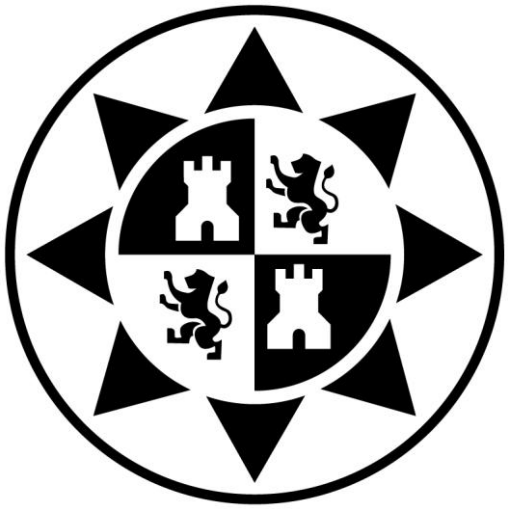
ESCUELA TÉCNICA SUPERIOR DE INGENIERÍA DE TELECOMUNICACIÓN
UNIVERSIDAD POLITÉCNICA DE CARTAGENA

Proyecto Fin de Carrera

Applet didáctico para cableado y comunicaciones Ethernet

AUTOR: Juan Pedro Saura Saura
DIRECTOR: Francesc Burrull i Mestres

Septiembre / 2012



Autor	Juan Pedro Saura Saura
E-mail del Autor	jpsaura_saura@hotmail.com
Director	Francesc Burrull i Mestres
E-mail del Director	francesc.burrull@upct.es
Titulo del PFC	Applet didáctico para cableado y comunicaciones Ethernet
Descriptores	Simulador: Applet Didáctico

Resumen

Se plantea el desarrollo de una herramienta didáctica para la docencia de Fundamentos de Telemática, donde una de las prácticas a desarrollar en el laboratorio consiste en comunicaciones Ethernet, en concreto la comprensión de los principios fundamentales de la tecnología Ethernet y su cableado.

Hasta el momento se da al alumno un enunciado de prácticas, y éste las realiza en laboratorio con instrumentos reales. Se pretende dotar de una alternativa con instrumentos virtuales (simulados) y el valor añadido de una ayuda contextual integrada.

Titulación	Ingeniero Técnico de Telecomunicaciones
Intensificación	Telemática
Departamento	Departamento de Tecnologías de Información y Comunicaciones
Fecha de Presentación	

Índice General

Capítulo 1.

1. Introducción

Se plantea el desarrollo de una herramienta didáctica para la docencia de Fundamentos de Telemática, donde una de las prácticas a desarrollar en el laboratorio consiste en comunicaciones Ethernet, en concreto la comprensión de los principios fundamentales de la tecnología Ethernet y su cableado.

Hasta el momento se da al alumno un enunciado de prácticas, y éste las realiza en laboratorio con instrumentos reales. Se pretende dotar de una alternativa con instrumentos virtuales (simulados) y el valor añadido de una ayuda contextual integrada.

En este proyecto se ha realizado un simulador didáctico para cableado y comunicaciones Ethernet, herramienta que servirá para la docencia en la Escuela Técnica Superior de Ingeniería de Telecomunicación (ETSIT) de la Universidad Politécnica de Cartagena (UPCT), disponiendo además del código fuente de la herramienta.

Dicho simulador es un Applet de java (JApplet), que consiste en una aplicación que se ejecuta en el navegador web, lo que hace que sea independiente de la plataforma y pueda funcionar perfectamente en cualquier máquina. El simulador se encuentra incrustado en un código HTML y es representado por una pequeña pantalla gráfica dentro del navegador.

Capítulo 2.

2. Objetivos

El objetivo del proyecto es obtener un applet simulador del cablemeter LAN Fluke 620 y un puesto de laboratorio dotado con tecnología Ethernet, para la compresión visual de dicho sistema de comunicaciones, y completar el enunciado de prácticas de la asignatura de **Fundamentos de Telemática**. Con ello se conseguirá una aplicación de la que se dispondrá el código fuente, y así en un futuro mejorar el autoaprendizaje por parte de los alumnos respecto a la asignatura:

- Facilitar la comprensión del programa mediante una aplicación Web que ayudará al alumno a poder usar dicha herramienta cuando lo necesite, viendo así la utilidad y la función de todos los componentes del programa.

- Modernización: se ha realizado una interfaz gráfica (Applet), haciendo más agradable el entorno visual al alumno.

- La portabilidad del programa: este consistirá en un archivo tipo "*.jar*", que se podrá ejecutar en cualquier plataforma, como Windows, Linux,……, etc.

- El software está desarrollado en la propia Universidad Politécnica de Cartagena, sin tener que pagar ningún tipo de licencia o tener problemas con ésta, y al ser software de la propia Universidad, estará en español, facilitando así la comprensión al alumno.

- El código se dejará abierto para posibles mejoras futuras del programa.

- Conocer el funcionamiento del cablemeter LAN Fluque 620, y entender el funcionamiento de sus distintos modos de trabajo.

- Estudiar la estructura a nivel físico y de enlace del comando de red "ping", así como los distintos campos que lo forman.

- Observar e identificar cuáles son los distintos tipos de averías que presentan los cables de par trenzado (en concreto UTP tipo EIA/TIA, 4 pares de categoría 5).

- Identificar los campos más importantes del comando de red "ping" visto en el osciloscopio.

Capítulo 3.

3. Análisis y Diseño

A continuación se hará una breve descripción de la práctica 3 de Fundamentos de telemática, donde se explicara en qué consiste y cuáles son sus principales características.

La práctica 3 se subdivide a su vez en dos: práctica 3a (Testeo y verificación de cableado) y práctica 3b (Estudio del nivel físico y del nivel enlace), con lo que se hará una pequeña descripción por separado de cada una de las partes.

3.1. Práctica 3a: Testeo y verificación de cableado

3.1.1. Elementos que intervienen

- Un cablemeter para redes LAN modelo "Fluke Cablemeter 620".
- Un polímetro digital convencional.
- Una placa simuladora de averías.
- Conectores de simulación de cortocircuitos y cortocircuitos resistivos.
- Un juego de cables con distintos conexionados con sus terminales.
- Dos cables patrón de un metro.

De los cuales solo el cablemeter ya correctamente configurado para su funcionamiento y el juego de cables se han incluido en el applet, ya que los mismos cables presentan las distintas averías que se simulan con las placas nombradas anteriormente.

3.1.2. Configuración del idioma y unidades de medida

Para comenzar a manejar el cablemeter hay que empezar por configurar el idioma y las unidades de medida. También se puede elegir el filtro de red y si el aparato realizará autodiagnóstico.

3.1.3. Configuración del tipo de cables

Se deberá configurar el cablemeter para realizar medidas y comprobaciones sobre cables del tipo UTP tipo EIA/TIA, 4 pares de categoría 5.

3.1.4. Calibración de los cables

Antes de realizar cualquier medida es necesario calibrar el cablemeter para que las medidas sean correctas.

3.1.5. Comprobación de cables.(Modo Test)

Se debe realizar el modo test a distintos cables usando y sin usar la prueba, anotando los resultados e interpretandolos.

3.1.6. Medidas de longitudes de cables.(Modo Length)

Con los mismos cables del punto 3.1.5 se debe realizar ahora el modo length usando y sin usar la prueba, y proceder igual que en el punto anterior.

3.1.7. Verificación de conexionado.(Modo Wire Map)

La comprobación de muestras de cables se realizarán con los mismos cables que se usaron en el apartado de comprobación de cables, debiéndose determinar su conexionado. Una vez acabado el test **MAP** de un cable se realizará un test con la opción **TEST** comentando, si se da el caso, la diferencia de alarmas al realizar los test en ambos modos.

3.1.8. Identificación de cables entre una sala dada y un cuarto de cableado

Este ejercicio trata de identificar dos extremos de un mismo cable en una instalación ya hecha. Para realizar el ejercicio se introducirá el accesorio ID en una de las tomas (una correspondiente al puesto de trabajo) del armario de comunicaciones. Después, con ayuda del cablemeter, se determinará la toma del puesto de trabajo que esta unida a la del armario de comunicaciones.

Nota1: los puntos 3.1.2, 3.1.3, 3.1.4 y 3.1.8 no se han implementado en el simulador ya que se centra en el estudio de las averías de los cables y de la correcta interpretación de los mensajes del cablemeter.

Nota2: la descripción de la práctica esta explicada más detalladamente en el manual de la práctica.

3.2. Práctica 3b: Estudio del nivel físico y del nivel de enlace

3.2.1 Elementos que intervienen

- PC's que actúan como cliente/servidor en un entorno Windows.
- Tarjetas de red Ethernet para bus PCI.
- Red de comunicaciones interna del laboratorio.
- POD's RJ-45 para el estudio del nivel físico.
- 1 switch de 24 puertos.
- 1 Osciloscopio digitales YOKOGAWA.
- 1 hub de 8 puertos.
- Cables de conexionado UPT 5 con terminales RJ-45 con conexión pin a pin.

Los equipos terminales de datos están formados por PC conectados a la red del laboratorio. Las tomas de las rosetas de cada uno de los puestos están conectadas, mediante el cableado interno del laboratorio a las correspondientes tomas del armario de comunicaciones. Cada uno de los puestos tiene 3 tomas para RJ-45 que se corresponden con otras tantas tomas en el armario de conexiones. La identificación de las tomas de cada puesto es sencilla pues en el armario de comunicaciones las tomas están numeradas según el puesto al que corresponden.

Mediante los cables de conexionado adecuados se pueden hacer conexiones punto a punto entre PC's, o conectar los ordenadores al switch para componer una red de varios usuarios.

3.2.2 El laboratorio

El laboratorio está dotado, en lo concerniente a redes, con dos redes de cableado estructurado. Además dispone de instrumentación telemática, que en el caso de esta práctica será un osciloscopio digital y un analizador de redes software.

- **UPCTNet:** La primera es la red general del edificio, no utilizada en esta práctica. Mediante esta red cada puesto de trabajo dispone dos rosetas con tomas para conectores RJ-45 a los que llegan una manguera de cuatro pares de categoría 5. A través de estas tomas la central de comunicaciones del edificio nos puede dar servicio de telefonía analógica, RDSI y acceso a la red del edificio.

- **Red interna:** Ésta será la que nos sirva para realizar las distintas comunicaciones durante las sesiones de prácticas. La arquitectura de esta red es idéntica a la cualquier otra red de comunicaciones, por ejemplo, la del edificio. Esta red interna realiza la conexión de cada una de las tomas que hay en los puestos con otras tantas tomas que hay en un armario de conexiones. Cada uno de los ordenadores de los puestos se puede conectar a tres tomas RJ-45. Cada una de estas tomas está conectada a otra que se encuentra en al armario de comunicaciones a través de una manguera de cuatro pares trenzados de categoría 5. Para realizar estas conexiones se dispondrá de los cables correspondientes junto con los conectores y conexiones adecuadas.

- **Armario de comunicaciones:** En este armario lo primero que nos encontramos es una serie de bocas que se corresponden con cada una de las tomas de los puestos de trabajo. Además existen 48 tomas adicionales para RJ-45 que están conectadas a la centralita telefónica del laboratorio. A ellas llegan líneas RDSI y líneas de telefonía analógica. Estas Líneas se reparten entre las tomas de la siguiente manera:

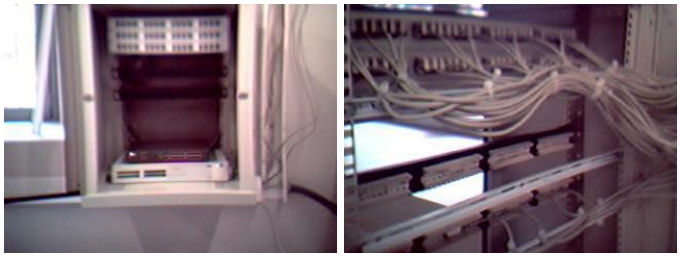


Figura 1. Armario de comunicaciones. Frontal y bastidor de conexiones

Dentro del armario de conexiones encontramos un hub de 16 puertos y un switch configurable por software de 24 puertos. Estos elementos nos permitirán construir distintas configuraciones de red LAN en el laboratorio. Además, cada banco dispondrá también de un hub de 8 puertos.

Figura 2. Hub de 8 puertos. Disponible 1 por banco

Durante la sesión de prácticas el alumno dispondrá de latiguillos de pares trenzados con conectores RJ-45, con los cuales realizará las conexiones adecuadas o bien en cada uno de los hubs que hay por banco o bien en el armario de comunicaciones para tener distintas configuraciones de red. Los latiguillos que se emplearán en la práctica son de dos tipos: conexión pin a pin (normal) y conexión cruzada. El alumno tendrá especial cuidado en la elección del cable a usar según la comunicación que deba realizar. Nótese que en el caso de utilizar el hub, se puede realizar una conexión cruzada utilizando adecuadamente la boca 8.

- **Analizador de redes software:** El analizador de redes que va a utilizarse consiste en un ordenador personal ejecutando el software **Ethereal**. Este programa analiza las tramas a partir del nivel MAC hacia arriba, por lo que será adecuado para la realización de la práctica.

3.2.3 Conexionado del ordenador a una red LAN

En este apartado se procede a configurar la tarjeta de red y los recursos de red en nuestro PC. Para ello habrá proceder como se indica en el guion de la práctica.

3.2.4 Estudio del nivel físico

Partiendo de la LAN aislada, se conectará el POD de RJ-45 entre el hub y uno de los PC's (es como un alargador a efectos de las comunicaciones), para de esta manera poder pinchar con las sondas del osciloscopio los pares 1-2 y 3-6 del conector. Para provocar las transmisiones de tramas por la red usaremos la utilidad "ping", que se puede ejecutar desde MS-DOS. Es conveniente ejecutar los pings desde el PC conectado al hub através del POD y ejecutar el Ethereal en el otro PC.

Una vez realizado el conexionado, habrá que observar el nivel físico, es decir, los niveles de tensión, el tiempo de bit, etc.

Para poder ver adecuadamente una trama Ethernet a nivel físico habrá que configurar el osciloscopio de la siguiente manera:

Base de tiempos: 10μs/div

CH1:
V/div: 2V/div
POSITION: Position to 2div
INPUT: Coupling DC, probe 10:1, invert OFF,
Acq HOLD OFF.

CH2:
V/div: 2V/div
POSITION: Position to -2div
INPUT: Coupling DC, probe 10:1, invert OFF,

TRIGGER:
TYPE: Edge
MODE: Single
LEVEL/SOURCE: CH1/↑, CH1 level -1V
POSITION/DELAY: trigger position to –4 div, trigger delay 0ns
COUPLING: coupling DC, HF_Rej OFF
HOLD OFF: OFF

ZOOM: ZOOM MODE: MainZoom

Así el osciloscopio capturará una trama completa, pudiéndose observar los detalles necesarios con la ventana de zoom.

- **Formato de una trama Ethernet:**

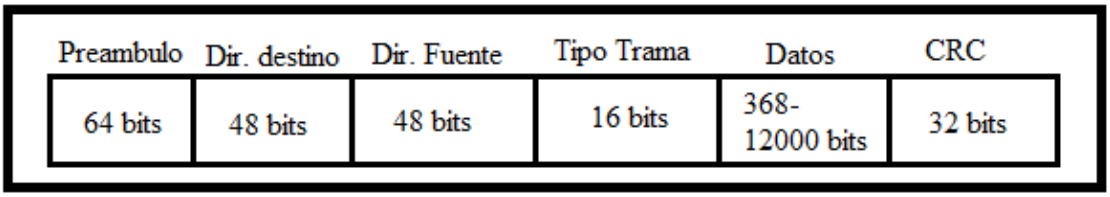


Figura 3. Formato trama Ethernet

- Datos: el área de datos contiene de 56 bytes a 1500 bytes, por lo que el tamaño de una trama en Ethernet es variable: no menor a 64 bytes ni mayor a 1518 bytes.
- Preámbulo: para permitir a nodos receptores sincronizarse (1's y 0's alternados).
- CRC (Cyclic Redundancy Check:) sirve para detectar errores en la transmisión.
- Tipo de Trama: utilizado para saber el tipo de información que transporta la trama o el protocolo de nivel superior a utilizar, no necesariamente TCP/IP.

3.2.5 Estudio del nivel de enlace

Para estudiar el nivel de enlace corroborando con el nivel físico se propone estudiar una trama que transporte un paquete con una solicitud de "ping". Para hacer este experimento se propone primero hacer dos veces "ping" desde el PC que tiene la línea de transmisión pinchada hasta el PC que tiene el programa analizador ejecutándose en él ping <IP destino> –n 1. Así el PC de origen actualiza sus tablas en el primer ping y ya no tendrá que usar ARP. En la segunda vez, se capturará el "ping" mediante el osciloscopio y el analizador de red, para así poder corroborar conceptos mediante las dos herramientas. Una vez se tenga la captura realizada, se trabajará con estos datos.

El primer campo que muestra el analizador serán las direcciones físicas destino y origen de la trama MAC. A continuación hay un byte, que en función del tipo de trama tendrá un significado u otro. A partir de aquí los datos son de niveles superiores.

- **Protocolos de nivel superior:** Si se desea profundizar dentro de la trama a niveles superiores, el programa Ethereal permite el análisis hasta el nivel de aplicación. Para ver tramas de niveles superiores podemos hacer una captura de tramas al hacer una conexión FTP con un servidor FTP del laboratorio.

Nota1: la descripción de la práctica esta explicada más detalladamente en el manual de la práctica.

Capítulo 4.

4. Ámbito de aplicación

4.1. Elección del lenguaje de desarrollo.

Se han barajado los siguientes lenguajes para desarrollar la implementación del programa:

4.1.1. C

El lenguaje por excelencia de programación de sistemas. Es un lenguaje de nivel medio (no se puede decir que sea un lenguaje de alto nivel, por incorporar muchos elementos propios del ensamblador), tremendamente ligado a UNIX (aunque es portable y hay compiladores para casi cualquier sistema operativo). Casi podríamos considerarlo como un ensamblador estructurado y portable. Difícil de aprender (no es recomendable como primer lenguaje, como sí lo podría ser Pascal), aunque tremendamente flexible. Bastante dado a errores, sobre todo entre programadores novatos.

4.1.2. C++

Es una evolución sobre el lenguaje de programación de C. C++ casi se puede considerar de alto nivel. Es orientado a objetos, relativamente difícil de aprender (muchas características, muy complicado), pero combina la potencia y flexibilidad de C con el valor añadido de C++ es orientado a objetos. Bastante utilizado. Dado a errores, aunque no tantos como en C.

4.1.3. Java

JAVA es el lenguaje de "moda", de sintaxis parecida al C. Orientado a objetos (JAVA te obliga, C++ sólo te da la posibilidad), mucho menos flexible que C++, pensado para hacer aplicaciones interactivas más que controladores de dispositivos y sistemas operativos. Mucha aceptación popular, muchos recursos, y posibilidad de incluir programas de JAVA en páginas HTML (los llamados "Applet").

JAVA es un lenguaje multiplataforma, los programas JAVA se pueden ejecutar en cualquier plataforma soportada (Windows, Unix, Mac, etc.), además es un lenguaje compilado e interpretado, ya que el compilador produce un código intermedio independiente del sistema llamado bytecode. Se necesita instalar en el ordenador la JVM (Java Virtual Machine), que es el intérprete que convierte el bytecode en código máquina. Sun Microsystems distribuye de forma gratuita el producto base, el llamado JDK (Java Development Kit.), también llamado J2SE (Java 2 Standard Edition), y se puede encontrar en la siguiente dirección: http://java.sun.com/.

4.1.4. Elección asumida

Finalmente me he decantado por el lenguaje de programación JAVA para la implementación del programa por los siguientes motivos:

i. Se ha elegido en detrimento de C al ser un lenguaje orientado a objetos y por su portabilidad.

ii. Frente a C++ tiene la ventaja de tener más portabilidad, se puede usar las clases compiladas en cualquier plataforma (Windows, Unix, etc.).

iii. Una razón muy importante fue el entorno gráfico resultante, Ya que JAVA proporciona elementos gráficos muy atractivos, sobre todo utilizando las componentes swing.

4.2. Elección del entorno de desarrollo.

Para el lenguaje de programación de Java hay múltiples entornos de desarrollo, entre los que se han barajado los siguientes:

4.2.1. JBuilder

Dirección: (Borland) **http://www.borland.com/jbuilder/**
Versión actual: 6.0
Plataformas: Windows, Linux, Solaris.
Licencia: La versión de evaluación, la personal, es gratis, las avanzadas, profesionales y Enterprise son de pago.

Jbuilder Foundation está diseñado para desarrolladores Java que quieren una alta productividad **IDE** (Entorno de Desarrollo Integrado) para crear más fácilmente aplicaciones multiplataforma para Linux, Solaris y Windows.

Finalmente, Jbuilder permite que los usuarios retoquen a su gusto y extiendan el entorno según sus necesidades de desarrollo usando **Open Tools API**, la cual facilita la integración de otros componentes adicionales.

Tiene la gran utilidad de visualizar los diagramas **UML**, además de poder desarrollar aplicaciones **Web** con **JSP** y **servlets**.

4.2.2. Kawa

Dirección:
Versión actual: 5.0
Plataforma: Windows.
Licencia: Como es habitual, las versiones profesionales y Enterprise son de pago. Pero también disponen de versiones de evaluación.

Kawa es un entorno de múltiples ventanas. Se parte de un proyecto, lleva dos módulos, uno con soporte **HTML** y otro para **JAVA**. Se pueden visualizar las clases, variables…

Sencillo y potente.

4.2.3. NetBeans

Dirección: **http://www.netbeans.org/**
Versión actual: 7.0.1 (Build 201107282000)
Plataforma: Todas las plataformas con **JVM**.
Licencia: Opensource.

NetBeans es una aplicación Opensource (el código del entorno está abierto a posibles modificaciones) de desarrollo escritas en Java, esto quiere decir que se puede modificar el entorno de acuerdo a ciertos parámetros de licencia. Soporta, aparte de Java, otros lenguajes, como C++.

NetBeans es la mejor opción a la hora de desarrollar en Swing y J2EE/EJB 3.0. Algunas de sus funciones y características son:

- Completado de código.
- Soporte para escritura de servlets.
- Ayudas con el código y ayuda on-line.

El Entorno de Desarrollo Integrado (**IDE**) NetBeans es un entorno de programación de varios lenguajes, incluyendo a Java y a C++. Este desarrollo es de fuente abierta, es decir, se proporciona el código fuente del entorno para que se pueda modificar de acuerdo a ciertos parámetros de licencia.

Es un entorno muy cómodo en cuanto a interfaz gráfica se refiere, ya que ahorra al usuario escribir código tan sólo con arrastrar los componentes gráficos que soporta Java y que están representados mediante iconos. Al añadir estos componentes, en la aplicación se genera el código correspondiente a dicha inclusión del elemento gráfico. Está recomendado por Sun Microsystems, aunque una de sus desventajas es que ocupa muchos recursos de memoria.

