concluyó que los SGBD no relacionales de clave-valor al igual que los orientados a columnas presentan limitaciones que podrían atentar contra el rendimiento y eficiencia de la solución, por lo que se desechó la posibilidad de seleccionar alguno de ellos en la solución. En cambio, los SGBD orientados a documentos por sus propias características pudieran brindar grandes beneficios en cuanto a rendimiento y escalabilidad; por tal motivo la investigación centró su atención en los gestores de bases de datos documentales.

Destaca entre estos, MongoDB debido a que posibilita el almacenamiento orientado a documentos con esquemas dinámicos, empleando el formato JSON que ofrecen simplicidad y poder a las aplicaciones. Brinda soporte completo de indexado sobre cualquier atributo al igual que la replicación proporcionando una alta disponibilidad. (Vázquez Ortíz, MsC. Yudisney; Sotolongo León, MsC. Anthony Rafael, 2014) Permite realizar una amplia variedad de consultas independientemente de si se trata de búsquedas por campos, rangos o expresiones regulares. Además de ejecutar consultas Map/Reduce [1]y permitir agregaciones flexibles que fortalecen el procesamiento de los datos; así como el almacenamiento GridFS[2] para archivos de cualquier tamaño. (Karl Seguin, 2011)

A continuación, se muestra un gráfico que representa las tendencias de búsqueda en *Google* para las entradas MongoDB y CouchDB a partir del año 2005 hasta la actualidad.

1 **Map/Reduce:** es una funcionalidad de consulta muy potente; se utiliza para desarrollar operaciones de agregación complejas y agrupaciones.

[2] **GridFS:** Sistema de almacenamiento de MongoDB que permite dividir archivos. Está incluido en los *drivers* de MongoDB y disponible para los lenguajes de programación que soporta.

*Serie Científica de la Universidad de las Ciencias Informáticas*
Vol. 9, No. 2, Febrero, 2016
ISSN: 2306-2495 | RNPS: 2343
http://publicaciones.uci.cu
Pág. 41-55


Figura 1. Tendencias de búsqueda en Google para MongoDB y CouchDB.

Los números del gráfico reflejan la cantidad de búsquedas que se han realizado acerca de ambos términos en relación con el número de búsquedas totales realizadas en *Google* a lo largo del tiempo. Una tendencia descendente no significa que haya descendido el número de búsquedas absoluto o total de ese término sino que ha disminuido su popularidad, como es el caso de CouchDB en la Figura 1.

En cambio, MongoDB presenta una tendencia ascendente. *DB-Engines* clasifica mensualmente en un *ranking* los sistemas de gestión de base de datos de acuerdo a su popularidad. Al realizar el estudio se ubicaba en la 4ta posición de 277 y en la 1ra de las de su tipo seguida por Cassandra en la 8va posición. (DB-Engines, 2014) Dado el auge que ha tomado este SGBD han surgido alternativas de desarrollo y cada vez son más los sistemas que lo requieren; como es el caso de MEAN.io [3] y MeteorJS[4].

[3] **Stack MEAN:** es una solución JavaScript utilizado para construir aplicaciones web rápidas, robustas y mantenibles usando MongoDB, Express, AngularJS y Node.js.

[4] **MeteorJS:** es una nueva plataforma con la que cualquier desarrollador va a poder escribir sus propias aplicaciones web

A partir del estudio anteriormente descrito se concluyó que el sistema de gestión documental MongoDB es el sistema de bases de datos no relacional indicado para incorporarle al Generador Dinámico de Reportes 2.0 con el fin de consultar orígenes de datos MongoDB a través de él y finalmente diseñar reportes con la información seleccionada de la fuente de datos de este tipo; debido a que admite la replicación de datos utilizando múltiples servidores para el almacenamiento de archivos, es escalable, permite realizar consultas complejas a las bases de datos, es de código abierto, posee una comunidad de desarrollo muy activa dada su popularidad, además de que la herramienta iReport 5.2 soporta MongoDB como único sistema de base de datos no relacional orientado a documentos, elemento que lo hace compatible con las tecnologías utilizadas en el Generador Dinámico de Reportes actualmente.

