**ESCUELA TÉCNICA SUPERIOR DE INGENIERÍA INFORMÁTICA**

**INGENIERÍA TÉCNICA EN INFORMÁTICA DE SISTEMAS**

**UN MEDIDOR DE RENDIMIENTO DE SERVIDORES DE BASES DE DATOS RELACIONALES**

**Realizado por**

**JOSE ANTONIO JAMILENA DAZA**

**Dirigido por**

**ANTONIO CÉSAR GÓMEZ LORA**

**Departamento**

**LENGUAJES Y CIENCIAS DE LA COMPUTACIÓN**

**UNIVERSIDAD DE MÁLAGA**

**MÁLAGA, (SEPTIEMBRE 2009)**

**UNIVERSIDAD DE MÁLAGA**

**ESCUELA TÉCNICA SUPERIOR DE INGENIERÍA INFORMÁTICA**

**INGENIERÍA TÉCNICA EN INFORMÁTICA DE SISTEMAS**

Reunido el tribunal examinador en el día de la fecha, constituido por:

Presidente/a Dº/Dª. \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Secretario/a Dº/Dª.\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Vocal Dº/Dª. \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

para juzgar el proyecto Fin de Carrera titulado:

**UN MEDIDOR DE RENDIMIENTO DE SERVIDORES DE BASES DE DATOS RELACIONALES.**

del alumno Dº. Jose Antonio Jamilena Daza

dirigido por Dº. Antonio César Gómez Lora

ACORDÓ POR \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

OTORGAR LA CALIFICACIÓN DE \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Y PARA QUE CONSTE, SE EXTIENDE FIRMADA POR LOS

COMPARECIENTES DEL TRIBUNAL, LA PRESENTE DILIGENCIA.

Málaga, a \_\_\_\_\_ de \_\_\_\_\_\_\_\_\_\_\_\_\_ del 200\_\_\_

|  |  |  |
| --- | --- | --- |
| El/La Presidente/a | El/La Secretario/a | El/La Vocal |
| Fdo: | Fdo: | Fdo: |

|  |
| --- |
|  |
| Un medidor de rendimiento de servidores de bases de datos relacionales |
| Realizado por Jose Antonio Jamilena Daza Dirigido por Antonio César Gómez Lora |

|  |
| --- |
|  |

Contenido

[1 Introducción 9](#_Toc235599085)

[1.1 Objetivos 10](#_Toc235599086)

[1.2 Metodologías y fases de trabajo 10](#_Toc235599087)

[1.3 Herramientas de desarrollo 11](#_Toc235599088)

[1.4 Calidad 13](#_Toc235599089)

[1.5 Capítulos 14](#_Toc235599090)

[2 Prospección en el mercado de productos que aborden el problema tratado 16](#_Toc235599091)

[3 Distintos tipos de soluciones para el mismo problema 23](#_Toc235599092)

[4 Herramientas para el desarrollo y la gestión del proyecto. 25](#_Toc235599093)

[5 Diseño de la solución 27](#_Toc235599094)

[6 Bibliografía 29](#_Toc235599095)

Tabla de ilustraciones

[Ilustración 1 18](#_Toc235598740)

[Ilustración 2 20](#_Toc235598741)

[Ilustración 3 21](#_Toc235598742)

[Ilustración 4 22](#_Toc235598743)

# Introducción

La mayoría de los sistemas gestores de bases de datos actuales disponen de herramientas propias o de terceros destinadas a medir el estado de salud y el rendimiento del propio sistema. Estos sistemas con capaces de proporcionar un extenso y valioso conjunto de información para administradores y auditores. Adicionalmente existen plataformas software destinadas a construir medidores de calidad de servicio, aunque, hoy por hoy, con una orientación muy específica hacia redes y servicios web.

Con este rico abanico de herramientas disponibles, y la necesidad de medir la calidad de los servicios prestados, es cada vez más común encontrar servidores dentro de una empresa u organización destinados a analizar la disponibilidad y rendimiento de los servicios ofertados (entendiendo el acceso a una base de datos como uno de estos servicios). Desgraciadamente estos servidores suelen estar situados estratégicamente en zonas clave de las intranets o de las infraestructuras de comunicación. Cuando los servicios están abiertos al exterior sólo se analiza el entorno próximo del servidor y nunca el entorno del cliente.