4.2.4. Eclipse

Dirección: **http://www.eclipse.org.**
Versión actual: Java 2 Technology Edition 5.0, SR4 (see caveat below).
Plataforma:

Eclipse es en el fondo, únicamente un armazón (workbench) sobre el que se pueden montar herramientas de desarrollo para cualquier lenguaje, mediante la implementación de los plugins adecuados.

La arquitectura de plugins de Eclipse permite, además de integrar ciertos lenguajes sobre un mismo IDE, introducir otras aplicaciones accesorias que pueden resultar útiles durante el proceso de desarrollo como: herramientas UML, editores visuales de interfaces, ayuda en línea para librerías, etc.

4.2.5. Elección asumida

El entorno principal elegido es el ***NetBeans***. Por su facilidad de uso y gran potencia (aunque consuma demasiada memoria.), es muy ideal para implementar distintos algoritmos, además de ser un entorno de ventanas, mediante el cual la programación se hace mas llevadera.

Para el diseño de la interfaz gráfica, también me he decantado por NetBeans, también por su gran facilidad de uso.

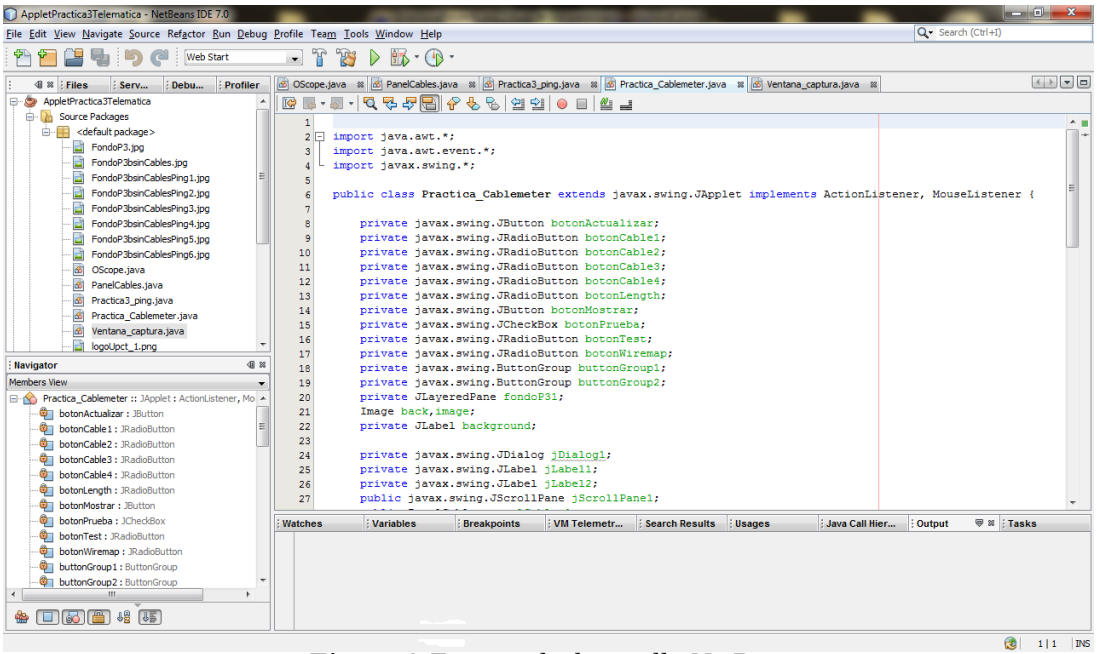


Figura 4: Entorno de desarrollo NetBeans

Capítulo 5.

5. Implementación de la aplicación

5.1. Página de inicio.

La aplicación comienza en un diseño web (al cual se puede acceder a través de la siguiente dirección: http://labit501.upct.es/~jpsaura), en la cual se muestra una página de inicio a través de la que se puede acceder al enunciado de cada una de las partes de la práctica (enunciado práctica 3a ó enunciado práctica 3b).

Figura 5: Web de inicio

En ambas partes se puede leer el enunciado de la práctica seleccionada y acceder al applet correspondiente.

Figura 6: Enunciado práctica 3a

Figura 7: Enunciado práctica 3b

Finalmente se muestra el applet correspondiente a dicha práctica, donde a su vez hay enlaces a los manuales de ayuda de uso del applet.

Figura 8: Applet práctica 3a

Figura 9: Applet práctica 3a

Ahora voy a explicar por que clases está formado cada applet de forma general, para después hacer una descripción más en profundidad de cada una de ellas. Antes de empezar con esta descripción resaltar que el applet se abre en el propio navegador web, como hemos podido observar anteriormente.

Práctica 3a: Testeo y verificación de cableado.

La clase principal de este applet es la clase *Práctica_Cablemeter*, la cual hereda de *JApplet.* Además es la que instancia todos los componentes y la que proporciona la interfaz grafica.

Esta clase también es la que implementa la mayoría de la funcionalidad del applet.

Hay otra clase por la que está formado el applet y es la clase *PanelCables* que hereda de *JPanel.* Esta clase lo que hace es mostrar la imagen de los cables y es la encargada de generar las averías de los mismos.

La interfaz grafica es la vista en la Figura 8, donde se pueden ver los distintos botones, el cablemeter y los cables.

Como ya he mencionado anteriormente es la clase Práctica_*Cablemeter* la que proporciona la interfaz grafica e instancia cuatro objetos *PanelCables*. En los siguientes apartados explicare con mayor profundidad como funciona este applet explicando el código.

Práctica 3b.Ethernet: estudio del nivel físico y del nivel de enlace.

La clase principal de este applet es la clase *Práctica3_ping* la cual hereda de *JApplet.* Además es la que instancia todos los demás componentes y la que proporciona la interfaz grafica.

Hay dos clases más por la que está formado el applet y son la clase *Oscope* y la clase *Ventana_captura*, que heredan de *Jframe.*

La clase *OScope* es la que implementa el osciloscopio, es la encargada de dibujar el ping entre otras cosas, como por ejemplo: modificar la anchura o altura de la señal para verla mejor, cambiar de color la señal para ver los distintos campos por los que está formada… Esta clase se lanza al hacer click en la imagen del osciloscopio (Figura 9), lo que sucede es que se abre una ventana en el escritorio.

La clase *Ventana_captura* es la que representa los datos del analizador de tráfico de redes (Wireshark), y básicamente lo que hace es mostrar en texto plano los datos del ping tal como lo haría Wireshark. Esta clase se lanza cuando se hace click en el botón Mostar Captura y al igual que en la clase *OScope* se abre una ventana en el escritorio.

A continuación se explicaran detalladamente cada una de las clases nombradas anteriormente (de ambas partes de la práctica), y cómo interactúan entre si.

5.2. Clases de la práctica 3a

5.2.1. Práctica_Cablemter

```
import java.awt.*;
import java.awt.event.*;
import javax.swing.*;

public class Práctica_Cablemeter extends javax.swing.JApplet implements
ActionListener, MouseListener {

    private javax.swing.JButton botonActualizar;
    private javax.swing.JRadioButton botonCable1;
    private javax.swing.JRadioButton botonCable2;
    private javax.swing.JRadioButton botonCable3;
    private javax.swing.JRadioButton botonCable4;
    private javax.swing.JRadioButton botonLength;
    private javax.swing.JButton botonMostrar;
    private javax.swing.JCheckBox botonPrueba;
    private javax.swing.JRadioButton botonTest;
    private javax.swing.JRadioButton botonWiremap;
    private javax.swing.ButtonGroup buttonGroup1;
    private javax.swing.ButtonGroup buttonGroup2;
    private JLayeredPane fondoP31;
    Image back,image;
    private JLabel background;

    private javax.swing.JDialog jDialog1;
    private javax.swing.JLabel jLabel1;
    private javax.swing.JLabel jLabel2;
    public javax.swing.JScrollPane jScrollPane1;
    public PanelCables panelCables1;
    public PanelCables panelCables2;
    public PanelCables panelCables3;
    public PanelCables panelCables4;
    public javax.swing.JTextPane panelTexto;

    private int[] propiedadesC1= new int[6];
    private int[] propiedadesC2= new int[6];
    private int[] propiedadesC3= new int[6];
    private int[] propiedadesC4= new int[6];
    public int[] propAux=new int[6];
    private char[] caracteres = new char[2];
    private int cable=0,banderaC=0;
    private double cableEnCorto;
    private boolean test=false,length=false,wire=false,prueba=false;
    private int longModoLen;

    @Override
    public void init() {}

    @Override
    public void actionPerformed(ActionEvent e) {}

    @Override
    public void mouseClicked(MouseEvent e) {}
```

```
    @Override
    public void mousePressed(MouseEvent e) {}

    @Override
    public void mouseReleased(MouseEvent e) {}

    @Override
    public void mouseEntered(MouseEvent e) {}

    @Override
    public void mouseExited(MouseEvent e) {}

    private void VerInfoCable() {}

    private void MostrarPulsado() {}

    public Image getImage(String im){}

}
```

El código que se muestra arriba es el de la clase *Práctica_Cablemeter*, ahora voy a explicar cada uno de los métodos que la forman.

En este primer trozo de código es donde se declaran las variables y los componentes que forman la clase, esto lo explicare con más detalle a continuación cuando vayan apareciendo en los distintos métodos.

```
    private javax.swing.JButton botonActualizar;
    private javax.swing.JRadioButton botonCable1;
    private javax.swing.JRadioButton botonCable2;
    private javax.swing.JRadioButton botonCable3;
    private javax.swing.JRadioButton botonCable4;
    private javax.swing.JRadioButton botonLength;
    private javax.swing.JButton botonMostrar;
    private javax.swing.JCheckBox botonPrueba;
    private javax.swing.JRadioButton botonTest;
    private javax.swing.JRadioButton botonWiremap;
    private javax.swing.ButtonGroup buttonGroup1;
    private javax.swing.ButtonGroup buttonGroup2;
    private JLayeredPane fondoP31;
    Image back,image;
    private JLabel background;
    private javax.swing.JLabel jLabel1;
    private javax.swing.JLabel jLabel2;
    public javax.swing.JScrollPane jScrollPane1;
    public PanelCables panelCables1;
    public PanelCables panelCables2;
    public PanelCables panelCables3;
    public PanelCables panelCables4;
    public javax.swing.JTextPane panelTexto;
    private int[] propiedadesC1= new int[6];
    private int[] propiedadesC2= new int[6];
    private int[] propiedadesC3= new int[6];
    private int[] propiedadesC4= new int[6];
    public int[] propAux=new int[6];
    private char[] caracteres = new char[2];
    private int cable=0,banderaC=0;
    private double cableEnCorto;
    private boolean test=false,length=false,wire=false,prueba=false;
    private int longModoLen;
```

- Metodo *Public void init().*

En este método es en el que se inicializan y se instancian los componentes declarados en la parte anterior, voy a ir explicando que es cada uno y en el caso de los que sean visibles, con que elemento de la interfaz grafica se corresponde.

```
Public void init(){

  buttonGroup1 = new javax.swing.ButtonGroup(); //se corresponde con G1
  buttonGroup2 = new javax.swing.ButtonGroup(); //se corresponde con G2
  fondoP31 = new JLayeredPane(); //panel donde se incrustan los componentes
  panelTexto = new javax.swing.JTextPane();//se corresponde con pantalla
osciloscopio
  jScrollPane1 = new javax.swing.JScrollPane();//scroll que a veces sale en
pantelTexto
  botonPrueba = new javax.swing.JCheckBox();//se corresponde con prueba
  botonTest = new javax.swing.JRadioButton();//se corresponde con test
  botonLength = new javax.swing.JRadioButton();//se corresponde con length
  botonWiremap = new javax.swing.JRadioButton();//se corresponde con wiremap
  panelCables2 = new PanelCables(this);//se corresponde con C2
  panelCables4 = new PanelCables(this);//se corresponde con C4
  panelCables3 = new PanelCables(this);//se corresponde con C3
  panelCables1 = new PanelCables(this);//se corresponde con C1
  botonCable1 = new javax.swing.JRadioButton();//se corresponde con BC1
  botonCable3 = new javax.swing.JRadioButton();//se corresponde con BC3
  botonCable4 = new javax.swing.JRadioButton();//se corresponde con BC4
  botonCable2 = new javax.swing.JRadioButton();//se corresponde con BC2
  botonActualizar = new javax.swing.JButton();// se corresponde con actualizar
  jLabel1 = new javax.swing.JLabel();//se corresponde con CABLES
  jLabel2 = new javax.swing.JLabel();//se corresponde con CABLEMETER
  botonMostrar = new javax.swing.JButton();//se corresponde con MOSTRAR
  Font f = new Font("Arial",Font.PLAIN,14);//Fuentes para letras botones
  Font f1 = new Font("Arial",Font.PLAIN,12);

//se instancia y se dan las propiedades del panel del fondo
  fondoP31.setBackground(Color.white);
  setBackground(Color.white);
  setSize(1010,617);
  fondoP31.setLayout(null);
  setVisible(true);

//se isntancia y se dan las propiedades de la etiqueta CABLES
  jLabel1.setFont(new java.awt.Font("Arial", 0, 24));
  jLabel1.setText("    CABLES");
  jLabel1.setBounds(600,37,137,40);
  jLabel1.setToolTipText("Zona de soluciones");
  jLabel1.setVisible(true);
  fondoP31.add(jLabel1);

//se isntancia y se dan las propiedades de la etiqueta CABLEMETER
  jLabel2.setFont(new java.awt.Font("Arial", 0, 24));
  jLabel2.setText(" CABLEMETER");
  jLabel2.setBounds(120,37,168,40);
  jLabel2.setToolTipText("Zona de prueba");
  jLabel2.setVisible(true);
  fondoP31.add(jLabel2);
```

```
//se instancia y se dan las propiedades a la pantalla del cablemeter
  panelTexto.setEditable(false);
  panelTexto.setFont(new java.awt.Font("Tahoma", 0, 16));
  jScrollPane1.setBounds(98,105,220,125);
  panelTexto.setVisible(true);
  jScrollPane1.setViewportView(panelTexto);
  fondoP31.add(jScrollPane1);

//se instancia y se dan las propiedades al botón MOSTRAR
  botonMostrar.setFont(f);
  botonMostrar.setText("MOSTRAR");
  botonMostrar.setBounds(140,245,138,38);
  botonMostrar.setVisible(true);
  botonMostrar.addActionListener(this);
  fondoP31.add(botonMostrar);

//se instancia y se dan las propiedades al botón TEST
   botonTest.setFont(f);
   botonTest.setBackground(new java.awt.Color(204, 204, 255));
   botonTest.setText("   TEST");
   botonTest.setBounds(155,305,98,28);
   botonTest.addActionListener(this);
   botonTest.setVisible(true);
   buttonGroup1.add(botonTest);
   fondoP31.add(botonTest);

//se instancia y se dan las propiedades al botón LENGTH
   botonLength.setFont(f);
   botonLength.setBackground(new java.awt.Color(204, 204, 255));
   botonLength.setText(" LENGTH");
   botonLength.setBounds(155,353,98,28);
   botonLength.addActionListener(this);
   botonLength.setVisible(true);
   buttonGroup1.add(botonLength);
   fondoP31.add(botonLength);

//se instancia y se dan las propiedades al botón WIRE_MAP
   botonWiremap.setFont(f);
   botonWiremap.setBackground(new java.awt.Color(204, 204, 255));
   botonWiremap.setText("WIRE_MAP");
   botonWiremap.setBounds(155,401,98,28);
   botonWiremap.addActionListener(this);
   botonWiremap.setVisible(true);
   buttonGroup1.add(botonWiremap);
   fondoP31.add(botonWiremap);

//se instancia y se dan las propiedades al botón prueba
   botonPrueba.setFont(f);
   botonPrueba.setBackground(new java.awt.Color(204, 204, 255));
   botonPrueba.setText("Prueba");
   botonPrueba.setBounds(110,467,85,20);
   botonPrueba.addActionListener(this);
   botonPrueba.setVisible(true);
   fondoP31.add(botonPrueba);
```

```
//se instancia y se dan las propiedades a C1
  panelCables1.setName("panelCables1");
  panelCables1.setOpaque(true);
  panelCables1.setBounds(468,167,125,120);
  panelCables1.setToolTipText("Pulsar para ver solución");
  panelCables1.addMouseListener(this);
  panelCables1.setVisible(true);
  fondoP31.add(panelCables1);

//se instancia y se dan las propiedades a C2
  panelCables2.setName("panelCables2");
  panelCables2.setOpaque(true);
  panelCables2.setBounds(763,167,125,120);
  panelCables2.setToolTipText("Pulsar para ver solución");
  panelCables2.addMouseListener(this);
  panelCables2.setVisible(true);
  fondoP31.add(panelCables2);

//se instancia y se dan las propiedades a C3
  panelCables3.setName("panelCables3");
  panelCables3.setOpaque(true);
  panelCables3.setBounds(468,357,125,120);
  panelCables3.setToolTipText("Pulsar para ver solución");
  panelCables3.addMouseListener(this);
  panelCables3.setVisible(true);
  fondoP31.add(panelCables3);

//se instancia y se dan las propiedades a C4
  panelCables4.setName("panelCables4");
  panelCables4.setOpaque(true);
  panelCables4.setBounds(763,357,125,120);
  panelCables4.setToolTipText("Pulsar para ver solución");
  panelCables4.addMouseListener(this);
  panelCables4.setVisible(true);
  fondoP31.add(panelCables4);

//se instancia y se dan las propiedades a BC1
  botonCable1.setBackground(new java.awt.Color(153, 255, 255));
  botonCable1.setBounds(593,217,20,20);
  botonCable1.addActionListener(this);
  botonCable1.setVisible(true);
  buttonGroup2.add(botonCable1);
  fondoP31.add(botonCable1);

//se instancia y se dan las propiedades a BC2
  botonCable2.setBackground(new java.awt.Color(153, 255, 255));
  botonCable2.setBounds(743,217,20,20);
  botonCable2.addActionListener(this);
  botonCable2.setVisible(true);
  buttonGroup2.add(botonCable2);
  fondoP31.add(botonCable2);

//se instancia y se dan las propiedades a BC3
  botonCable3.setBackground(new java.awt.Color(153, 255, 255));
  botonCable3.setBounds(593,407,20,20);
  botonCable3.addActionListener(this);
  botonCable3.setVisible(true);
  buttonGroup2.add(botonCable3);
  fondoP31.add(botonCable3);
```

```
//se instancia y se dan las propiedades a BC4
  botonCable4.setBackground(new java.awt.Color(153, 255, 255));
  botonCable4.setBounds(743,407,20,20);
  botonCable4.addActionListener(this);
  botonCable4.setVisible(true);
  buttonGroup2.add(botonCable4);
  fondoP31.add(botonCable4);

//se instancia y se dan las propiedades al botón de actualizar
  botonActualizar.setFont(f1);
  botonActualizar.setText("Generar Nuevas Averías");
  botonActualizar.setBounds(588,308,185,25);
  botonActualizar.setToolTipText("Al pulsar este boton las averias de los cables
cambian");

   botonActualizar.addActionListener(this);
   botonActualizar.setVisible(true);
   fondoP31.add(botonActualizar);


//se pone la imagen de fondo
   back =getImage("FondoP3.jpg");
   ImageIcon bgicon = new ImageIcon(back);
   background = new JLabel(bgicon);
   background.setOpaque(true);
   background.setBounds(0,0,bgicon.getIconWidth(), bgicon.getIconHeight());
   fondoP31.add(background,new Integer(0));
   getContentPane().setBackground(Color.white);
   getContentPane().add(fondoP31);

//aqui lo que se hace es rellenar los arrays de propiedades y para ello se llama al
metodo propiedades de los cables (se explicara mas adelante como funciona dicho
metodo)

//los bucles vacios es para dejar pasar algo de tiempo ya que las propiedades de los
cables se generan con una variable aleatoria que depende de este

   for(int a=0;a<2;a++){
       propiedadesC1=panelCables1.propiedades(2);
       for(int i=0;i<1000000;i++){}
       propiedadesC2=panelCables2.propiedades(8);
       for(int i=0;i<2000000;i++){}
       propiedadesC3=panelCables3.propiedades(3);
       for(int i=0;i<4000000;i++){}
       propiedadesC4=panelCables4.propiedades(1);
   }
//array en el que se guardan los caracteres que aparecen en el cablemeter que indican
si se ha pulsado el botón prueba o no

   caracteres[0]='-';
   caracteres[1]='-';
}
```

Este método se lanza nada mas ejecutarse el applet, y como vemos, es el que genera toda la interfaz grafica y el que instancia todos los componentes y añade a estos los controladores de los eventos. Con lo que, los demás métodos son llamados siempre desde esta clase, ya sea directamente o cuando el usuario ejecute alguna acción (p.e: pulsar un boton).

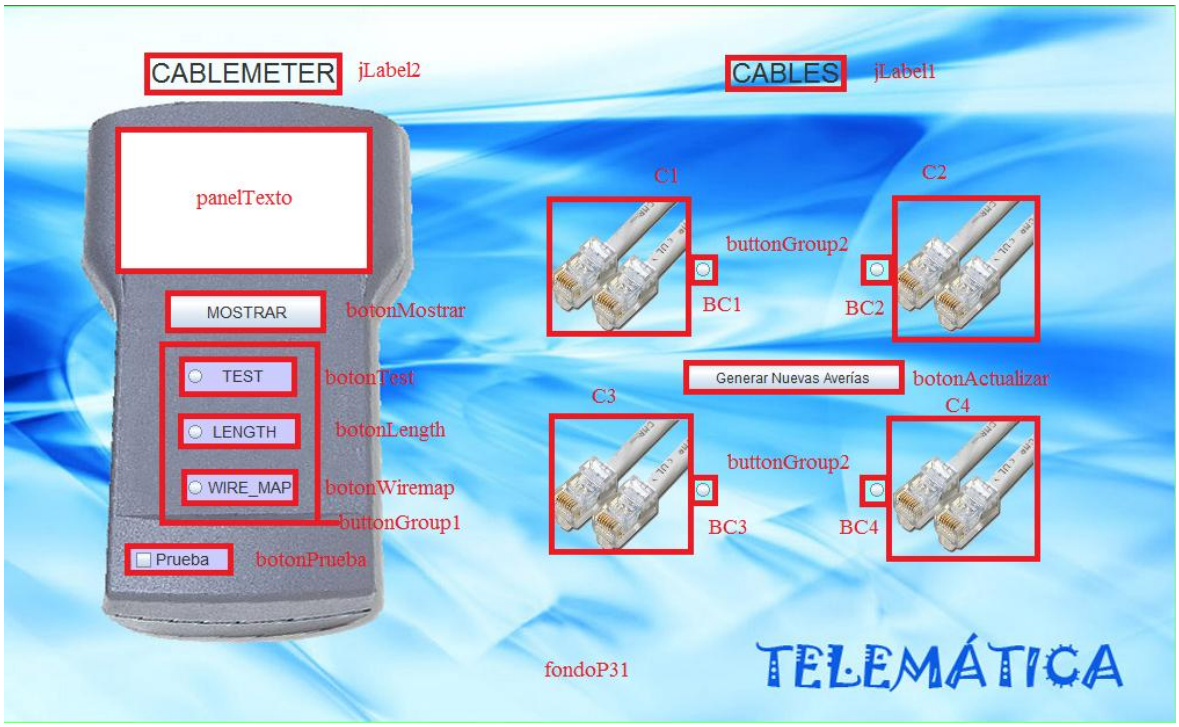


Figura 10

- Metodo *public void actionPerformed(ActionEvent e).*

A este metodo solo se accede cuando se produce un evento asociado a un ActionListener.