## Generador Dinámico de Reportes 2.0

El GDR 2.0 es una aplicación web que provee una forma transparente al usuario para realizar consultas a bases de datos y obtener información de ellas en forma de reporte. Está desarrollado sobre el *framework* Symfony 2.0 para facilitar la gestión de la información. Presenta un área de trabajo adecuada para el diseño de los reportes. Soporta varios orígenes de datos relacionales, proporcionando la generación de reportes en los formatos PDF, HTML, XLS, CSV, XLSX, PPTX, ODS, ODT, RTF, TXT, XML, XHTML y DOCX de forma rápida y segura. Para la protección y auditoría de los recursos de la aplicación le han sido asegurados los principios de seguridad y el trabajo con las listas de control de acceso. La aplicación incluye potenciales avances en sus interfaces ya que emplea el novedoso *framework* Lycan en su diseño basado en ExtJS, brindando una serie de comportamientos y apariencia que favorecen la interacción con el sistema.

## Arquitectura de la solución

El GDR 2.0 está diseñado con una arquitectura modular y distribuida que ayuda a obtener tanto escalabilidad como flexibilidad. Los procesos se distribuyen entre varios componentes que se pueden ampliar e integrar con soluciones personalizadas. Una típica aplicación para la generación de informes atraviesa por las tres etapas del ciclo de vida de los reportes: creación, administración y entrega; el GDR ofrece las herramientas necesarias para llevar a cabo estos procesos.

Para soportar el ciclo de creación de los reportes cuenta con seis módulos. El ciclo se inicia con el diseñador de modelos, que es donde se realiza la conexión a uno de los gestores de base de datos soportados por el sistema, para posteriormente diseñar los modelos semánticos que contienen la información en forma de metadatos y los que serán utilizados en la confección de los reportes.

En el diseñador de consultas se encuentra el área donde se diseñan las consultas; para ello la aplicación cuenta con las vistas básicas, la vista de diseño y la vista SQL; en la vista de diseño se elabora la consulta de forma gráfica mostrando los documentos, sus campos y las relaciones entre ellos; mientras que en la vista SQL se muestra el código perteneciente a la consulta diseñada.

Con el fin de obtener información de los sistemas de gestión a través de utilizar plantillas como vía más rápida para diseñar y gestionar los reportes, se define la estructura interna de cada uno posibilitando entre otras funciones, su creación, exportación e importación en el diseñador de reportes al igual que la visualización de los ya existentes.

El administrador de reportes se encarga de administrar las suscripciones y reportes por categorías, logrando la organización y actualización del estado de cada uno mediante el envío de correos electrónicos dado un tiempo establecido por el usuario.

El visor de reportes permite realizar varias funcionalidades como la visualización y exportación del reporte a diferentes formatos. El Generador Dinámico de Reportes 2.0 implementa el proceso de entrega en el diseñador de reportes y en el visor de reportes, pues ambos módulos posibilitan la visualización y exportación del informe a diferentes formatos.

## Solución

El GDR actualmente brinda soporte para diferentes tipos de base de datos relacionales, está diseñado de modo que el usuario sin poseer grandes conocimientos acerca de las particularidades del tipo de base de datos que esté utilizando, puede interactuar con esta a través del sistema. El uso de una base de datos u otra no difiere mucho pues el sistema aprovecha mecanismos comunes existentes en estos tipos de bases de datos relacionales. Al interactuar con una base de datos no relacional como MongoDB la forma de gestionar orígenes de este tipo es totalmente diferente a como se realiza el mismo proceso con una base de datos relacional, tanto en la forma de conectarse de forma remota a un origen de datos de este tipo como en el diseño de las consultas. La nueva forma de interactuar con una base de datos de este tipo implicaría grandes modificaciones a los módulos metadatos (gestiona orígenes de datos) y diseñador de consultas (diseña las consultas para el reporte).

