UNIVERSIDAD POLITÉCNICA DE VALENCIA

ESCUELA POLITÉCNICA SUPERIOR DE GANDIA

Grado en Ing. Sist. de Telecom., Sonido e Imagen

"Diseño y desarrollo de una aplicación para operadora de Telecomunicaciones para monitorizar la conexión a Internet."

TRABAJO FINAL DE GRADO

Autor:

José Rubén Sáez Romero

Tutor UPV:

José Marín-Roig Ramón

Tutor empresa:

Oscar Silvestre Jover

GANDIA, 2016

Resumen

El creciente desarrollo de Internet y el avance de la ciencia y la tecnología que ampara esta plataforma motiva al autor para diseñar y crear una aplicación útil que sirva como proyecto de fin de carrera y herramienta, sencilla y clara, de ayuda a una operadora de telecomunicaciones tener funcionalidades como:

- La comprobación de conectividad a una o varias direcciones IP.
- La detección de posibles cortes continuos de la conexión a Internet.
- La indicación del momento producido por el corte.
- La posibilidad de recopilar los datos producidos por la monitorización.

La empresa y recientemente operadora, "Instalaciones y Comunicaciones de la Safor, Digitel S.L", tiene como objetivo dar servicio global de comunicación tanto al público en general como a empresas y autónomos. Además, avala su experiencia y buen hacer en el asesoramiento, comercialización, instalación y soporte de sistemas de comunicaciones.

La óptima gestión es básica en la estrategia de la compañía para contribuir al bienestar de la sociedad.

Así pues, con esta aplicación, la empresa puede visualizar y determinar, de una manera eficaz, si ofrece a sus clientes una buena calidad de servicio de conexión a Internet e incidir en aquellas posibles averías de una forma agilizada para su reparación, con la intención de lograr adelantarse a quejas de usuarios.

Palabras clave: Monitorización, Conectividad, Java, MySQL, Reporte.

Abstract

The increasing development of the Internet and the advancement of science and technology that protects this platform motivates the author to design and create a useful application that serve as final course project and assistance, simple and clear, to a telecommunications operator to have functionalities such as:

- Checking connectivity to one or more IP addresses.
- The detection of possible continuous cuts of Internet connection.
- The indication of the moment produced by the cut.
- The ability to collect the data produced by the monitoring.

The company and recently operator, "Instalaciones y Comunicaciones de la Safor, Digitel S.L ", aims to provide global communication service to both the general public and businesses and freelancers. In addition, the company endorses its vast experience and good work in consulting, marketing, installation and support of communications systems.

The optimal management is the key strategy of the company to optimize the contribution to the common good and welfare of the society.

So, with this application, the company can view and determine in an efficient manner if offers its customers a good quality of service Internet connection and influence in those possible faults of a streamlined way for repair, in order to achieve to anticipate complaints from users.

Keywords: Connectivity, Monitoring, Java, MySQL, Report.

Tabla de contenido

Resumen 3
Abstract 4
Tabla de contenido 5
Tabla de ilustraciones y figuras 7
Capítulo 1 8
Introducción 8
1.1. Motivación del proyecto 8
1.2. Estructura del proyecto 9
Capítulo 2 10
Tecnologías 10
2.1. Herramientas empleadas 10
2.1.1. Java 10
Características del lenguaje Java 11
2.1.2. Java Development Kit 12
Componentes del JDK 12
2.1.3. Java Runtime Environment 13
Componentes del JRE 13
2.1.4. Java Virtual Machine 14
2.1.5. JDBC 15
JDBC frente a ODBC y otros API's 15
2.1.6. NetBeans IDE 16
Características de NetBeans IDE 16
2.1.7. MySQL 17
Características de MySQL 17
2.1.8. WampServer 18
Características de WampServer 18
phpMyAdmin 19
2.1.9. JasperReports 19
2.1.10. Launch4j 20
2.1.11. Inno Setup Compiler 20

Capítulo 3 21
Conceptos 21
3.1. La idea que forma el entendimiento 21
3.1.1. El nivel IP 22
OSI frente TCP/IP 22
Capa de red 24
Protocolo de Internet 25
Direcciones IP 26
Protocolo ICMP 26
Comando Ping 28
3.1.2. Dominio 29
3.1.3. DDNS 29
Capítulo 4 30
Desarrollo 30
4.1. Estructura de la aplicación 30
4.2. Ápices sobre la implementación 31
4.2.1. Implementación Ping 31
4.2.2. Implementación pestaña Consola 31
4.2.3. Implementación Recordatorio 32
4.2.4. Implementación Tiempo de Repetición 33
4.2.5. Implementación Conexión con la base de datos. 34
4.2.6. Implementación pestaña Principal e Historial 37
4.2.7. Implementación JCheckBox 39
4.2.8. Implementación Ordenar y Filtrar tabla 40
4.2.9. Implementación Interfaz Inicio 41
4.2.10. Implementación Reporte 43
Capítulo 5 48
Conclusiones 48
Capítulo 6 49
Referencia bibliográfica 49

Tabla de ilustraciones y figuras

Ilustración 1. Logotipo Java 10
Ilustración 2. Logotipo Netbeans IDE 16
Ilustración 3. Logotipo de MySQL 17
Ilustración 4. Logotipo PHP 18
Ilustración 5. Logotipo MySQL 18
Ilustración 6. Logotipo Apache 18
Ilustración 7. Logotipo WampServer 18
Ilustración 8. Logotipo phpMyAdmin 19
Ilustración 9. Logotipo JasperReports 19
Ilustración 10. Logotipo Launch4j 20
Ilustración 11. Logotipo Inno Setup Compiler 20
Ilustración 12. Nivel de Red 24
Ilustración 13. Distribución del proyecto en NetBeans IDE 30
Ilustración 14. Pestaña Consola de la aplicación. 32
Ilustración 15. Creación BBDD con phpMyAdmin 34
Ilustración 16. Estructura de la tabla "tabla_ping". 35
Ilustración 17. Estructura de la tabla "tabla_ping2". 36
Ilustración 18. Agregar JAR de MySQL 36
Ilustración 19. JCheckBox 39
Ilustración 20. Orden ascendente y descendente. 40
Ilustración 21. Captura Interfaz Inicio 43
Ilustración 22. Agregar libreria de JasperReports. 43
Ilustración 23. Librería JasperReports. 44
Ilustración 24. Icono Connections/Datasources. 44
Ilustración 25. Database JDBC connection 1. 45
Ilustración 26. Database JDBC connection. 45
Ilustración 27. Sentencia SQL del reporte. 46
Ilustración 28. Grupo con campos del Reporte 46
Ilustración 29. Diseño Reporte Principal. 47
Ilustración 30. Diseño Reporte Historial. 47

Figura 1. Esquema conceptual JRE 13
Figura 2. Arquitectura general de un programa en ejecución en una JVM 14
Figura 3. Clases en el paquete java.sql 15
Figura 4. Servicios de Operadores 21
Figura 5. Primitivas entre niveles OSI 22
Figura 6. Modelo OSI 23
Figura 7. Modelo TCP/IP 23
Figura 8. Encapsulado de mensajes ICMP 27

Capítulo 1

Introducción

1.1. Motivación del proyecto

El acelerado avance tecnológico pone a disposición del usuario sistemas cada vez más desarrollados y confiables. La extensa cobertura posibilita que las comunicaciones alcancen cada vez lugares más distantes, brindando más servicios y mejorando el ancho de banda con tasas de transmisión más altas. Y el reducido coste y las múltiples utilidades ofrecidas por los sistemas de comunicación, ofrecen una interacción constante y bidireccional entre la tecnología y la sociedad.

Actualmente, la gestión de redes de ordenadores y su soporte conlleva en diversas circunstancias extenuar el intelecto humano de una manera innecesaria pues pueden surgir diversos errores de forma continua y los administradores, aún teniendo un plano de la topología y arquitectura de la red, tardan en detectar y poner solución a estos inconvenientes.

Así pues, los administradores hacen uso de herramientas de diagnóstico de red con la finalidad de conseguir una supervisión que establezca una comunicación sólida y efectiva en las redes.

Los objetivos principales para este propósito deben ser la monitorización del tráfico de la red, la correcta configuración de los recursos de seguridad que doten los equipos y mantener estaciones de trabajo y servidores, correspondientes al dominio o grupo de trabajo, en activo.

Es por ello que se requieren herramientas eficientes para la detección de posibles conflictos que proporcionen el buen servicio del sistema administrado. Esta es la causa que provoca que la labor de analizar y supervisar las redes cobre gran importancia, pues la supervisión de una red es una actividad fundamental ya que permite captar con anticipación posibles caídas del sistema y tener estrategias de contingencia para poder solucionar incidentes de forma apropiada.

En esta memoria se presenta una aplicación elaborada en lenguaje de programación Java, conectada a una base de datos mediante MySQL, capaz de comprobar el estado de conexión a una o varias direcciones IP gracias a la implementación de una interfaz gráfica que permite la generación de un informe en base a los resultados obtenidos, todo ello construido en un entorno de desarrollo integrado llamado NetBeans IDE y ofreciendo un pequeño apoyo al diagnóstico y mantenimiento de redes.

1.2. Estructura del proyecto

"El Marco Teórico de trabajo se puede definir como aquel modelo conceptual en donde se describe la forma cómo se teorizan las relaciones entre varios factores que han sido identificados como importantes para el problema. Por teoría se entiende un conjunto de conceptos interrelacionados sistemáticamente, propuestas, definiciones y variables son expuestas anticipadamente con el fin de explicar y predecir los hechos que se van a investigar". (Eyssautier, 2002).

Con ello, para abordar el diagnóstico de redes y predecir cualquier conflicto que mane sobre la herramienta creada se establecen cuatro partes primordiales para la presente memoria.

- Parte 1: ubica el marco de la aplicación creada, exponiendo a que concierne el proyecto, qué objetivos persigue y cuál es su motivación.
- Parte 2: ofrece una descripción de las tecnologías referentes a la herramienta creada, detallando características y utilidades.
- Parte 3: esclarece con contundencia para comprender más a fondo la aplicación aquellos conceptos que se estiman de importancia.
- Parte 4: ilustra ápices de código con el fin de orientar al usuario como ha sido creada la aplicación.
- Parte 5: expone una conclusión sobre la experiencia de programar este tipo de utilidad.
- Parte 6: anexa el código fuente además de iconos y el estético aspecto visual final de la interfaz.

Para concluir, en la bibliografía se pueden localizar todos los puntos tratados en la memoria con la virtud de indagar en aquellas incisiones que inquieten al lector.

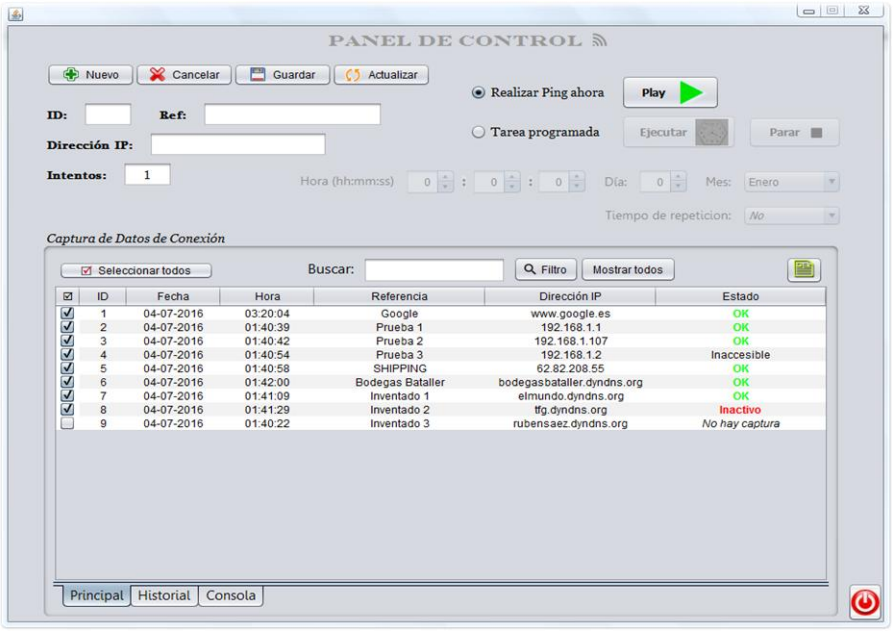

Ilustración 1. Aspecto final de Interfaz

Capítulo 2

Tecnologías

2.1. Herramientas empleadas

Como es de costumbre, en el desarrollo de programas se hace uso de diferentes ámbitos y especialidades, ya que se han de considerar aspectos como el lenguaje, interactividad, sistemas de control, adquisición y almacenamiento de datos.

Un aspecto importante en el desarrollo de un proyecto informático es la elección del lenguaje más apropiado para su implementación y las herramientas a utilizar. Existen múltiples herramientas informáticas y múltiples lenguajes de programación dependiendo del área de estudio. Una vez elegido el lenguaje, se ha de recabar información sobre las herramientas disponibles y realizar un estudio acerca del cual se adapta mejor a las necesidades del proyecto a realizar.

En esta parte, se especifican de forma genérica los componentes clave que resultan imprescindibles para el desarrollo e implementación que conforma la aplicación creada.

2.1.1. Java

El lenguaje de programación elegido, Java, fue originalmente diseñado en 1990 por James Gosling, de la compañía Sun MicroSystems, como software para dispositivos electrónicos de consumo así como calculadoras, microondas y la televisión interactiva.

Java inicialmente se denominó Oak (roble en inglés), luego pasó a denominarse Green pero tras descubrir que era ya una marca comercial registrada finalmente se renombró a Java.

Java es un lenguaje de programación de propósito general y orientado a objetos desarrollado por Sun MicroSystems en los primeros años de la década de los noventa. A diferencia de los lenguajes C y C++, tiene un modelo de objetos más robusto eliminando así herramientas de bajo nivel, que suelen tener muchos errores, como la manipulación directa de punteros o memoria.

Ilustración 2. Logotipo Java

Sun MicroSystems desarrolló la implementación de referencia original para compiladores de Java, máquinas virtuales, y librerías de clases en 1991 y las publicó por primera vez en 1995.

Fue en 2009 cuando Oracle Corportation, compañía de software que desarrolla bases de datos y sistemas de gestión de bases de datos, adquiere Sun MicroSystems.

Características del lenguaje Java

- Todo programa en Java ha de compilarse y el código que se genera (bytecodes), es interpretado por una máquina virtual. De este modo se consigue la independencia de la máquina, el código compilado se ejecuta en máquinas virtuales que si son dependientes de la plataforma.
- Existen varios niveles de seguridad en Java, desde el ámbito del programador, hasta el ámbito de la ejecución en la máquina virtual.
- Con respecto al programador, Java realiza la comprobación estricta de tipos durante la compilación, evitando con ello problemas tales como el desbordamiento de la pila. Pero, es durante la ejecución donde se encuentra el método adecuado según el tipo de la clase receptora del mensaje; aunque siempre es posible forzar un enlace estático declarando un método como final.
- Todas las instancias de una clase se crean con el operador new(), de manera que un recolector de basura se encarga de liberar la memoria ocupada por los objetos que ya no están referenciados. La máquina virtual de Java gestiona la memoria dinámicamente.
- Una fuente común de errores en programación proviene del uso de punteros. En Java se han eliminado los punteros, el acceso a las instancias de clase se hace a través de referencias.
- El programador siempre está obligado a tratar las posibles excepciones que se produzcan en tiempo de ejecución. Java define procedimientos para tratar estos errores.
- Java posee mecanismos para garantizar la seguridad durante la ejecución comprobando, antes de ejecutar código, que este no viola ninguna restricción de seguridad del sistema donde se va a ejecutar.
- También cuenta con un cargador de clases, de modo que todas las clases cargadas a través de la red tienen su propio espacio de nombres para no interferir con las clases locales.
- Además, Java está preparado para la programación concurrente sin necesidad de utilizar ningún tipo de biblioteca.
- Finalmente, Java posee un gestor de seguridad con el que poder restringir el acceso a los recursos del sistema.