Los elementos que pueden capturar un evento de este tipo son: botonMostrar, botonTest, botonLength, botonWiremap, botonPrueba, botonCable1, botonCable2, botonCable3, botonCable4, botonActualizar. A continuación se muestra el código:

```
public void actionPerformed(ActionEvent e) {
        if(e.getSource()==botonMostrar){
            MostrarPulsado();//este metodo se explicara más adelante
        }
        if(e.getSource()==botonTest){
            test=true;//indica que esta seleccionado el boton test para
                      luego usarlo en el metodo MostrarPulsado.
            length=false;
            wire=false;
            panelTexto.setText("\n");
        }
        if(e.getSource()==botonLength){
            test=false;
            length=true; ;//indica que esta seleccionado el boton length
                           para luego usarlo en el metodo MostrarPulsado.
            wire=false;
            panelTexto.setText("\n");
        }
        if(e.getSource()==botonWiremap){
            test=false;
            length=false;
            wire=true; ;//indica que esta seleccionado el boton wiremap para
                           luego usarlo en el metodo MostrarPulsado.
            panelTexto.setText("\n");
        }
        if(e.getSource()==botonPrueba){
            panelTexto.setText("\n");
            if(botonPrueba.isSelected()){//si esta seleccionado se pone
prueba a verdadero para usarlo luego en el metodo MostrarPulsado. Además se
rellena el array caracteres con los caracteres adecuados cuando el botón
esta seleccionado.
                prueba=true;
                caracteres[0]='#';
                caracteres[1]='1';
            }
            else{//se pone prueba a falso para usarlo luego en el metodo
MostrarPulsado. Además se rellena el array caracteres con los caracteres
adecuados cuando el botón no está seleccionado.
                prueba=false;
                caracteres[0]='-';//para modo test
                caracteres[1]='-';
            }
        }
```

```
        if(e.getSource()==botonCable1){
//se pone el valor de cable para luego utilizarlo en el metodo MostrarPulsado.
            panelTexto.setText("\n");
            cable=1;
        }
        if(e.getSource()==botonCable2){
//se pone el valor de cable para luego utilizarlo en el metodo MostrarPulsado.

            panelTexto.setText("\n");
            cable=2;
        }
        if(e.getSource()==botonCable3){
//se pone el valor de cable para luego utilizarlo en el metodo MostrarPulsado.

            panelTexto.setText("\n");
            cable=3;
        }
        if(e.getSource()==botonCable4){
//se pone el valor de cable para luego utilizarlo en el metodo MostrarPulsado.

            panelTexto.setText("\n");
            cable=4;
        }
        if(e.getSource()==botonActualizar){
//se rellenan las variables propiedadesCX(x=1,2,3,4) llamando al método de
porpiedades de panelcables. Como este proceso es aleatorio cada vez que entre aquí
cambiaran las propiedades de los cables
            propiedadesC1=panelCables1.propiedades(1);
            for(int i=0;i<1000000;i++){}
            propiedadesC2=panelCables2.propiedades(5);
            for(int i=0;i<2000000;i++){}
            propiedadesC3=panelCables3.propiedades(9);
            for(int i=0;i<4000000;i++){}
            propiedadesC4=panelCables4.propiedades(7);
            panelTexto.setText("\n");
        }
}
```

Aquí aparecen las variables de tipo boolean test, length y wire, que indican que modo esta seleccionado. La variable prueba es también un boolean que indica si se está usando la prueba o no y, en función de esto se rellena con un valor u otro el array de char caracteres. Y por ultimo aparece la variable cable de tipo int que indica que cable esta seleccionado.

- Metodo `public void mouseClicked(MouseEvent e) {}`
- Metodo `public void  mouseReleased(MouseEvent e){}`
- Metodo `public void mouseEntered(MouseEvent e){}`
- Metodo `void mouseExited(MouseEvente) {}`

Estos métodos están vacios porque no son de utilidad, pero es necesario añadirlos ya que la clase implementa la interfaz MouseListener.

- Metodo `public void mousePressed(MouseEvent e) {}`

A este metodo solo se accede cuando se produce un evento asociado a un MouseListener.

Los elementos que pueden capturar un evento de este tipo son: panelCables1, panelCables2, panelCables3, panelcalbes4. A continuación se muestra el código:

```
public void mousePressed(MouseEvent e) {

        if(e.getSource()==panelCables1){
            for(int i=0;i<5;i++){//inicializacion del bucle auxiliar
                propAux[i]=0;
            }
            banderaC=1;
            propAux[0]=propiedadesC1[0];
            propAux[1]=propiedadesC1[1];
            propAux[2]=propiedadesC1[2];
            propAux[3]=propiedadesC1[3];
            propAux[4]=propiedadesC1[4];
            propAux[5]=propiedadesC1[5];
            VerInfoCable();
        }
        if(e.getSource()==panelCables2){
            for(int i=0;i<5;i++){//inicializacion del bucle auxiliar
                propAux[i]=0;
            }
            banderaC=2;
            propAux[0]=propiedadesC2[0];
            propAux[1]=propiedadesC2[1];
            propAux[2]=propiedadesC2[2];
            propAux[3]=propiedadesC2[3];
            propAux[4]=propiedadesC2[4];
            propAux[5]=propiedadesC2[5];
            VerInfoCable();
        }
        if(e.getSource()==panelCables3){
            for(int i=0;i<5;i++){//inicializacion del bucle auxiliar
                propAux[i]=0;
            }
            banderaC=3;
            propAux[0]=propiedadesC3[0];
            propAux[1]=propiedadesC3[1];
            propAux[2]=propiedadesC3[2];
            propAux[3]=propiedadesC3[3];
            propAux[4]=propiedadesC3[4];
            propAux[5]=propiedadesC3[5];
            VerInfoCable();
        }
```

```
if(e.getSource()==panelCables4){
             for(int i=0;i<5;i++){//inicializacion del bucle auxiliar
                 propAux[i]=0;
             }
            banderaC=4;
            propAux[0]=propiedadesC4[0];
            propAux[1]=propiedadesC4[1];
            propAux[2]=propiedadesC4[2];
            propAux[3]=propiedadesC4[3];
            propAux[4]=propiedadesC4[4];
            propAux[5]=propiedadesC4[5];
            VerInfoCable();
        }
    }
```

Aquí en función del cable clicado se rellenan los valores de la variable *banderaC* y del array *propAux* y luego se llama al método *VerInfocable* (se explicara más adelante).

La variable *banderaC* se utiliza para comprobar que el cable seleccionado es el mismo que el que se quiere ver la solución,es decir, que el botón clicado es el mismo que esta seleccionado,por eso cuando entra a este método se rellena con el valor del cable clicado (1, 2, 3 o 4).

El array *propAux* se rellena con el valor de las propiedades del cable clicado, qué significa cada campo se verá cuando se proceda a utilizar el array de propiedades en los métodos siguientes.

- Metodo `prívate void VerInfoCable(){}`

Este método es el encargado de mostrar la solución. Al hacer click sobre un cable, si es el seleccionado, se abre una ventana indicando el tipo de avería.

A este método se accede cuando es llamado desde alguna parte del código y, eso sucede cuando se hace click en la imagen de un cable. El código de este método es:

```
private void VerInfoCable() {

        if(cable!=0){
              if(cable==banderaC){

                if(propAux[0]==0){//No esta averiado
                   JOptionPane.showMessageDialog(null,"Cable UTP tipo EIA/TIA, 4 pares
de categoria5\n\n"+"El cable esta en perfecto estado y tiene una longitud
de"+propAux[5]+" m","Solucion",1);
                }

                else if((propAux[0]==1)){//Cortocircuito lejano mismo par
                    JOptionPane.showMessageDialog(null,"Cable UTP tipo EIA/TIA, 4 pares
de categoria5\n\n"+"El cable tiene un cortocircuito lejano"+" en los cables de un mismo
par,\npar "+propAux[1],"Solucion",1);
                }

                else if(propAux[0]==2){//Cortocircuito lejano distinto par
                    JOptionPane.showMessageDialog(null,"Cable UTP tipo EIA/TIA, 4 pares
de categoria5\n\n"+"El cable tiene un cortocircuito lejano"+" en los cables de distinto
par,\ncorto cables "+propAux[1],"Solucion",1);
                }

                else if(propAux[0]==3){//Cortocircuito cercano mismo par
                    JOptionPane.showMessageDialog(null,"Cable UTP tipo EIA/TIA, 4 pares
de categoria5\n\n"+"El cable tiene un cortocircuito cercano"+" en los cables de un mismo
par,\npar "+propAux[1],"Solucion",1);
                }

               else if(propAux[0]==4){//Cortocircuito cercano distinto par
                    JOptionPane.showMessageDialog(null,"Cable UTP tipo EIA/TIA, 4 pares
de categoria5\n\n"+"El cable tiene un cortocircuito cercano"+" en los cables de distinto
par,\ncorto cables "+propAux[1],"Solucion",1);
                }

                else if(propAux[0]==5){//Un cable abierto

                  if(propAux[3]==1){//Abierto en el extremo cercano
                    JOptionPane.showMessageDialog(null,"Cable UTP tipo EIA/TIA, 4 pares
de categoria5\n\n"+"Circuito abierto en el cable "+propAux[2]+" en el extremo
cercano","Solucion",1);
                  }

                  else if(propAux[3]==2){//Abierto en el extremo lejano
                    JOptionPane.showMessageDialog(null,"Cable UTP tipo EIA/TIA, 4 pares
de categoria5\n\n"+"Circuito abierto en el cable "+propAux[2]+" en el extremo
lejano","Solucion",1);
                  }
                }
```

```
                else if(propAux[0]==6){//Par abierto

                  if(propAux[3]==1){//Abierto en el extremo cercano
                    JOptionPane.showMessageDialog(null,"Cable UTP tipo EIA/TIA, 4 pares
de categoria5\n\n"+"Circuito abierto en el par "+propAux[1]+" en el extremo
cercano","Solucion",1);
                  }

                  else if(propAux[3]==2){//Abierto en el extremo lejano
                    JOptionPane.showMessageDialog(null,"Cable UTP tipo EIA/TIA, 4 pares
de categoria5\n\n"+"Circuito abierto en el par "+propAux[1]+" en el extremo
lejano","Solucion",1);
                  }
                }

                else{//propAux[0]==7; Cables cambiados

                  if(propAux[3]==1){//Cambiados en el extremo cercano
                    JOptionPane.showMessageDialog(null,"Cable UTP tipo EIA/TIA, 4 pares
de categoria5\n\n"+"Par "+propAux[1]+"  cambiados en el extremo cercano","Solucion",1);
                  }

                  else if(propAux[3]==2){//Cambiados en el extremo lejano
                    JOptionPane.showMessageDialog(null,"Cable UTP tipo EIA/TIA, 4 pares
de categoria5\n\n"+"Par "+propAux[1]+" mal conectado en el extremo
lejano","Solucion",1);
                  }
                }
           }

           else{//cable!=banderaC
             JOptionPane.showMessageDialog(null,"Solo puedes ver la solucion del cable
seleccionado","Aviso",0);
           }

        }

        else{//cable==0
            JOptionPane.showMessageDialog(null,"Debes de seleccionar un cable para poder
ver la solucion","Aviso",0);
        }
}
```

En este método lo primero que se hace es comprobar si hay algún cable seleccionado. En el caso de que no lo haya (*cable==0*) se muestra un mensaje advirtiendo que es necesario seleccionar un cable para poder ver la solución. En el caso de que si haya un cable seleccionado (cable*!=0*) se pasa a comprobar que el cable clicado coincida con el seleccionado.

Si no coinciden (*cable!=banderaC*) se muestra un mensaje indicando que solo se puede ver la solución del cable seleccionado.

Si coinciden hay que comprobar que avería presente el cable seleccionado para mostrar la solución. Esto se hace comprobando el valor de algunos de los campos del array de *propiedades*, en este caso *propAux* que ha sido rellenado con los mismos valores.

En este método solo se comprueba el valor de la posición cero y tres de dicho array, de forma que *propAux[0]* indica el tipo de avería y *propAux[3]* si es en extremo cercano o lejano, en el caso de que sea necesario comprobarlo. En el código anterior se ve claramente que significa cada valor de *propAux[0]* y de *propAux[3]*.

En los mensajes también aparecen *propAux[1]*, *propAux[2]* y *propAux[5]*.

propAux[1] indica que cables están en corto, abiertos y cambiados, el tipo de avería lo indica *propAux[0]* y si es en el extremo lejano o cercano *propAux[3]*.

propAux[2] indica que cable está abierto (en el caso de que sea uno solo), y siempre coincidirá con uno de los cables que indica *propAux[1]*.

propAux[5] simplemente indica la longitud del cable.

- Metodo `private void MostrarPulsado() {}`

Este método es el encargado de mostrar los mensajes en la pantalla del Cablemeter, esto lo hace en función del cable seleccionado, del modo y de si está o no pulsada la prueba.