El módulo Metadatos se utiliza para gestionar los orígenes relacionales de forma remota mediante PDO[5]; que no soporta base de datos no relacionales (como MongoDB). En su lugar se utiliza MongoClient[6], lo cual hace posible el

5 **PDO:** (por sus siglas en inglés, *PHP Data Objects*) es una extensión que provee una capa de abstracción de acceso a datos para PHP5 para SGBD relacionales.

6 **MongoClient:** es el driver de PHP5 para MongoDB; se utiliza para administrar las conexiones entre ambos.

*Serie Científica de la Universidad de las Ciencias Informáticas*
Vol. 9, No. 2, Febrero, 2016
ISSN: 2306-2495 | RNPS: 2343
http://publicaciones.uci.cu
Pág. 41-55

acceso a los orígenes de datos MongoDB ya que está dado por una clase de abstracción cuyos métodos son redefinidos por los diferentes orígenes como se muestra en la Figura 2.

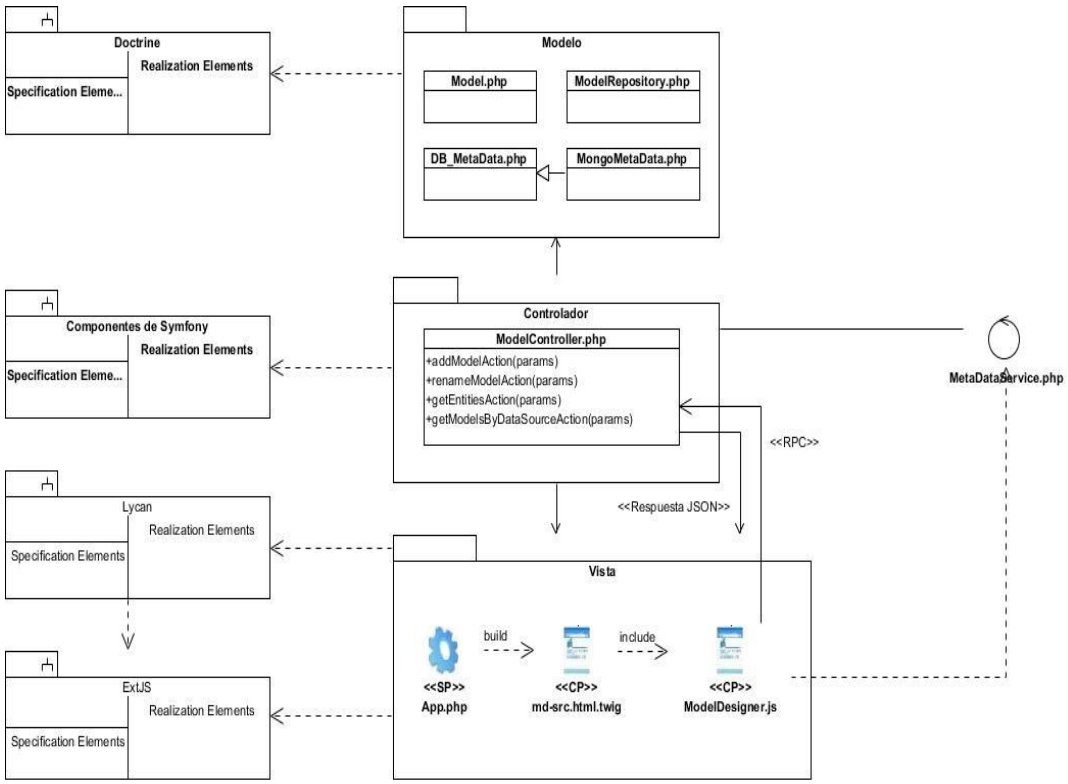


Figura 2. Diagrama de clases del diseño para el caso de uso Diseñar modelo.