2.1.2. Java Development Kit

El Kit de Desarrollo de Java (JDK) es un conjunto de herramientas, utilidades, documentación y ejemplos para la creación de aplicaciones Java. Puede instalarse en un equipo local o en una unidad de red.

En un equipo puede haber varios JDK de distintas versiones instaladas, y éstos pueden ser utilizados a voluntad siempre y cuando se configure correctamente la herramienta de desarrollo, sin embargo y por lo general, el sistema operativo únicamente poseerá el JRE más actual.

JDK consta de una serie de componentes para realizar cada una de las tareas de las que es capaz de encargarse.

A continuación se explican más en profundidad cada uno de los componentes.

Componentes del JDK

- **Intérprete en tiempo de ejecución (JRE):** Permite la ejecución de los programas Java (*.class).
- **Compilador (javac):** Se utiliza para compilar archivos de código fuente Java (habitualmente *.java), en archivos de clases Java ejecutables (*.class). Se crea un archivo de clase para cada clase definida en un archivo fuente.
- **Visualizador de applets (appletviewer):** Es una herramienta que sirve como campo de pruebas de applets, visualizando cómo se mostrarían en un navegador en lugar de tener que esperar.

  Al ser activado desde una línea de órdenes se abre una ventana en la que muestra el contenido de la applet.
- **Depurador (jdb):** Es una utilidad de línea de comandos que permite depurar aplicaciones Java.

  No es un entorno de características visuales, pero permite encontrar y eliminar los errores de los programas Java con mucha exactitud.
- **Desensamblador de archivo de clase (javap):** Su salida, por defecto, muestra los atributos y métodos públicos de la clase desensamblada, pero con la opción –c también desensambla los códigos de bytes, mostrándolos por pantalla. Es útil cuando se quiere saber cómo fue codificada una clase sin tener el código fuente.
- **Generador de cabecera y archivo apéndice (javah):** Se utiliza para generar archivos fuentes y cabeceras C para implementar métodos Java en C (código nativo). Esto se consigue mediante la generación de una estructura C cuya distribución coincide con la correspondiente clase Java.

  El generador de cabeceras javah, crea los ficheros de cabecera C/C++ para implementar en esos lenguajes los métodos nativos que presenta un programa Java.

- **Generador de documentación (javadoc):** Es útil para la generación de documentación sobre la interfaz de programación de aplicaciones, o API, directamente desde el código fuente Java. Genera páginas HTML basadas en las declaraciones y comentarios javadoc, con el formato /** comentarios */.

  La documentación que genera es del mismo estilo que la documentación que se obtiene con el JDK y las etiquetas, que se indican con una arroba, aparecerán resaltadas en la documentación generada.

- **Applets de demostración:** El JDK incluye una serie de applets de demostración, con su código fuente. El código fuente de la API se instala de forma automática, cuando se descomprime el JDK, aunque permanece en formato comprimido en un archivo llamado "src.zip" localizado en el directorio Java que se creó durante la instalación.

2.1.3. Java Runtime Environment

El Entorno de Ejecución Java (JRE) es un conjunto de utilidades que permite la ejecución de programas Java. Es decir, es el requerimiento mínimo de software que debe tener un equipo para poder ejecutar o correr aplicaciones desarrolladas en Java.

Normalmente el JRE está destinado a usuarios finales que no requieren el JDK, pues a diferencia de este no contiene los programas necesarios para crear aplicaciones en el lenguaje Java, así pues, el JRE se puede instalar sin necesidad de instalar el JDK, pero al instalar el JDK, este siempre cuenta en su interior con el JRE.

Componentes del JRE

El JRE está compuesto básicamente por:

- Las librerías de clases estándar que conforman la interfaz de programación de aplicaciones provista por los creadores del lenguaje de programación Java (API Java).
- Un componente llamado Java Virtual Machine.

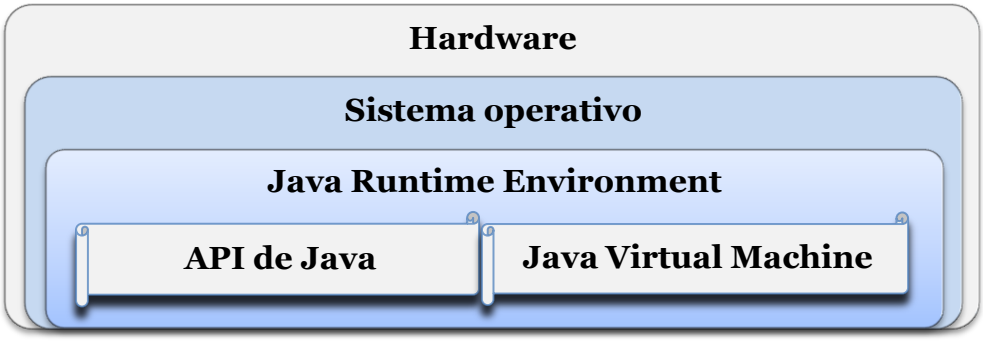

Figura 1. Esquema conceptual JRE

2.1.4. Java Virtual Machine

La Máquina Virtual Java, o JVM, es el entorno en el que se ejecutan los programas Java y tiene como misión principal garantizar la portabilidad de las aplicaciones Java.

Con el compilador se convierte el código fuente que reside en archivos cuya extensión es ".java", a un conjunto de instrucciones que recibe el nombre de "bytecodes" y que se guardan en un archivo cuya extensión es ".class".

El intérprete ejecuta cada una de estas instrucciones independientes del tipo de ordenador. Solamente es necesario, por tanto, compilar una vez el programa, pero se interpreta cada vez que se ejecuta en un ordenador.

Figura 2. Arquitectura general de un programa en ejecución en una JVM

Por tanto, cada intérprete Java es una implementación de la Máquina Virtual Java. Los "bytecodes" posibilitan el objetivo de "write once, run anywhere", es decir, escribir el programa una vez y que se pueda correr en cualquier plataforma que disponga de una implementación de la JVM.

Las principales tareas de la JVM son las siguientes:

- Reservar espacio en memoria para los archivos creados ".class".
- Liberar la memoria no usada (garbage collection).
- Asignar variables a registros y pilas.
- Llamar al sistema huésped para ciertas funciones, como los accesos a los dispositivos.
- Verificar el código de bytes para así asegurar que sean válidos y que no violen las restricciones de las normas de seguridad de las aplicaciones Java.

Además, las propias especificaciones del lenguaje Java contribuyen a que:

- Las referencias a arrays son verificadas en el momento de la ejecución del programa.
- No hay manera de manipular de forma directa los punteros.
- La JVM gestiona automáticamente el uso de la memoria, de modo que no queden huecos.
- No se permiten realizar ciertas conversiones (casting) entre distintos tipos de datos.

2.1.5. JDBC

El API Java Database Connectivity se basa en un conjunto de clases e interfaces escritas en el lenguaje de programación Java para la ejecución de sentencias SQL y así proporcionar un acceso uniforme a una gran variedad de bases de datos relacionales, independientemente del sistema operativo donde se ejecute o de la base de datos a la cual se accede. Así pues, con un único programa escrito y usando el API JDBC, el programa será capaz de enviar sentencias SQL a la base de datos apropiada.

Eso sí, el uso de JDBC con un sistema gestor de base de datos requiere disponer del driver JDBC adecuado que haga de intermediario entre los dos componentes. Este driver puede estar escrito en Java puro, o ser una mezcla de Java y métodos nativos JNI (Java Native Interface).

El resultado es un conjunto de clases e interfaces, localizadas en el paquete java.sql, que pueden ser utilizadas con cualquier base de datos que disponga del driver JDBC apropiado. La utilización de este driver significa que dicha aplicación podrá utilizarse con una base de datos diferente cambiando simplemente el driver JDBC por el que ofrezca el fabricante del servidor al que se desea acceder.

En el paquete java.sql existen las siguientes clases para trabajar con bases de datos:

Driver Manager	• Carga un driver.
Connection	• Establece conexiones con las bases de datos.
Statement	• Ejecuta sentencias SQL y las envia a las BBDD.
PreparedStatement	• Contiene una sentencia SQL ya compilada.
ResultSet	• Almacena el resultado de la consulta.

Figura 3. Clases en el paquete java.sql

JDBC frente a ODBC y otros API's

El ODBC de Microsoft (Open Database Connectvity), es el API probablemente más extendido para el acceso a bases de datos relacionales. Ofrece la posibilidad de conectar a la mayoría de las bases de datos en casi todas las plataformas pero, tal vez nos estemos preguntando qué motivos hizo que se desarrollara JDBC existiendo OBDC. Las respuestas son las siguientes:

- ODBC no es apropiado para su uso directo con Java porque utiliza una interface C. Las llamadas desde Java a código nativo C tienen un número de inconvenientes en la seguridad, la implementación, la robustez y en la portabilidad de las aplicaciones.
- Cuando se usa ODBC, el gestor de drivers de ODBC y los drivers deben instalarse manualmente en cada máquina cliente, en cambio los drivers de JDBC como están completamente escritos en Java, el código JDBC es automáticamente instalable, portable y seguro en todas las plataformas Java.

2.1.6. NetBeans IDE

NetBeans IDE es un software libre desarrollado inicialmente por Sun Microsystems. Es una herramienta muy útil para desarrollar aplicaciones para internet, dispositivos móviles, animaciones, soluciones de negocios, etc. usando lenguajes de programación como Java, PHP, C/C++, JavaScript, Groovy y Ruby. Puede ser instalado en sistemas operativos como Windows, Mac, Linux y Solaris.

NetBeans es un proyecto público que consiste en un entorno integrado de desarrollo, IDE (Integrated Development Enviroment), de código abierto con una gran base de usuarios y con una comunidad en constante crecimiento.

Ilustración 3. Logotipo Netbeans IDE

Características de NetBeans IDE

- **Gran soporte:** Cualquier novedad del lenguaje es rápidamente soportada por NetBeans.
- **Asistentes:** Dispone de asistentes para la creación y configuración de distintos proyectos, incluida la elección de algunos "frameworks".
- **Buen editor de código:** Con el habitual coloreado y sugerencias de código, acceso a clases pinchando en el código, control de versiones, localización de ubicación de la clase actual, comprobaciones sintácticas y semánticas, plantillas de código, herramientas de refactorización, etc.
- **Simplifica la gestión de grandes proyectos:** Con el uso de diferentes vistas estructura la visualización de manera ordenada.
- **Herramientas para depurado de errores:** El "Debugger" que incluye el IDE es bastante útil para encontrar dónde fallan las cosas. Se pueden definir puntos de ruptura en la línea de código que nos interese y monitorizar en tiempo real los valores de propiedades y variables.
- **Optimización de código:** Por su parte, el "Profiler" ayuda a optimizar las aplicaciones e intenta hacer que se ejecuten más rápido y con el mínimo uso de memoria. Se puede ver el comportamiento de nuestra aplicación y obtener indicadores e información de cómo y cuántos recursos consume, cuántos objetos se crean y obtener capturas del estado del sistema en diferentes momentos.
- **Acceso a base de datos:** Desde el propio NetBeans podemos conectarnos a distintos sistemas gestores de bases de datos, como pueden ser Oracle, MySQL y demás, ver las tablas, realizar consultas y modificaciones, y todo ello integrado en el propio IDE.
- **Fácilmente extensible:** A través de plugins.

2.1.7. MySQL

MySQL es un sistema de administración de bases de datos relacional, o RDBMS, desarrollado y proporcionado por MySQL AB. MySQL AB fue adquirida por Sun Microsystems en 2008, y ésta fue comprada por Oracle Corporation en 2010, la cual ya era dueña desde 2005 de Innobase Oy, empresa finlandesa desarrolladora del motor InnoDB para MySQL.

Ilustración 4. Logotipo de MySQL

MySQL es un sistema de gestión de bases de datos que permite agregar, acceder y procesar los datos almacenados con gestión de usuarios y passwords manteniendo la seguridad en los datos.

Una base de datos es una colección estructurada de datos y la información que puede almacenar puede ser tan simple como la de una agenda, un contador, o un libro de visitas, ó tan vasta como la de una tienda en línea o la información generada en una red corporativa.

Características de MySQL

- Utiliza el lenguaje estructurado de consulta SQL (Structured Query Language) que es el estándar de consulta a bases de datos a nivel mundial.
- Es una base de datos relacional que almacena los datos en tablas separadas en lugar de poner los datos en un solo lugar. Así las tablas pueden ser enlazadas al definir las relaciones que hacen posible combinar datos de varias tablas cuando se necesitan consultar datos. Esto agrega velocidad y flexibilidad.
- MySQL está considerada la base de datos "open source" más popular del mundo. "Open source" significa que cualquier persona puede descargar, usar y modificar MySQL.
- MySQL es un servidor multiusuario muy rápido y robusto de ejecución de instrucciones en paralelo, es decir, que múltiples usuarios distribuidos a lo largo de una red local o Internet podrán ejecutar distintas tareas sobre las bases de datos localizadas en un mismo servidor.
- MySQL usa la licencia GPL (Licencia Pública General GNU). GNU GPL garantiza a los usuarios finales (personas, organizaciones, compañías) la libertad de usar, estudiar, compartir y modificar el software. Su propósito es declarar que el software cubierto por esta licencia es software libre pero protegido de intentos de apropiación por parte de los usuarios.
- Las plataformas que utiliza son varias y entre ellas podemos mencionar LAMP, MAMP, SAMP, BAMP y WAMP, aplicables a Mac, Windows, Linux, BSD, Open Solaris, Perl y Phyton entre otras.

2.1.8. WampServer

Un servidor WAMP, LAMP o MAMP es un paquete compuesto de software que contiene un entorno de desarrollo web completo y elemental, en concreto Apache MySQL y PHP, que da pie a los desarrolladores instalar de una manera sencilla todo el software necesario para crear aplicaciones web. La diferencia que existe entre WAMP, LAMP y MAMP está en la plataforma o sistema operativo en el cual se aplica, por lo que se traduce WAMP a Windows Apache MySQL PHP, LAMP a Linux Apache MySQL PHP y MAMP a MAC Apache MySQL PHP.

Ilustración 7. Logotipo Apache

Ilustración 6. Logotipo MySQL

Ilustración 5. Logotipo PHP

Características de WampServer

Como se ha dicho anteriormente, WampServer es un entorno de desarrollo web para el sistema operativo Windows, que además es un proyecto de código abierto, completamente gratuito, con licencia GPL y que nos permite:

- Disponer de un servidor web instalado en nuestro equipo.
- El uso de la herramienta adicional phpMyAdmin.
- Administrar las configuraciones de servidores Apache.
- Programar con el lenguaje PHP.
- Ejecutar archivos con extensión ".php" de manera local.
- Tener una vista previa de la realización de los proyectos web por parte de los desarrolladores antes de subirse al servidor web en Internet.
- Gestionar bases de datos MySQL.
- Tener acceso a sus registros y a sus archivos de configuración.
- Disponer de accesos para el arranque y la parada de los servicios.

Ilustración 8. Logotipo WampServer

phpMyAdmin

Administrar y gestionar las bases de datos de MySQL deriva ser en ocasionas un tanto laboriosa, más para usuarios acostumbrados a usar herramientas con interfaz gráfica de usuario.

Por suerte existen alternativas para la administración de las bases de datos que propician ser más intuitivas y sencillas de utilizar. En concreto, una de las más conocidas es la herramienta gratuita phpMyAdmin.