Al igual que el anterior a este método solo se accede cuando es llamado desde alguna parte del código, y esto sucede cuando se pulsa el botón Mostrar del Cablemeter.

```
private void MostrarPulsado() {
        for(int i=0;i<6;i++){//inicializacion del bucle
            propAux[i]=0;
        }

        if(cable==0){//no se ha seleccionado ningun cable
            JOptionPane.showMessageDialog(null,"No se ha seleccionado ningun
cable","Aviso",0);
        }
        else{//se ha seleccionado algun cable

            if(cable==1){
                propAux[0]=propiedadesC1[0];
                propAux[1]=propiedadesC1[1];
                propAux[2]=propiedadesC1[2];
                propAux[3]=propiedadesC1[3];
                propAux[4]=propiedadesC1[4];
                propAux[5]=propiedadesC1[5];
            }
            else if(cable==2){
                propAux[0]=propiedadesC2[0];
                propAux[1]=propiedadesC2[1];
                propAux[2]=propiedadesC2[2];
                propAux[3]=propiedadesC2[3];
                propAux[4]=propiedadesC2[4];
                propAux[5]=propiedadesC2[5];
            }
```

```
            else if(cable==3){
                propAux[0]=propiedadesC3[0];
                propAux[1]=propiedadesC3[1];
                propAux[2]=propiedadesC3[2];
                propAux[3]=propiedadesC3[3];
                propAux[4]=propiedadesC3[4];
                propAux[5]=propiedadesC3[5];

            }
            else if(cable==4){
               propAux[0]=propiedadesC4[0];
               propAux[1]=propiedadesC4[1];
               propAux[2]=propiedadesC4[2];
               propAux[3]=propiedadesC4[3];
               propAux[4]=propiedadesC4[4];
               propAux[5]=propiedadesC4[5];

            }
            cableEnCorto=((double)propAux[1]/10)+0.01;//Cuando el cable en corto es de
distinto par
                    while(cableEnCorto>0.9){//se hace esto para separar los cables
                        cableEnCorto=cableEnCorto-1;//para luego poder escribirlos asi:
cable1&cable2
                    }
                    cableEnCorto=cableEnCorto*10;

             if(test){//si esta seleccionado test

              if(propAux[0]==0){//El cable no esta averiado
                 panelTexto.setText("\n  PASA\t\tID"+caracteres[0]+caracteres[1]+"\n
              }

               else{//El cable esta averiado

                  if(propAux[0]==1||propAux[0]==2){//Cortocircuito lejano del mismo o
distinto par con o sin prueba
                   panelTexto.setText("\n FALLO\t\tID"+caracteres[0]+caracteres[1]+"\n
"+(int)(propAux[1]/10)+"&"+(int)cableEnCorto+"   CORTO <\t"+propAux[5]+" m");
                  }

                 else if(propAux[0]==3||propAux[0]==4){//Cortocircuito cercano del
mismo o distinto par con y sin prueba
                  panelTexto.setText("\n  FALLO\t\tID"+caracteres[0]+caracteres[1]+"\n
"+(int)(propAux[1]/10)+"&"+(int)cableEnCorto+"   CORTO <\t"+propAux[4]+" m");
                  }

                  else if(propAux[0]==5){//Cable abierto
                    if(propAux[3]==1&prueba){//Abierto en el extremo cercano con prueba
                        panelTexto.setText("\n  FALLO\t\t
ID"+caracteres[0]+caracteres[1]+"\n  "+propAux[2]+"   ABIERTO @  0.0 m");
                    }
```

```
                    else if(propAux[3]==1&!prueba){//Abierto en el extremo cercano sin
prueba
                         if(propAux[2]!=(int)cableEnCorto){
                            panelTexto.setText("\n  FALLO\t\t
ID"+caracteres[0]+caracteres[1]+"\n  "+propAux[2]+"&"+(int)cableEnCorto+"  ABIERTO @
0.0 m");//BIEN
                        }
                        else{
                            panelTexto.setText("\n  FALLO\t\t
ID"+caracteres[0]+caracteres[1]+"\n  "+(int)(propAux[1]/10)+"&"+(int)cableEnCorto+"
ABIERTO @  0.0 m");
                        }

                    }
                    else if(propAux[3]==2&prueba){//Abierto en el extremo lejano con
prueba
                        panelTexto.setText("\n  FALLO\t\t
ID"+caracteres[0]+caracteres[1]+"\n    "+propAux[2]+"  ABIERTO @ "+propAux[4]+" m");
                    }

                    else if(propAux[3]==2&!prueba){//Abierto en el extremo lejano sin
prueba
                        panelTexto.setText("\n
PASA\t\tID"+caracteres[0]+caracteres[1]+"\n  "+propAux[5]+" m");
                    }
                }

                else if(propAux[0]==6){//Par abierto
                    if(propAux[3]==1){//Abierto en el extremo cercano
                        panelTexto.setText("\n  FALLO\t\t
ID"+caracteres[0]+caracteres[1]+"\n  "+(int)(propAux[1]/10)+"&"+(int)cableEnCorto+"
ABIERTO @  0.0 m");
                    }

                    else if(propAux[3]==2){//Abierto en el extremo lejano

                        if(prueba){
                            panelTexto.setText("\n  FALLO\t\t
ID"+caracteres[0]+caracteres[1]+"\n  "+(int)(propAux[1]/10)+"&"+(int)cableEnCorto+"
ABIERTO @ "+propAux[4]+" m");
                        }

                        else{
                            panelTexto.setText("\n
PASA\t\tID"+caracteres[0]+caracteres[1]+"\n  "+propAux[5]+" m");
                        }
                    }
                }

                else{//propAux[0]==7; Pares cambiados
                    if(prueba){//Cambiados en el extremo lejano y cercano con prueba
                        if(propAux[1]==12){
                            panelTexto.setText("   1 2  3 6  4 5  7 8  ");
                            panelTexto.setText("\n   2 1  3 6  4 5  7 8
                               \n"+panelTexto.getText());
                        }

                        else if(propAux[1]==36){
                            panelTexto.setText("   1 2  3 6  4 5  7 8
                              "+panelTexto.getText());
                            panelTexto.setText("\n   1 2  6 3  4 5  7 8
                               \n"+panelTexto.getText());
                        }
```

```
                        else if(propAux[1]==45){
                            panelTexto.setText("   1 2  3 6  4 5  7 8
                               "+panelTexto.getText());
                            panelTexto.setText("\n   1 2  3 6  5 4  7 8
                               \n"+panelTexto.getText());
                        }
                        else if(propAux[1]==78){
                            panelTexto.setText("   1 2  3 6  4 5  7 8
                               "+panelTexto.getText());
                            panelTexto.setText("\n   1 2  3 6  4 5  8 7
                               \n"+panelTexto.getText());
                        }
                        panelTexto.setText("\n  FALLO\t\t
ID"+caracteres[0]+caracteres[1]+"\n MALA CONEXION\n\n  "+panelTexto.getText());
                    }
                    else{//Cambiados en el extremo lejano y cercano sin prueba
                        panelTexto.setText("\n
PASA\t\tID"+caracteres[0]+caracteres[1]+"\n  "+propAux[5]+" m");
                    }
                }
            }
        }

        else if(length){//length seleccionado
            if(propAux[0]==0){//si no esta averiado
                panelTexto.setText("  7 8\t\t"+propAux[5]+" m");
                panelTexto.setText("  4 5\t\t"+propAux[5]+"
                   m\n"+panelTexto.getText());
                panelTexto.setText("  3 6\t\t"+propAux[5]+"
                   m\n"+panelTexto.getText());
                panelTexto.setText("  1 2\t\t"+propAux[5]+"
                   m\n"+panelTexto.getText());
            }
            else if(propAux[0]==1||propAux[0]==3){//corto mismo par
                if(propAux[0]==1){//corto lejano mismo par con y sin prueba
                    longModoLen=propAux[5];
                }
                else if(propAux[0]==3){//corto cercano mismo par con y sin prueba
                    longModoLen=propAux[4];
                }
                if(propAux[1]==12){
                    panelTexto.setText("  7 8\t\t"+propAux[5]+" m");
                    panelTexto.setText("  4 5\t\t"+propAux[5]+"
                           m\n"+panelTexto.getText());
                    panelTexto.setText("  3 6\t\t"+propAux[5]+"
                           m\n"+panelTexto.getText());
                    panelTexto.setText("  1&2 CORTO < "+longModoLen+"
                       m\n"+panelTexto.getText());
                }
                else if(propAux[1]==36){
                    panelTexto.setText("  7 8\t\t"+propAux[5]+" m");
                    panelTexto.setText("  4 5\t\t"+propAux[5]+"
                           m\n"+panelTexto.getText());
                    panelTexto.setText("  1 2\t\t"+propAux[5]+"
                           m\n"+panelTexto.getText());
                    panelTexto.setText("  3&6 CORTO < "+longModoLen+"
                           m\n"+panelTexto.getText());
                }
```

```
            else if(propAux[1]==45){
                panelTexto.setText("  7 8\t\t"+propAux[5]+" m");
                panelTexto.setText("  3 6\t\t"+propAux[5]+"
                        m\n"+panelTexto.getText());
                panelTexto.setText("  1 2\t\t"+propAux[5]+"
                        m\n"+panelTexto.getText());
                panelTexto.setText("  4&5 CORTO < "+longModoLen+"
                        m\n"+panelTexto.getText());
            }
            else{
                panelTexto.setText("  4 5\t\t"+propAux[5]+" m");
                panelTexto.setText("  3 6\t\t"+propAux[5]+"
                        m\n"+panelTexto.getText());
                panelTexto.setText("  1 2\t\t"+propAux[5]+"
                        m\n"+panelTexto.getText());
                panelTexto.setText("  7&8 CORTO < "+longModoLen+"
                        m\n"+panelTexto.getText());
            }
        }
        else if(propAux[0]==2||propAux[0]==4){//corto distinto par
            if(propAux[0]==2){//corto lejano distinto par con y sin prueba
                longModoLen=propAux[5];
            }
            else if(propAux[0]==4){//corto cercano distinto par con y sin prueba
                longModoLen=propAux[4];
            }
            if(propAux[1]==13&&prueba){
                panelTexto.setText("  7 8\t\t"+propAux[5]+" m");
                panelTexto.setText("  4 5\t\t"+propAux[5]+"
                        m\n"+panelTexto.getText());
                panelTexto.setText("  1&3 CORTO < "+longModoLen+"
                        m\n"+panelTexto.getText());
            }
            else if(propAux[1]==13&&!prueba){
                panelTexto.setText("  7 8\t\t"+propAux[5]+" m");
                panelTexto.setText("  4 5\t\t"+propAux[5]+"
                        m\n"+panelTexto.getText());
                panelTexto.setText("   6  ABIERTO @\n"+panelTexto.getText());
                panelTexto.setText("   2  ABIERTO @\n"+panelTexto.getText());
                panelTexto.setText("  1&3 CORTO < "+longModoLen+"
                        m\n"+panelTexto.getText());
            }
            else if(propAux[1]==34&&prueba){
                panelTexto.setText("  7 8\t\t"+propAux[5]+" m");
                panelTexto.setText("  1 2\t\t"+propAux[5]+"
                        m\n"+panelTexto.getText());
                panelTexto.setText("  3&4 CORTO < "+longModoLen+"
                        m\n"+panelTexto.getText());
            }
            else if(propAux[1]==34&&!prueba){
                panelTexto.setText("  7 8\t\t"+propAux[5]+" m");
                panelTexto.setText("   5  ABIERTO @\n"+panelTexto.getText());
                panelTexto.setText("   6  ABIERTO @\n"+panelTexto.getText());
                panelTexto.setText("  1 2\t\t"+propAux[5]+"
                        m\n"+panelTexto.getText());
                panelTexto.setText("  3&4 CORTO < "+longModoLen+"
               m\n"+panelTexto.getText());
            }
```

```
                 else if(propAux[1]==58&&prueba){
                    panelTexto.setText("  3 6\t\t"+propAux[5]+" m");
                    panelTexto.setText("  1 2\t\t"+propAux[5]+"
                           m\n"+panelTexto.getText());
                    panelTexto.setText("  5&8 CORTO < "+longModoLen+"
                           m\n"+panelTexto.getText());
                }
                else if(propAux[1]==58&&!prueba){
                    panelTexto.setText("   7  ABIERTO @");
                    panelTexto.setText("   5  ABIERTO @\n"+panelTexto.getText());
                    panelTexto.setText("  3 6\t\t"+propAux[5]+"
                           m\n"+panelTexto.getText());
                    panelTexto.setText("  1 2\t\t"+propAux[5]+"
                           m\n"+panelTexto.getText());
                    panelTexto.setText("  5&8 CORTO < "+longModoLen+"
                           m\n"+panelTexto.getText());
                }
                else if(propAux[1]==27&&prueba){
                    panelTexto.setText("  4 5\t\t"+propAux[5]+" m");
                    panelTexto.setText("  3 6\t\t"+propAux[5]+"
                           m\n"+panelTexto.getText());
                    panelTexto.setText("  2&7 CORTO < "+longModoLen+"
                           m\n"+panelTexto.getText());
                }
                else if(propAux[1]==27&&!prueba){
                    panelTexto.setText("   8  ABIERTO @");
                    panelTexto.setText("  4 5\t\t"+propAux[5]+"
                           m\n"+panelTexto.getText());
                    panelTexto.setText("  3 6\t\t"+propAux[5]+"
                           m\n"+panelTexto.getText());
                    panelTexto.setText("   1  ABIERTO @"+panelTexto.getText());
                    panelTexto.setText("  2&7 CORTO < "+longModoLen+"
                           m\n"+panelTexto.getText());
                }
            }

            else if(propAux[0]==5){//cable abierto
                if(propAux[3]==1){// Abierto en el extremo cercano
                    if(prueba){// Con prueba
                        if(propAux[2]==1||propAux[2]==2){
                            panelTexto.setText("  7 8\t\t"+propAux[5]+" m");
                            panelTexto.setText("  4 5\t\t"+propAux[5]+"
                                   m\n"+panelTexto.getText());
                            panelTexto.setText("  3 6\t\t"+propAux[5]+"
                                   m\n"+panelTexto.getText());
                            panelTexto.setText("  "+propAux[2]+" ABIERTO @ 0.0
                                   m\n"+panelTexto.getText());
                        }
                        else if(propAux[2]==3||propAux[2]==6){
                            panelTexto.setText("  7 8\t\t"+propAux[5]+" m");
                            panelTexto.setText("  4 5\t\t"+propAux[5]+"
                                   m\n"+panelTexto.getText());
                            panelTexto.setText("  1 2\t\t"+propAux[5]+"
                                   m\n"+panelTexto.getText());
                            panelTexto.setText("  "+propAux[2]+" ABIERTO @ 0.0
                                   m\n"+panelTexto.getText());
                        }
```

```
                    else if(propAux[2]==4||propAux[2]==5){
                        panelTexto.setText("  7 8\t\t"+propAux[5]+" m");
                        panelTexto.setText("  3 6\t\t"+propAux[5]+"
                                m\n"+panelTexto.getText());
                        panelTexto.setText("  1 2\t\t"+propAux[5]+"
                                m\n"+panelTexto.getText());
                        panelTexto.setText("  "+propAux[2]+" ABIERTO @ 0.0
                                m\n"+panelTexto.getText());
                    }
                    else if(propAux[2]==7||propAux[2]==8){
                        panelTexto.setText("  4 5\t\t"+propAux[5]+" m");
                        panelTexto.setText("  3 6\t\t"+propAux[5]+"
                                m\n"+panelTexto.getText());
                        panelTexto.setText("  1 2\t\t"+propAux[5]+"
                                m\n"+panelTexto.getText());
                        panelTexto.setText("  "+propAux[2]+" ABIERTO @ 0.0
                                m\n"+panelTexto.getText());
                    }
                }

                else{// Sin prueba
                    if(propAux[2]==1||propAux[2]==2){
                        panelTexto.setText("  7 8\t\t"+propAux[5]+" m");
                        panelTexto.setText("  4 5\t\t"+propAux[5]+"
                                m\n"+panelTexto.getText());
                        panelTexto.setText("  3 6\t\t"+propAux[5]+"
                                m\n"+panelTexto.getText());
                        panelTexto.setText("  1&2 ABIERTO @ 0.0
                                m\n"+panelTexto.getText());
                    }
                    else if(propAux[2]==3||propAux[2]==6){
                        panelTexto.setText("  7 8\t\t"+propAux[5]+" m");
                        panelTexto.setText("  4 5\t\t"+propAux[5]+"
                                m\n"+panelTexto.getText());
                        panelTexto.setText("  1 2\t\t"+propAux[5]+"
                                m\n"+panelTexto.getText());
                        panelTexto.setText("  3&6 ABIERTO @ 0.0
                                m\n"+panelTexto.getText());
                    }
                    else if(propAux[2]==4||propAux[2]==5){
                        panelTexto.setText("  7 8\t\t"+propAux[5]+" m");
                        panelTexto.setText("  3 6\t\t"+propAux[5]+"
                                m\n"+panelTexto.getText());
                        panelTexto.setText("  1 2\t\t"+propAux[5]+"
                                m\n"+panelTexto.getText());
                        panelTexto.setText("  4&5 ABIERTO @ 0.0
                                m\n"+panelTexto.getText());
                    }
                    else if(propAux[2]==7||propAux[2]==8){
                        panelTexto.setText("  4 5\t\t"+propAux[5]+" m");
                        panelTexto.setText("  3 6\t\t"+propAux[5]+"
                                m\n"+panelTexto.getText());
                        panelTexto.setText("  1 2\t\t"+propAux[5]+"
                                m\n"+panelTexto.getText());
                        panelTexto.setText("  7&8 ABIERTO @ 0.0
                                m\n"+panelTexto.getText());
                    }
                }
            }
```

```
                    else if(propAux[3]==2){//Abierto en el extremo lejano
                       if(prueba){// Con prueba
                           if(propAux[2]==1||propAux[2]==2){
                               panelTexto.setText("  7 8\t\t"+propAux[5]+" m");
                               panelTexto.setText("  4 5\t\t"+propAux[5]+"
                                     m\n"+panelTexto.getText());
                               panelTexto.setText("  3 6\t\t"+propAux[5]+"
                                     m\n"+panelTexto.getText());
                               panelTexto.setText("  "+propAux[2]+" ABIERTO @
                                     "+propAux[4]+" m\n"+panelTexto.getText());
                           }
                           else if(propAux[2]==3||propAux[2]==6){
                               panelTexto.setText("  7 8\t\t"+propAux[5]+" m");
                               panelTexto.setText("  4 5\t\t"+propAux[5]+"
                                     m\n"+panelTexto.getText());
                               panelTexto.setText("  1 2\t\t"+propAux[5]+"
                                     m\n"+panelTexto.getText());
                               panelTexto.setText("  "+propAux[2]+" ABIERTO @
                                     "+propAux[4]+" m\n"+panelTexto.getText());
                           }
                           else if(propAux[2]==4||propAux[2]==5){
                               panelTexto.setText("  7 8\t\t"+propAux[5]+" m");
                               panelTexto.setText("  3 6\t\t"+propAux[5]+"
                                     m\n"+panelTexto.getText());
                               panelTexto.setText("  1 2\t\t"+propAux[5]+"
                                     m\n"+panelTexto.getText());
                               panelTexto.setText("  "+propAux[2]+" ABIERTO @
                                     "+propAux[4]+" m\n"+panelTexto.getText());
                           }
                           else if(propAux[2]==7||propAux[2]==8){
                               panelTexto.setText("  4 5\t\t"+propAux[5]+" m");
                               panelTexto.setText("  3 6\t\t"+propAux[5]+"
                                     m\n"+panelTexto.getText());
                               panelTexto.setText("  1 2\t\t"+propAux[5]+"
                                     m\n"+panelTexto.getText());
                               panelTexto.setText("  "+propAux[2]+" ABIERTO @
                                     "+propAux[4]+" m\n"+panelTexto.getText());
                           }
                       }
                       else{// Sin prueba
                           panelTexto.setText("  7 8\t\t"+propAux[5]+" m");
                           panelTexto.setText("  4 5\t\t"+propAux[5]+"
                                     m\n"+panelTexto.getText());
                           panelTexto.setText("  3 6\t\t"+propAux[5]+"
                                     m\n"+panelTexto.getText());
                           panelTexto.setText("  1 2\t\t"+propAux[5]+"
                                     m\n"+panelTexto.getText());
                       }
                    }
                }
```

```
                else if(propAux[0]==6){//par abierto
                  if(propAux[3]==1){
                      if(propAux[2]==1||propAux[2]==2){
                              panelTexto.setText("  7 8\t\t"+propAux[5]+" m");
                              panelTexto.setText("  4 5\t\t"+propAux[5]+"
                                    m\n"+panelTexto.getText());
                              panelTexto.setText("  3 6\t\t"+propAux[5]+"
                                    m\n"+panelTexto.getText());
                              panelTexto.setText("  1&2  ABIERTO @ 0.0
                                    m\n"+panelTexto.getText());
                          }
                          else if(propAux[2]==3||propAux[2]==6){
                              panelTexto.setText("  7 8\t\t"+propAux[5]+" m");
                              panelTexto.setText("  4 5\t\t"+propAux[5]+"
                                    m\n"+panelTexto.getText());
                              panelTexto.setText("  1 2\t\t"+propAux[5]+"
                                    m\n"+panelTexto.getText());
                              panelTexto.setText("  3&6  ABIERTO @ 0.0
                                    m\n"+panelTexto.getText());
                          }
                          else if(propAux[2]==4||propAux[2]==5){
                              panelTexto.setText("  7 8\t\t"+propAux[5]+" m");
                              panelTexto.setText("  3 6\t\t"+propAux[5]+"
                                    m\n"+panelTexto.getText());
                              panelTexto.setText("  1 2\t\t"+propAux[5]+"
                                    m\n"+panelTexto.getText());
                              panelTexto.setText("  4&5  ABIERTO @ 0.0
                                    m\n"+panelTexto.getText());
                          }
                          else if(propAux[2]==7||propAux[2]==8){
                              panelTexto.setText("  4 5\t\t"+propAux[5]+" m");
                              panelTexto.setText("  3 6\t\t"+propAux[5]+"
                                    m\n"+panelTexto.getText());
                              panelTexto.setText("  1 2\t\t"+propAux[5]+"
                                    m\n"+panelTexto.getText());
                              panelTexto.setText("  7&8  ABIERTO @ 0.0
                                    m\n"+panelTexto.getText());
                          }
                  }
```

```
                    if(propAux[3]==2){
                      if(prueba){
                          if(propAux[2]==1||propAux[2]==2){
                              panelTexto.setText("  7 8\t\t"+propAux[5]+" m");
                              panelTexto.setText("  4 5\t\t"+propAux[5]+"
                                    m\n"+panelTexto.getText());
                              panelTexto.setText("  3 6\t\t"+propAux[5]+"
                                    m\n"+panelTexto.getText());
                              panelTexto.setText("  1&2  ABIERTO @ "+propAux[4]+"
                                    m\n"+panelTexto.getText());
                          }
                          else if(propAux[2]==3||propAux[2]==6){
                              panelTexto.setText("  7 8\t\t"+propAux[5]+" m");
                              panelTexto.setText("  4 5\t\t"+propAux[5]+"
                                    m\n"+panelTexto.getText());
                              panelTexto.setText("  1 2\t\t"+propAux[5]+"
                                    m\n"+panelTexto.getText());
                              panelTexto.setText("  3&6  ABIERTO @ "+propAux[4]+"
                                    m\n"+panelTexto.getText());
                          }
                          else if(propAux[2]==4||propAux[2]==5){
                              panelTexto.setText("  7 8\t\t"+propAux[5]+" m");
                              panelTexto.setText("  3 6\t\t"+propAux[5]+"
                                    m\n"+panelTexto.getText());
                              panelTexto.setText("  1 2\t\t"+propAux[5]+"
                                    m\n"+panelTexto.getText());
                              panelTexto.setText("  4&5  ABIERTO @ "+propAux[4]+"
                                    m\n"+panelTexto.getText());
                          }
                          else if(propAux[2]==7||propAux[2]==8){
                              panelTexto.setText("  4 5\t\t"+propAux[5]+" m");
                              panelTexto.setText("  3 6\t\t"+propAux[5]+"
                                    m\n"+panelTexto.getText());
                              panelTexto.setText("  1 2\t\t"+propAux[5]+"
                                    m\n"+panelTexto.getText());
                              panelTexto.setText("  7&8  ABIERTO @ "+propAux[4]+"
                                    m\n"+panelTexto.getText());
                          }
                      }
                      else{
                          panelTexto.setText("  7 8\t\t"+propAux[5]+" m");
                          panelTexto.setText("  4 5\t\t"+propAux[5]+"
                                    m\n"+panelTexto.getText());
                          panelTexto.setText("  3 6\t\t"+propAux[5]+"
                                    m\n"+panelTexto.getText());
                          panelTexto.setText("  1 2\t\t"+propAux[5]+"
                                    m\n"+panelTexto.getText());
                      }

                    }
                }
```

```
                    else{//propAux[1]==7 pares cambiados
                     if(prueba){
                         if(propAux[1]==12){
                             panelTexto.setText("  7 8\t\t"+propAux[5]+" m");
                             panelTexto.setText("  4 5\t\t"+propAux[5]+"
                                       m\n"+panelTexto.getText());
                             panelTexto.setText("  3 6\t\t"+propAux[5]+"
                                       m\n"+panelTexto.getText());
                             panelTexto.setText("  2 1\tERROR "+propAux[5]+"
                                       m\n"+panelTexto.getText());
                         }
                         else if(propAux[1]==36){
                             panelTexto.setText("  7 8\t\t"+propAux[5]+" m");
                             panelTexto.setText("  4 5\t\t"+propAux[5]+"
                                       m\n"+panelTexto.getText());
                             panelTexto.setText("  1 2\t\t"+propAux[5]+"
                                       m\n"+panelTexto.getText());
                             panelTexto.setText("  6 3\tERROR "+propAux[5]+"
                                       m\n"+panelTexto.getText());
                         }
                         else if(propAux[1]==45){
                             panelTexto.setText("  7 8\t\t"+propAux[5]+" m");
                             panelTexto.setText("  3 6\t\t"+propAux[5]+"
                                       m\n"+panelTexto.getText());
                             panelTexto.setText("  1 2\t\t"+propAux[5]+"
                                       m\n"+panelTexto.getText());
                             panelTexto.setText("  5 4\tERROR "+propAux[5]+"
                                       m\n"+panelTexto.getText());
                         }
                         else if(propAux[1]==78){
                             panelTexto.setText("  4 5\t\t"+propAux[5]+" m");
                             panelTexto.setText("  3 6\t\t"+propAux[5]+"
                                       m\n"+panelTexto.getText());
                             panelTexto.setText("  1 2\t\t"+propAux[5]+"
                                       m\n"+panelTexto.getText());
                             panelTexto.setText("  8 7\tERROR "+propAux[5]+"
                                       m\n"+panelTexto.getText());
                         }
                     }
                     else{
                         panelTexto.setText("  7 8\t\t"+propAux[5]+" m");
                         panelTexto.setText("  4 5\t\t"+propAux[5]+"
                                       m\n"+panelTexto.getText());
                         panelTexto.setText("  3 6\t\t"+propAux[5]+"
                                       m\n"+panelTexto.getText());
                         panelTexto.setText("  1 2\t\t"+propAux[5]+"
                                       m\n"+panelTexto.getText());
                     }
                 }
             }
```

```
            else if(wire){//wiremap seleccionado
              if(propAux[0]==0){//no esta averiado con y sin prueba
                panelTexto.setText("   1 2  3 6  4 5  7 8
                     "+caracteres[0]+caracteres[1]);
                panelTexto.setText("\n   1 2  3 6  4 5  7 8
                     ID\n"+panelTexto.getText());
              }
              else if(propAux[0]==1||propAux[0]==3){//Cortocircuito lejano o
cercano mismo par con y sin prueba
                   if(propAux[1]==12){//par 12
                         panelTexto.setText("   c c
                            "+caracteres[0]+caracteres[1]);
                         panelTexto.setText("\n   1 2  3 6  4 5  7 8
                            ID\n"+panelTexto.getText());
                   }
                   else if(propAux[1]==36){//par 36
                         panelTexto.setText("   c c
                            "+caracteres[0]+caracteres[1]);
                         panelTexto.setText("\n   3 6  1 2  4 5  7 8
                            ID\n"+panelTexto.getText());
                   }
                   else if(propAux[1]==45){//par 45
                         panelTexto.setText("   c c
                            "+caracteres[0]+caracteres[1]);
                         panelTexto.setText("\n   4 5  1 2  3 6  7 8
                            ID\n"+panelTexto.getText());
                   }
                   else{//par 78
                         panelTexto.setText("   c c
                            "+caracteres[0]+caracteres[1]);
                         panelTexto.setText("\n   7 8  1 2  3 6  4 5
                            ID\n"+panelTexto.getText());
                   }
              }
              else if(propAux[0]==2||propAux[0]==4){
                   if(propAux[1]==13){
                         panelTexto.setText("   c c
                            "+caracteres[0]+caracteres[1]);
                         panelTexto.setText("\n   1 3  2  6  4 5  7 8
                            ID\n"+panelTexto.getText());
                   }
                   else if(propAux[1]==34){
                         panelTexto.setText("   c c
                            "+caracteres[0]+caracteres[1]);
                         panelTexto.setText("\n   3 4  1 2  6  5  7 8
                            ID\n"+panelTexto.getText());
                   }
                   else if(propAux[1]==58){
                         panelTexto.setText("   c c
                            "+caracteres[0]+caracteres[1]);
                         panelTexto.setText("\n   5 8  1 2  3 6  4  7
                            ID\n"+panelTexto.getText());
                   }
                   else{//27
                         panelTexto.setText("   c c
                            "+caracteres[0]+caracteres[1]);
                         panelTexto.setText("\n   2 7  1  3 6  4 5  8
                            ID\n"+panelTexto.getText());
                   }
              }
```

```
                else if(propAux[0]==5){//Cable abierto
                   if(propAux[3]==1){//Abierto extremo cercano
                     if(propAux[2]==1){//Abierto cable1
                         if(prueba){
                           panelTexto.setText("     2  3 6  4 5  7 8  #1");
                           panelTexto.setText("\n   a 2  3 6  4 5  7 8
                              ID\n"+panelTexto.getText());
                         }
                         else{
                           panelTexto.setText("                                --");
                           panelTexto.setText("\n   a a  3 6  4 5  7 8
                              ID\n"+panelTexto.getText());
                         }
                     }
                     else if(propAux[2]==2){//Abierto cable2
                         if(prueba){
                           panelTexto.setText("   1    3 6  4 5  7 8  #1");
                           panelTexto.setText("\n   1 a  3 6  4 5  7 8
                              ID\n"+panelTexto.getText());
                         }
                         else{
                           panelTexto.setText("                                --");
                           panelTexto.setText("\n   a a  3 6  4 5  7 8
                              ID\n"+panelTexto.getText());
                         }
                     }
                     else if(propAux[2]==3){//Abierto cable3
                         if(prueba){
                           panelTexto.setText("   1 2    6  4 5  7 8  #1");
                           panelTexto.setText("\n   1 2  a 6  4 5  7 8
                              ID\n"+panelTexto.getText());
                         }
                         else{
                           panelTexto.setText("                                --");
                           panelTexto.setText("\n   1 2  a a  4 5  7 8
                              ID\n"+panelTexto.getText());
                         }
                     }
                     else if(propAux[2]==4){//Abierto cable4
                         if(prueba){
                           panelTexto.setText("   1 2  3 6    5  7 8  #1");
                           panelTexto.setText("\n   1 2  3 6  a 5  7 8
                              ID\n"+panelTexto.getText());
                         }
                         else{
                           panelTexto.setText("                                --");
                           panelTexto.setText("\n   1 2  3 6  a a  7 8
                              ID\n"+panelTexto.getText());
                         }
                     }
                     else if(propAux[2]==5){//Abierto cable5
                         if(prueba){
                           panelTexto.setText("   1 2  3 6  4    7 8  #1");
                           panelTexto.setText("\n   1 2  3 6  4 a  7 8
                              ID\n"+panelTexto.getText());
                         }
                         else{
                           panelTexto.setText("                                --");
                           panelTexto.setText("\n   1 2  3 6  a a  7 8
                              ID\n"+panelTexto.getText());
                         }
                     }
```

```
                    else if(propAux[2]==6){//Abierto cable6
                         if(prueba){
                           panelTexto.setText("   1 2  3    4 5  7 8  #1");
                           panelTexto.setText("\n   1 2  3 a  4 5  7 8
                              ID\n"+panelTexto.getText());
                         }
                         else{
                           panelTexto.setText("                              --");
                           panelTexto.setText("\n   1 2  a a  4 5  7 8
                              ID\n"+panelTexto.getText());
                         }
                     }
                     else if(propAux[2]==7){//Abierto cable7
                         if(prueba){
                           panelTexto.setText("   1 2  3 6  4 5    8  #1");
                           panelTexto.setText("\n   1 2  3 6  4 5  a 8
                              ID\n"+panelTexto.getText());
                         }
                         else{
                           panelTexto.setText("                              --");
                           panelTexto.setText("\n   1 2  3 6  4 5  a a
                              ID\n"+panelTexto.getText());
                         }
                     }
                     else{//Abierto cable8
                         if(prueba){
                           panelTexto.setText("   1 2  3 6  4 5  7    #1");
                           panelTexto.setText("\n   1 2  3 6  4 5  7 a
                              ID\n"+panelTexto.getText());
                         }
                         else{
                           panelTexto.setText("                              --");
                           panelTexto.setText("\n   1 2  3 6  4 5  a a
                              ID\n"+panelTexto.getText());
                         }
                     }
                   }
                   else if(propAux[3]==2){//Abierto extremo lejano
                     if(propAux[2]==1){
                         if(prueba){
                           panelTexto.setText("   a 2  3 6  4 5  7 8  #1");
                           panelTexto.setText("\n   1 2  3 6  4 5  7 8
                              ID\n"+panelTexto.getText());
                         }
                         else{
                           panelTexto.setText("                              --");
                           panelTexto.setText("\n   1 2  3 6  4 5  7 8
                              ID\n"+panelTexto.getText());
                         }
                     }
                     else if(propAux[2]==2){
                         if(prueba){
                           panelTexto.setText("   1 a  3 6  4 5  7 8  #1");
                           panelTexto.setText("\n   1 2  3 6  4 5  7 8
                              ID\n"+panelTexto.getText());
                         }
                         else{
                           panelTexto.setText("                              --");
                           panelTexto.setText("\n   1 2  3 6  4 5  7 8
                              ID\n"+panelTexto.getText());
                         }
                     }
```

```
                    else if(propAux[2]==3){
                        if(prueba){
                          panelTexto.setText("   1 2  a 6  4 5  7 8  #1");
                          panelTexto.setText("\n   1 2  3 6  4 5  7 8
                             ID\n"+panelTexto.getText());
                        }
                        else{
                          panelTexto.setText("                                --");
                          panelTexto.setText("\n   1 2  3 6  4 5  7 8
                             ID\n"+panelTexto.getText());
                        }
                    }
                    else if(propAux[2]==4){
                        if(prueba){
                          panelTexto.setText("   1 2  3 6  a 5  7 8  #1");
                          panelTexto.setText("\n   1 2  3 6  4 5  7 8
                             ID\n"+panelTexto.getText());
                        }
                        else{
                          panelTexto.setText("                                --");
                          panelTexto.setText("\n   1 2  3 6  4 5  7 8
                             ID\n"+panelTexto.getText());
                        }
                    }
                    else if(propAux[2]==5){
                        if(prueba){
                          panelTexto.setText("   1 2  3 6  4 a  7 8  #1");
                          panelTexto.setText("\n   1 2  3 6  4 5  7 8
                             ID\n"+panelTexto.getText());
                        }
                        else{
                          panelTexto.setText("                                --");
                          panelTexto.setText("\n   1 2  3 6  4 5  7 8
                             ID\n"+panelTexto.getText());
                        }
                    }
                    else if(propAux[2]==6){
                        if(prueba){
                          panelTexto.setText("   1 2  3 a  4 5  7 8  #1");
                          panelTexto.setText("\n   1 2  3 6  4 5  7 8
                             ID\n"+panelTexto.getText());
                        }
                        else{
                          panelTexto.setText("                                --");
                          panelTexto.setText("\n   1 2  3 6  4 5  7 8
                             ID\n"+panelTexto.getText());
                        }
                    }
```

```
                    else if(propAux[2]==7){
                      if(prueba){
                        panelTexto.setText("   1 2  3 6  4 5  a 8  #1");
                        panelTexto.setText("\n   1 2  3 6  4 5  7 8
                           ID\n"+panelTexto.getText());
                      }
                      else{
                        panelTexto.setText("                                --");
                        panelTexto.setText("\n   1 2  3 6  4 5  7 8
                           ID\n"+panelTexto.getText());
                      }
                    }
                    else{
                      if(prueba){
                        panelTexto.setText("   1 2  3 6  4 5  7 a  #1");
                        panelTexto.setText("\n   1 2  3 6  4 5  7 8
                           ID\n"+panelTexto.getText());
                      }
                      else{
                        panelTexto.setText("                                --");
                        panelTexto.setText("\n   1 2  3 6  4 5  7 8
                           ID\n"+panelTexto.getText());
                      }
                    }

                  }

                }

                else if(propAux[0]==6){//Par abierto
                  if(propAux[3]==1){//Abierto extremo cercano
                    if(propAux[1]==12){
                      if(prueba){
                        panelTexto.setText("        3 6  4 5  7 8  #1");
                        panelTexto.setText("\n   a a  3 6  4 5  7 8
                           ID\n"+panelTexto.getText());
                      }
                      else{
                        panelTexto.setText("                                --");
                        panelTexto.setText("\n   a a  3 6  4 5  7 8
                           ID\n"+panelTexto.getText());
                      }
                    }
                    else if(propAux[1]==36){
                      if(prueba){
                        panelTexto.setText("   1 2       4 5  7 8  #1");
                        panelTexto.setText("\n   1 2  a a  4 5  7 8
                           ID\n"+panelTexto.getText());
                      }
                      else{
                        panelTexto.setText("                                --");
                        panelTexto.setText("\n   1 2  a a  4 5  7 8
                           ID\n"+panelTexto.getText());
                      }
                    }
```

```
                    else if(propAux[1]==45){
                      if(prueba){
                        panelTexto.setText("   1 2  3 6       7 8  #1");
                        panelTexto.setText("\n   1 2  3 6  a a  7 8
                          ID\n"+panelTexto.getText());
                      }
                      else{
                        panelTexto.setText("                                --");
                        panelTexto.setText("\n   1 2  3 6  a a  7 8
                          ID\n"+panelTexto.getText());
                      }
                    }
                    else{
                      if(prueba){
                        panelTexto.setText("   1 2  3 6  4 5       #1");
                        panelTexto.setText("\n   1 2  3 6  4 5  a a
                          ID\n"+panelTexto.getText());
                      }
                      else{
                        panelTexto.setText("                                --");
                        panelTexto.setText("\n   1 2  3 6  4 5  a a
                          ID\n"+panelTexto.getText());
                      }
                    }
                  }
                  else if(propAux[3]==2){//abierto extremo lejano
                    if(propAux[1]==12){
                      if(prueba){
                        panelTexto.setText("   a a  3 6  4 5  7 8  #1");
                        panelTexto.setText("\n   1 2  3 6  4 5  7 8
                          ID\n"+panelTexto.getText());
                      }
                      else{
                        panelTexto.setText("                                --");
                        panelTexto.setText("\n   1 2  3 6  4 5  7 8
                          ID\n"+panelTexto.getText());
                      }
                    }
                    else if(propAux[1]==36){
                      if(prueba){
                        panelTexto.setText("   1 2  a a  4 5  7 8  #1");
                        panelTexto.setText("\n   1 2  3 6  4 5  7 8
                          ID\n"+panelTexto.getText());
                      }
                      else{
                        panelTexto.setText("                                --");
                        panelTexto.setText("\n   1 2  3 6  4 5  7 8
                          ID\n"+panelTexto.getText());
                      }
                    }
```

```
                    else if(propAux[1]==45){
                        if(prueba){
                          panelTexto.setText("   1 2  3 6  a a  7 8  #1");
                          panelTexto.setText("\n   1 2  3 6  4 5  7 8
                             ID\n"+panelTexto.getText());
                        }
                        else{
                          panelTexto.setText("                                --");
                          panelTexto.setText("\n   1 2  3 6  4 5  7 8
                             ID\n"+panelTexto.getText());
                        }
                    }
                    else{
                        if(prueba){
                          panelTexto.setText("   1 2  3 6  4 5  a a  #1");
                          panelTexto.setText("\n   1 2  3 6  4 5  7 8
                             ID\n"+panelTexto.getText());
                        }
                        else{
                          panelTexto.setText("                                --");
                          panelTexto.setText("\n   1 2  3 6  4 5  7 8
                             ID\n"+panelTexto.getText());
                        }
                    }
                }
                else{//Cables cambiados
                    if(prueba){//Cambiados extremo cercano
                      if(propAux[1]==12){
                            panelTexto.setText("   1 2  3 6  4 5  7 8  #1");
                            panelTexto.setText("\n   2 1  3 6  4 5  7 8
                               ID\n"+panelTexto.getText());
                      }
                      else if(propAux[1]==36){
                            panelTexto.setText("   1 2  3 6  4 5  7 8  #1");
                            panelTexto.setText("\n   1 2  6 3  4 5  7 8
                               ID\n"+panelTexto.getText());
                      }
                      else if(propAux[1]==45){
                            panelTexto.setText("   1 2  3 6  4 5  7 8  #1");
                            panelTexto.setText("\n   1 2  3 6  5 4  7 8
                               ID\n"+panelTexto.getText());
                      }
                      else{
                            panelTexto.setText("   1 2  3 6  4 5  7 8  #1");
                            panelTexto.setText("\n   1 2  3 6  4 5  8 7
                               ID\n"+panelTexto.getText());
                      }
                    }
                    else{
                        panelTexto.setText("   1 2  3 6  4 5  7 8  --");
                        panelTexto.setText("\n   1 2  3 6  4 5  7 8
                               ID\n"+panelTexto.getText());
                    }
                }
            }
            else{//si no hay ningun modo
                JOptionPane.showMessageDialog(null,"SELECCIONA UN MODO","Aviso",1);
            }
        }
    }
```