Para un usuario simple que accede desde su equipo a un servidor de bases de datos el rendimiento viene marcado por el rendimiento de todos y cada uno de los equipos, dispositivos y enlaces involucrados en el procesamiento y el transporte de la información. En una sesión común esto puede significar la participación de decenas, cientos e incluso miles de elementos hardware y software participantes, la mayoría de los cuales son transparentes al usuario. Y en caso de ser visible alguno de estos participantes generalmente no se tiene la capacidad ni los permisos suficientes para analizar su rendimiento individual.

Se nos plantea pues la necesidad de que un usuario pudiera ejecutar un sencillo programa que le permitiera medir el rendimiento de determinadas tareas en un servidor de bases de datos. Este sistema analizará el rendimiento de estas tareas de forma periódica siguiendo una planificación dada para cumplir su objetivo principal, que es detectar la aparición de posibles problemas. Aunque el diagnóstico del problema queda fuera de los objetivos de esta aplicación, sí es cierto que se añadirá la capacidad de clasificar las diferentes tareas según características particulares para agrupar sus resultados y así asistir al usuario en la interpretación de los datos y estadísticos para que se pueda realizar un diagnóstico. Este programa sencillo tiene un doble destino: primero el ser utilizado por usuarios aislados; y segundo el poder ser instalado en múltiples equipos y permitir recolectar posteriormente sus datos de forma sencilla (preferiblemente en forma de un archivo simple).

## Objetivos

Los objetivos que este proyecto plantea son los que siguen:

* Elaborar un software que ejecute sobre sistemas gestores de bases de datos relacionales, bajo una planificación dada, unos scripts de prueba, para a partir de los tiempos de ejecución generar indicativos sobre su comportamiento.
* Construir una interfaz gráfica que sea capaz de recoger los datos de la herramienta descrita anteriormente y generar unas estadísticas que sean mostradas de forma visual para que el usuario las interprete. Dichas estadísticas se agruparan según los diferentes tipos de scripts lanzados.
* Generar un conjunto base de scripts que puedan ser clasificados en conjuntos no disjuntos de categorías o propiedades. Con esto se consiguen estadísticas según las posibles características evaluar.

## Metodologías y fases de trabajo

En el desarrollo del proyecto se empleará el paradigma de programación orientado a objetos y para modelar el software el Lenguaje Unificado de Modelado o UML, iniciales que corresponden al acrónimo inglés *Unified Modeling Language*.

El modelo de desarrollo será el Desarrollo Iterativo Incremental ].

En cuanto a las fases en las que se dividirá el desarrollo software éstas serán:

* Una prospección inicial del desarrollo en el que barajaremos posibles lenguajes para desarrollar el proyecto, tecnologías a emplear, etc. Así también como la obtención de los objetivos que ha de cumplir el desarrollo software que formará el proyecto. Tras la cual habremos obtenido el análisis de requisitos.
* Se realizará la especificación de los distintos componentes que formarán parte del desarrollo software.
* Una vez que tenemos los componentes que debemos implementar pasaremos a una fase de diseño de la arquitectura de la solución. En esta fase del desarrollo se realizará el diseño de los distintos componentes del sistema software, así como el diseño de la jerarquía de clases de los distintos componentes del desarrollo software.
* Posteriormente pasaremos a la implementación del diseño. En esta fase se puede llegar a tener que hacer cambios en el diseño si la naturaleza del desarrollo así lo exige.
* Finalmente se realizarán las pruebas oportunas para verificar el correcto comportamiento del sistema.

Durante todo el proceso se seguirá el Desarrollo Iterativo Incremental ], se irán obteniendo prototipos de desarrollo sobre los que se irán haciendo pruebas de funcionamiento y de completitud de requisitos.

Durante todas estas fases se llevará un amplio proceso de documentación, tanto para la escritura de la memoria final como para guardar el conocimiento adquirido durante el desarrollo.

## Herramientas de desarrollo

El desarrollo se hará sobre el lenguaje de programación Java, concretamente haciendo uso de JDK 1.6.0\_14-b08 de Sun Microsystems, la última versión de este leguaje de programación orientado a objetos ].

El interfaz elegido para el desarrollo será Netbeans 6.5, un entorno integrado de desarrollo especializado en Java ].

Todo el proyecto se guardará con el sistema de control de versiones Subversion. Este se encarga de mantener históricos de ficheros durante los desarrollos software ]. Subversión se instala como un software servidor al que se accede mediante un cliente. El cliente elegido es TortoiseSVN, un cliente que se integra tanto en la shell del sistema operativo como en el propio Netbeans ].