Ilustración 9. Logotipo phpMyAdmin

La aplicación phpMyAdmin es un conjunto de páginas escritas en PHP y que son copiadas directamente en el escritorio que aloja las páginas web del servidor. Mediante las diferentes páginas se pueden crear y eliminar bases de datos, crear, eliminar y alterar tablas, realizar consultas, insertar, editar y borrar registros, ejecutar cualquier sentencia SQL, administrar privilegios y exportar datos en varios formatos.

Es necesario disponer de un servidor web con soporte PHP y MySQL para poder utilizar la herramienta phpMyAdmin. Por supuesto, este pack es brindado por WampServer, mencionado anteriormente.

2.1.9. JasperReports

JasperReports es una librería "open source" que nos permite la generación y el desarrollo de informes desde Java sobre bases de datos. La publicación y exportación de los informes pueden ser en formatos PDF, RTF, XML, XLS, CSV, HTML, XHTML, DOCX o TXT.

Ilustración 10. Logotipo JasperReports

El funcionamiento consiste en escribir un xml donde se recogen las particularidades del informe. La extensión de documentos de reportes generados por JasperReports es ".jrxml". Este documento xml lo tratan las clases del Jasper para obtener una salida en los formatos mencionados anteriormente.

Para generar el documento xml, o el archivo compilado ".jasper", se recomienda utilizar la herramienta iReport. Esta herramienta trabaja de manera integrada con JasperReports, por lo tanto no se requiere una instalación adicional de Jasper.

iReport es un entorno gráfico de código abierto implementado en Java para crear la parte visual de los reportes así como formatos, tipos de letra, estilos, imágenes, decoraciones, etc.

2.1.10. Launch4j

Launch4j es un software libre y multiplataforma bajo licencia GPL que se utiliza para crear ejecutables ligeros y nativos de Windows con extensión ".exe" a partir de aplicaciones Java distribuidas como archivos con extensión ".jar".

Ilustración 11. Logotipo Launch4j

Con dicho ejecutable es posible configurarlo para que utilice una determinada versión del JRE o introducir una URL para descargar en caso de no encontrarse la versión apropiada, además de establecer el tiempo de ejecución o el tamaño tanto inicial como máximo de la memoria. Por otro lado, el empaquetador ofrece una mejor experiencia de usuario al incluir un icono de aplicación, un nombre de fichero de ejecutable para identificar fácilmente la aplicación personalizada y una pantalla de bienvenida con tiempo de espera mostrada hasta que la aplicación Java comienza.

2.1.11. Inno Setup Compiler

El compilador Inno Setup es un software libre y gratuito para crear asistentes de instalación de una aplicación. Inno Setup es creado en Delphi por Jordan Rusell y soportado por todas las plataformas y versiones de Windows.

Ilustración 12. Logotipo Inno Setup Compiler

Inno Setup Compiler funciona mediante el uso de un asistente de configuración que permite personalizar el proceso de instalación o bien, en modo más avanzado, con la programación de un script, que básicamente es un conjunto de secuencias de comandos, o órdenes guardadas en un archivo de texto y que es ejecutado por lotes o línea a línea.

Los scripts de Inno Setup son archivos de texto con extensión ".iss". La función del script es controlar aspectos de la instalación como especificar qué archivos van a ser instalados y dónde, qué menús y carpetas deben ser incluidos y qué recursos van a ser requeridos por la aplicación a instalar. Además con Inno Setup es posible la creación de accesos directos en cualquier directorio, incluso en el menú de Inicio, e incorporar un desinstalador.

Los archivos de secuencias de comandos suelen ser editados desde dentro del programa de configuración del compilador. Una vez terminado de escribir el script o guión, la siguiente y última fase es compilar. Esto crea una solución completa, lista para ejecutar el programa de configuración en base a su guión.

Gracias a este software es posible crear, editar y compilar el archivo script de extensión ".iss" con el objetivo de generar un archivo ejecutable con extensión ".exe" que contenga un programa completo listo para la distribución y para ejecutar su instalación.

Capítulo 3

Conceptos

3.1. La idea que forma el entendimiento

En un mundo interconectado en el que cada vez son más notorias las tecnologías de la información y la comunicación impulsa, a escala urbana, la necesidad de infiltrarse en la confluencia tecnológica.

Sobre este razonamiento variante, flexible, inestable y discontinuo en el tiempo, se halla el requisito para tener acceso al ciberespacio. En este entorno, la conectividad, se manifiesta como un suceso espacial y tecnológico, distribuido y organizado a partir de los dispositivos conectados al medio y al hombre con la finalidad de optimizar la comunicación, creando así el espacio de flujos.

La conectividad, que consta de los elementos tecnológicos necesarios para el enlace con la red global de informaciones, debe cumplir las exigencias del cliente respecto a la calidad de servicio. En caso contrario, nos veremos envueltos en una deficiencia en las prestaciones de este, así como no suministrar una red destinada a la alta velocidad asta incluso no mejorar la calidad de vida de las personas.

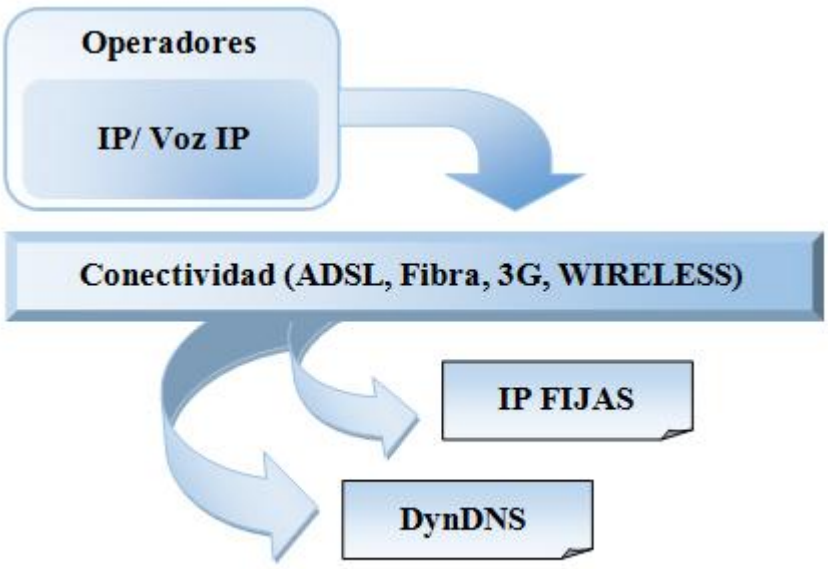

Figura 4. Servicios de Operadores

3.1.1. El nivel IP

OSI frente TCP/IP

Las primeras soluciones de red eran propietarias de cada fabricante e incompatibles entre sí, por lo que surgió la necesidad de una normalización. Como resultado, en los años 70, la Organización Internacional de Normalización (ISO - International Organization for Standardization) desarrolla un modelo de referencia para la interconexión de sistemas abiertos (OSI), que proporcionó la base para definir el proceso global de las comunicaciones entre equipos de datos, aceptado por la ITU-T en su recomendación X.200.

Se puede resumir del modelo OSI las siguientes características generales:

- En OSI se especifican un total de 7 capas. Cada capa utiliza los servicios ofrecidos por la capa inmediatamente inferior, y ofrece sus propios servicios a la capa inmediatamente superior.
- La frontera o interfaz entre dos niveles o capas residentes en el mismo sistema se define en términos de primitivas, que definen completamente el servicio ofrecido. Estas primitivas pueden ser de cuatro tipos:
  - Petición: utilizada por la capa usuario para invocar una función o servicio de la capa proveedora de servicio.
  - Indicación: utilizada por la capa proveedora para notificar que una función ha sido invocada.
  - Respuesta: utilizada por la capa usuaria para completar la función invocada mediante una primitiva de petición previa.
  - Confirmación: mediante la cual, la capa proveedora confirma que una función previamente invocada se ha completado.

Cualquier servicio se ofrece como combinación de estos cuatro tipos de primitivas, que además contendrán un conjunto de parámetros específicos del servicio a ofrecer.

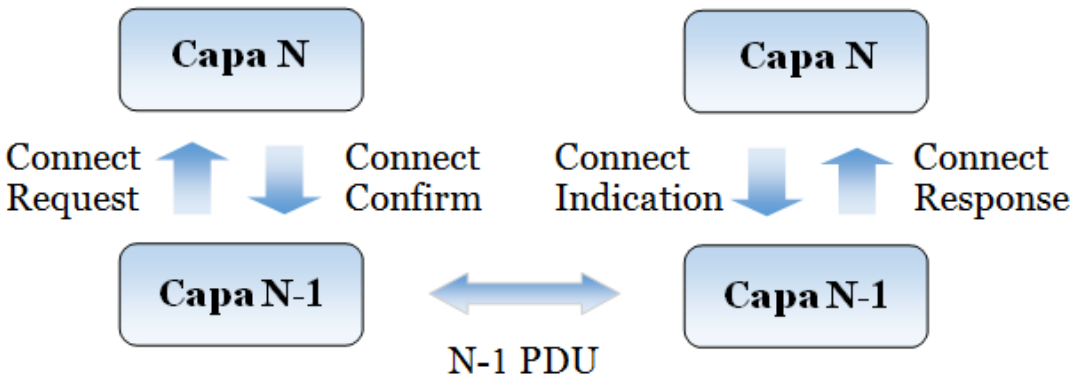

Figura 5. Primitivas entre niveles OSI

Por otro lado, se desarrolla un proyecto encaminado a proporcionar una red de datos robusta de arquitectura distribuida. Dicha red nace en 1977-79 y se denomina ARPA Net. Se inicia en el departamento de defensa de EEUU, y progresivamente va incorporando universidades y entidades públicas, de forma que en 1983 se divide en dos redes: MilNET, con fines militares, y ARPA Net, para propósitos de investigación. Esta última fue creciendo e incorporando nuevos equipos hasta formar lo que hoy conocemos como Internet.

Internet se apoya para su funcionamiento en una serie de protocolos abiertos, públicos y gratuitos conocidos generalmente como TCP/IP. La torre de protocolos TCP/IP está compuesta por cinco niveles: físico, enlace, red, transporte, y aplicación.

El Conjunto de Protocolos TCP/IP y su correspondiente pila han sido utilizados antes de que se estableciera el modelo OSI y desde entonces el modelo TCP/IP ha sido comparado con el modelo OSI tanto en libros como en instituciones educativas. Ambas se relacionan pero no son equiparables.

Y aunque entre los dos modelos se observan similitudes, las diferencias que destacan son que TCP/IP combina las funciones de la capa de presentación y de sesión en la capa de aplicación, además de combinar la capa de enlace de datos y la capa física del modelo OSI en una sola capa llamada acceso de red.

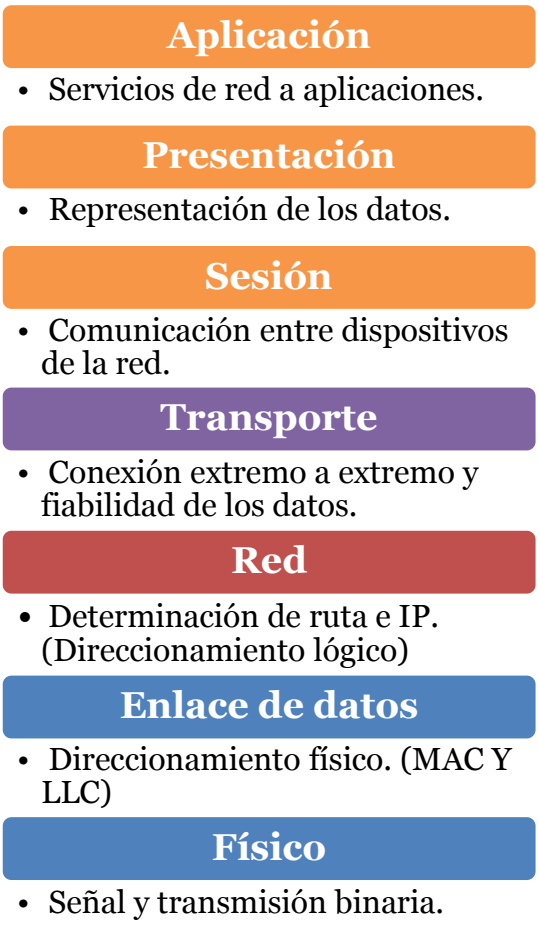

Figura 6. Modelo OSI

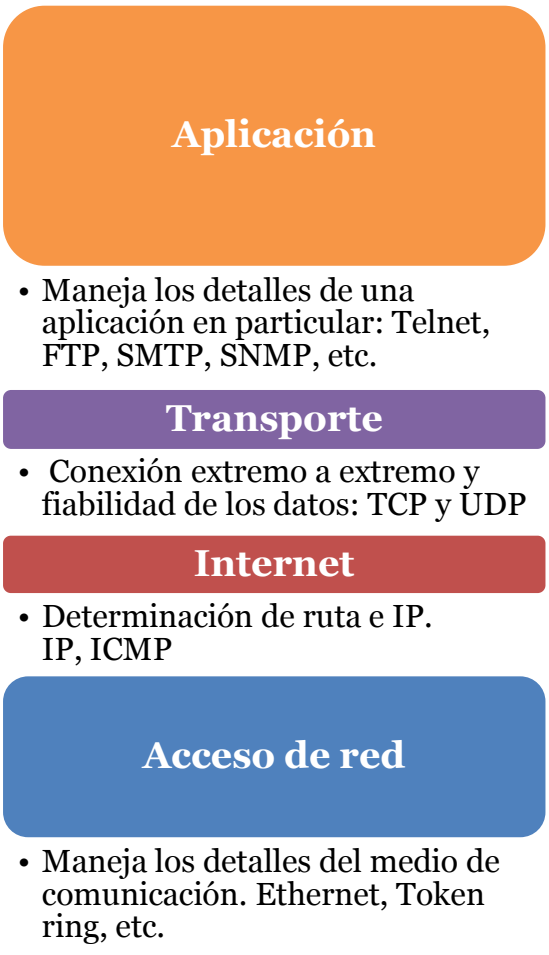

Figura 7. Modelo TCP/IP

Capa de red

En base al modelo de referencia de OSI o TCP/IP, la capa o nivel de red se denomina capa de interred en la arquitectura de red. Esta capa se ubica por encima de la capa de enlace. Esto implica que en esta capa es donde se produce la gestión de encaminamiento. La información que le entrega la capa de enlace, y que ésta a su vez ha recibido de la capa física, es encaminada hacia su respectivo destino.

La función de encaminamiento de la red debe optimizar el camino de envío de los paquetes, de manera que se emplee el menor tiempo posible, se minimicen las pérdidas de información y no se congestione la red. La capa de red debe incluir, por lo tanto, funciones de control de la congestión.

Además, el nivel de red debe contar con funciones de contabilización del tráfico permitiendo así la tarificación por volumen de tráfico y, en general, debe ajustar el paso de unas redes a otras distintas incorporando funciones tales como la interpretación de direcciones y, de ser necesaria, la corrección del tamaño de los paquetes.

En Internet, el protocolo empleado en la capa de red es el denominado protocolo de internet o IP.