Este método es el más extenso de todos, voy a explicar cómo funciona de forma general.

Lo primero que se hace es inicializar el array *propAux* a cero. A continuación se comprueba si hay algún cable seleccionado, si no lo hay (cable=0) se muestra un mensaje advirtiéndolo, en el caso de que si lo haya se identifica cual es y se llena el array *propAux* con las propiedades del cable seleccionado.

En el siguiente paso se procede a rellenar la variable *cableEnCorto* que se rellena con el segundo valor de *propAux[1]* que es utilizado para poder escribir en la pantalla del Cablemeter los cables en abierto con el formato "*num1&num2"*, donde *num1* sería el primer número de *propAux[1]* y *num2 cableEnCorto*.

Tras esto se comprueba si alguno de los modos esta seleccionado, si no hay ninguno seleccionado se muestra un mensaje indicándolo, luego, en función del modo seleccionado, del tipo de avería y de si esta seleccionada la prueba o no se muestran los mensajes en la pantalla del Cablemeter (para más información ojear el código).

Para finalizar decir que en este método aparecen las variables *propAux[4]* y *LongModoLen*, donde *propAux[4]* tiene la longitud de la mitad del cable para usarlo cuando la avería es cercana, y *LongModoLen* tiene la longitud del cable en modo Length.

- Metodo `public Image getImage(String im){}`

Este método lo único que hace es devolver una imagen, es decir, coge la imagen que tiene por nombre la que indica el *String* que se le pasa como parámetro, si el nombre no existe lanzará una excepción.

Este método es usado por esta misma clase(*Práctica_Cablemeter*) y por la clase *PanelCables*, ya que la clase *Práctica_Cablemeter* al heredar de *JApplet* es la única que tiene permisos para abrir imágenes en las carpetas locales.

```
public Image getImage(String im){

      return getImage(getDocumentBase(),im);
}
```

5.2.2. PanelCables

El código de la clase, desde una perspectiva general, es el que se muestra a continuación.

```
import java.awt.*;
import java.util.Random;

public class PanelCables extends javax.swing.JPanel {

    private int longitud;
    private int averiado;
    private int tipoAveria;
    private int paresCorto[]=new int[8];
    private Random rnd = new Random(); // crea la semilla
    private int propiedades[]=new int[6];
    private int paridad=0;
    private double aux,prop2;
    private Image fondo;

    public PanelCables(Práctica_Cablemeter ap){}

    public void paint(Graphics g) {}

    public int[] propiedades(int j){}

}
```

Ahora se explicaran en qué consiste cada uno de los métodos y para que se utilizan las variables definidas.

- Constructor `public PanelCables(Práctica_Cablemeter ap){}`

Este es el constructor de la clase que en este caso se trata de un *JPanel*, el código es el siguiente.

```
public PanelCables(Práctica_Cablemeter ap){
        this.setSize(100,100);
        fondo=ap.getImage("rj45.png");
        //inicializacion del buffer de averias
        //se inicializa aqui porque es fijo
        paresCorto[0]=12;
        paresCorto[1]=13;
        paresCorto[2]=36;
        paresCorto[3]=34;
        paresCorto[4]=45;
        paresCorto[5]=58;
        paresCorto[6]=78;
        paresCorto[7]=27;
    }
```

Como se puede observar al constructor se le pasa un objeto de la clase *Práctica_Cablemeter*, esto se hace para cargar la imagen de los cables, que como ya dijimos solo la clase que hereda de *JApplet* puede hacerlo.

En el constructor lo que se hace es dar un tamaño al *JPanel*, se carga la imagen en la variable *fondo* y se rellena el array de *paresCorto*, que es el que utilizaremos para decir que cables (del mismo o de distinto par) están averiados, ya sea cortocircuito, circuito abierto o cualquier otra avería.

En las posiciones pares de este array están situados los cables del mismo par, y en las impares cables de distinto par.

- Metodo `public void paint(Graphics g) {}`

Este método es el encargado de dibujar el fondo en el *JPanel*, en este caso de poner por fondo la imagen de los cables.

```
public void paint(Graphics g) {
        Dimension tamanio = getSize();
        g.drawImage(fondo, 0, 0, tamanio.width, tamanio.height, null);
        this.setOpaque(false);
        super.paint(g);
        this.repaint();
    }
```

Este método simplemente pone de fondo del *JPanel* la imagen que hay cargada en la variable *fondo*.

- Método `public int[] propiedades(int j){}`

Este método es el utilizado por la clase *Práctica_Cablemeter* para dotar de propiedades a los cables, como se observa recibe un entero como parámetro de entrada, esto es para hacer mas aleatorias las propiedades que generan los cables.

Este método no es muy eficiente porque tiene que generar una cantidad elevada de numero aleatorios y en alguna ocasión el numero aleatorio generado deberá de ser par o impar, de forma que hasta que no se genere el numero deseado no podrá continuar con la ejecución del código, lo cual puede hacer que el applet se relentice, aunque como norma general no se aprecia esa espera.

A continuación se muestra el código del método.

```
public int[] propiedades(int j){

        rnd.setSeed(System.currentTimeMillis()*j);
        longitud=(int)(rnd.nextDouble()*20+4);//se genera la longitud del cable

          /*
        *numero aleatorio entre 0 y 2
        *para generar si hay averia o no
        *NO averia si <1, Si averia si >=1
        */

        if((int)(rnd.nextDouble()*3)<1){
            averiado=0; // NO averiado
            //Se generan la longitud de los pares

            propiedades[0]=averiado;
            propiedades[1]=longitud;//longPares12;
            propiedades[2]=longitud;//longPares36;
            propiedades[3]=longitud;//longPares45;
            propiedades[4]=longitud;//longPares78;
            propiedades[5]=longitud;
            return propiedades;
        }
        else{
            rnd.setSeed((System.currentTimeMillis()*j));
            tipoAveria=(int)(rnd.nextDouble()*15);


            /*
             tipoAveria € [0,2)--> Cortocircuito lejano mismo par--> averiado=1
             *tipoAveria € [2,4)--> Cortocircuito lejano distinto par-->averiado=2
             *tipoAveria € [4,6)--> Cortocircuito cercano mismo par--> averiado=3
             *tipoAveria € [6,8)--> Cortocircuito cercano distino par-->averiado=4
             *tipoAveria € [8,10)--> Cable en circuito abierto--> averiado=5
             *tipoAveria € [10,12)--> Par en circuito abierto--> averiado=6
             *tipoAveria € [12,14]--> Cambio en las conexiones de los cables-->
                                                                    averiado=7
             */
            if(tipoAveria<2){
                averiado=1;
                do{
                   rnd.setSeed((System.currentTimeMillis()*j));
                   paridad=(int)((rnd.nextDouble()*8));
                }while(paridad!=0&&paridad!=2&&paridad!=4&&paridad!=6);
                //La paridad indica que cables del mismo par estan en corto
            }
            else if(tipoAveria>=2&&tipoAveria<4){
                averiado=2;
                do{
                   rnd.setSeed((System.currentTimeMillis()*j));
                   paridad=(int)(rnd.nextDouble()*8);
                }while(paridad!=1&&paridad!=3&&paridad!=5&&paridad!=7);
                //La paridad indica que cables del mismo par estan en corto
            }
```

```
            else if(tipoAveria>=4&&tipoAveria<6){
                averiado=3;
                do{
                   rnd.setSeed((System.currentTimeMillis()*j));
                   paridad=(int)(rnd.nextDouble()*8);
                }while(paridad!=0&&paridad!=2&&paridad!=4&&paridad!=6);
            }
            else if(tipoAveria>=6&&tipoAveria<8){
                averiado=4;
                do{
                   rnd.setSeed((System.currentTimeMillis()*j));
                   paridad=(int)(rnd.nextDouble()*8);
                }while(paridad!=1&&paridad!=3&&paridad!=5&&paridad!=7);
            }
            else if(tipoAveria>=8&&tipoAveria<10){
                averiado=5;
                if((int)(rnd.nextDouble()*9)<4){

                    propiedades[3]=1;//indica que el cable abierto lo esta en el
                                     extremo cercano
                    //el cable esta en propiedades[2]
                 }
                 else{
                    propiedades[3]=2;//indica que el cable abierto lo esta en el
                                     extremo lejano
                    //el cable esta en propiedades[2]
                 }
            }
            else if(tipoAveria>=10&&tipoAveria<12){
                averiado=6;
                do{// Genera el par que va a estar en circuito abierto
                   rnd.setSeed((System.currentTimeMillis()*j));
                   paridad=(int)((rnd.nextDouble()*8));
                }while(paridad!=0&&paridad!=2&&paridad!=4&&paridad!=6);

                 if((int)(rnd.nextDouble()*9)<3){

                    propiedades[3]=1;//indica que el par que esta abierto lo esta
                                     en el extremo cercano

                 }
                 else{
                    propiedades[3]=2;//indica que el par que esta abierto lo esta
                                     en el extremo cercano
                 }
            }
            else if(tipoAveria>=12&&tipoAveria<14){
                averiado=7;
                do{// Genera el par que va a estar mal conectado
                   rnd.setSeed((System.currentTimeMillis()*j+1));
                   paridad=(int)((rnd.nextDouble()*8));
                }while(paridad!=0&&paridad!=2&&paridad!=4&&paridad!=6);
            }

                 rnd.setSeed((System.currentTimeMillis()));//semilla para la próxima
                                                 generacion de un numero aleatorio
                 propiedades[0]=averiado;//indica el tipo de averia
                 propiedades[1]=paresCorto[paridad];//coge del array un par
                                                 aleatorio de los que estan en corto
                 aux=rnd.nextDouble()*9;
```

```
            if((int)aux<5){//En propiedades[2] va el cable que esta abierto
                           //Pero a veces muestra el par como abierto
                          //por lo que para que coincidan utilizo el par
                         //para seleccionar el cable
              propiedades[2]=(int)(propiedades[1]/10);
            }
            else if(aux>=5){
                   prop2=((double)propiedades[1]/10)+0.01;
                   while(prop2>0.9){
                        prop2=prop2-1;
                    }
                   propiedades[2]=(int)(prop2*10);
            }
              // propiedades[3] ya esta relleno arriba
               propiedades[4]=(longitud/2);//La mitad de la longitud para las
                                           averias cercanas
               propiedades[5]=longitud;//Indica la longitud
               return propiedades;
        }
}
```

De forma general lo que hace este método es dotar de propiedades a los cables, para ello primero genera un numero aleatorio entre cero y dos, si el numero generado es menor que uno el cable no estará averiado y se rellenan los campos del array *propiedades* con su valores correspondientes, si por el contrario el numero generado es mayor que uno el cable esta averiado y se rellenaran los campos del array con los valores correspondientes a las averías. Cómo se implementa esto esta explicado en el código, y se hace de forma aleatoria en todo momento.

Las variables que quedan por comentar son:

- *longitud*: indica la longitud del cable, que es como máximo 24 metros y como minimo de 4, y es calculada de forma aleatoria.
- *averiado*: esta variable indica si esta averiado el cable o no, también se elige de forma aleatoria.
- *tipoAveria*: indica con un numero el tipo de averia y al igual que las anteriores se calcula de forma aleatoria.
- *paridad*: es un numero aleatorio que se utiliza para seleccionar uno de los campos del array paresCorto.
- *aux* y *prop2*: son dos variables aleatorias que se utilizan para cálculos intermedios.