Se realizará el diseño de la aplicación sobre Enterprise Architect y sobre un complemento específico de Netbeans para estos menesteres. Enterprise Architect es una herramienta CASE, Computer Aided Software Engineering o Ingeniería de Software Asistida por Ordenador, y en el diseño de este proyecto en cuestión se guardarán los datos de diseño en un sistema de bases de datos MySQL ]]. En cuanto al componente de Netbeans lo almacenará con estructura de archivos de proyecto de Netbeans.

El proyecto tendrá integrada su propia base de datos de estadísticas. Para dicho cometido se empleará SQLite, un sistema gestor de bases de datos empotrado, rápido y de poco peso en memoria ]. Para comunicarse con Java se empleará la librería SQLiteJDBC ]. Para facilitar el diseño en dicho sistema gestor de bases de datos se empleará la herramienta SQLite Management Studio, una herramienta para gestión gráfica de ficheros SQLite ].

Para velar por la integridad y conocer inequívocamente los fallos que pueden acaecer, tanto en la ejecución normal como en las sesiones de pruebas, se empleará Apache Log4J. Apache Log4J es un sistema de diario de ejecución para aplicaciones Java ]. Dicho sistema nos da la capacidad de generar ficheros de diario de ejecución con las trazas de la misma e incluso la capacidad de que llegado el caso, si el sistema al que estamos evaluando se llegara a colapsar, enviar un email advirtiendo este hecho.

Para el componente encargado de mostrar las estadísticas de las pruebas se empleara un interfaz gráfico Java implementado con Swing. Como sistema visualizador de datos y estadísticas emplearemos la biblioteca JFreeChart. JFreeChart es un sistema de generación de graficas para el entorno grafico Swing de Java ].

Las conexiones entre el proyecto y los diferentes gestores de bases de datos se harán mediante la tecnología JDBC (Java Data Base Connectivity). JDBC es una API que permite la ejecución de operaciones sobre bases de datos desde el lenguaje de programación Java, independientemente del sistema operativo donde se ejecute o de la base de datos a la cual se accede, utilizando el dialecto SQL del modelo de base de datos que se utilice.

Toda la planificación temporal del proyecto será llevada acabo con la ayuda de OpenProj ]. Y la documentación se generará a partir de Enterprise Architect y se editará el texto con OpenOffice ] y con Microsoft Visio ] los diagramas que fueran necesarios.

## Calidad

La calidad debería ser un principio fundamental en el desarrollo software, y en especial si el desarrollo lo estamos haciendo en un lenguaje como Java que dispone de multitud de estándares y herramientas de auditoría que nos permiten garantizar que el código implementado cumple todos los requisitos como para considerarlo estable y seguro.

Para la completitud de los propósitos de calidad el desarrollo se ha empleado herramientas de verificación de código estático y de chequeo de código.

Para la verificación de código estático se ha empleado PMD. PMD analiza el código fuente de Java y busca posibles problemas como:

* Posibles errores: sentencias try / catch / finally / switch vacías.
* Código de muertos: variables locales, parámetros y métodos privador sin usar.
* Optimiza código: controla el despilfarro de uso de String / StringBuffer.
* Detecta expresiones excesivamente complejas, declaraciones innecesaria o bucles sin condiciones de salida.
* Detecta código duplicado. El copiar / pegar en el código significa copiar / pegar los errores.

Para garantizar que la escritura de código es estándar y portable se emplea Checkstyle. Checkstyle es una herramienta de desarrollo para ayudar a los programadores escribir código Java que se adapte a una norma de codificación.

Automatiza el proceso de verificación y control de código Java ya que para los seres humanos de esto, a pesar de ser una tarea importante, resulta muy aburrido.

Esto hace que sea ideal para los proyectos que quieren hacer cumplir una norma de codificación.

Checkstyle es altamente configurable y se puede hacer para apoyar casi cualquier norma de codificación. Por defecto este proyecto final de carrera emplea los convenios de Sun para el desarrollo de código Java.

## Capítulos

La memoria consta de x capítulos en los que se abordan los siguientes temas en cada uno de ellos.