Las clases *BDMetadata.php* y *MongoMetaData.php* que contienen las acciones necesarias para realizar la conexión a la base de datos MongoDB al igual que todas las funcionalidades que permiten obtener la información de la misma para el correcto funcionamiento del sistema; permitiendo manejar orígenes de datos MongoDB e integrar las funciones del componente con la lógica de negocio establecida por el GDR.

*Serie Científica de la Universidad de las Ciencias Informáticas*
Vol. 9, No. 2, Febrero, 2016
ISSN: 2306-2495 | RNPS: 2343
http://publicaciones.uci.cu
Pág. 41-55

```php
abstract class DBMetaData {
    public function createDSN($dbms, $host, $port, $db) {
        $dsn = "$dbms:host=$host;port=$port;dbname=$db";
        return $dsn;
    }
    public function configureConnection() {
        $dbms = $this->getDbms();
        $host = $this->dataSource->getHost();
        $port = $this->dataSource->getPort();
        $user = $this->dataSource->getUsername();
        $db = $this->dataSource->getDatabase();
        $pass = $this->dataSource->getPassword();
        $dsn = $this->createDSN($dbms, $host, $port, $db);
        if (!$dsn) {
            throw new \Exception('Error');
        } else {
            return new \PDO($dsn, $user, $pass);
        }
    }
```

```php
class MongoMetaData extends DBMetaData {
    public function createDSN($dbms, $host, $port, $db) {
        $userPass = '';
        if ($this->getDataSource()->getUsername() != '') {
            $user = $this->getDataSource()->getUsername();
            $pass = $this->getDataSource()->getPassword();
            $userPass = "$user:$pass@";
        }
        return "mongodb://$userPass$host:$port";
    }
    public function configureConnection() {
        $dbms = $this->getDbms();
        $host = $this->getDataSource()->getHost();
        $port = $this->getDataSource()->getPort();
        $user = $this->getDataSource()->getUsername();
        $db = $this->getDataSource()->getDatabase();
        $pass = $this->getDataSource()->getPassword();
        $dsn = $this->createDSN($dbms, $host, $port, $db);
        if (!$dsn) {
            throw new \Exception('Error');
        } else {
            return new \MongoClient($dsn);
        }
```

Figura 3. Reimplementación del método *configureConnection()*.

En la Figura 3 se muestra la reimplementación del método *configureConnection()*. A la izquierda, para el caso de los orígenes de base de datos relacionales mientras que, a la derecha, aparece cómo se realiza para MongoDB.

```
INSERT INTO "nombre_tabla" ("columna1", "columna2", ...)
VALUES ("valor1", "valor2", ...);
```

Figura 4. Consulta SQL de la operación Insertar.

```
db.coleccion.insert(documento)
```

Figura 5. Consulta realizada utilizando MongoDB para la operación Insertar.

El módulo diseñador de consultas es el encargado de realizar las consultas, cuyos resultados serán mostrados en los reportes diseñados, el principal inconveniente de este módulo al trabajar con una base de datos MongoDB es que la forma de realizar las consultas sobre este tipo de base de datos difiere en gran medida como se muestra en la Figura 4 y la Figura 5.

Investigaciones posteriores sobre este tipo de base de datos determinaron que esta es la base de datos no relacional más enfocada a lo que se conoce como SQL; ello es se debe al JDBC[7] para MongoDB. A la hora de integrar este

[7] **JDBC:** (por sus siglas en inglés, *Java Database Connectivity)*, es una API que permite la ejecución de operaciones sobre bases de datos desde el lenguaje de programación Java.