5.3. Clases de la práctica 3b

5.3.1. Práctica3_ping

El código que se muestra a continuación es el de la clase desde un punto de vista general, y al igual que en el punto anterior se explicaran uno a uno los distintos métodos que la forman.

```
import java.awt.Color;
import java.awt.Font;
import java.awt.Image;
import java.io.FileNotFoundException;
import java.util.Random;
import javax.swing.JOptionPane;
import javax.swing.Timer;
import java.awt.event.*;
import java.io.BufferedReader;
import java.io.FileReader;
import java.net.URL;
import javax.swing.ImageIcon;
import javax.swing.JLabel;
import javax.swing.JLayeredPane;


public class Práctica3_ping extends javax.swing.JApplet implements
ActionListener, MouseListener{

    private int i=0,numeroCaptura=0;
    private ActionListener lista[];
    private Timer timer;
    private Ventana_captura captura;
    private Random rnd = new Random(); // crea la semilla
    private JLayeredPane fondoP3b;
    private javax.swing.JButton EnviarPing;
    private javax.swing.JButton MostrarC;
    Image back,image;
    OScope osc;
    ImageIcon bgicon;
    private JLabel background,osciloscopio;
```

```
private int []
ping1={1,0,1,0,1,0,1,0,1,0,1,0,1,0,1,0,1,0,1,0,1,0,1,0,1,0,1,0,1,0,1,0
,1,0,1,0,1,0,1,0,1,0,1,0,1,0,1,0,1,0,1,0,1,0,1,0,1,0,1,0,1,0,0,0,0,0,0,0,0,0,1
,1,1,0,0,0,1,1,1,1,1,1,0,1,0,0,0,1,0,0,1,1,0,1,1,0,0,0,0,1,1,1,1,0,1,1,1,0,0,0
,0,0,0,0,0,0,1,0,0,0,0,1,0,0,1,1,0,1,0,1,1,0,0,0,1,0,1,1,1,0,0,0,0,0,0,1,1,1,0
,0,1,1,0,1,1,0,0,0,0,1,0,0,0,0,0,0,0,0,0,0,0,0,0,1,0,1,0,0,0,1,0,0,0,0,0,0,0,0
,0,0,0,0,0,0,0,0,0,0,1,1,1,1,0,0,0,0,0,0,0,0,1,0,1,1,0,1,1,1,1,1,0,0,0,0,0,0,0
,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,1,1,0,0,0,0,0,0,0,1,1,1,1,1,0,0,1,1,1,1,0,0,0
,0,0,0,0,1,1,0,1,0,1,1,0,0,0,0,0,0,0,1,0,0,0,0,0,0,0,1,1,0,1,1,0,0,1,0,0,1,1,0
,1,0,1,1,0,0,0,0,0,0,0,1,0,0,0,0,0,0,0,1,0,0,0,0,1,0,0,0,0,0,1,0,0,0,0,0,0,0,0
,0,0,0,0,1,0,1,1,0,0,1,0,0,0,0,1,1,0,0,0,0,0,0,0,0,0,0,1,0,0,0,0,0,0,0,0,0,0,0
,0,0,0,0,0,1,1,0,0,0,0,1,0,1,0,0,0,0,1,1,0,0,1,0,0,0,1,1,0,1,1,0,0,0,1,1,0,0,0
,1,0,0,1,1,0,1,0,1,0,0,1,1,0,0,1,1,0,0,1,1,0,1,1,1,0,0,1,1,0,0,0,0,1,0,1,1,0,1
,0,0,1,0,1,1,0,0,1,0,1,0,1,1,0,1,1,0,1,0,1,1,0,0,0,1,1,0,1,1,0,1,0,1,1,0,1,1,0
,0,1,1,1,0,1,1,0,1,1,1,1,1,0,1,1,0,0,0,0,0,1,1,1,0,1,0,0,0,1,1,1,0,0,1,0,0,1,1,1
,0,1,1,0,0,1,1,1,0,0,0,1,0,1,1,1,0,1,0,1,0,1,1,1,0,0,1,1,0,1,1,1,0,1,1,1,0,1,1
,1,0,1,0,0,0,0,1,1,0,0,1,0,0,0,1,1,0,1,1,0,0,0,1,1,0,0,0,1,0,0,1,1,0,1,0,1,0,0
,1,1,0,0,1,1,0,0,1,1,0,1,1,1,0,0,1,1,0,0,0,0,1,0,1,1,0,1,0,0,1,0,1,1,0,0,0,0,1
,0,0,0,1,0,0,0,1,1,0,1,1,1,1,0,1,0,1,1,1,1,0,0,0,1,0,0,1};//ping captura1;

private int []
ping2={1,0,1,0,1,0,1,0,1,0,1,0,1,0,1,0,1,0,1,0,1,0,1,0,1,0,1,0,1,0,1,0
,1,0,1,0,1,0,1,0,1,0,1,0,1,0,1,0,1,0,1,0,1,0,1,0,1,0,1,0,1,0,0,0,0,0,0,0,0,0,1
,1,1,0,0,0,1,1,1,1,1,1,0,1,0,0,0,1,0,0,1,1,0,1,1,0,0,0,0,1,1,1,1,0,1,1,1,0,0,0
,0,0,0,0,0,0,1,0,0,0,0,1,0,0,1,1,0,1,0,1,1,0,0,0,1,0,1,1,1,0,0,0,0,0,0,1,1,1,0
,0,1,1,0,1,1,0,0,0,0,1,0,0,0,0,0,0,0,0,0,0,0,0,0,1,0,1,0,0,0,1,0,0,0,0,0,0,0,0
,0,0,0,0,0,0,0,0,0,0,1,1,1,1,0,0,0,0,1,0,0,0,1,0,1,0,1,0,0,0,1,0,0,0,0,0,0,0,0
,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,1,1,0,0,0,0,0,0,0,1,1,0,1,1,0,0,1,1,0,1,1,1,1
,0,1,0,0,1,1,0,1,0,1,1,0,0,0,0,0,0,0,1,0,0,0,0,0,0,0,1,1,0,1,1,0,0,1,0,0,1,1,0
,1,0,1,1,0,0,0,0,0,0,0,1,0,0,0,0,0,0,0,1,0,0,0,0,1,0,0,0,0,0,1,0,0,0,0,0,0,0,0
,0,0,0,0,1,0,1,1,0,0,1,0,0,0,0,0,1,0,0,0,0,0,0,0,0,0,0,1,0,0,0,0,0,0,0,0,0,0,0
,0,0,0,0,0,1,1,0,1,0,0,1,0,1,0,0,0,0,1,1,0,0,1,0,0,0,1,1,0,1,1,0,0,0,1,1,0,0,0
,1,0,0,1,1,0,1,0,1,0,0,1,1,0,0,1,1,0,0,1,1,0,1,1,1,0,0,1,1,0,0,0,0,1,0,1,1,0,1
,0,0,1,0,1,1,0,0,1,0,1,0,1,1,0,1,1,0,1,0,1,1,0,0,0,1,1,0,1,1,0,1,0,1,1,0,1,1,0
,0,1,1,1,0,1,1,0,1,1,1,1,1,0,1,1,0,0,0,0,0,1,1,1,0,1,0,0,0,1,1,1,0,0,1,0,0,1,1,1
,0,1,1,0,0,1,1,1,0,0,0,1,0,1,1,1,0,1,0,1,0,1,1,1,0,0,1,1,0,1,1,1,0,1,1,1,0,1,1
,1,0,1,0,0,0,0,1,1,0,0,1,0,0,0,1,1,0,1,1,0,0,0,1,1,0,0,0,1,0,0,1,1,0,1,0,1,0,0
,1,1,0,0,1,1,0,0,1,1,0,1,1,1,0,0,1,1,0,0,0,0,1,0,1,1,0,1,0,0,1,0,1,1,0,0,0,0,1
,1,0,0,1,1,1,1,1,1,1,0,1,0,1,0,0,0,0,1,0,0,0,0,1,0,0,1,0};para captura2

//Aqui irian private int[] ping3, prívate int[] ping4 y private int[] ping5,
que serian muy similares a estos, por ahorrar espacio no los he añadido

private int [][] pings={ping1,ping2,ping3,ping4,ping5};

public void init(){}

public Image getImage(String im){}

public void actionPerformed(ActionEvent e) {}

public void mouseClicked(MouseEvent e) {}
public void mousePressed(MouseEvent e) {}//vacio
public void mouseReleased(MouseEvent e) {}//vacio
public void mouseEntered(MouseEvent e) {}//vacio
public void mouseExited(MouseEvent e) {}//vacio
```

- Metodo `public void init(){}`

En este método es en el que se inicializan y se instancian los componentes declarados en la parte anterior, voy a ir explicando que es cada uno y en el caso de los que sean visibles, con que elemento de la interfaz grafica se corresponde.

```
public void init(){
        EnviarPing = new javax.swing.JButton();//se corresponde con Enviar
                                                                    Ping
        MostrarC = new javax.swing.JButton();//se corresponde con Mostrar
                                                                  Captura
        fondoP3b = new JLayeredPane();//fondo de la clase
        osciloscopio=new javax.swing.JLabel();//muestra la imagen del
                                                          osciloscopio
        Font f = new Font("Arial",Font.PLAIN,14);
        fondoP3b.setBackground(Color.white);//configuracion del fondo
        setBackground(Color.white);
        setSize(1010,617);
        fondoP3b.setLayout(null);
        setVisible(true);
        //configuracion de la etiqueta osciloscopio(que es la imagen)
        osciloscopio.setIcon(new
       javax.swing.ImageIcon(getClass().getResource("osciloscopio.png")));
        osciloscopio.setBounds(500,150,450,300);
        osciloscopio.setToolTipText("Pulsar para ver el Ping");
        osciloscopio.addMouseListener(this);
        osciloscopio.setVisible(true);
        fondoP3b.add(osciloscopio);
        EnviarPing.setFont(f);//configuración botón Enviar Ping
        EnviarPing.setText("Enviar Ping");
        EnviarPing.setBounds(195,110,170,40);
        EnviarPing.setVisible(true);
        EnviarPing.addActionListener(this);
        fondoP3b.add(EnviarPing);
        MostrarC.setFont(f);//configuracion boton Mostrar Captura
        MostrarC.setText("Mostrar Captura");
        MostrarC.setBounds(195,415,170,40);
        MostrarC.setVisible(false);
        MostrarC.addActionListener(this);
        fondoP3b.add(MostrarC);
        lista=EnviarPing.getActionListeners();//esto se utiliza para
                                permitir que se cree el efecto de la flecha
        timer=new Timer(120,lista[0]);

        back =getImage("FondoP3bsinCables.jpg");//se termina de configurar
                                                          el fondo
        bgicon = new ImageIcon(back);
        background = new JLabel(bgicon);
        background.setOpaque(true);
        background.setBounds(0,0,bgicon.getIconWidth(),
       bgicon.getIconHeight());
        fondoP3b.add(background,new Integer(0));
        getContentPane().setBackground(Color.white);
        getContentPane().add(fondoP3b);
    }
```

Antes de enviar el ping la interfaz grafica que se muestra es la siguiente:

Figura 11

Una vez enviado el ping:

Figura 12

- Metodo `public Image getImage(String im){}`

Este método lo único que hace es devolver una imagen, es decir, coge la imagen que tiene por nombre la que indica el *String* que se le pasa como parámetro, si el nombre no existe lanzará una excepción.

Este método es usado por esta misma clase(*Práctica3_ping*) y lo usa para poner la imagen de fondo y para poner la imagen del osciloscopio.

```
public Image getImage(String im){

      return getImage(getDocumentBase(),im);

}
```

- Metodo `public void actionPerformed(ActionEvent e) {}`

A este método se accede cuando se pulsa el botón *Enviar Ping* o el botón *Mostrar Captura* que son los que capturan eventos del tipo ActionListener.

Hay una tercera causa por la que se puede entrar a este método y es cuando es llamado a sí mismo (esto lo explicare tras ver el código).

```
public void actionPerformed(ActionEvent e) {
        if(e.getSource()==EnviarPing){
           i++;
           fondoP3b.remove(background);
           numeroCaptura= (int)(rnd.nextDouble()*4+1);
           timer.setInitialDelay(120);
           timer.start();
           back =getImage("FondoP3bsinCablesPing"+i+".jpg");
           bgicon.setImage(back);
           background.setIcon(bgicon);
           background.setOpaque(true);
           background.setBounds(0,0,bgicon.getIconWidth(),
          bgicon.getIconHeight());
           fondoP3b.add(background,new Integer(0));
           MostrarC.setVisible(false);
           i++;
        }
        else if(e.getSource()==MostrarC){
            rnd.setSeed(System.currentTimeMillis());
            captura=new Ventana_captura(numeroCaptura,this);
           MostrarC.setVisible(false);
            captura.setVisible(true);
        }
```

```
        else{
            if(i==7){
                timer.stop();
                i=0;
                fondoP3b.remove(background);
                back =getImage("FondoP3bsinCables.jpg");
                bgicon.setImage(back);
                background.setIcon(bgicon);
                background.setOpaque(true);
                background.setBounds(0,0,bgicon.getIconWidth(),
                bgicon.getIconHeight());
                fondoP3b.add(background,new Integer(0));
                setContentPane(fondoP3b);
                MostrarC.setVisible(true);
                JOptionPane.showMessageDialog(null,"Ping enviado
                correctamente","Ping Correcto",1);
                EnviarPing.setText("Reenviar Ping");
            }
            else{
                fondoP3b.remove(background);
                back =getImage("FondoP3bsinCablesPing"+i+".jpg");
                bgicon.setImage(back);
                background.setIcon(bgicon);
                background.setOpaque(true);
                background.setBounds(0,0,bgicon.getIconWidth(),
                bgicon.getIconHeight());
                fondoP3b.add(background,new Integer(0));
                i++;
             }
        }
    }
```

En principio solo se muestra el botón *Enviar Ping*, luego, a este método se accede primero tras pulsar este botón.

Cuando este botón es pulsado entra en el condicional *if(e.getSource()==EnviarPing{})* y lo primero que se hace es incrementar la variable *i* (que en un principio vale 0 y valdra como máximo 7) que será usada para cambiar de fondo de forma que esta variable indica el numero de imagen que se va a coger, ya que el nombre de las imágenes sigue el formato: "FondoP3bsinCablesPing"+i+".jpg". Tras esto la variable *i* tendrá como valor 1.

Después lo que se hace es borrar el fondo y generar un numero aleatorio entre 1 y 5 (*numeroCaptura*) que se le pasara como argumento a las clases *OScope* y *Ventana_captura*, de forma que este número es el que indica el ping a mostrar (hay 5 diferentes).

En el siguiente paso lo que se hace es inicializar la variable *timer*. Esta variable es de tipo *Timer* que ya ha sido inicializada en el método *init()*, a esta variable se le pasan como argumentos un tiempo y una variable que pueda capturar un evento *ActionListener*.

Esta variable debe de pertenecer a un array de *ActionListeners* para poder pasarla como argumento a la clase *Timer*. Lo que hace *timer* es llamar al evento *ActionListener* asociado a la variable cada cierto tiempo, que es el que se le pasa como argumento (en milisegundos), esto lo hace hasta que se detenga el *timer.*

Lo que se hace a continuación es poner de fondo la primera imagen que simula el movimiento de la flecha, también se hace invisible el botón *Mostrar Captura* (esto es para cuando ya se ha pulsado alguna vez el botón) y se incrementa la variable *i* para dejarla preparada para cargar la siguiente imagen.

Tras esto es cuando empieza a actuar el *timer*, y el método comienza a llamarse a sí mismo. Ahora entra en el condicional else{}, dentro de esto condicional vuelve a entrar en el condicional else{} que tiene en su interior, de forma que cada vez que entre lo que hara es:

Borrar el fondo, cargar la siguiente imagen y aumentar la variable *i*.

El método seguirá llamándose a sí mismo hasta que la variable *i* alcance el valor 7, de esta forma entrara en el condicional *if(i==7){}*, donde lo que se hace es:

Parar el *timer*, poner a cero la variable *i*, borrar el fondo y poner el fondo original, se hace visible el botón *Mostrar Captura*, se muestra un mensaje indicando que el ping se ha enviado correctamente y se le cambia el nombre al botón *Enviar Ping* por *Reenviar Ping*.

La otra posibilidad que queda para entrar a este método es que sea pulsado el botón *MostrarC*, en cuyo caso lo que se hace es crear un objeto del tipo *Ventana_captura* y volver a ocultar el botón pulsado. Al constructor de esta clase hay que pasarle un entero, y un *JApplet Práctica3_ping*, este ultimo en este caso solo es usado para poner el icono de la ventana.

- Metodo `public void mouseClicked(MouseEvent e) {}`

A este metodo solo se accede cuando se produce un evento asociado a un MouseListener. En este caso la única variable que puede lanzar un evento de este tipo es el JLabel osciloscopio.

Este método lo único que hace es abrir el osciloscopio, es decir, crear un objeto de la clase *OScope*, pero solo lo hace si el ping ha sido ya enviado, en caso contrario mostrará un mensaje indicando que el ping debe de ser enviado.

```
public void mouseClicked(MouseEvent e) {
        if(e.getSource()==osciloscopio){
            if(EnviarPing.getText()=="Enviar Ping"){
                JOptionPane.showMessageDialog(null,"Debes enviar el
ping para poder verlo en el osciloscopio","Aviso",2);
            }
            else{
                osc=new OScope(this,pings[numeroCaptura-1]);
                osc.repaint();
            }
        }
    }
```

De este código comentar que, la forma de saber si se ha enviado el ping es comprobando el nombre que aparece en el botón *EnviarPing*, ya que si el nombre es *Enviar Ping* significa que no se ha enviado ya que si esto hubiera ocurrido pondría *Reenviar Ping*.

Al constructor *OScope* hay que pasarle como parámetros un *JApplet Práctica3_ping* y un array de enteros. El *JApplet* es necesario para que la clase *OScope* pueda cargar imágenes, y el array de enteros es un array que contiene unos y ceros, que serán usados para la representación del ping.

- Metodo `public void mousePressed(MouseEvent e) {}`
- Metodo `public void mouseReleased(MouseEvent e) {}`
- Metodo `public void mouseEntered(MouseEvent e) {}`
- Metodo `public void mouseExited(MouseEvent e) {}`

Estos métodos están vacios porque no son de utilidad, pero es necesario añadirlos ya que la clase implementa la interfaz MouseListener.

5.3.2. Ventana_captura

Esta clase es la encargada de mostrar la captura de wireshark. Lo que hace es abrir una ventana y mostrar en texto plano la captura, donde se pueden ver los distintos campos de un ping.

El código es el que se muestra a continuación:

```
import javax.swing.JScrollPane;
import javax.swing.JTextPane;

public class Ventana_captura extends javax.swing.JFrame {

    private int n;
    private JScrollPane scroll;
    private JTextPane panel;

    public Ventana_captura(int i,Práctica3_ping ap){
        n=i;
        scroll = new javax.swing.JScrollPane();
        panel = new javax.swing.JTextPane();
        panel.setEditable(false);
        setSize(1024,512);
        setResizable(false);
        super.setIconImage(ap.getImage("logoUpct_1.png"));
        setTitle("Datos del Ping enviado");
        panel.setFont(new java.awt.Font("Lucida Console", 0,14));
        panel.setBounds(0,0,1024,512);
        scroll.setViewportView(panel);
        this.add(scroll);

        switch (n)  {
            case 1 : panel.setText(/*aqui va el texto de la captura*/)break;

            case 2 : panel.setText(/*aqui va el texto de la captura*/)break;

            case 3 : panel.setText(/*aqui va el texto de la captura*/)break;

            case 4 : panel.setText(/*aqui va el texto de la captura*/)break;

            case 5 : panel.setText(/*aqui va el texto de la captura*/)break;
        }
     }
}
//código simplificado, para ver el texto de la captura mirar el código.
```

Esta clase lo que hace es crear una ventana (*JFrame*) con el texto de la captura. La captura a mostrar la indica la variable *i* que se le pasa al constructor, hay cinco capturas distintas.

La ventana generada tiene el siguiente aspecto:

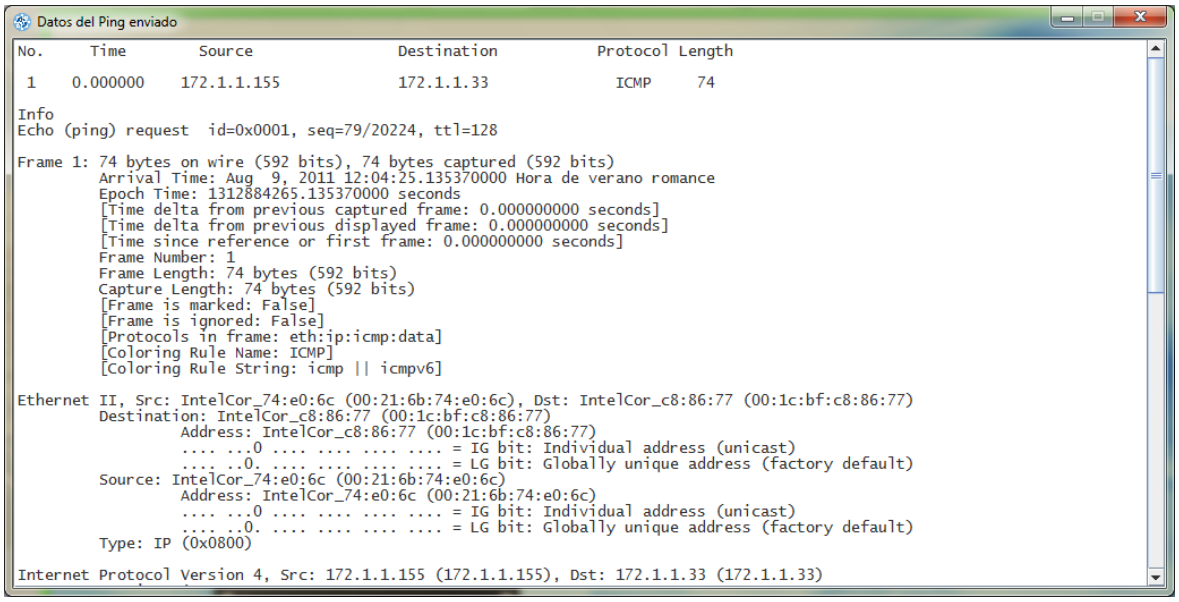


Figura 13

5.3.3. OScope

Esta clase es la encargada de dibujar el ping y además es la más compleja de este applet.