1. En el primer capítulo nos encontramos la introducción.
2. En el segundo capítulo analizamos el mercado de este tipo de soluciones y comparamos con la llevada a cabo en este proyecto final de carrera.
3. En el tercer capítulo, nos dedicamos a abordar los distintos tipos de soluciones por las que podía haber abordado el problema, plantearemos sus ventajas e inconvenientes, y nos decantaremos por una que es la que llevaremos a su culminación.
4. En el cuarto capítulo nos embarcamos en las distintas herramientas o bibliotecas para el desarrollo y la gestión del proyecto. Indagaremos sobre cuales vamos a emplear y el porqué.
5. En el quinto capítulo empezamos con el diseño de la solución.
6. Conclusiones.

# Prospección en el mercado de productos que aborden el problema tratado

El análisis y la métrica de rendimiento en los sistemas gestores de bases de datos relacionales es una práctica muy extendida. Actualmente todos los sistemas gestores de bases de datos, de una forma u otra, disponen de herramientas que ayudan tanto a su configuración como a la monitorización y supervisión de los mismos. Entonces, ¿por qué diseña otro medidor de rendimiento?

La explicación es bien sencilla, y la existencia de sistemas de métrica y supervisión de sistemas gestores de bases de datos de terceros nos dan la razón. Normalmente, las herramientas que se incluyen en con el propio sistema gestor de bases de datos están orientadas a analizar parámetros propios de rendimiento, de estado del servicio dentro del sistema operativo, de la apertura o no de los canales de comunicación por los que el sistema gestor de bases de datos se comunica con los clientes ya sean usuarios o software de terceros. Estas métricas son concienzudas y han sido realizadas por los propios creadores del gestor de bases de datos, y nadie mejor que ellos para analizar en modo teórico todo lo relacionado con dicho gestor.

Pero, ¿qué ocurre cuando por ejemplo, aquí en la escuela, en un examen de la asignatura de bases de datos trescientos alumnos se enfrentan a un examen en el que todos de modo concurrente tienen que lanzar consultas contra un determinado gestor de bases de datos y todo va mal? Sí se consultan las herramientas de supervisión y se encuentra algún problema, es fácil de solucionar. Pero ¿y si esto no ocurre? ¿De dónde viene el fallo? ¿Es fallo del sistema gestor de bases de datos y las herramientas de supervisión no lo reflejan o por lo menos de un modo claro? ¿Puede ser causado por la red? ¿Puede ser del sistema operativo del servidor que está actualizando algún componente, verbigracia el parseador de XML y el planificador de sistema operativo lo considera de más prioridad que el servicio del sistema gestor de bases de datos? ¿Puede ser un alumno rezagado que no ha estudiado las sentencias JOIN y ha creado sin saberlo un producto cartesiano inmenso que desborda la caché y los tiempos de CPU del gestor de bases de datos?

Con este proyecto final de carrera no se pretende hacer una herramienta que prevenga cualquier tipo de fallos y los solucione. Lo que se pretende desarrollar es un sistema muy simple para medir rendimientos, que se pueda ejecutar de una forma eficiente y casi imperceptible para el usuario y que nos dé una idea de que algo anormal está ocurriendo.

En el marcado ya existe herramientas con estas capacidades, otras que desarrollan funciones similares.

Transaction Processing Performance Council (TPC)

TCP es una organización sin fines de lucro fundada en 1988 para definir los proceso de transacciones y herramientas de métrica de base de datos además de difundir los resultado de la información de forma objetiva y verificable para la industria.

Los sistemas de métrica de TPC son ampliamente utilizados hoy en la evaluación del rendimiento de los sistemas, los resultados se publican en el sitio web de TPC.

Estos sistemas de métrica son tipo benchmark y están más orientado a la comparación de distintos sistemas gestores de bases de datos.

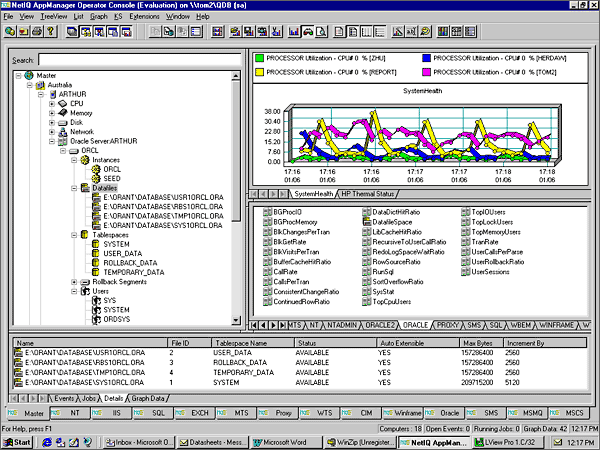
No los podemos incluir en el mismo saco que este proyecto, ya que los objetivos pretendidos son distintos, pero fue la primera toma de contacto con este tipo de soluciones. Además de disponer de múltiple documentación de cómo hacer pruebas de rendimiento para poder encontrar posibles fallos de rendimiento o de degradación en el comportamiento normal del sistema. También emplean metodologías de desarrollo de sus benchmark de forma altamente eficiente.