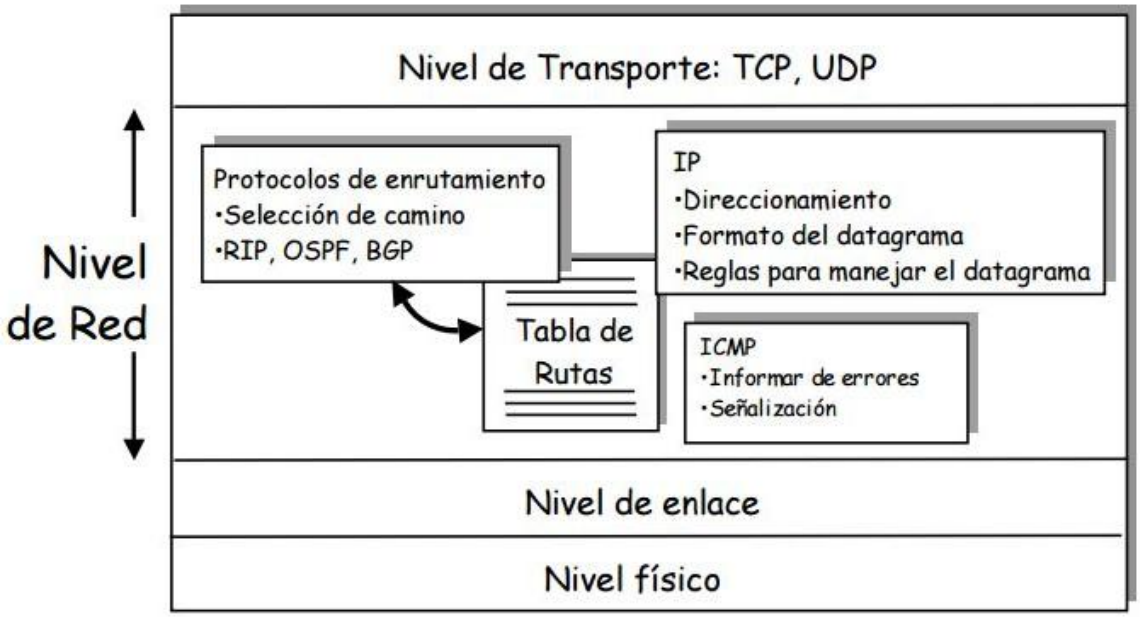

Ilustración 13. Nivel de Red

El servicio proporcionado por el nivel IP es el siguiente:

- No orientado a la conexión.
- No fiable.
- Best – effort.

Necesita protocolos de niveles superiores que proporcionen la calidad de servicio.

Protocolo de Internet

El protocolo de Internet, o IP, tiene como propósito ofrecer los mecanismos necesarios para la transmisión de unidades de datos desde el origen al destino. Las unidades de datos se denominan paquetes o datagramas de IP y, gracias a las direcciones de las cabeceras de los paquetes IP, los routers de Internet determinan la selección del camino más adecuado para un paquete dado. Los hosts implicados en una sesión pueden hacer uso de diferentes campos de la cabecera IP para fragmentar y reensamblar los paquetes IP cuando éstos deban atravesar tramos de red que requieran un tamaño más pequeño para las tramas del nivel de enlace. IP no garantiza la entrega fiable de los datagramas al host destino pues no implementa ningún mecanismo dirigido a ofrecer garantías "end to end" como TCP.

El protocolo de Internet utiliza cuatro campos clave para prestar su servicio:

1. El Tipo de Servicio (TOS - Type Of Service), conjunto de parámetros utilizados para determinar de qué modo hay que tratar a cada uno de los paquetes y así indicar la calidad del servicio deseado.
2. El Tiempo de Vida (TTL - Time To Live) fijado por el remitente, indica el periodo de vida de un paquete IP. Este valor se reduce en una unidad en todos los routers por los que va atravesando. Si el tiempo de vida alcanza un valor de cero antes de que el paquete llegue a su destino, éste se descartará.
3. Las Opciones, funciones de control que incluyen recursos para marcas de tiempo, seguridad y encaminamiento especial, etc.
4. La Suma de Control de Cabecera, una verificación de que la información contenida en el paquete ha sido transmitida correctamente al incluir cierta información redundante que contiene una suma de comprobación de los campos de cabecera. Si al recibir un nuevo paquete la suma de comprobación es inválida, el paquete es descartado inmediatamente por la entidad que detecta el error.

En resumen, las características principales de este protocolo son las siguientes:

- Protocolo orientado a no conexión.
- Segmentación y reensamblado de paquetes si es necesario.
- Plan de numeración y direccionamiento por medio de direcciones lógicas IP de 32 bits.
- Si un paquete no es recibido, este permanecerá en la red durante un tiempo finito.
- Realiza el "mejor esfuerzo" para el transporte de paquetes pero no garantiza su entrega, no hay corrección de errores ni control de la congestión.
- Tamaño máximo del paquete de 65635 bytes.
- Solo se realiza la verificación por suma al encabezado del paquete, no a los datos que éste contiene.

Direcciones IP

Una dirección IP es un número de 32 bits que identifica de manera lógica y jerárquica a un dispositivo dentro de una red que utilice el protocolo IP. Dicho número no se ha de confundir con la dirección MAC, que es un número asignado a la tarjeta de red del propio ordenador e impuesto por el fabricante de la tarjeta.

La dirección IP está dividida internamente en dos partes: un ID de red, que identifica un único segmento de red dentro de un conjunto de redes, o red de redes, y un ID de host, el cual identifica un nodo TCP/IP dentro de cada red.

Las direcciones IP se clasifican en:

- Direcciones IP públicas, visibles desde cualquier otro ordenador conectado a Internet. Para conectarse a Internet es necesario tener una dirección IP pública, la cual identifica el equipo en Internet y es única.
- Direcciones IP privadas, o reservadas, visibles para hosts dentro de una red de área local (LAN - Local Area Network), dentro de una empresa o una red doméstica.

A su vez, las direcciones IP pueden ser:

- Direcciones IP estáticas, o fijas pues no cambian al reconectar Internet, son propias de sitios de Internet que por su función necesitan estar conectados permanentemente a la red. Es el caso de servidores de correo, DNS, FTP públicos y grandes servidores de páginas web con objeto de que sean siempre localizables por los usuarios de Internet.
- Direcciones IP dinámicas, o una IP distinta asignada cada vez que un host establece una conexión. Esta dirección está activa para nosotros exclusivamente durante la sesión establecida con el ISP, o proveedor de acceso a internet, desde el momento de la conexión hasta la desconexión. Esta dirección IP nos permite, por supuesto, poder acceder a toda la red.

Protocolo ICMP

El protocolo IP al no proporcionar mecanismos de control, se hizo necesario implementar el protocolo ICMP (Internet Control Message Protocol). Este protocolo, definido en la RFC 792, no corrige errores, sólo se ocupa de informar sobre las incidencias producidas en la red ofreciendo a los nodos intermedios el envío de mensajes de control a los equipos que enviaron la información.

Cabe decir que dicha implementación sólo informa de errores al origen del datagrama, pues es la única fuente fiable conocida ya que el destino puede existir o estar inactivo en un momento dado. De este modo se evita la sobrecarga de la red con tráfico innecesario.

Debido a que el protocolo IP no es fiable puede darse el caso de que un mensaje ICMP se pierda o se dañe. Si esto llega a ocurrir no se creará un nuevo mensaje ICMP sino que el primero se descartará sin más.

Los mensajes ICMP se envían encapsulados dentro del campo de datos de un datagrama IP, como se muestra en la siguiente ilustración.

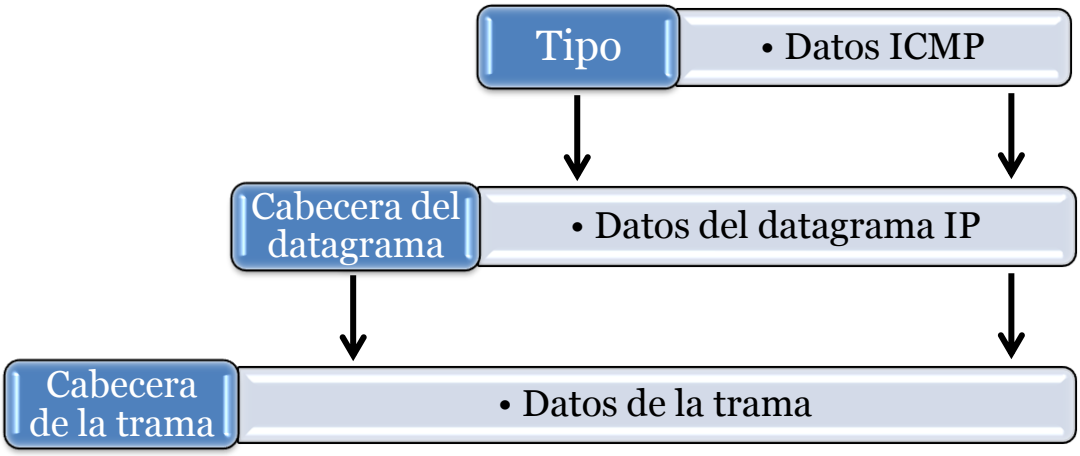

Figura 8. Encapsulado de mensajes ICMP

Los mensajes ICMP comienzan con un campo de 8 bits que identifica al tipo de mensaje ICMP. A continuación se describen los mensajes ICMP más comunes:

- Confirmación de host (Echo Request/ Echo Response): Se utiliza para comprobar la existencia de comunicación entre 2 hosts. Estos mensajes de eco son la base del funcionamiento de los comandos ping y traceroute.
- Destino inaccesible (Destination Unreachable): Informa a un equipo que el destino o servicio es inalcanzable. Entre los códigos de destino inalcanzable se encuentran: 0 = red inalcanzable y 1 = host inalcanzable, respuesta de un encaminador cuando no puede enviar un paquete o cuando el último encaminador de la ruta que lleva desde un origen a un destino recibe un paquete para el cual posee una ruta conectada pero no puede enviar el paquete al host destino en la red conectada; 2 = protocolo inalcanzable y 3 = puerto inalcanzable, utilizados por un equipo final para indicar que el segmento TCP o el datagrama UDP en un paquete no puede ser enviado al servicio de capa superior o porque la seguridad del equipo no permite el acceso al servicio.
- Redireccionamiento de ruta (Redirect): Un encaminador puede usar un mensaje de redireccionamiento de ICMP para notificar a los equipos de una red acerca de una mejor ruta disponible para un destino en particular.
- Disminución del tráfico desde el origen (Source Quench): Informa al origen de que disminuya la velocidad a la que envía los datagramas porque hay congestión ya que los datagramas llegan demasiado rápido para ser procesados por un nodo de la red que no posee suficiente espacio en búfer para almacenar los paquetes entrantes y procede al descarte de éstos.
- Tiempo excedido (Time Exceeded): Indica al host de origen que el tiempo de vida (TTL – Time To Live) del datagrama IP ha expirado.

Comando Ping

Ping, acrónimo de "Packet Internet Groper", es un comando o herramienta de diagnóstico para verificar la conectividad de un host local con un equipo remoto contemplado en una red de tipo TCP/IP. Ping, gracias al uso del protocolo ICMP nombrado en el apartado anterior, se basa en el envío de un mensaje "IP ICMP Echo Request" por parte del origen cuando inicia el ping, y la respuesta "IP ICMP Echo Reply", por parte del destino cuando recibe el mensaje recibido. Además, esta herramienta puede ser ejecutada desde múltiples sistemas operativos.

El comando ping acepta dos formas de uso: ping [dirección IP] o ping [nombre]. Cuando se utiliza la dirección IP, el paquete ICMP es creado directamente con la IP del host que se indica, por lo contrario, cuando se usa un nombre, el host, antes de crear el paquete ICMP hace una consulta DNS a dicho nombre con el objetivo de averiguar su IP y una vez averiguada se crea el paquete ICMP utilizando como dirección IP de destino la que se obtiene en la respuesta del servidor DNS al realizar la consulta.

El comando ping presenta las siguientes posibilidades más comunes:

- -t: Efectúa el ping enviando mensajes de solicitud de eco al destino especificado hasta que se le interrumpa.
- -a: Resuelve direcciones a nombres de host.
- -n *cantidad*: Especifica el número de solicitudes de eco a enviar. El valor predeterminado es 4.
- -l *tamaño*: Especifica la longitud en bytes, del campo de datos en los mensajes de solicitud de eco a enviar. El valor predeterminado es 32. El tamaño máximo es 65.527.
- -f: Establece la no fragmentación del paquete, siendo solo posible en la IPV4.
- -i *TTL*: Especifica el valor del campo TTL del encabezado IP del mensaje de solicitud de eco enviado.
- -v *TOS*: Determina el valor del campo TOS (Tipo de servicio) del encabezado IP del mensaje de solicitud de eco enviado, siendo solo posible en la IPV4. TOS se especifica como un valor decimal que oscila entre 0 y 255, siendo 0 el valor predeterminado.
- -r *Recuento*: Registra la ruta de saltos que toma el mensaje de solicitud de eco y el correspondiente mensaje de respuesta de eco, siendo solo posible en IPV4. El recuento debe ser un mínimo de 1 y un máximo de 9.
- -s *Recuento*: Registra la hora de llegada para el mensaje de solicitud de eco y respuesta de eco correspondiente para cada salto, siendo solo posible en IPV4. El recuento debe ser un mínimo de 1 y un máximo de 4.
- -j *Lista de hosts*: Establece la opción ruta de origen variable en la lista de hosts.
- -k *Lista de hosts*: Especifica la opción ruta de origen estricta en la lista de hosts.
- -w *Tiempo de espera*: Tiempo de espera de respuesta en milisegundos.

3.1.2. Dominio

Un dominio o nombre de dominio es el nombre que identifica un sitio web. Cada dominio tiene que ser único en Internet. Un solo servidor web puede servir múltiples páginas web de múltiples dominios, pero un dominio sólo puede apuntar a un servidor.

El principal propósito de los nombres de dominio y del sistema de nombres de dominio (DNS – Domain Name Server), es la traducción de direcciones, de cada nodo activo en la red, a términos memorizables y fáciles de encontrar proporcionando la utilidad de poder moverse por los distintos lugares geográficos en la red Internet aún cuando esta movilización implique tener una dirección IP distinta. Sin la ayuda del sistema de nombres de dominio, los usuarios de Internet tendrían que acceder a cada servicio web utilizando la dirección IP del nodo.

A continuación se presentan los tipos de dominio más comunes de la amplia variedad que existe en la actualidad:

- -.biz, para negocios.
- -.com, para sitios comerciales.
- -.es, para servicios de España.
- -.edu, para servicios de Educación.
- -.gov y .gob, para organismos gubernamentales o entidades públicas.
- -.info, para información.
- -.int, para entidades internacionales, organizaciones como la ONU.
- -.mil para Dependencias Militares Oficiales de los Estados.
- -.name, para aspectos personales.
- -.net, para infraestructura de sistemas y redes.
- -.org, para organizaciones.

3.1.3. DDNS

El Sistema Dinámico de Nombres de Dominio o DDNS es un servidor conectado a Internet que dispone de una base de datos que vincula una dirección IP pública con un nombre de dominio.

Es útil si se desea configurar un servidor web, ftp, montar una VPN, etc. ya que hay que tener localizado nuestro router en Internet para disponer de acceso y como la mayoría de operadoras ofrece a sus clientes direcciones IP dinámicas, de esta forma mediante la función DDNS, tenemos la posibilidad de configurar el router y asociar un nombre de dominio a una dirección IP sin preocuparnos de que la dirección IP pública cambie cada vez que nos conectemos, se produzca un corte de línea o se resetee el router.

Los servidores más conocidos son: www.dyndns.org, www.no-ip.com, etc.

Capítulo 4

Desarrollo

4.1. Estructura de la aplicación

Esta parte se centra básicamente en la distribución y desarrollo del proyecto y se hace una inmersión en las distintas clases creadas dentro de cada paquete de Java, desde el panel de NetBeans IDE, el cual mantiene la organización de los elementos que constituyen la aplicación y que se puede observar en la siguiente captura.