*Serie Científica de la Universidad de las Ciencias Informáticas*
Vol. 9, No. 2, Febrero, 2016
ISSN: 2306-2495 | RNPS: 2343
http://publicaciones.uci.cu
Pág. 41-55

último a la solución para gestionar orígenes MongoDB realizando consultas en formato SQL se hicieron pequeños ajustes en la redefinición del método *query()* como se muestra en la Figura 6. Esto posibilitó no tener que efectuar grandes cambios en la implementación del diseñador de consultas.

```php
abstract class DBMetaData {
    public function query($query, $con) {
        $stament = $con->prepare($query);
        if (!$stament->execute()) {
            $this->setErrorInfo(TRUE);
            return $stament->errorInfo();
        }
        return $con->query($query);
    }
```

```php
class MongoMetaData extends DBMetaData {
    public function query($query, $con) {
        if (is_array($query)) {
            return $query;
        }
        if ($this->sql_query($query)) {
            try {
                return $this->sql_to_mongo_query($query);
            } catch (\Exception $exc) {
                $this->setErrorInfo(TRUE);
                return $exc->getTraceAsString();
            }
        }
        return $con->$query();
    }
```

Figura 6. Reimplementación del método *query()* en la clase *MongoMetadata.php*.

Como se muestra en la reimplementación del método *query()* en la clase *MongoMetadata.php*, cuando la consulta tiene formato SQL se obtienen entonces los datos del origen mediante un servicio web utilizando SOAP[8] que permite ejecutar consultas de este tipo sobre base de datos MongoDB, este servicio web constituye una aplicación java web que funciona de forma independiente y esta tiene incorporadas las librerías necesarias para acometer dicha tarea como lo es en este caso el JDBC de MongoDB.

## Imágenes de la solución

[8] **SOAP:** (por sus siglas en inglés, *Simple Object Access Protocol*) es un protocolo estándar que define cómo dos objetos en diferentes procesos pueden comunicarse por medio de intercambio de datos XML.

*Serie Científica de la Universidad de las Ciencias Informáticas*
Vol. 9, No. 2, Febrero, 2016
ISSN: 2306-2495 | RNPS: 2343
http://publicaciones.uci.cu
Pág. 41-55


Figura 7. Selección del SGBD MongoDB.

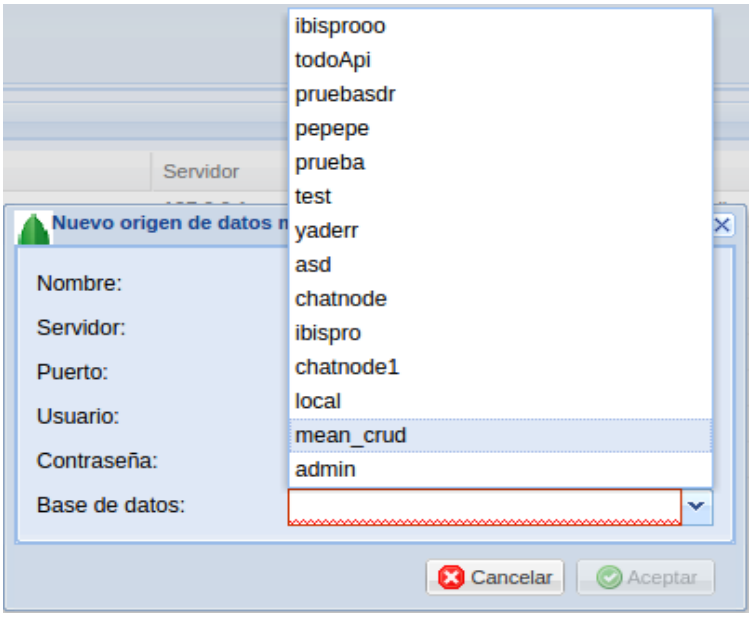


Figura 8. Lista de las bases de datos del SGBD MongoDB disponibles.

## Marco de trabajo

### Symfony 2.0

Symfony 2.0 es un marco de trabajo de código abierto, rápido, flexible y fácil de aprender que permite a los desarrolladores construir aplicaciones web mantenibles. Ha sido desarrollado teniendo en cuenta el rendimiento como

mayor prioridad. Se construye a base de *bundles* que permiten configurar y personalizar el sistema de una forma limpia. (Sitio Oficial de Symfony, 2007) Está construido utilizando un contenedor de inyección de dependencias. Todos los detalles de implementación los mantiene ocultos detrás de un buen sistema de configuración que permite personalizar todo a través de archivos .yml o .xml o a través de código PHP.