NetIQ AppManager for Oracle Database RDBMS Server

Dispone de un sistema de administración y supervisión de rendimiento. Proporciona herramientas de gestión de bases de datos Oracle optimizando su rendimiento y disponibilidad. Disponible en tiempo real de sistemas de diagnóstico y se almacenan en un repositorio de históricos.

Proporciona un sistema de notificación proactiva y un sistema automático que aplica medidas correctivas. Además de un sistema de aviso de todas las ocurrencias que se producen.

Orientado para gestores de bases de datos Oracle. A pesar de todo, también se pueden adquirir componentes para hacerlo trabajar con Microsoft SQL Server.



Ilustración

Es un software propietario. Y a diferencia de los anteriores no dispone de ningún conjunto de bibliotecas para el desarrollo de complementos.

Sólo se puede implantar en entornos Microsoft Windows.

Nagios

Nagios es un sistema de código abierto para la supervisión de redes. Nagios está muy extendido. Nagios se emplea para la supervisión de equipos y servicios.

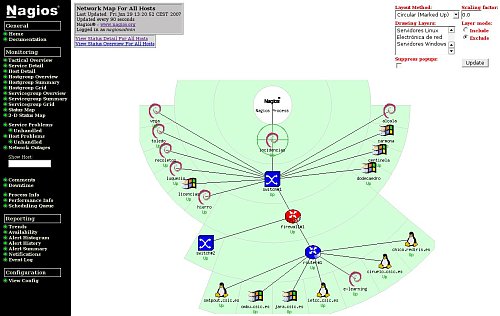
Nagios se distribuye bajo licencia GNU General Public License Version 2.

Es un software que proporciona una gran variedad de parámetros a consultar y supervisar en el sistema y genera alertas según pautas preestablecidas.

Entre sus virtudes figura la capacidad de supervisar protocolos de red tales como SMTP, POP3, HTTP y SNMP entre otros, la supervisión del hardware de los equipos conectados en la red y dispone de un conjunto de bibliotecas para el desarrollo de complementos para el sistema.

El diseño de dichos componentes o plugins se puede hacer mediante Bash, C++, Perl, Ruby, Python, PHP, C#,...

En cuanto a su implantación, como aplicación es bastante pesada, por lo cual normalmente necesita un host dedicado a este fin.



Ilustración

Webmin

Webmin un sistema de administración remota mediante HTTP para sistemas UNIX. Webmin permite configurar aspectos del sistema desde un navegador.

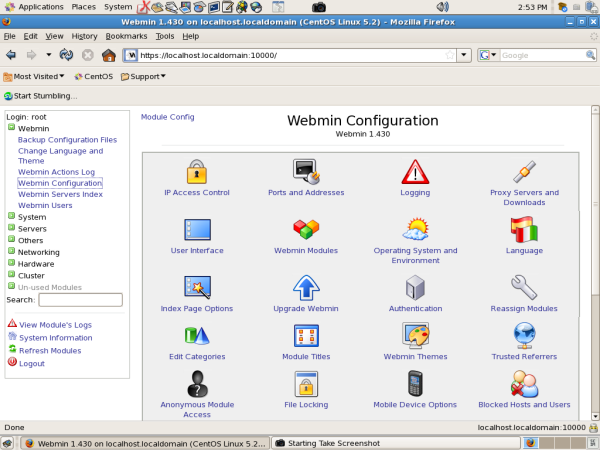
Webmin se distribuye bajo Licencia BSD.

Webmin está escrito en Perl, versión 5, y tiene su propio servidor HTTP ligero y con poco consumo de recursos.

Webmin es un sistema modular, y el usuario puede decidir que quiere y que no quiere cargar en la administración. Dispone a sí mismo un conjunto de bibliotecas para el desarrollo de estos módulos.