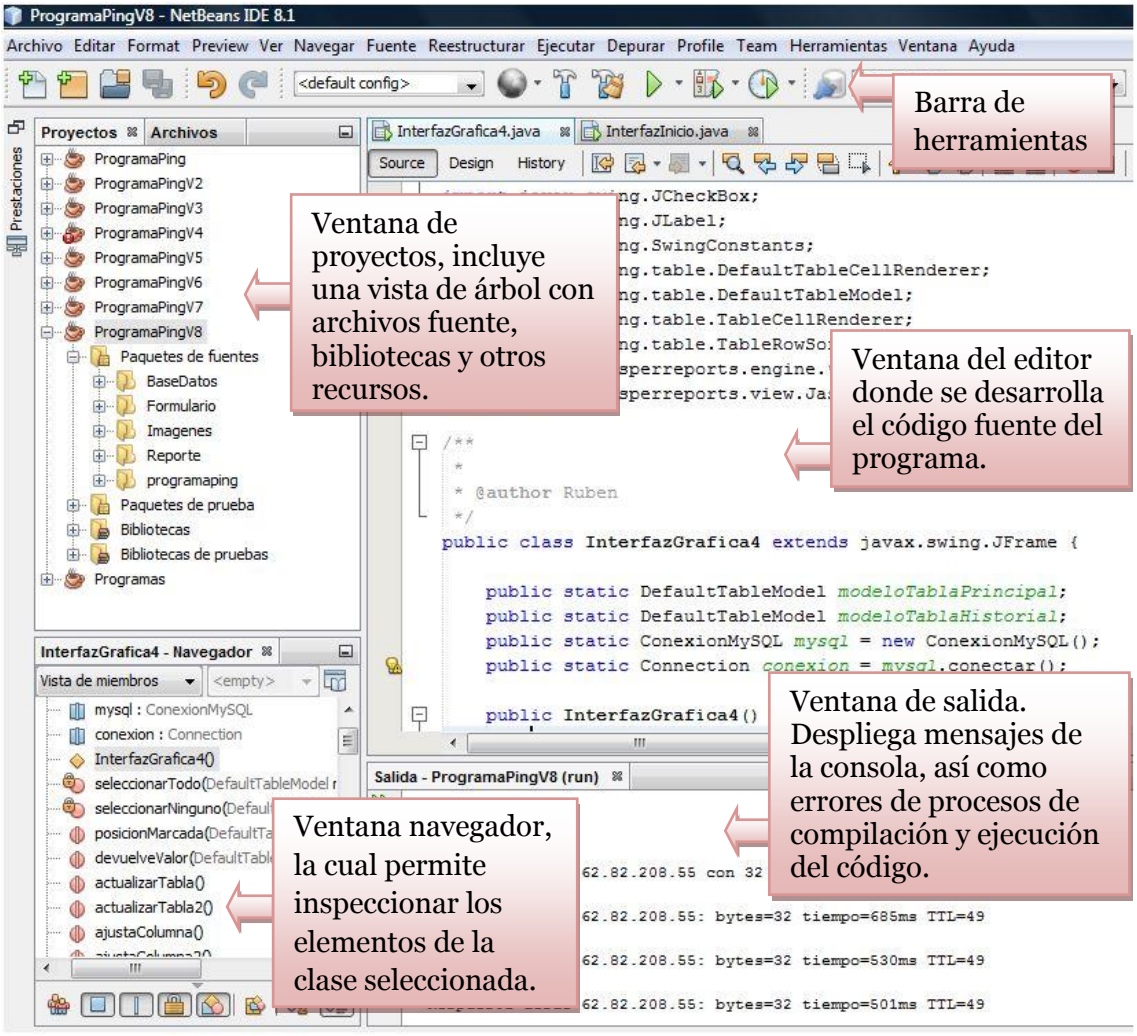

Ilustración 14. Distribución del proyecto en NetBeans IDE

4.2. Ápices sobre la implementación

A continuación, en los apartados siguientes se detallan pequeñas sentencias de código que serán clave para la realización de las partes más destacadas de la aplicación.

4.2.1. Implementación Ping

Aunque en la práctica no es muy aconsejable utilizar instrucciones para llamar a ejecutables externos puesto que se pierde el objetivo básico de Java, que es la ejecución del programa en cualquier plataforma, por motivos de fiabilidad y eficacia se decide llamar a la clase "Runtime" de Java y efectuar la ejecución del comando Ping del sistema operativo Windows por medio de la siguiente instrucción.

```
Process ejecutarComando = Runtime.getRuntime().exec("ping "+ip);
```

Así pues, para poder leer la respuesta o salida del ejecutable externo, se debe usar el "Process" que devuelve la instrucción con el método "Runtime.exec()" y posteriormente, mediante el método "getInputSream()", se crea el flujo de datos que inicializa la lectura y conversión de secuencias de bytes en secuencias de caracteres.

Aún así, leer secuencias de caracteres es algo enrevesado, por ello, se utiliza la clase "BufferedReader" que contiene el método "readLine()" que permite leer una línea de texto en forma de "String".

```
BufferedReader flujoEntrada = new BufferedReader(new
InputStreamReader(ejecutarComando.getInputStream()));

      String linea;
      while((linea = flujoEntrada.readLine()) != null) {
            System.out.println(linea);
      }
```

4.2.2. Implementación pestaña Consola

Para que la aplicación dote de una pestaña con la consola de Java, o salida de sucesos cronológicos del programa, con el objetivo de poder visualizar los datos que se procesan y de este modo identificar si el proceso a tratar se codifica correctamente, se utiliza la paleta de componentes visuales de "Swing" y "AWT", que ofrece el editor NetBeans IDE para crear interfaces gráficas de usuario (GUI) con espaciado y alineación automática.

Para ello, se ingresa en la interfaz un nueva contenedor de tipo "JScrollPane" que proporcione una ventana desplegable para contenido dinámico. Una vez creada, en el interior se añade desde la paleta un "JTextArea", que no es más que un área de varias líneas que muestra texto sin adornos.

Una vez efectuada esta fase, se crea la instrucción que llama al método "append()" para que anexe de forma automática texto al final del contenido del área de texto.

Además, mencionar que el parámetro que se le pasa al método es de tipo “String” y que la existencia de los caracteres “\n” permite crear un salto de línea automático.

Esta instrucción se añade debajo del código implementado en el apartado anterior y queda de la siguiente forma.

```
InterfazGrafica.TextAreaConsola.append("\n"+linea);
```

Este procedimiento, es gracias a la aportación de pequeños tutoriales de la web “Chuwiki”, donde el contenido de ésta lo difiere la gente que lo visita. En concreto, la fuente proviene de http://chuwiki.chuidiang.org/index.php?title=JScrollPane y, para finalizar con este apartado, el aspecto visual de la pestaña Consola en la aplicación es la mostrada a continuación.

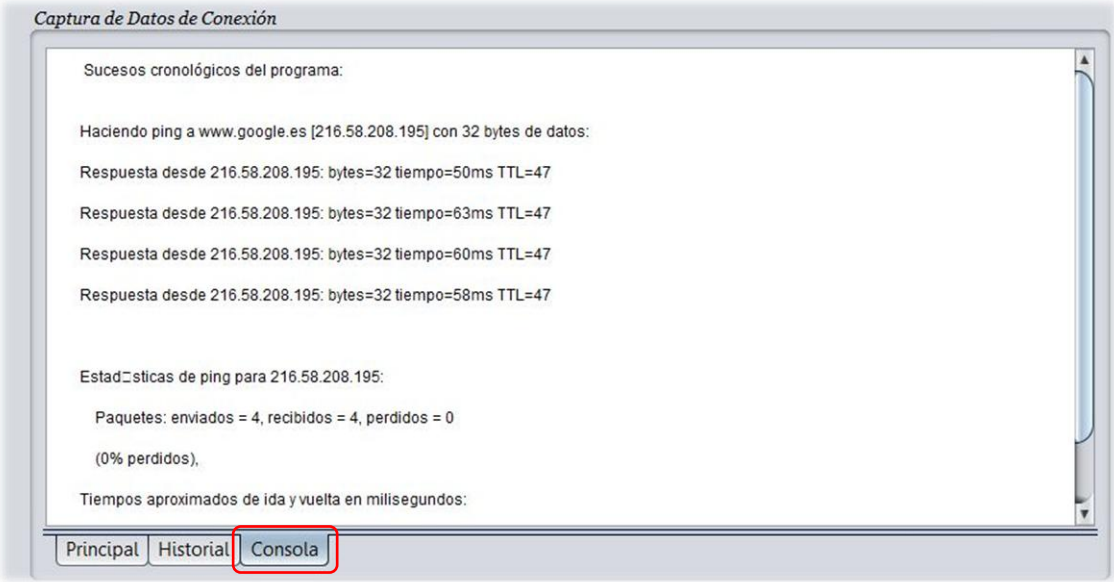

Ilustración 15. Pestaña Consola de la aplicación.

4.2.3. Implementación Recordatorio

La implementación de esta clase requiere dos partes, por un lado trabajar con fechas haciendo uso de la clase “java.util.Calendar”, y por otra parte el temporizador.

“Calendar” es una clase abstracta con la que se puede representar y manejar fechas , o objetos de tipo Date, así como la modificación y obtención de sus datos enteros tales como “YEAR”, “MONTH”, “DAY”, etc. Como es una clase abstracta, no puede ser instanciada con un constructor típico de tipo new Calendar(), por lo tanto instanciaremos un objeto de clase “Calendar”, mediante el método “getInstance()”, cuyos campos se inicializan con la fecha y la hora actuales, o bien, mediante el constructor de la clase “GregorianCalendar”, subclase de “java.util.Calendar” que facilita un calendario estándar utilizado en la mayor parte del mundo.

En este caso se utiliza la instancia del objeto mediante el método “getInstance()”.

```
Calendar calendario = Calendar.getInstance();
```

Por otro lado, como no se desea obtener y sí modificar los atributos día del mes, mes, hora, minutos y segundos, se emplea el método "set(int field, int value)", donde el parámetro de tipo entero "field" es el atributo de la fecha que queremos modificar, y el parámetro de tipo entero "value" el nuevo valor a asignar.

```
calendario.set(Calendar.DAY_OF_MONTH, dia);
calendario.set(Calendar.MONTH, mes);
calendario.set(Calendar.HOUR_OF_DAY, hora);
calendario.set(Calendar.MINUTE, min);
```

Una vez creado el calendario y modificado sus parámetros se procede a convertir "Calendar" a un objeto de tipo "Date" para poder dar una salida legible.

```
Date time = calendario.getTime();
```

La siguiente parte a realizar es la tarea periódica basada en la espera temporizada. Para poder manejar una tarea de manera automática y repetitiva se tiene que crear una clase que herede de la clase "TimerTask" y sobrescribir el método "run()".

Expresar que "TimerTask" es una clase abstracta que define una tarea planificada por la ejecución de un "timer".

```
public class RemindTask extends TimerTask {

     @Override
     public void run() {
          System.out.format("La tarea programada empieza… ");
          Ping.ping(dir_ip, num_intento);
          //timer.cancel();
     }
}
```

4.2.4. Implementación Tiempo de Repetición

Una vez implementado el apartado anterior, se necesita hacer uso de un método llamado "schedule()" de la clase "Timer". A este método se le pasan tres parámetros:

1. La clase creada que hereda de la clase TimerTask: "new RemindTask()".
2. La clase Date que hará que se ejecute el temporizador en el calendario especificado: `Date time = calendario.getTime();`
3. El tiempo de ejecución del timer en milisegundos: `int tiempo1min = 60000;`

Por tanto, la instrucción queda de la siguiente forma:

```
Timer temporizador = new Timer();
temporizador.schedule(new RemindTask(), time, tiempoDeRepeticion);
```

Se debe tener en cuenta que el tercer parámetro de esta instrucción no se debe programar con un tiempo de ejecución igual a 0, ya que al compilar habría un error. Como solución no pasar esta variable en el método y solucionado.

```
temporizador.schedule(new RemindTask(), time);
```

4.2.5. Implementación Conexión con la base de datos.

En este apartado se explica cómo conectar la aplicación Java con la base de datos, en este caso, de tipo MySQL utilizando el IDE NetBeans.

Primero que nada, en nuestro equipo debe estar instalado MySQL, o tener una base de datos que permita la conexión remota, ya que si no se tiene instalado MySQL ni se dispone de una conexión remota entonces se recomienda la instalación de paquetes como WampServer, MAMP, XAMP, etc.

Como ya se mencionó anteriormente, en el capítulo Tecnologías de la memoria, estos paquetes de software de fácil instalación, contienen un conjunto de herramientas, más detalladamente MySQL, APACHE y PHP, que brindan la posibilidad de crear y gestionar tus bases de datos mediante una interfaz muy sencilla de utilizar llamada "phpMyAdmin". Para esta aplicación se utiliza el paquete de software WampServer y la creación de la base de datos con phpMyAdmin.

Acto seguido, se muestra la creación de la base de datos al seleccionar la pestaña "Bases de Datos" e introducir el nombre de la base de datos que se desee en el campo de texto. En nuestro caso, ésta se denomina "programa_ping".

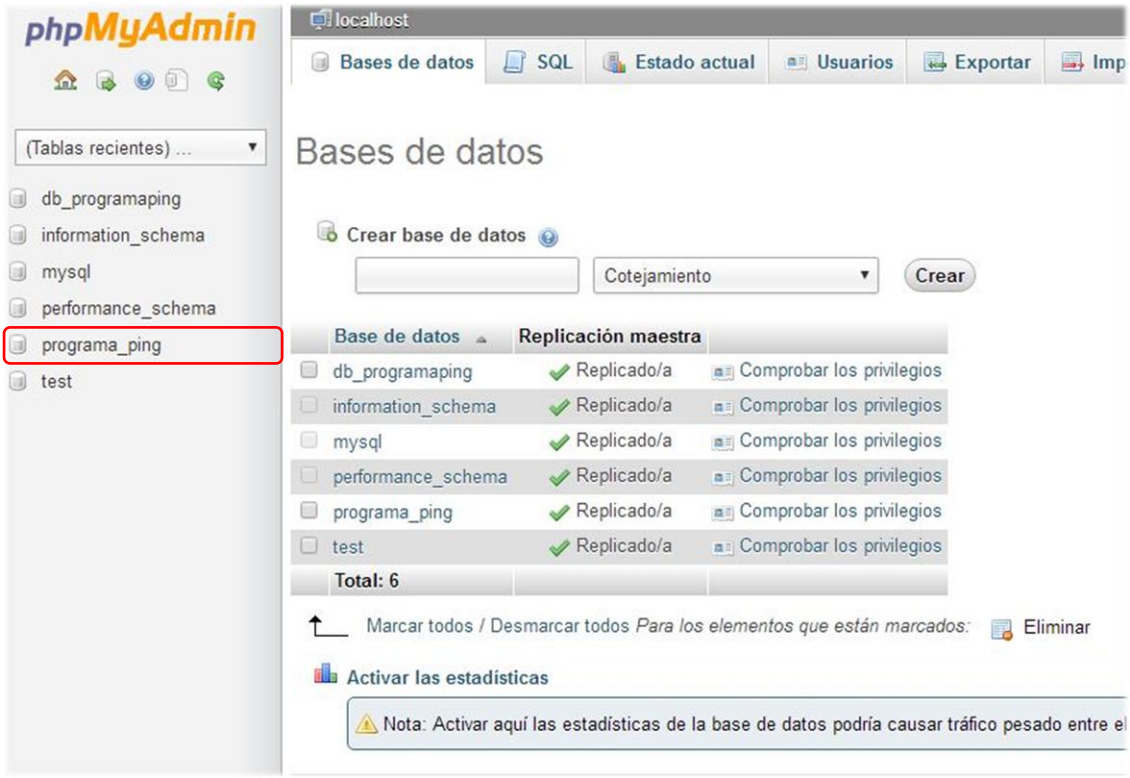

Ilustración 16. Creación BBDD con phpMyAdmin

Posteriormente, se crean las tablas con los campos que necesite la aplicación. En este caso, se crean dos tablas de datos, llamadas "tabla_ping", para la monitorización principal de la red con los datos más actualizados, y "tabla_ping2" que contendrá únicamente el historial de la monitorización efectuada por la tarea programada.