**ExtJS 3.4**

ExtJS es un marco de trabajo escrito en JavaScript con la finalidad de asistir el desarrollo de Aplicaciones Enriquecidas para Internet (RIA, por sus siglas en inglés). Se basa en componentes soportados por recursos para la programación orientada a objetos en JavaScript.

**Lycan**

Lycan es una librería de ExtJS desarrollada por DATEC. La misma surge por la necesidad de diseñar componentes de ExtJS de manera intuitiva, rápida y cómoda, promoviendo la usabilidad y elevando la productividad de los desarrolladores. Esta librería fue pensada para contribuir también con otros proyectos externos al centro que también trabajaran con ExtJS en la UCI colaborando así con el éxito de los mismos. Además, implementa buenas prácticas de arquitectura y diseño contribuyendo a la calidad del desarrollo de las aplicaciones web y se ha optimizado para su ejecución en Mozilla Firefox.

**ORM Doctrine**

Doctrine es un potente y completo sistema de Mapeo Objeto-Relacional (ORM, por sus siglas en inglés) para PHP 5.2 con una Capa de Abstracción de Bases de Datos (DBAL, por sus siglas en inglés) incorporada. Brinda la posibilidad de exportar una base de datos a las clases PHP correspondientes y también realizar el proceso inverso. Doctrine es un ORM para PHP 5.2.3 y posterior. Cuenta con su Lenguaje de Consultas de Doctrine (DQL, por sus siglas en inglés) siendo esta una de sus principales ventajas.

## Ambiente de desarrollo

**Metodología de desarrollo OpenUP**

La metodología utilizada para desarrollar el componente de conexión a MongoDB para el Generador Dinámico de Reportes 2.0 es OpenUP por decisión del proyecto. La misma se define como un proceso de desarrollo de software unificado dentro de un ciclo de vida estructurado basado en Proceso Racional Unificado (RUP, por sus siglas en inglés). Es dirigida por casos de uso y centrada en la arquitectura, iterativa e incremental.

**Lenguaje de modelado UML 2.0**

El Lenguaje Unificado de Modelado (UML, por sus siglas en inglés) es el lenguaje estándar de propósito general más utilizado para especificar y documentar cualquier sistema de forma precisa. Proporciona una gran flexibilidad y

expresividad a la hora de modelar sistemas. UML es una consolidación de muchas de las notaciones y conceptos más usados orientados a objetos.

**Herramienta de modelado Visual Paradigm 6.1**

Visual Paradigm permite diseñar diagramas de clases, código inverso, generar código desde diagramas y generar documentación. Visual Paradigm soporta el ciclo de vida completo del desarrollo de software: análisis y diseño orientados a objetos, construcción, pruebas y despliegue. El UML facilita la construcción de aplicaciones de calidad con un costo relativamente pequeño. Es fácil de usar, soporta ingeniería inversa y exportación/importación XML.

**Java 6.1**

Java es un lenguaje de programación de propósito general, concurrente y orientado a objetos que fue diseñado específicamente para tener tan pocas dependencias de implementación como fuera posible. (CODEHERO, 2013) De modo que al ejecutar el código en una plataforma no tiene que ser recompilado para correr en otra.

**JavaScript 1.8.5**

JavaScript es un lenguaje interpretado que se utiliza principalmente en su forma del lado del cliente. Todos los navegadores modernos interpretan el código JavaScript integrado en las páginas web. (Universidad de Sevilla, 2014) Es un lenguaje basado en objetos y es además orientado a eventos. Esto implica que gran parte de la programación en JavaScript se centra en describir objetos y escribir funciones que respondan a movimientos del ratón, pulsación de teclas, apertura y cerrado de ventanas o carga de una página, entre otros eventos.