El código de forma general es el que se muestra a continuación:

```
public class OScope extends JFrame implements ActionListener,
ChangeListener{

    boolean sol=false;
    double x1=250,y1=210;
    double escalaX=1,escalaY=1;
    private JButton plusVolt; //+voltage
    private JButton minusVolt;//-votage
    private JButton plusTime;//+tiempo
    private JButton minusTime;//-tiempo
    private JButton solucion;//solucion
    private JButton nosolucion;
    private JLabel preambulo,sfd,datos,crc;
    private JSlider xslider, yslider;//barras para mover el ejee Y y el X
    private int xOffset, yOffset;
    private Color color=Color.yellow;
    private TextField tVolt, tTime;
    Image back,image;
    private JLabel background;
    private JLayeredPane pane;
    Container cp;
    int vPointer = 1;
    int value;
    int tPointer = 3;
    int resolXslider[]= {15000,30000,40000,150000};
    int indiceXslider=0;
    double volt[] = {0.6,0.3,0.15,0.075};
    double time[] = {2.5,5,10,20,40,80};
    int ping[];
    Práctica3_ping applet;

    public OScope(Práctica3_ping ap,int [] p){}

    public void stateChanged(ChangeEvent e){}

    public void paint(Graphics g){}

    public void actionPerformed(ActionEvent evt){}

    private String getVolt(){}

    private String getTime(){}

}
```

- Constructor `public OScope(Práctica3_ping ap,int [] p){}`

Este es el constructor de la clase y es el que se encarga de inicializar las variables. A continuación se muestra el código. Al igual que antes se mostrará la correspondecia entre este código y la interfaz grafica.

```
public OScope(Práctica3_ping ap,int [] p){

        Font f = new Font("Arial",Font.PLAIN,14);
        Font f1 = new Font("Arial",Font.PLAIN,11);

        applet=ap;
        this.setResizable(false);
        pane = new JLayeredPane();//es el fondo
        pane.setBackground(Color.white);
        setBackground(Color.white);
        setSize(850,500);
        pane.setLayout(null);
                                //barra ejeX para desplazar la señal
        xslider = new JSlider(-15000,15000,0);
       xslider.setBounds(130,350,250,25);
        xslider.setVisible(true);
        xslider.setOpaque(false);
        xslider.addChangeListener(this);
                                //barra ejeY para desplazar la señal
        yslider = new JSlider(JSlider.VERTICAL,-300,300,0);
        yslider.setBounds(460,90,25,250);
        yslider.setVisible(true);
        yslider.setOpaque(false);
        yslider.addChangeListener(this);

        pane.add(xslider);
        pane.add(yslider);
        setVisible(true);
                                //boton aumentar voltage/division
        plusVolt = new JButton("+");
        plusVolt.setFont(f);
        plusVolt.setBounds(560,80,60,25);
        pane.add(plusVolt);
        plusVolt.addActionListener(this);
                                //boton disminuir voltage/division
        minusVolt = new JButton("-");
        minusVolt.setFont(f);
        minusVolt.setBounds(625,80,60,25);
        pane.add(minusVolt);
        minusVolt.addActionListener(this);
                                //boton aumentar tiempo/division
        plusTime = new JButton("+");
        plusTime.setFont(f);
        plusTime.setBounds(560,155,60,25);
        pane.add(plusTime);
        plusTime.addActionListener(this);
                                //boton disminuir tiempo/division
        minusTime = new JButton("-");
        minusTime.setFont(f);
        minusTime.setBounds(625,155,60,25);
        pane.add(minusTime);
        minusTime.addActionListener(this);
```

```
        solucion = new JButton("Pulsar para ver solucion");//botón sol
        solucion.setFont(f1);
        solucion.setBounds(555,280,200,25);
        pane.add(solucion);
        solucion.addActionListener(this);
                                                //botón no sol
        nosolucion = new JButton("Pulsar para ocultar solucion");
        nosolucion.setFont(f1);
        nosolucion.setBounds(555,280,200,25);
        pane.add(nosolucion);
        nosolucion.setVisible(false);
        nosolucion.addActionListener(this);
                                                //etiqueta preambulo
        preambulo=new JLabel("PREAMBULO");
        preambulo.setBounds(610,280,200,100);
        preambulo.setForeground(Color.ORANGE);
        pane.add(preambulo);
        preambulo.setVisible(false);
        sfd=new JLabel("SFD");          //etiqueta sfd
        sfd.setBounds(610,300,200,100);
        sfd.setForeground(Color.RED);
        pane.add(sfd);
        sfd.setVisible(false);
        datos=new JLabel("DATOS");      //etiqueta datos
        datos.setBounds(610,320,200,100);
        datos.setForeground(Color.BLUE);
        pane.add(datos);
        datos.setVisible(false);
        crc=new JLabel("CRC");          //etiqueta crc
        crc.setBounds(610,340,200,100);
        crc.setForeground(Color.PINK.darker());
        pane.add(crc);
        crc.setVisible(false);

        tVolt = new TextField(); //area texto para mostrar voltage/division
        tVolt.setFont(new java.awt.Font("Tahoma", 0, 14));
        tVolt.setEditable(false);
        tVolt.setBounds(690,80,60,25);
        pane.add(tVolt);
        tVolt.setText(getVolt());

        tTime = new TextField();//area texto para mostrar tiempo/division
        tTime.setFont(new java.awt.Font("Tahoma", 0, 14));
        tTime.setEditable(false);
        tTime.setBounds(690,155,60,25);
        pane.add(tTime);
        tTime.setText(getTime());

        super.setIconImage(applet.getImage("logoUpct_1.png"));
        setTitle("Osciloscopio");
        ping=p;
        back=applet.getImage("osc2.gif");
        ImageIcon bgicon = new ImageIcon(back);
        background = new JLabel(bgicon);
        background.setOpaque(true);
        background.setBounds(0,0,bgicon.getIconWidth(),
        bgicon.getIconHeight());
        pane.add(background,new Integer(0));
        getContentPane().setBackground(Color.white);
        getContentPane().add(pane);
        this.repaint();
    }
```

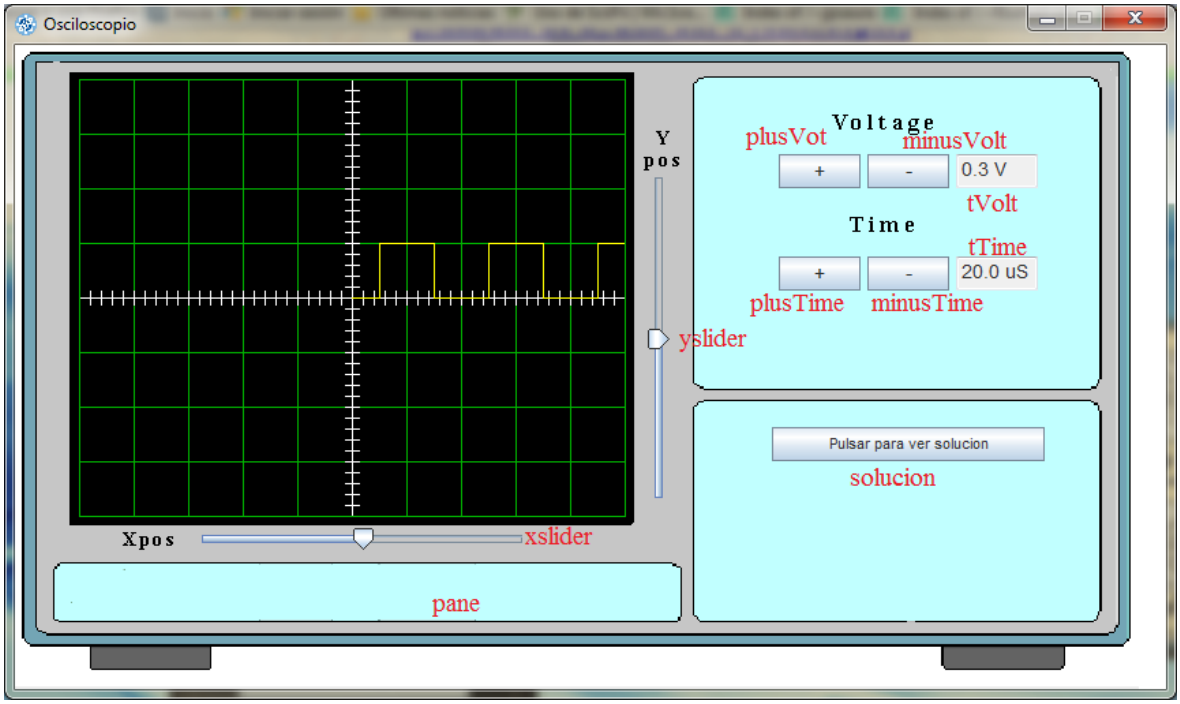


Figura 14

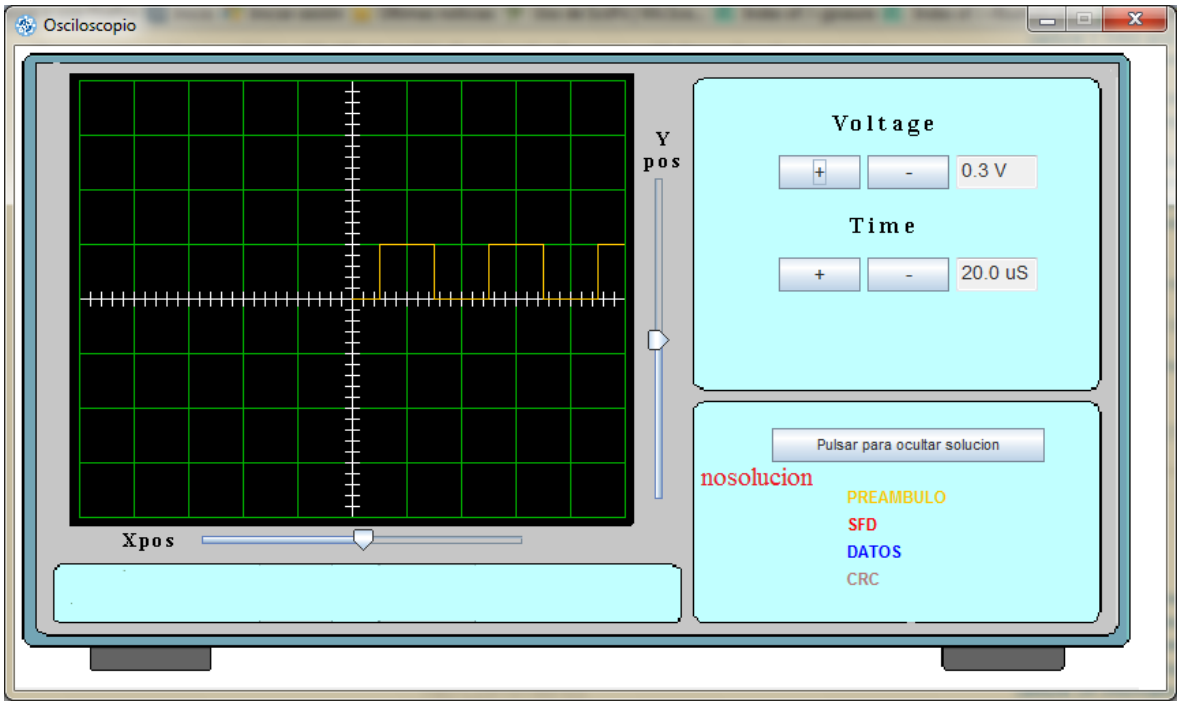


Figura 15

- Metodo `public void stateChanged(ChangeEvent e){}`

A este metodo solo se accede cuando se produce un evento asociado a un *ChangeListener*. En este caso las variables que pueden lanzar un evento de este tipo son *xslider* e *yslider*.

Lo que hace este método es mover la señal, es decir, se cambian los valores de los variables *yOffset* y *xOffset* (asociadas a *yslider* y a *xslider* respectivamente) que se utilizan el metodo *paint()* para dibujar el ping.

```
public void stateChanged(ChangeEvent e){

        JSlider source = (JSlider)e.getSource();

        if (source==yslider){
            yOffset =(-1)* (int)source.getValue();
            repaint();
        }
        else if(source==xslider){
           xOffset = (-1)*(int)source.getValue();
           repaint();
        }
     }
```

- Metodo `public void paint(Graphics g){}`

Este metodo es el más importante de esta clase ya que es el encargado de dibujar la señal. A continuacion se muestra el codigo y luego explicare de forma general cómo funciona este metodo, ya que es algo extenso.

```
public void paint(Graphics g){

        super.paint(g);
        int gridSize = 400;
        int ygridSize = 320;
        int border = 50;
        int gb = gridSize+border;
        int inc = (gridSize/10) / 5;
```

```
//-- dibujo pantalla ------------------------------------------------------

        g.setColor((Color.green).darker());

        for(int x=border;x<=gb;x+=gridSize/10)// dibujo cuadricula
                g.drawLine(x,border,x,ygridSize+border);

        for(int y=border;y<=(ygridSize+border);y+=gridSize/10)
                g.drawLine(border,y,gb,y);

        g. setColor(Color.white);
        for(int xi=border;xi<=gb;xi+=inc)  // dibujo incrementos
                g.drawLine(xi,ygridSize/25+border,xi,ygridSize/2+5+border);

        for(int yi=border;yi<=(ygridSize+border);yi+=inc)
                g.drawLine(gridSize/2-5+border,yi,gridSize/2+5+border,yi);

        g.drawLine(250,50,250,370);
        g.drawLine(50,210,450,210);
```

Esto lo que hace es dibujar la pantalla del osciloscopio, es decir, la cuadricula y las separaciones. En el siguiente codigo se muestra como se dibuja el ping.

```
//-- dibujar ping ---------------------------------------------------------
        g.setColor(color);

        if(sol){
           g.setColor(Color.ORANGE);
        }
    //inicio dibujar primer bit
        int xo=(int)(x1+xOffset);//posicion actual x mas desplazamiento
        int yo=(int)(y1+yOffset);//posicion actual y mas desplazamiento
int dx=(int)(20*escalaX),dy=(int)(40*escalaY),anchoPing=(int)(40*escalaX);
        if(xo>=50-xOffset&&xo+anchoPing<=450-xOffset){//dentro del marco segun  x
            if(yo-dy>=50-yOffset&&yo<370-yOffset){//dentro del marco segun y
                if(ping[0]==1){//se dibuja un 1
                   g.drawLine(xo+xOffset,yo+yOffset,xo+dx+xOffset,yo+yOffset);
                   g.drawLine(xo+dx+xOffset,yo+yOffset,xo+dx+xOffset,yo
                       dy+yOffset);
                    g.drawLine(xo+dx+xOffset, yo-dy+yOffset,
                       xo+2*dx+xOffset,yo-dy+yOffset);
                    xo=xo+2*dx+xOffset;
                    yo=yo-dy+yOffset;
                }
                else{//se dibuja un cero
                   g.drawLine(xo+xOffset,yo+yOffset,xo+xOffset,yo-dy+yOffset);
                   g.drawLine(xo+xOffset,yo-dy+yOffset,xo+dx+xOffset,yo
                       dy+yOffset);
                   g.drawLine(xo+dx+xOffset,yo
                       dy+yOffset,xo+dx+xOffset,yo+yOffset);
                   g.drawLine(xo+dx+xOffset, yo+yOffset,
                       xo+2*dx+xOffset,yo+yOffset);
                   xo=xo+2*dx+xOffset;
                   yo=yo+yOffset;
                }
             }
```

```
//el primer bit se esta saliendo por arriba (pero esta dentro segun x)
             else if(yo-dy<50-yOffset&&yo-dy>=50-yOffset-dy){
                 if(ping[0]==1){

                     g.drawLine(xo+xOffset,yo+yOffset,xo+dx+xOffset,yo+yOffset);
                     g.drawLine(xo+dx+xOffset,yo+yOffset,xo+dx+xOffset,50);
                     xo=xo+2*dx+xOffset;
                     yo=yo-dy+yOffset;
                 }
                 else{
                     g.drawLine(xo+xOffset,yo+yOffset,xo+xOffset,50);
                     g.drawLine(xo+dx+xOffset,50,xo+dx+xOffset,yo+yOffset);
                     g.drawLine(xo+dx+xOffset, yo+yOffset,
                        xo+2*dx+xOffset,yo+yOffset);
                     xo=xo+2*dx+xOffset;
                     yo=yo+yOffset;
                 }
             }

//el primer bit se esta saliendo por abaja (pero esta dentro segun x)
             else if(yo>=370-yOffset&&yo<=370-yOffset+dy){
               if(ping[0]==1){
                     g.drawLine(xo+dx+xOffset,370,xo+dx+xOffset,yo-dy+yOffset);
                     g.drawLine(xo+dx+xOffset, yo-dy+yOffset,
                        xo+2*dx+xOffset,yo-dy+yOffset);
                     xo=xo+2*dx+xOffset;
                     yo=yo-dy+yOffset;
               }
               else{
                     g.drawLine(xo+xOffset,370,xo+xOffset,yo-dy+yOffset);
                     g.drawLine(xo+xOffset,yo-dy+yOffset,xo+dx+xOffset,yo
                        dy+yOffset);
                     g.drawLine(xo+dx+xOffset,yo-dy+yOffset,xo+dx+xOffset,370);
                     xo=xo+2*dx+xOffset;
                     yo=yo+yOffset;
               }
             }

//el primer bit esta totalment fuera del marco
             else if(yo-dy<=50-yOffset-dy||yo>=370-yOffset+dy){
                 if(ping[0]==1){
                        xo=xo+2*dx+xOffset;
                        yo=yo-dy+yOffset;
                     }
                 else{
                        xo=xo+2*dx+xOffset;
                        yo=yo+yOffset;
                 }
             }
         }
```

```
     else if((xo<50-xOffset&&xo>=(50-xOffset-dx))){//la primera mitad del
primer bit esta saliendo por la izquierda(según x)
              if(yo-dy>=50-yOffset&&yo<370-yOffset){//esta dentron segun y
                 if(ping[0]==1){
                     g.drawLine(50,yo+yOffset,xo+dx+xOffset,yo+yOffset);
                     g.drawLine(xo+dx+xOffset,yo+yOffset,xo+dx+xOffset,yo
                        dy+yOffset);
                     g.drawLine(xo+dx+xOffset, yo-dy+yOffset,
                        xo+2*dx+xOffset,yo-dy+yOffset);
                     xo=xo+2*dx+xOffset;
                     yo=yo-dy+yOffset;
                 }
                 else{
                     g.drawLine(50,yo-dy+yOffset,xo+dx+xOffset,yo-dy+yOffset);
                     g.drawLine(xo+dx+xOffset,yo
                        dy+yOffset,xo+dx+xOffset,yo+yOffset);
                     g.drawLine(xo+dx+xOffset, yo+yOffset,
                        xo+2*dx+xOffset,yo+yOffset);
                     xo=xo+2*dx+xOffset;
                     yo=yo+yOffset;
                 }
              }
//empieza a salirse por arriba
              else if(yo-dy<50-yOffset&&yo-dy>=50-yOffset-dy){
                  if(ping[0]==1){
                     g.drawLine(50,yo+yOffset,xo+dx+xOffset,yo+yOffset);
                     g.drawLine(xo+dx+xOffset,yo+yOffset,xo+dx+xOffset,50);
                     xo=xo+2*dx+xOffset;
                     yo=yo-dy+yOffset;
                 }
                 else{
                     g.drawLine(xo+dx+xOffset,50,xo+dx+xOffset,yo+yOffset);
                     g.drawLine(xo+dx+xOffset, yo+yOffset,
xo+2*dx+xOffset,yo+yOffset);
                     xo=xo+2*dx+xOffset;
                     yo=yo+yOffset;
                 }
              }
//empieza a salirse por abajo
              else if(yo>=370-yOffset&&yo<=370-yOffset+dy){
                if(ping[0]==1){
                     g.drawLine(xo+dx+xOffset,370,xo+dx+xOffset,yo-dy+yOffset);
                     g.drawLine(xo+dx+xOffset, yo-dy+yOffset,
                        xo+2*dx+xOffset,yo-dy+yOffset);
                     xo=xo+2*dx+xOffset;
                     yo=yo-dy+yOffset;
                 }
                 else{
                     g.drawLine(50,yo-dy+yOffset,xo+dx+xOffset,yo-dy+yOffset);
                     g.drawLine(xo+dx+xOffset,yo-dy+yOffset,xo+dx+xOffset,370);
                     xo=xo+2*dx+xOffset;
                     yo=yo+yOffset;
                 }
              }
```

```
                else {//se sale totalmente
                    if(ping[0]==1){
                            xo=xo+2*dx+xOffset;
                            yo=yo-dy+yOffset;
                        }
                   else{
                           xo=xo+2*dx+xOffset;
                           yo=yo+yOffset;
                    }
                }
           }
//se sale la segunda mitad del primer bit por la izquierda
          else if(xo<(50-xOffset-dx)&&xo>(50-xOffset-2*dx)){
                if(yo-dy>=50-yOffset&&yo<370-yOffset){//dentro segun y
                   if(ping[0]==1){
                       g.drawLine(50, yo-dy+yOffset,xo+2*dx+xOffset,yo
                          dy+yOffset);
                       xo=xo+2*dx+xOffset;
                       yo=yo-dy+yOffset;
                   }
                   else{
                       g.drawLine(50, yo+yOffset,xo+2*dx+xOffset,yo+yOffset);
                       xo=xo+2*dx+xOffset;
                       yo=yo+yOffset;
                   }
                }
                else if(yo-dy<50-yOffset&&yo-dy>=50-yOffset-dy){//se sale por
                                                                   ariba
                    if(ping[0]==1){
                   }
                   else{
                       g.drawLine(50, yo+yOffset,xo+2*dx+xOffset,yo+yOffset);
                       xo=xo+2*dx+xOffset;
                       yo=yo+yOffset;
                   }
                }
                else{//esta totalmente fuera por arriba o por abajo
                    if(ping[0]==1){
                            xo=xo+2*dx+xOffset;
                            yo=yo-dy+yOffset;
                        }
                   else{
                            xo=xo+2*dx+xOffset;
                            yo=yo+yOffset;
                   }
                }
          }
```

```
//la segunda parte del primer bit se sale por la derecha
          else if((xo+anchoPing>450-xOffset&&xo+anchoPing<=450-xOffset+dx)){
              if(yo-dy>=50-yOffset&&yo<370-yOffset){//dentro segun y
                 if(ping[0]==1){
                 g.drawLine(xo+xOffset,yo+yOffset,xo+dx+xOffset,yo+yOffset);
                     g.drawLine(xo+dx+xOffset,yo+yOffset,xo+dx+xOffset,yo
                        dy+yOffset);
                     g.drawLine(xo+dx+xOffset, yo-dy+yOffset, 450,yo
                        dy+yOffset);
                     xo=xo+2*dx+xOffset;
                     yo=yo-dy+yOffset;
                 }
                 else{
                    g.drawLine(xo+xOffset,yo+yOffset,xo+xOffset,yo
                        dy+yOffset);
                    g.drawLine(xo+xOffset,yo-dy+yOffset,xo+dx+xOffset,yo
                        dy+yOffset);
                    g.drawLine(xo+dx+xOffset,yo
                        dy+yOffset,xo+dx+xOffset,yo+yOffset);
                    g.drawLine(xo+dx+xOffset, yo+yOffset, 450,yo+yOffset);
                    xo=xo+2*dx+xOffset;
                    yo=yo+yOffset;
                 }
              }
                //se sale por arriba
              else if(yo-dy<50-yOffset&&yo-dy>=50-yOffset-dy){
                  if(ping[0]==1){
                  g.drawLine(xo+xOffset,yo+yOffset,xo+dx+xOffset,yo+yOffset);
                     g.drawLine(xo+dx+xOffset,yo+yOffset,xo+dx+xOffset,50);
                     xo=xo+2*dx+xOffset;
                     yo=yo-dy+yOffset;
                 }
                 else{
                    g.drawLine(xo+xOffset,yo+yOffset,xo+xOffset,50);
                    g.drawLine(xo+dx+xOffset,50,xo+dx+xOffset,yo+yOffset);
                    g.drawLine(xo+dx+xOffset, yo+yOffset, 450,yo+yOffset);
                    xo=xo+2*dx+xOffset;
                    yo=yo+yOffset;
                 }
              }
                //se sale por abajo
              else if(yo>=370-yOffset&&yo<=370-yOffset+dy){
                  if(ping[0]==1){
                     g.drawLine(xo+dx+xOffset,370,xo+dx+xOffset,yo
                        dy+yOffset);
                     g.drawLine(xo+dx+xOffset, yo-dy+yOffset, 450,yo
                        dy+yOffset);
                     xo=xo+2*dx+xOffset;
                     yo=yo-dy+yOffset;
                 }
                 else{
                    g.drawLine(xo+xOffset,370,xo+xOffset,yo-dy+yOffset);
                    g.drawLine(xo+xOffset,yo-dy+yOffset,xo+dx+xOffset,yo
                        dy+yOffset);
                    g.drawLine(xo+dx+xOffset,yo-dy+yOffset,xo+dx+xOffset,370);
                    xo=xo+2*dx+xOffset;
                    yo=yo+yOffset;
                 }
```

```
              else {//esta totalmente fuera por arriba o por abajo
                  if(ping[0]==1){
                          xo=xo+2*dx+xOffset;
                          yo=yo-dy+yOffset;
                      }
                  else{
                          xo=xo+2*dx+xOffset;
                          yo=yo+yOffset;
                  }
              }
            }
        }
//la primera parte del primer bit se sale por la derecha
        else if((xo+anchoPing>450-xOffset+dx&&xo+anchoPing<=450xOffset+2*dx)){
              if(yo-dy>=50-yOffset&&yo<370-yOffset){//dentro segun y
                  if(ping[0]==1){
                      g.drawLine(xo+xOffset,yo+yOffset,450,yo+yOffset);
                      xo=xo+2*dx+xOffset;
                      yo=yo-dy+yOffset;
                  }
                  else{
                      g.drawLine(xo+xOffset,yo+yOffset,xo+xOffset,yo
                         dy+yOffset);
                      g.drawLine(xo+xOffset,yo-dy+yOffset,450,yo-dy+yOffset);
                      xo=xo+2*dx+xOffset;
                      yo=yo+yOffset;
                  }
              }
                //se sale por arriba
              else if(yo-dy<50-yOffset&&yo-dy>=50-yOffset-dy){
                  if(ping[0]==1){
                      g.drawLine(xo+xOffset,yo+yOffset,450,yo+yOffset);
                      xo=xo+2*dx+xOffset;
                      yo=yo-dy+yOffset;
                  }
                  else{
                     g.drawLine(xo+xOffset,yo+yOffset,xo+xOffset,50);//esta
                     xo=xo+2*dx+xOffset;
                     yo=yo+yOffset;
                  }
              }
                //se sale por abajo
              else if(yo>=370-yOffset&&yo<=370-yOffset+dy){
                  if(ping[0]==1){
                     g.drawLine(xo+xOffset,yo+yOffset,450,yo+yOffset);
                      xo=xo+2*dx+xOffset;
                      yo=yo-dy+yOffset;
                  }
                  else{
                     g.drawLine(xo+xOffset,370,xo+xOffset,yo-dy+yOffset);//esta
                     g.drawLine(xo+xOffset,yo-dy+yOffset,450,yo-dy+yOffset);
                     xo=xo+2*dx+xOffset;
                     yo=yo+yOffset;
                  }
              }
```

```
            //totalmente fuera
            else{
                if(ping[0]==1){
                        xo=xo+2*dx+xOffset;
                        yo=yo-dy+yOffset;
                    }
                else{
                        xo=xo+2*dx+xOffset;
                        yo=yo+yOffset;
                }
            }
        }
        else{// totalmente fuera según x e y
            xo=xo+2*dx+xOffset;
            if(ping[0]==1){
                yo=yo-dy+yOffset;
            }
            else{
                yo=yo+yOffset;
            }
        }
//fin dibujar primer bit
```