La diferencia radica en la estructura de ambas tablas partiendo de qué tabla exige un mayor o menor número de campos o columnas según la necesidad. Entre los campos o columnas a destacar se encuentra "SELECCIÓN" e "ID".

El campo "SELECCIÓN" de tipo "Boolean" es creado con la intención de que las filas de cada dirección IP queden permanentemente seleccionadas con la virtud de que, cuando inicie la aplicación o se ejecute un ping con tiempo de repetición, no haya que volver a realizar el proceso de selección de filas.

Ahora bien, al ser obligatorio que en cada tabla se presencie un campo más importante que el resto, "ID" será el campo representativo y elegido por la tabla "tabla_ping" para ser el campo que guarde datos irrepetibles. Esto se realiza activando la clave Primaria en dicho campo y puede servir para agilizar la consulta de sentencias SQL como actualizar y eliminar filas específicas como la siguiente.

```
String sentenciaSQL = "";
sentenciaSQL = "DELETE FROM tabla_ping WHERE ID ='"+ indice +"'";
```

Como se puede visualizar en la captura, el campo con la clave Primaria en activo habitualmente se manifiesta como subrayado en la estructura de la tabla. Se debe tener en cuenta que solo un único campo puede ser el campo clave.

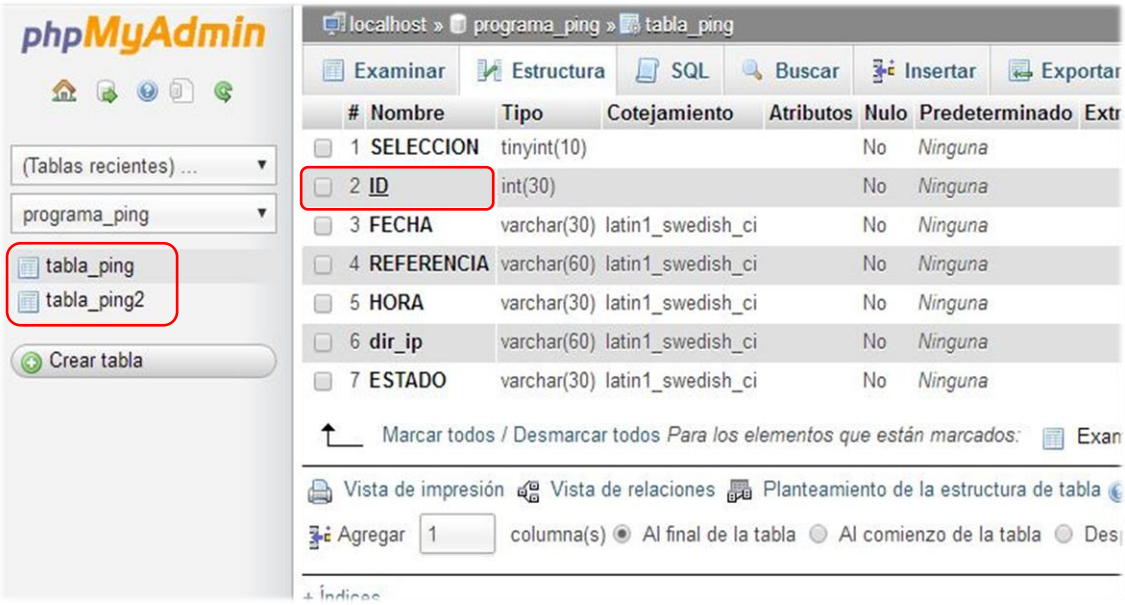

Ilustración 17. Estructura de la tabla "tabla_ping".

Por otra parte, el campo idóneo que será etiquetado como campo clave en "tabla_ping2" será el campo "HORA". La razón de esta acción es porque en esta tabla se omite el campo "ID", ya que no hay necesidad de modificar sus datos, y en el caso de que se quiera eliminar filas específicas será el campo idóneo al no encontrarse dos mismas horas.

```
String sentenciaSQL = "";
sentenciaSQL = "DELETE FROM tabla_ping2 WHERE HORA ='"+ hora +"'";
```

En este caso, hay que tener en cuenta que se trata de la tabla que pertenece al historial, y por ello se prescinde de direccionar al campo "HORA" para direccionar al campo "dir_ip" en la sentencia SQL, ya que es inconcebible que se presencien dos mismas horas para dos pings ejecutados en momentos distintos.

La razón de esta acción es no eliminar varias filas sin tener conocimiento de causa, pues en el momento de eliminar una fila puede hallarse dos o más direcciones IP o dominios iguales en la tabla del historial, si se tiene en cuenta que cada vez que se ejecuta una tarea programada o hay una nueva repetición de la tarea, se ubica una nueva entrada de datos a la tabla haciendo referencia al mismo identificativo.

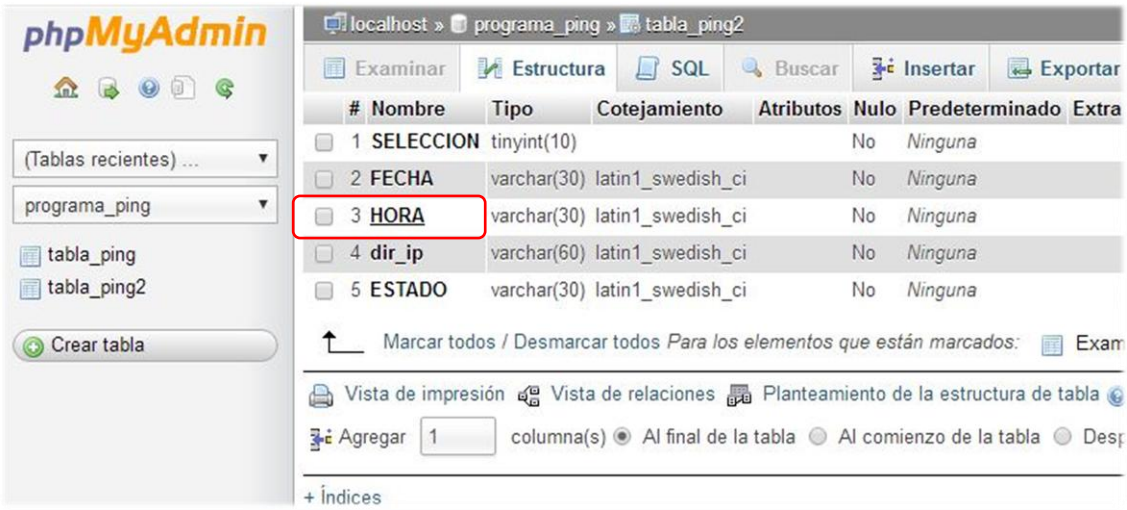

Ilustración 18. Estructura de la tabla "tabla_ping2".

Posteriormente, para conectar la aplicación con la base de datos en NetBeans IDE es imprescindible añadir la librería "java.sql.*", que contiene las clases que conforman el API JDBC y servirá para cargar los drivers, además de descargarnos el conector. Para cada tipo de motor de base de datos se necesita su respectiva librería o conector que, de forma habitual, se ofrece en la página principal del fabricante.

En este caso se utiliza una base de datos de tipo MySQL y se añade el conector con extensión ".jar" que contiene el driver de MySQL (mysql-connector-java-5.1.6-bin.jar). Esta acción se realiza al presionar con el botón derecho sobre "Biblioteca" y al seleccionar en la ventana desplegable la opción "Agregar archivo JAR/Carpeta" como se observa en la captura.

Si todo va bien, se obtiene una nueva librería externa con el nombre del conector añadido.

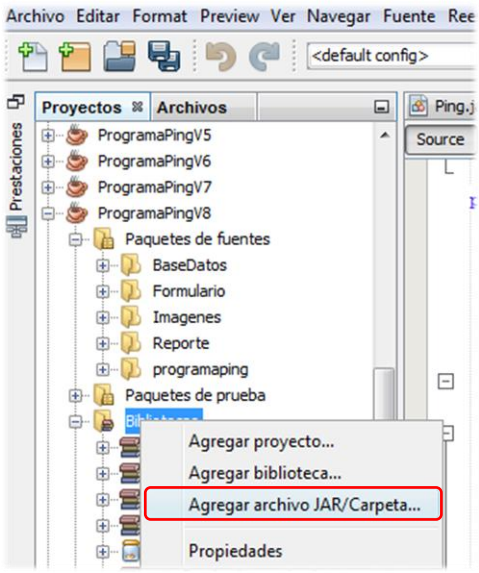

Ilustración 19. Agregar JAR de MySQL

Una vez hemos adquirido los ingredientes para realizar la conexión, se crea la clase para establecer ésta. La clase constará de cuatro atributos que sirven para especificar la ruta de la base de datos junto su nombre, así como detallar el nombre de usuario y la contraseña. En nuestro caso el usuario y la contraseña son por defecto.

```
private static String db = "programa_ping";
private static String url = "jdbc:mysql://localhost/"+db;
private static String user = "root";
private static String pass = "";
```

Después, gracias al método "DriverManager.getConnection()", pasándole como argumentos el servidor, el usuario y la contraseña, se termina de crear una conexión abierta para realizar sentencias SQL al controlador de la base de datos especificado en la URL del JDBC.

```
Connection link = null;
try {
  Class.forName("org.gjt.mm.mysql.Driver");
  link = DriverManager.getConnection(this.url,this.user,this.pass);
}catch(Exception e) {
  JOptionPane.showMessageDialog(null,e);
}
return link;
```

4.2.6. Implementación pestaña Principal e Historial

En el momento de elaborar el formulario o la interfaz gráfica de usuario (GUI) de la aplicación, además de emplear botones (JButton), campos de texto (JTextField) o etiquetas (JLabel) dentro del contenedor de componentes (JFrame), se tropieza con la necesidad o el requerimiento de acumular los datos en una tabla (JTable).

Un "JTable" es un componente visual de Java que permite dibujar una tabla con sus respectivas filas y columnas e ingresar, en el interior de las celdas, datos ingresados por nosotros mismos o provenientes de una base de datos.

Una de las formas de llenar un "JTable" y servirse de todas sus funcionalidad para modificar los datos en su interior, así como añadir filas o columnas y darle a cada columna el nombre que se desee, es a partir de la instancia de un modelo de tabla (DefaultTableModel). El aporte para llevar a cabo este propósito se presenta en la página web http://www.chuidiang.com/java/tablas/tablamodelo/tablamodelo.php, entre otras.

Para esta coyuntura se añaden los objetos "DefaultTableModel" como atributos a la interfaz gráfica que hereda de la biblioteca gráfica "javax.swing", que permite incluir componentes y controles visuales tales como cajas de texto, botones, desplegables y tablas en una interfaz gráfica de usuario.

```
public class InterfazGrafica extends javax.swing.JFrame {

      public static DefaultTableModel modeloTablaPrincipal;
      public static DefaultTableModel modeloTablaHistorial;
```

Acto seguido, creamos un método para cada tabla. Dentro de estos métodos se implementará el código indispensable para crear y dar nombre al encabezado de las columnas gracias a un vector o "array" de objetos, pasándole el número de columnas que contenga cada tabla. La diferencia entre ambos métodos reside en la sustitución de "modeloTablaPrincipal" y "tablaCaptura" por "modeloTablaHistorial" y tablaHistorial".

```
public static void mostrarDatos1() {

      modeloTablaPrincipal = new DefaultTableModel();

      modeloTablaPrincipal.addColumn("☑");
      modeloTablaPrincipal.addColumn("ID");
      modeloTablaPrincipal.addColumn("Fecha");
      modeloTablaPrincipal.addColumn("Hora");
      modeloTablaPrincipal.addColumn("Referencia");
      modeloTablaPrincipal.addColumn("Dirección IP");
      modeloTablaPrincipal.addColumn("Estado");

      tablaCaptura.setModel(modeloTablaPrincipal);

      String sentenciaSQL = "";
      sentenciaSQL = "SELECT * FROM tabla_ping";

      Object[] registro = new Object[7];
```

Una vez llegado hasta aquí, es el momento de realizar transacciones con la base de datos con el fin de efectuar una inserción, consulta, modificación o borrado de datos. Esto se alcanza con los métodos "createStatement()" y "prepareStatement()". Gracias a la contribución de páginas como http://chuidiang.org/content/transacciones-con-base-de-datos se puede encontrar la distinción entre ambas.

En este caso, se elaboran operaciones en la base de datos creando una instrucción que encapsula la sentencia SQL con "Statement", a partir de la conexión. Después, se ejecuta la consulta "SELECT" con el método "executeQuery()" que devuelve un objeto "ResulSet" que encapsula el conjunto de valores de cada campo gracias al método "getX("campo"), correspondiendo "X" al tipo de valor SQL cuando se creó la estructura de la tabla con phpMyAdmin.

Es preciso encerrar la transacción dentro de un bloque "try - catch" para la captura de posibles excepciones que se puedan originar tras la ejecución.

```
try {
     Statement st = conexion.createStatement();
     ResultSet rs;
     rs = st.executeQuery(sentenciaSQL);

     while(rs.next()) {
          registro[0] = rs.getBoolean("SELECCION");
          registro[1] = rs.getInt("ID");
          registro[2] = rs.getString("FECHA");
          registro[3] = rs.getString("HORA");
          registro[4] = rs.getString("REFERENCIA");
          registro[5] = rs.getString("dir_ip");
          registro[6] = rs.getString("ESTADO");
          modeloTablaPrincipal.addRow(registro);
     }
```

```
            rs.close();
            tablaCaptura.setModel(modeloTablaPrincipal);

        }catch(Exception e) {
            JOptionPane.showMessageDialog(null,"Error mostrardatos.");
        }
```

Para concluir, con la idea de mostrar los datos cuando se inicialice la aplicación se añade al constructor de la interfaz gráfica creada los métodos implementados así, desde el primer momento, los datos serán visibles en las tablas.

```
public InterfazGrafica() {

     mostrarDatos1("");
     mostrarDatos2("");
}
```

4.2.7. Implementación JCheckBox

Con el inconveniente de que una tabla de datos de Java, o "JTable", puede llegar a ser visualmente tediosa o, en vez de utilizar las celdas de la tabla tal cual son, surja la necesidad de cambiar aspectos significativos de ésta, se requiere sobrescribir métodos de superclases o clases padre que configuren los tipos de componentes que integran las celdas.

En este apartado se decide convertir la primera columna de las tablas de la base de datos en un "JCheckBox", o casilla de verificación, de tipo "Boolean".

Este complemento se logra gracias al aporte de páginas web como http://www.jc-mouse.net/java/jcheckbox-dentro-jtable-con-netbeans, que alegan el código básico para alcanzar este fin.