**PHP 5.3.10**

Es un lenguaje *script* de código abierto que posee un intérprete que lee el código y se encarga de ejecutar las instrucciones que contiene para el desarrollo de páginas web dinámicas del lado del servidor. PHP incluye muchas ventajas entre las que se encuentran las siguientes:

- Integración de base de datos: PHP dispone de una conexión a diferentes tipos de servidores de bases de datos tanto SQL como NoSQL; tales como MySQL, PostgreSQL, Oracle, ODBC, DB2, Microsoft SQL Server, Firebird, SQLite, MongoDB, Hyperwave, Informix, InterBase y Sybase, entre otras.

**Herramienta de desarrollo NetBeans 8.0**

NetBeans IDE es un entorno de desarrollo escrito en Java. Es un editor de código multilenguaje con sugerencias de código, acceso a clases ejecutándose en el código, control de versiones, localización de ubicación de la clase actual, herramientas de refactorización, comprobaciones sintácticas y semánticas. Es un producto libre y gratuito sin restricciones de uso.

## Conclusiones

Culminando el desarrollo de la solución se arriba a las siguientes conclusiones:

- Se le incorporó soporte al Generador Dinámico de Reportes para el SGBD no relacional MongoDB permitiendo a sus usuarios interactuar con el sistema sin tener conocimientos previos sobre este, pues la solución se comporta de igual forma que con los demás SGBD relacionales soportados anteriormente.
- Se implementó un servicio web que permite obtener la información resultante de realizar las consultas a bases de datos MongoDB en formato SQL aprovechando las potencialidades del JDBC utilizado.

## Referencias

CODEHERO. 2013. CODEHERO. [En línea] 12 de mayo de 2013. [Citado el: 18 de enero de 2016.] http://codehero.co/series/java-desde-cero.html.

Consultores, WebMining. 2013. WebMining Consultores. [En línea] 8 de julio de 2013. [Citado el: 2 de noviembre de 2015.] http://www.webmining.cl/2012/06/como-elegir-una-base-de-datos-para-su-empresa/.

DB-Engines. 2014. DB-Engines. [En línea] 17 de enero de 2014. [Citado el: 15 de febrero de 2016.] http://db-engines.com/en/ranking.

Karl Seguin. 2011. The_little_MongoDB_book. [En línea] 2011. [Citado el: 3 de noviembre de 2013.] http://creativecommons.org/licenses/by-nc/3.0/deed.es_ES.

Leavitt, Neal. 2011. Will NoSQL Databases Live Up to Their Promise. Cambridge : s.n, 2011.

QuantumMode. 2014. QuantumMode. [En línea] 27 de abril de 2014. [Citado el: 1 de junio de 2015.] http://quantummode.com/las-bases-de-datos-nosql-ii-el-modelo-de-datos/.

Sitio Oficial de Symfony. 2007. sf Symfony. [En línea] 25 de junio de 2007. [Citado el: 15 de noviembre de 2014.] http://symfony.com/what-is-symfony.

Sitio Oficial del proyecto MongoDB. 2012. MongoBD FOR Giant IDEAS. [En línea] 6 de febrero de 2012. [Citado el: 12 de noviembre de 2014.] https://docs.mongodb.org/manual/.

Universidad de Sevilla. 2014. Dpto.Ciencias de la Computación e Inteligencia Artificial de la Universidad de Sevilla. [En línea] 12 de noviembre de 2014. [Citado el: 9 de diciembre de 2015.] http://www.cs.us.es/cursos/i1e-2014/temas/tema-01HTML.pdf.

Vázquez Ortíz, MsC. Yudisney; Sotolongo León, MsC. Anthony Rafael. 2014. Mirada a bases de datos NoSQL de código abierto orientadas a documentos. La Habana : s.n., 2014. s.n.

*Serie Científica de la Universidad de las Ciencias Informáticas*
Vol. 9, No. 2, Febrero, 2016
ISSN: 2306-2495 | RNPS: 2343
http://publicaciones.uci.cu
Pág. 41-55