No está orientado al desarrollo de módulos no orientados a administración de sistemas.

Se implanta en sistemas UNIX, aunque existen proyecto para portarlo a Windows estas no son oficiales.



Ilustración

Ventajas e inconvenientes de este Proyecto Final de Carrera con respecto a los anteriores

Ahora pasamos a mostrar una tabla comparativa de los sistemas anteriormente descritos y la comparación de sus capacidades con las del proyecto final de carrera al que pertenece estas memorias.

La tabla comparativa no incluye los sistemas de supervisión de los distintos sistemas gestores de bases de datos, ya que estos pueden ser muy diferentes entre sí, y enumerarlas todas es una tarea ardua y como hemos comentado con anterioridad estas herramientas tiene capacidades superiores a las analizadas, ya que su ámbito de actuación es mucho más amplio y complejo.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | Nagios | Webmin | NetIQ AppManager for Oracle Database RDBMS Server | PFC |
| Conjunto de bibliotecas para desarrollo | SI | SI | NO | NO |
| Multiplataforma | NO (Host dedicado) | NO (Sistemas UNIX) | NO (Sistemas Windows) | SI (JVM) |
| Funcionalidad de propósito especifico | NO | NO | SI | SI |
| Implementado | NO | NO | SI | SI |
| Licencia | GPL v.2 | BSD | Propietaria | Apache 2.0 |
| Distribución | Gratuita | Gratuita | De pago | - |
| Generación de gráficas | - | - | SI | SI (Analizador) |
| Supervisión en tiempo real | - | - | SI | SI (Chartserver) |
| Servicio de sistema | NO (Host dedicado) | SI | SI | SI (Servicio) |
| Lenguaje nativo | C y otros | Perl | - | Java 1.6.0\_14 |
| Múltiples SGBD | - | - | NO (Oracle) | SI (JDBC) |

Ilustración

# Distintos tipos de soluciones para el mismo problema

# Herramientas para el desarrollo y la gestión del proyecto.

# 

# Diseño de la solución

# Conclusiones

# Bibliografía

[] Wikipedia. Iterative and incremental development. [En línea] http://en.wikipedia.org/wiki/Iterative\_development.

[] Sun Microsystems. Java™ Platform, Standard Edition 6. [En línea] http://java.sun.com/javase/6/docs/api/.

[] Sun Microsystem. Netbeans. [En línea] http://www.netbeans.org/.

[] Visualsvn. Visualsvn. [En línea] http://www.visualsvn.com/server/.

[] TortoiseSVN. TortoiseSVN The coolest Interface to (Sub)Version Control. [En línea] http://tortoisesvn.net/.

[] Sparx Systems. Enterprise Architect - UML for Business, Software and Systems. [En línea] http://www.sparxsystems.com.au/.

[] Sun Microsystems. MySQL - The world's most popular open source database. [En línea] http://www.mysql.com/.

[] SQLite. SQLite - Small. Fast. Reliable. [En línea] http://www.sqlite.org/.

[] SQLiteJDBC. SQLiteJDBC. [En línea] www.zentus.com/sqlitejdbc/.

[] Osenxpsuite. SQLite2009 Pro Enterprise Manager. [En línea] http://link.osenxpsuite.net/?uid=homepage&id=sqlite2009pro.zip.

[] Apache Software Foundation. Logging Services. [En línea] 1999-2007. http://logging.apache.org/log4j/1.2/index.html.

[] JFree.org. JFreeChart. [En línea] http://www.jfree.org/jfreechart/.

[] Serena. OpenProj is a free, open source project management solution. [En línea] http://openproj.org/openproj.

[] Sun Microsystems. OpenOffice - the free and open productivity suite. [En línea] http://www.openoffice.org/.

[] Microsoft Corporation. Microsoft Office Visio 2007. [En línea] http://office.microsoft.com/es-es/visio/FX100487863082.aspx.

[] Dr. Bert Scalzo, Claudia Fernandez, Donald K. Burleson, Mike Ault, Kevin Kline. Database Benchmarking: Practical Methods for Oracle & SQL Server. s.l. : Rampant TechPress, 2007. ISBN 0977671534, 9780977671533.

[] Darwin, Ian F. Java cookbook. s.l. : O'Reilly, 2001. ISBN 0596001703, 9780596001704.