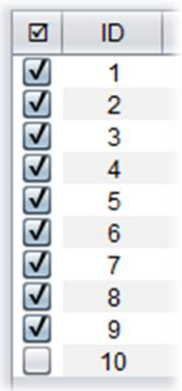

Ilustración 20. JCheckBox

La implementación para este propósito requiere de 2 fases:

1. La manipulación de los valores de la celda.

   En este punto, la meta es indicar el tipo de componente que debe utilizar "JTable" cuando se introducen los datos. Esto se logra sobrescribiendo el método "getTableCellEditorComponent()" al crear una nueva clase, "CheckBox_Editor", que herede de "DefaultCellEditor" e implemente "TableCellRenderer".

```
class CheckBox_Editor extends DefaultCellEditor implements
TableCellRenderer {

    public Component getTableCellEditorComponent(JTable table,
    Object value, boolean isSelected, int row, int column)

    { ... }

}
```

2. Pintar el componente "JCheckBox", o casilla de verificación de Java.

   Para ser pintado el componente por "JTable" se requiere realizar una nueva clase que herede de "JCheckBox" e implemente "TableCellRenderer" para sobrescribir el método getTableCellRendererComponent(). Gracias a este método se convierte y devuelve el valor del parámetro que recibe "Object value" en un "Component" de Java según si esta seleccionado o no.

```
class CheckBox_Render extends JCheckBox implements
TableCellRenderer {

    public Component getTableCellRendererComponent(JTable table,
    Object value, boolean isSelected, boolean hasFocus, int row,
    int column) {
          boolean b = ((Boolean) value).booleanValue();
          ((JCheckBox) component).setSelected(b);
          return ((JCheckBox) component);
    }

}
```

Para finalizar, una vez se han implementado estas dos clases, se realizan las llamadas creando un objeto "JCheckBox" y se especifica la columna indicada donde se desee la ubicación de éste dentro del método "getColumn()".

```
JCheckBox check = new JCheckBox();

tablaCaptura.getColumnModel().getColumn(0).setCellEditor(new
DefaultCellEditor(check));
tablaCaptura.getColumnModel().getColumn(0).setCellRenderer(new
CheckBox_Render());
```

4.2.8. Implementación Ordenar y Filtrar tabla

Una propiedad que se suma a nuestra tabla es la cualidad de poseer columnas ordenables, tanto de forma ascendente como descendente alternativamente. Esto mejora la percepción visual de los registros almacenados en la tabla.

Para implementar esta acción hay que hacer uso de la clase "TableRowSorter" de Java, la cual mediante la instancia de ésta y pasándole el "JTable" como parámetro, el usuario tendrá la aptitud de ordenar visualmente la tabla clicando sobre la cabecera de las columnas.

Ilustración 21. Orden ascendente y descendente.

Esta información es facilitada gracias a la concurrencia de páginas web como http://chuwiki.chuidiang.org/index.php?title=JTable:_Ordenar_y_filtrar_filas.

```
TableRowSorter ordena = new TableRowSorter(modeloTablaPrincipal);
tablaCaptura.setRowSorter(ordena);
```

Por otra parte, si se desea hacer un filtrado exhaustivo de los resultados se exige crear un "JTextField" para almacenar en éste aquello que se desee buscar. Una vez creado basta con pasarle una variable de tipo "String" al método que muestra los datos de la tabla para que con la condición "if - else" cambie de sentencia. Esta variable recoge el valor almacenado en el campo de texto donde se ha introducido la búsqueda y es el botón "Filtro" quien llama al método.

El método con la implementación mencionada anteriormente es el siguiente.

```
public static void mostrarDatos1(String valor) {
    String sentenciaSQL = "";
    if(valor.equals("")) {
         sentenciaSQL = "SELECT * FROM tabla_ping";
    } else {
         sentenciaSQL = "SELECT SELECCION,ID,FECHA,HORA,REFERENCIA,
         dir_ip,ESTADO FROM tabla_ping WHERE CONCAT (SELECCION,'
         ',ID,' ',FECHA,' ',HORA,' ',REFERENCIA,' ',dir_ip,'
         ',ESTADO) LIKE '%" +valor+ "%'";
    }
```

4.2.9. Implementación Interfaz Inicio

La interfaz de la aplicación es la interacción directa que hay entre ésta y el usuario, y por este motivo, es importante cuidar el aspecto visual con el fin de alcanzar un estilo estéticamente satisfactorio. Así pues, el trabajo del diseñador de la aplicación consiste en interpretar su propia visión para conseguir, además de un uso sencillo y situar los elementos bien organizados, la distinción sobre las demás aplicaciones.

Con esto se decide crear una nueva interfaz que se basará en una ventana de desvanecimiento conforme una barra de progreso vaya incrementando su valor.

Para este propósito, se elabora un hilo de ejecución que se ocupe de esta tarea con la creación de una nueva clase que herede de la clase "Thread" que tendrá un "JProgressBar" como atributo.

```
public class InicioCargar extends Thread {
    JProgressBar BarraDeProgreso;

    public InicioCargar(JProgressBar barraDeProgreso) {
        super();
        this.BarraDeProgreso = barraDeProgreso;
    }

    public JProgressBar getBarra() {
        return BarraDeProgreso;
    }
```

Posteriormente, se crea el método "pausa()", que provoca la detección de la ejecución durante el tiempo indicado en ms, y se redefine el método run() que será invocado cuando se inicie el "thread" mediante la llamada al método "start()" de la clase "Thread".

```
public void pausa(int mlsegundos) {

    try {
        Thread.sleep(mlsegundos);
    } catch (Exception e) {
        JOptionPane.showMessageDialog(null,"Error Pausa().");
    }
}

@Override
public void run() {

    for(int i=1; i<=100; i++) {
        BarraDeProgreso.setValue(i);
        pausa(40);  //Una pausa de 1000 ms = 1 segundo
    }
}
```

Por otro lado, se crea un nuevo formulario con una imagen de fondo importada en el programa. Este formulario será la ventana de desvanecimiento y el inicio de la aplicación, pues a partir de este formulario, una vez la barra de progreso se haya completado, se desplegará el formulario principal o panel de control.

Siguiendo con el proceso, en el constructor del formulario se le pasa el método "iniciar()". Dicho método contiene la instancia a la clase creada anteriormente, con el parámetro de tipo "JProgressBar" que se añade como componente a la interfaz, y la llamada a los métodos getBarra() y start().

```
public final class InterfazInicio extends javax.swing.JFrame {
    InicioCargar hilo;

    public InterfazInicio() {

        initComponents();
        iniciar();
    }

    public void iniciar() {

       //Se centra la ventana al iniciar la aplicación
        setLocationRelativeTo(null);

        hilo = new InicioCargar(BarraDeProgreso);
        hilo.getBarra();

       //Disparamos el Thread
        hilo.start();
        hilo = null;
    }
```

La ventana de desvanecimiento resultante, tras capturar el valor de la barra de progreso al incrementarse, gracias a la utilización del evento "State Changed", y la instrucción con el método "setWindowOpacity(ventana, opacidad)" que crea el efecto de translucidez para hacer una ventana más transparente cuanto menor es el valor de opacidad, es la mostrada a continuación.

Ilustración 22. Captura Interfaz Inicio

4.2.10. Implementación Reporte

Una de las principales bazas que habitualmente poseen las aplicaciones es la función de exportar los datos que se han ido generando y almacenando en la base de datos con la intención, entre otras, de tener una extracción de manera impresa.

Este punto se centra en un pequeño tutorial de programación Java de la página web http://www.elprogramador.com.mx/utilizar-irepots-con-netbeans donde se detallan los pasos a seguir para generar un reporte mediante las herramientas "JasperReports" y "iReports" en NetBeans IDE, y utilizando MySQL como motor de base de datos predeterminado.

Reiterando el proceso de importación de las librerías de la misma forma que se realizó en el apartado "Implementación Conexión con la base de datos", se presiona el botón derecho sobre "Biblioteca" y se selecciona en la ventana desplegable la opción "Agregar archivo JAR/Carpeta" como se observa en la captura.

En este caso se ha descargado y extraído el .zip (jasperreports-5.6.0.zip), pues versiones más actuales han dado problemas hasta el último momento en la visualización del reporte.

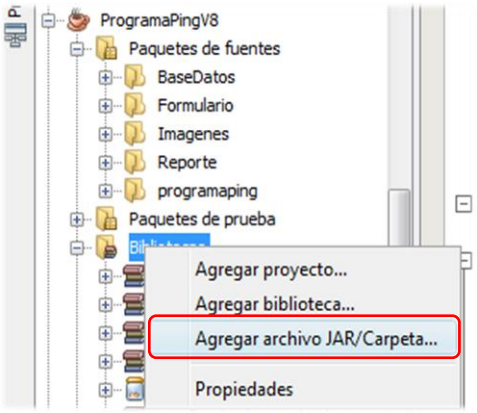

Ilustración 23. Agregar libreria de JasperReports.

La fuente para descargarse las librerías de JasperReports es la página oficial de JasperReports http://community.jaspersoft.com/download o, el siguiente enlace recomendable, https://sourceforge.net/projects/jasperreports/files/jasperreports/ donde se encuentra más de una versión.

Los archivos JAR que requiere el reporte son los presentados en la siguiente captura. Si al descargar el .zip no se encuentran los archivos mostrados a continuación, se buscarían en el siguiente enlace: http://www.java2s.com/Code/Jar/CatalogJar.htm

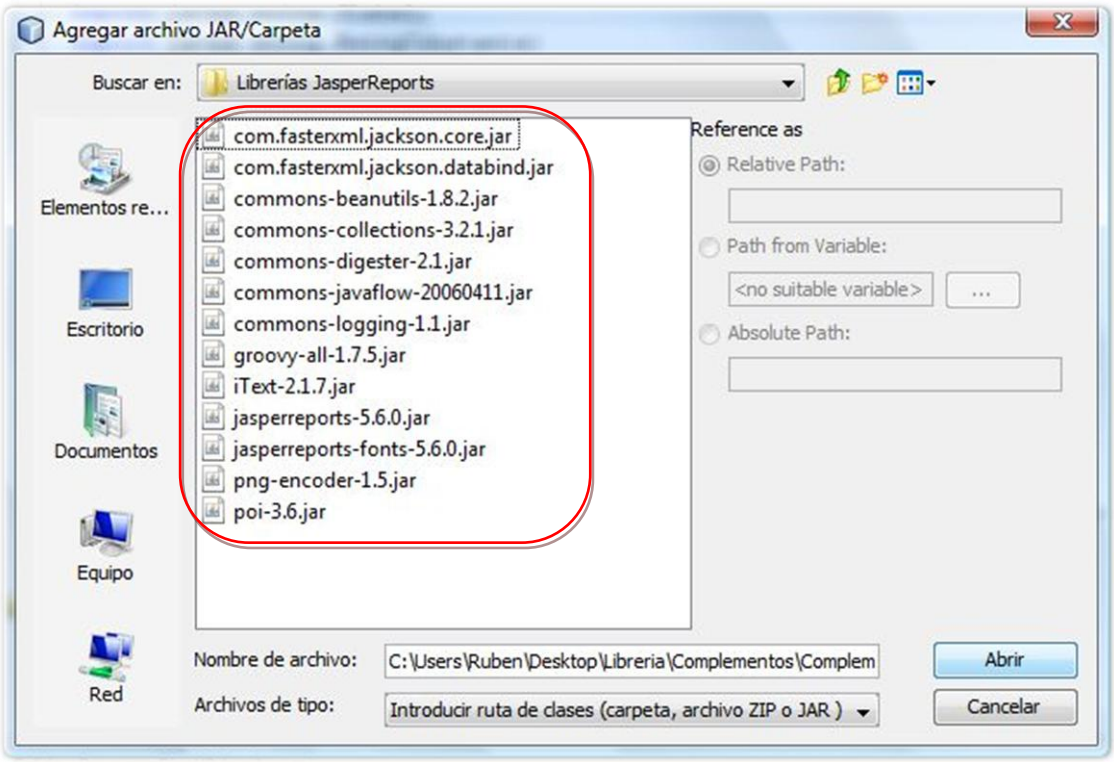

Ilustración 24. Librería JasperReports.

Por otro lado, vamos a la página oficial de NetBeans, más concretamente en la sección Plugins iReport, http://plugins.netbeans.org/plugin/4425/ireport, y se procede a descargar el plugin 7.4.

Posteriormente, en NetBeans IDE nos situamos en la siguiente localización: NetBeans>Herramientas>Plugins>Descargados, y al presionar el botón "Agregar Plugins" instalamos el plugin descargado. Una vez se instala es en ese momento cuando aparece en la interfaz de NetBeans el siguiente icono de creación de una BBDD.

Ilustración 25. Icono Connections/Datasources.

Ahora se clica en dicho icono y una vez abierta la ventana denominada "Connections/Datasources" se presiona sobre el botón "new". En la siguiente ventana se selecciona "Database JDBC connection" para una conexión con MySQL, y "next".

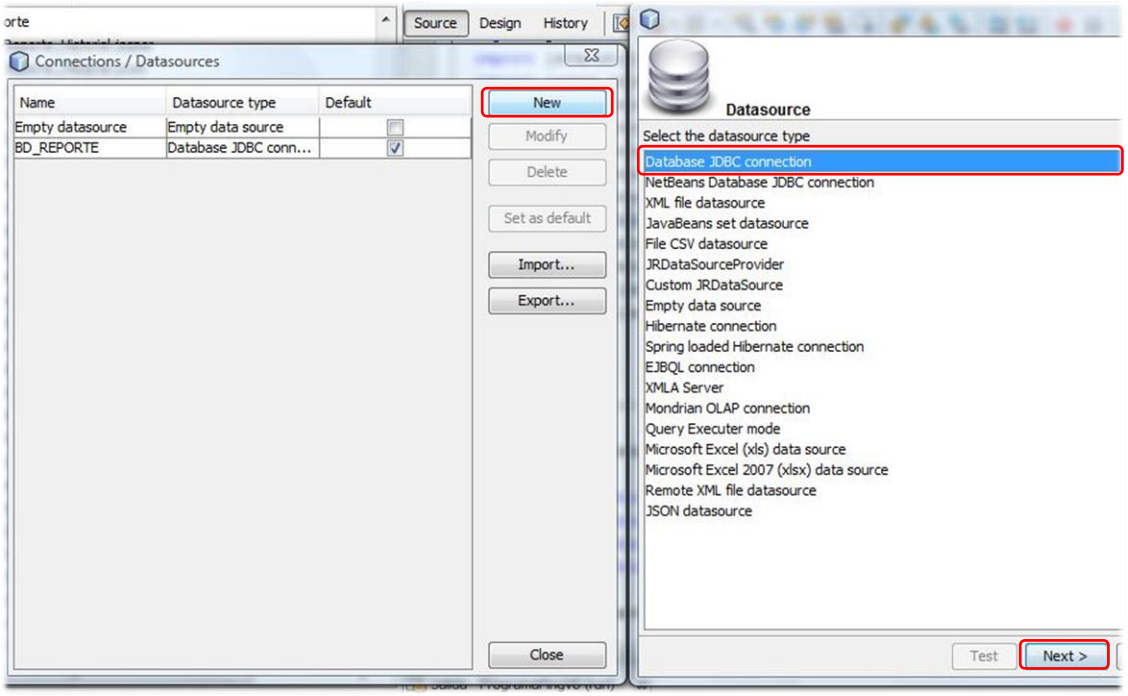

Ilustración 26. Database JDBC connection 1.

Una vez surja la ventana siguiente, se rellenan los campos como el nombre a elegir de la base de datos, se selecciona el JDBC driver apropiado a la base de datos MySQL, y se rellena la URL con el nombre de la base de datos que fue creada en phpMyAdmin, además del nombre de usuario por defecto y la contraseña que en este caso no tiene.

Para comprobar si la conexión creada puede ser establecida, se presiona el botón "Test" y, si la conexión funciona de manera correcta, aparecerá una nueva ventana con el siguiente mensaje emergente: "Connection test succesful!".

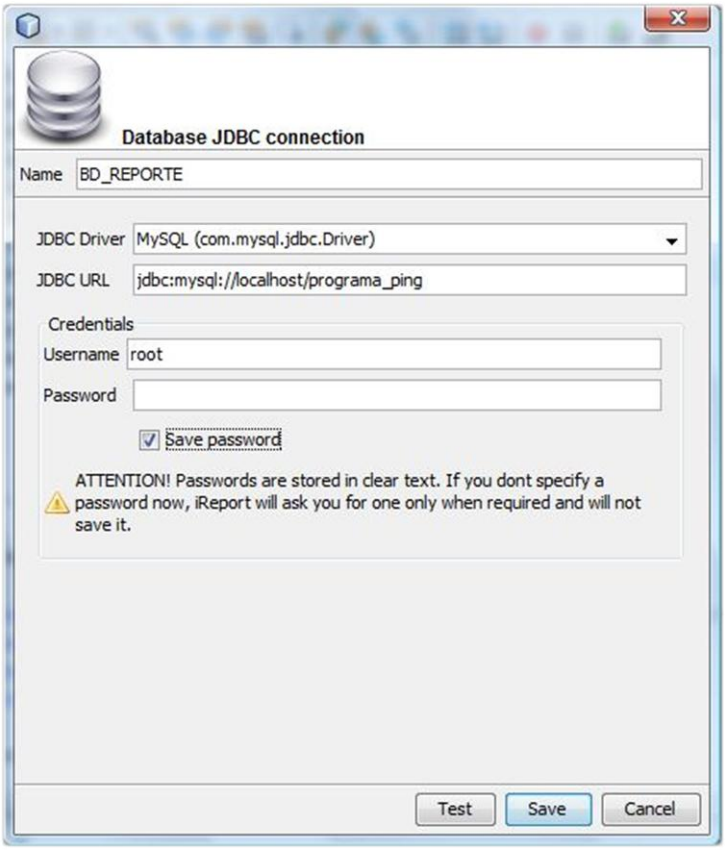

Ilustración 27. Database JDBC connection.

Una vez llegados a este punto, ya se puede crear el reporte, básicamente al hacer clic derecho en un nuevo "Java Package" que se haya designado para los reportes para mantener así el orden que conforma el programa. En la ventana desplegable se selecciona "Report Wizar" y, en base a la elección entre las plantillas de ejemplo creadas de forma predeterminada por NetBeans, se modifica dicha plantilla para ajustarla a nuestras necesidades.

Entre la configuración del reporte destacan dos ventanas que pueden ocasionar conflictos. La primera es la ventana que crea la sentencia SQL al presionar sobre el botón "Design query". Esta acción despliega una nueva ventana que, al clicar dos veces sobre la tabla que se desee, crea dicha sentencia para importar campos en el reporte.

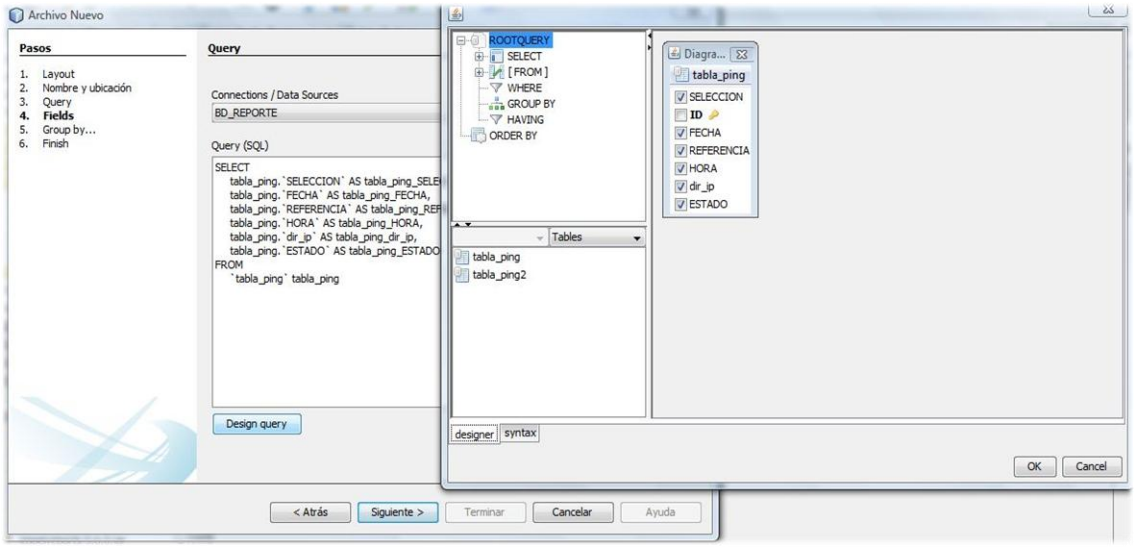

Ilustración 28. Sentencia SQL del reporte.

La segunda ventana es muy simple, tan solo hay que pasar los campos que se deseen para que los contenga el reporte. Esto se lleva a cabo al trasladar dichos campos desde el cuadro izquierdo al derecho. Con ello, ya estará prácticamente listo para poder comenzar con el diseño.

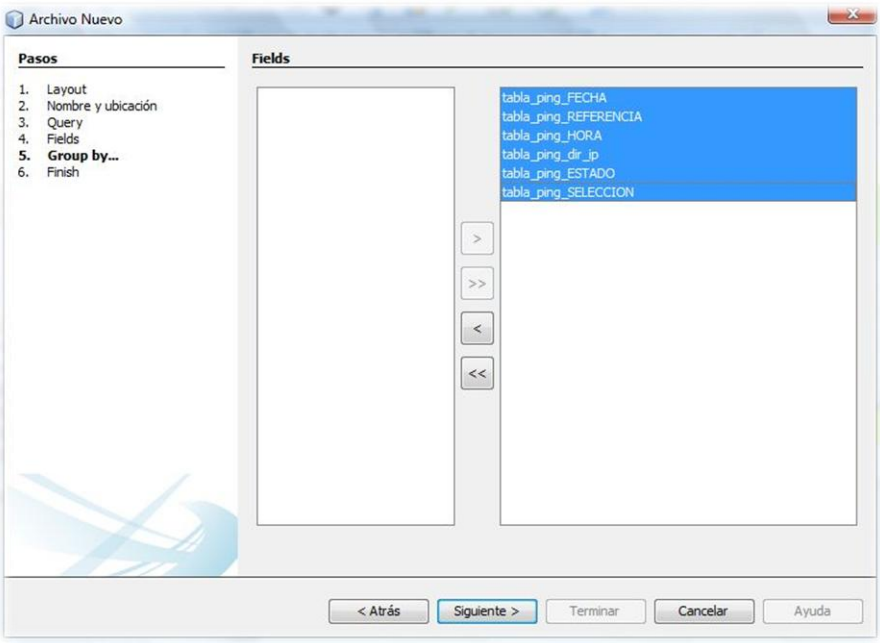

Ilustración 29. Grupo con campos del Reporte

Cabe decir que es preferible la creación de un reporte por cada tabla. En este caso se han creado un reporte para la tabla de datos de la pestaña principal y otro reporte para la tabla de datos de la pestaña historial. Esto requiere también crear dos botones para hacer llamadas a los respectivos reportes.

En los dos reportes se ha añadido el logotipo junto la marca de la empresa para la cual se ha realizado esta aplicación. Además, mencionar que en el pie de página se ubica tanto la fecha de la creación como el número de páginas que contiene el reporte.

Ilustración 30. Diseño Reporte Principal.

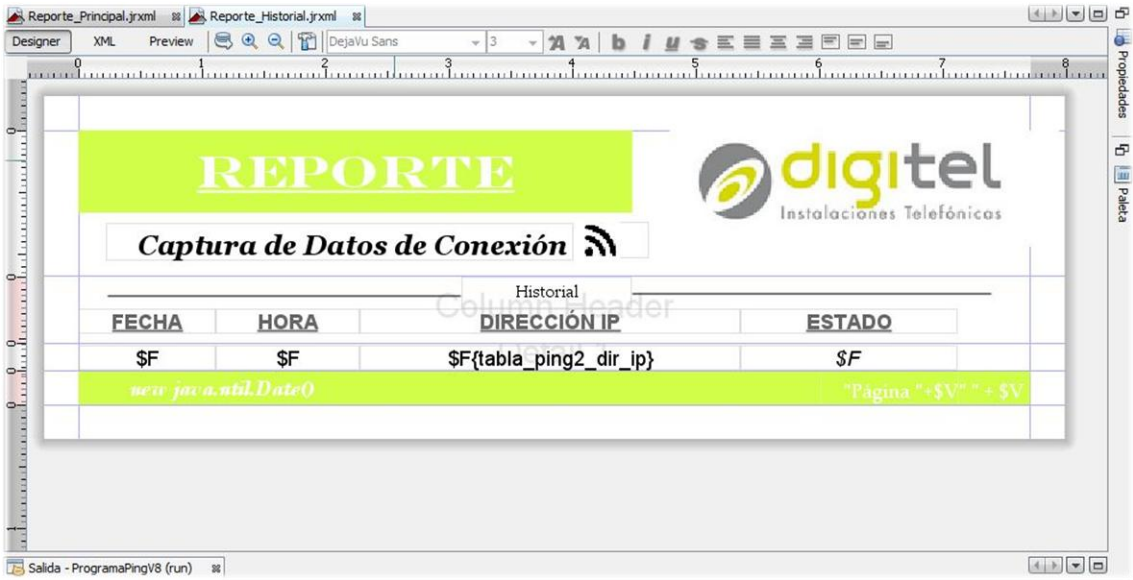

Ilustración 31. Diseño Reporte Historial.

Capítulo 5

Conclusiones

Las conclusiones que conciernen este apartado se enfocan a dos tesituras. La primera es la realización de prácticas curriculares en empresa, y la segunda es la elaboración del trabajo de fin de grado gracias a las prácticas curriculares en empresa.

Como causa primitiva, tener que realizar prácticas curriculares me ha servido para aprender a priorizar las cosas que son realmente importantes para poder avanzar con el trabajo, adaptar y saber gestionar mi horario para poder cumplir los objetivos laborales marcados, así como las necesidades propias, porque programar requiere tiempo y paciencia, afán y dedicación.

He podido desarrollar mis conocimientos, aptitudes y capacidades dentro de la empresa para contribuir al desarrollo tanto de esta como el mío, a nivel personal y profesional, ocasionando en mi, alzar la atención de elaborar y desarrollar con gran esfuerzo una aplicación sólida y funcional de ayuda al diagnóstico y mantenimiento de redes para una operadora de Telecomunicaciones que cumpla con los objetivos expuestos en el resumen de esta memoria.

- La comprobación de conectividad a una o varias direcciones IP.
- La detección de posibles cortes continuos de la conexión a Internet.
- La indicación del momento producido por el corte.
- La posibilidad de recopilar los datos producidos por la monitorización.

Y me sentía irrefutable al pensar que aquellas asignaturas que suspendes a lo largo de la carrera son las que deciden tu camino, ya que suspender la asignatura de Programación del lenguaje Java, y más si cabe, la asignatura de Arquitectura y Redes Telemáticas provocan un repaso doble de los conocimientos adquiridos la primera vez, pero estaba equivocado. Dichas asignaturas que son la base de esta aplicación, y que una vez posees la labor de interactuar con ellas, hacen percibir cuantos conocimientos quedan por afianzar.

Gran frase aquella que dice *"No confundas tu camino con tu destino"* (Anónimo), pues la creación del trabajo realizado es un paso más a aquella meta donde se quiere llegar.

Capítulo 6

Referencia bibliográfica

- Elliotte Rusty Harold, "**Java Network Programming**", O'Reilly, 2005. ISBN: 0-596-00721-3 (735 páginas).
- Bogdan Ciubotaru, "**Advanced Network Programming-Principles and Techniques**", Springer, 2013. ISBN: 9781447152927 (250 páginas).
- Clases de biblioteca de Java. http://docs.oracle.com/javase/8/docs/api/
- Foros online de Java
  http://www.lawebdelprogramador.com/foros/Java/index1.html
  http://www.javahispano.org/java-se/
- Oscar Belmonte Fernández, **"Introducción al lenguaje de programación Java.** http://www3.uji.es/~belfern/pdidoc/IX26/Documentos/introJava.pdf
- JVM-JDK-JRE. Conceptos fundamentales de la Programación Orientada a Objetos
  http://gl-epn-programacion-ii.blogspot.com.es/2010/03/jvm-jdk-jre-conceptos-fundamentales-de.html
- JDK (Java Development Kit)
  https://www.clubensayos.com/Tecnolog%C3%ADa/JDK-Java-Development-Kit/65848.html
- JDBC: acceso a bases de datos
  http://www.vc.ehu.es/jiwotvim/ISOFT2009-2010/Teoria/BloqueIV/JDBC.pdf
  http://profesores.fi-b.unam.mx/sun/Downloads/Java/jdbc.pdf
- Juan Manuel Gimeno, José Luis González, **"Introducción a NetBeans".**
  http://ocw.udl.cat/enginyeria-i-arquitectura/programacio-2/continguts-1/1-introduccioi81n-a-netbeans.pdf
- Uso de NetBeans
  https://www.fdi.ucm.es/profesor/luis/fp/devtools/NetBeansUso.html
- Luis Alberto Casillas, Marc Gibert Ginestà, Óscar Pérez, **"Bases de Datos en MySQL".** http://ocw.uoc.edu/computer-science-technology-and-multimedia/bases-de-datos/bases-de-datos/P06_M2109_02151.pdf
- Características de MySQL. https://packo.wikispaces.com/Caracteristicas+de+MYSQL
- Juan Benítez, **"Mini Guía Básica: Servidores WAMP- LAMP-MAMP".**
  http://www.tecnopedia.net/wp-content/uploads/2015/07/Mini-Guia-B%C3%A1sica-Servidores-WAMP-LAMP-MAMP1.pdf?eb27c9

- Instalación WampServer
  http://aulasne.navarra.es/pluginfile.php/4847/mod_page/content/31/instalar_wamp.pdf
- Instalación y configuración phpMyAdmin
  http://www.ite.educacion.es/formacion/materiales/85/cd/linux/m5/instalacin_y_configuracin_de_phpmyadmin.html
- JasperReports y iReports en NetBeans
  https://jossjack.wordpress.com/2014/06/15/jasperreport-ireport-en-netbeans/
- Launch4j
  http://svn.apache.org/repos/asf/cayenne/main/tags/1.2.3/cayenne/cayenne-ant/lib/windows/launch4j-2.1.1/web/index_es.html
- Crear instaladores con Inno Setup
  http://es.ccm.net/faq/8148-crear-un-setup-de-instalacion-con-inno-setup-compiler
- Nivel de Red en Internet
  https://www.tlm.unavarra.es/~daniel/docencia/rc_itig/rc_itig04_05/slides/clase16y17-NiveldeRed.pdf
- Introducción a TCP/IP
  http://www.alcancelibre.org/staticpages/index.php/introduccion-tcp-ip
- Protocolo IP
  http://www.ecured.cu/Protocolo_IP
- Direcciones IP
  https://msdn.microsoft.com/es-es/library/cc787434(v=ws.10).aspx
- Protocolo ICMP
  http://www.ietf.org/rfc/rfc792.txt
  https://msdn.microsoft.com/es-es/library/cc758065(v=ws.10).aspx
- Comando Ping
  https://bitsnocturnos.wordpress.com/2009/01/28/el-comando-ping/
- Dominio
  http://www.monografias.com/trabajos93/dominios-internet/dominios-internet.shtml
- DDNS
  http://www.adslayuda.com/smc7804-configuracion_dyndns.html
- Implementación Consola
  http://chuwiki.chuidiang.org/index.php?title=JScrollPane
- Implementación Pestaña Principal e Historial
  http://www.chuidiang.com/java/tablas/tablamodelo/tablamodelo.php
- Implementación JCheckBox
  http://www.jc-mouse.net/java/jcheckbox-dentro-jtable-con-netbeans
- Implementación Ordenar y Filtrar Tabla
  http://chuwiki.chuidiang.org/index.php?title=JTable:_Ordenar_y_filtrar_filas
- Implementación Reporte
  http://www.elprogramador.com.mx/utilizar-irepots-con-netbeans/
  https://sergiosoftware.wordpress.com/2015/02/01/creacion-reportes-jasperreports-ireports/