El primer bit hay que dibujarlo "manualmente", ya que a partir de este los demás se generarán de forma automática dentro de un bucle.

Para dibujar el ping hay que estar constantemente controlando que cuando el trazo llega a los bordes se deje de dibujar parte del ping, para así simular el movimiento de la señal cuando se desplaza.

Una vez dibujado este primer bit los otros se generan a partir de este, diferenciado cuatro casos posibles, que son: que el primer bit sea un 1 y el siguiente un 0, que el primer bit sea un 1 y el siguiente un 1, que el primer bit sea un 0 y el siguiente un 0 o que sea el primer bit sea un 0 y el segundo un 1.

Entonces en función de lo anterior se procede de la misma forma que para el primer bit, se va comprobando cuando los trazos llegan a los bordes y así ir calculando hasta donde hay que ir dibujando, de forma que de sensacion de movimiento y que no se vea señal dibujada fuera del marco.

El codigo restante es el siguiente:

```
for(int i=1; i<ping.length;i++){
        if(sol){
            if(i<56){
                g.setColor(Color.ORANGE);
            }
            else if(i>=56&&i<64){
                g.setColor(Color.RED);
            }
            else if(i>=ping.length-16){
                g.setColor(Color.PINK);
            }
            else{
                g.setColor(Color.BLUE);
            }
        }
       if(ping[i]==1&ping[i-1]==1){
           if(xo>=50&&xo+anchoPing<=450){
               if(yo>=50&&yo+dy<370){
                   g.drawLine(xo,yo,xo,yo+dy);
                   g.drawLine(xo,yo+dy,xo+dx,yo+dy);
                   g.drawLine(xo+dx, yo+dy, xo+dx,yo);
                   g.drawLine(xo+dx, yo, xo+2*dx,yo);
                   if(i==ping.length-1){
                       g.drawLine(xo+2*dx,yo,xo+2*dx,yo+dy);
                   }
               }
               else if(yo<50&&yo>50-dy){
                   g.drawLine(xo,50,xo,yo+dy);
                   g.drawLine(xo,yo+dy,xo+dx,yo+dy);
                   g.drawLine(xo+dx, yo+dy, xo+dx,50);
                   if(i==ping.length-1){
                       g.drawLine(xo+2*dx,50,xo+2*dx,yo+dy);
                   }
               }
               else if(yo+dy>=370&&yo+dy<370+dy){
                   g.drawLine(xo,yo,xo,370);
                   g.drawLine(xo+dx, 370, xo+dx,yo);
                   g.drawLine(xo+dx, yo, xo+2*dx,yo);
                   if(i==ping.length-1){
                       g.drawLine(xo+2*dx,yo,xo+2*dx,370);
                   }
               }
               else if(yo<=50-dy||yo+dy>=370+dy) {

                }
           }
```

```
                else if(xo<50&&xo>=(50-dx)){
                    if(yo>=50&&yo+dy<370){
                        g.drawLine(50,yo+dy,xo+dx,yo+dy);
                        g.drawLine(xo+dx, yo+dy, xo+dx,yo);
                        g.drawLine(xo+dx, yo, xo+2*dx,yo);
                        if(i==ping.length-1){
                            g.drawLine(xo+2*dx,yo,xo+2*dx,yo+dy);
                        }
                    }
                    else if(yo<50&&yo>50-dy){
                        g.drawLine(50,yo+dy,xo+dx,yo+dy);
                        g.drawLine(xo+dx, yo+dy, xo+dx,50);
                        if(i==ping.length-1){
                            g.drawLine(xo+2*dx,50,xo+2*dx,yo+dy);
                        }
                    }
                    else if(yo+dy>=370&&yo+dy<370+dy){
                        g.drawLine(xo+dx, 370, xo+dx,yo);
                        g.drawLine(xo+dx, yo, xo+2*dx,yo);
                        if(i==ping.length-1){
                            g.drawLine(xo+2*dx,yo,xo+2*dx,370);
                        }
                    }
                    else if(yo<=50-dy||yo+dy>=370+dy) {
                     }
                }
                else if(xo<(50-dx)&xo>=50-2*dx){
                    if(yo>=50&&yo+dy<370){
                        g.drawLine(50, yo, xo+2*dx,yo);
                        if(i==ping.length-1){
                            g.drawLine(xo+2*dx,yo,xo+2*dx,yo+dy);
                        }
                    }
                    else if(yo<50&&yo>50-dy){
                        if(i==ping.length-1){
                            g.drawLine(xo+2*dx,50,xo+2*dx,yo+dy);
                        }
                    }
                    else if(yo+dy>=370&&yo+dy<370+dy){
                        g.drawLine(50, yo, xo+2*dx,yo);
                        if(i==ping.length-1){
                            g.drawLine(xo+2*dx,yo,xo+2*dx,370);
                        }
                    }
                    else if(yo<=50-dy||yo+dy>=370+dy) {
                     }
                }
```

```
            else if(xo+anchoPing>450&&xo+anchoPing<=(450+dx)){
                 if(yo>=50&&yo+dy<370){
                     g.drawLine(xo,yo,xo,yo+dy);
                     g.drawLine(xo,yo+dy,xo+dx,yo+dy);
                     g.drawLine(xo+dx, yo+dy, xo+dx,yo);
                     g.drawLine(xo+dx, yo, 450,yo);
                 }
                 else if(yo<50&&yo>50-dy){
                     g.drawLine(xo,50,xo,yo+dy);
                     g.drawLine(xo,yo+dy,xo+dx,yo+dy);
                     g.drawLine(xo+dx, yo+dy, xo+dx,50);
                 }
                 else if(yo+dy>=370&&yo+dy<370+dy){
                     g.drawLine(xo,yo,xo,370);
                     g.drawLine(xo+dx, 370, xo+dx,yo);
                     g.drawLine(xo+dx, yo, 450,yo);
                 }
                 else if(yo<=50-dy||yo+dy>=370+dy) {
                  }
             }
              else if(xo+anchoPing>=(450+dx)&&xo+anchoPing<=(450+2*dx)){
                 if(yo>=50&&yo+dy<370){
                     g.drawLine(xo,yo,xo,yo+dy);
                     g.drawLine(xo,yo+dy,450,yo+dy);
                 }
                 else if(yo<50&&yo>50-dy){
                     g.drawLine(xo,50,xo,yo+dy);
                     g.drawLine(xo,yo+dy,450,yo+dy);
                 }
                 else if(yo+dy>=370&&yo+dy<370+dy){
                     g.drawLine(xo,yo,xo,370);
                 }
                 else if(yo<=50-dy||yo+dy>=370+dy) {
                  }
             }
            else if(xo+anchoPing>450+2*dx){
            }

             xo=xo+2*dx;
             yo=yo;
        }
```

```
        else if(ping[i]==0&ping[i-1]==1){
                if(xo>=50&&xo+anchoPing<=450){
                    if(yo>=50&&yo+dy<370){
                        g.drawLine(xo,yo,xo+dx,yo);
                        g.drawLine(xo+dx,yo,xo+dx,yo+dy);
                        g.drawLine(xo+dx, yo+dy, xo+2*dx,yo+dy);
                    }
                    else if(yo<50&&yo>50-dy){
                       g.drawLine(xo+dx,50,xo+dx,yo+dy);
                       g.drawLine(xo+dx, yo+dy, xo+2*dx,yo+dy);
                    }
                    else if(yo+dy>=370&&yo+dy<370+dy){
                        g.drawLine(xo,yo,xo+dx,yo);
                        g.drawLine(xo+dx,yo,xo+dx,370);
                    }
                     else if(yo<=50-dy||yo+dy>=370+dy) {
                     }
                }
                else if(xo<50&&xo>=(50-dx)){
                    if(yo>=50&&yo+dy<370){
                        g.drawLine(50,yo,xo+dx,yo);
                        g.drawLine(xo+dx,yo,xo+dx,yo+dy);
                        g.drawLine(xo+dx, yo+dy, xo+2*dx,yo+dy);
                    }
                    else if(yo<50&&yo>50-dy){
                        g.drawLine(xo+dx,50,xo+dx,yo+dy);
                        g.drawLine(xo+dx, yo+dy, xo+2*dx,yo+dy);
                    }
                    else if(yo+dy>=370&&yo+dy<370+dy){
                        g.drawLine(50,yo,xo+dx,yo);
                        g.drawLine(xo+dx,yo,xo+dx,370);
                    }
                     else if(yo<=50-dy||yo+dy>=370+dy) {
                     }
                }
                else if(xo<(50-dx)&xo>=50-2*dx){
                    if(yo>=50&&yo+dy<370){
                        g.drawLine(50, yo+dy, xo+2*dx,yo+dy);
                    }
                    else if(yo<50&&yo>50-dy){
                        g.drawLine(50, yo+dy, xo+2*dx,yo+dy);
                    }
                    else if(yo+dy>=370&&yo+dy<370+dy){
                    }
                     else if(yo<=50-dy||yo+dy>=370+dy) {
                     }
                }
```

```
                else if(xo+anchoPing>450&&xo+anchoPing<=(450+dx)){
                    if(yo>=50&&yo+dy<370){
                        g.drawLine(xo,yo,xo+dx,yo);
                        g.drawLine(xo+dx,yo,xo+dx,yo+dy);
                        g.drawLine(xo+dx, yo+dy, 450,yo+dy);
                    }
                    else if(yo<50&&yo>50-dy){
                        g.drawLine(xo+dx,50,xo+dx,yo+dy);
                        g.drawLine(xo+dx, yo+dy, 450,yo+dy);
                    }
                    else if(yo+dy>=370&&yo+dy<370+dy){
                        g.drawLine(xo,yo,xo+dx,yo);
                        g.drawLine(xo+dx,yo,xo+dx,370);
                    }
                     else if(yo<=50-dy||yo+dy>=370+dy) {
                     }
                }
                else if(xo+anchoPing>=(450+dx)&&xo+anchoPing<=(450+2*dx)){
                    if(yo>=50&&yo+dy<370){
                        g.drawLine(xo,yo,450,yo);
                    }
                    else if(yo<50&&yo>50-dy){
                    }
                    else if(yo+dy>=370&&yo+dy<370+dy){
                        g.drawLine(xo,yo,450,yo);
                    }
                     else if(yo<=50-dy||yo+dy>=370+dy) {
                     }
                }
                xo=xo+2*dx;
                yo=yo+dy;
            }
            else if(ping[i]==1&ping[i-1]==0){
                if(xo>=50&&xo+anchoPing<=450){
                    if(yo-dy>=50&&yo<370){
                        g.drawLine(xo,yo,xo+dx,yo);
                        g.drawLine(xo+dx,yo,xo+dx,yo-dy);
                        g.drawLine(xo+dx, yo-dy, xo+2*dx,yo-dy);
                        if(i==ping.length-1){
                            g.drawLine(xo+2*dx,yo-dy,xo+2*dx,yo);
                        }
                    }
                    else if(yo-dy<50&&yo-dy>50-dy){
                        g.drawLine(xo,yo,xo+dx,yo);
                        g.drawLine(xo+dx,yo,xo+dx,50);
                        if(i==ping.length-1){
                            g.drawLine(xo+2*dx,50,xo+2*dx,yo);
                        }
                    }
                    else if(yo>=370&&yo<370+dy){
                        g.drawLine(xo+dx,370,xo+dx,yo-dy);
                        g.drawLine(xo+dx, yo-dy, xo+2*dx,yo-dy);
                        if(i==ping.length-1){
                            g.drawLine(xo+2*dx,yo-dy,xo+2*dx,370);
                        }
                    }
                    else if(yo-dy<=50-dy||yo>=370+dy) {
                     }
                }
```

```
                else if(xo<50&&xo>=(50-dx)){
                     if(yo-dy>=50&&yo<370){
                         g.drawLine(50,yo,xo+dx,yo);
                         g.drawLine(xo+dx,yo,xo+dx,yo-dy);
                         g.drawLine(xo+dx, yo-dy, xo+2*dx,yo-dy);
                         if(i==ping.length-1){
                             g.drawLine(xo+2*dx,yo-dy,xo+2*dx,yo);
                         }
                     }
                     else if(yo-dy<50&&yo-dy>50-dy){
                         g.drawLine(50,yo,xo+dx,yo);
                         g.drawLine(xo+dx,yo,xo+dx,50);
                         if(i==ping.length-1){
                             g.drawLine(xo+2*dx,50,xo+2*dx,yo);
                         }
                     }
                     else if(yo>=370&&yo<370+dy){
                         g.drawLine(xo+dx,370,xo+dx,yo-dy);
                         g.drawLine(xo+dx, yo-dy, xo+2*dx,yo-dy);
                         if(i==ping.length-1){
                             g.drawLine(xo+2*dx,yo-dy,xo+2*dx,370);
                         }
                     }
                     else if(yo-dy<=50-dy||yo>=370+dy) {
                      }
                 }
                 else if(xo<(50-dx)&xo>=50-2*dx){
                     if(yo-dy>=50&&yo<370){
                         g.drawLine(50, yo-dy, xo+2*dx,yo-dy);
                         if(i==ping.length-1){
                             g.drawLine(xo+2*dx,yo-dy,xo+2*dx,yo);
                         }
                     }
                     else if(yo-dy<50&&yo-dy>50-dy){
                         if(i==ping.length-1){
                             g.drawLine(xo+2*dx,50,xo+2*dx,yo);
                         }
                     }
                     else if(yo>=370&&yo<370+dy){
                        g.drawLine(50, yo-dy, xo+2*dx,yo-dy);
                        if(i==ping.length-1){
                             g.drawLine(xo+2*dx,yo-dy,xo+2*dx,370);
                         }
                     }
                     else if(yo-dy<=50-dy||yo>=370+dy) {
                      }
                 }
```

```
            else if(xo+anchoPing>450&&xo+anchoPing<=(450+dx)){
                 if(yo-dy>=50&&yo<370){
                     g.drawLine(xo,yo,xo+dx,yo);
                     g.drawLine(xo+dx,yo,xo+dx,yo-dy);
                     g.drawLine(xo+dx, yo-dy, 450,yo-dy);
                 }
                 else if(yo-dy<50&&yo-dy>50-dy){
                    g.drawLine(xo,yo,xo+dx,yo);
                    g.drawLine(xo+dx,yo,xo+dx,50);
                 }
                 else if(yo>=370&&yo<370+dy){
                    g.drawLine(xo+dx,370,xo+dx,yo-dy);
                    g.drawLine(xo+dx, yo-dy, 450,yo-dy);
                 }
                 else if(yo-dy<=50-dy||yo>=370+dy) {
                  }
             }
             else if(xo+anchoPing>=(450+dx)&&xo+anchoPing<=(450+2*dx)){
                 if(yo-dy>=50&&yo<370){
                     g.drawLine(xo,yo,450,yo);
                 }
                 else if(yo-dy<50&&yo-dy>50-dy){
                    g.drawLine(xo,yo,450,yo);
                 }
                 else if(yo>=370&&yo<370+dy){
                 }
                 else if(yo-dy<=50-dy||yo>=370+dy) {
                  }
             }
             xo=xo+2*dx;
             yo=yo-dy;
         }
        else if(ping[i]==0&ping[i-1]==0){
             if(xo>=50&&xo+anchoPing<=450){
                 if(yo-dy>=50&&yo<370){
                     g.drawLine(xo,yo,xo,yo-dy);
                     g.drawLine(xo,yo-dy,xo+dx,yo-dy);
                     g.drawLine(xo+dx, yo-dy, xo+dx,yo);
                     g.drawLine(xo+dx, yo, xo+2*dx,yo);
                 }
                 else if(yo-dy<50&&yo-dy>50-dy){
                     g.drawLine(xo,yo,xo,50);
                     g.drawLine(xo+dx, 50, xo+dx,yo);
                     g.drawLine(xo+dx, yo, xo+2*dx,yo);
                 }
                  else if(yo>=370&&yo<370+dy){
                     g.drawLine(xo,370,xo,yo-dy);
                     g.drawLine(xo,yo-dy,xo+dx,yo-dy);
                     g.drawLine(xo+dx, yo-dy, xo+dx,370);
                 }
                 else if(yo-dy<=50-dy||yo>=370+dy) {
                  }
             }
```

```
            else if(xo<50&&xo>=(50-dx)){
                  if(yo-dy>=50&&yo<370){
                      g.drawLine(50,yo-dy,xo+dx,yo-dy);
                      g.drawLine(xo+dx, yo-dy, xo+dx,yo);
                      g.drawLine(xo+dx, yo, xo+2*dx,yo);
                  }
                  else if(yo-dy<50&&yo-dy>50-dy){
                      g.drawLine(xo+dx, 50, xo+dx,yo);
                      g.drawLine(xo+dx, yo, xo+2*dx,yo);
                  }
                   else if(yo>=370&&yo<370+dy){
                      g.drawLine(50,yo-dy,xo+dx,yo-dy);
                      g.drawLine(xo+dx, yo-dy, xo+dx,370);
                  }
                  else if(yo-dy<=50-dy||yo>=370+dy) {
                   }
              }
              else if(xo<(50-dx)&xo>=50-2*dx){
                  if(yo-dy>=50&&yo<370){
                      g.drawLine(50, yo, xo+2*dx,yo);
                  }
                  else if(yo-dy<50&&yo-dy>50-dy){
                      g.drawLine(50, yo, xo+2*dx,yo);
                  }
                   else{

                  }
              }
              else if(xo+anchoPing>450&&xo+anchoPing<=(450+dx)){
                  if(yo-dy>=50&&yo<370){
                      g.drawLine(xo,yo,xo,yo-dy);
                      g.drawLine(xo,yo-dy,xo+dx,yo-dy);
                      g.drawLine(xo+dx, yo-dy, xo+dx,yo);
                      g.drawLine(xo+dx, yo, 450,yo);
                  }
                  else if(yo-dy<50&&yo-dy>50-dy){
                      g.drawLine(xo,yo,xo,50);
                      g.drawLine(xo+dx, 50, xo+dx,yo);
                      g.drawLine(xo+dx, yo, 450,yo);
                  }
                   else if(yo>=370&&yo<370+dy){
                      g.drawLine(xo,370,xo,yo-dy);
                      g.drawLine(xo,yo-dy,xo+dx,yo-dy);
                      g.drawLine(xo+dx, yo-dy, xo+dx,370);
                  }
                  else if(yo-dy<=50-dy||yo>=370+dy) {
                   }
              }
```

```
            else if(xo+anchoPing>=(450+dx)&&xo+anchoPing<=(450+2*dx)){
                 if(yo-dy>=50&&yo<370){
                     g.drawLine(xo,yo,xo,yo-dy);
                     g.drawLine(xo,yo-dy,450,yo-dy);
                 }
                 else if(yo-dy<50&&yo-dy>50-dy){
                     g.drawLine(xo,yo,xo,50);
                 }
                  else if(yo>=370&&yo<370+dy){
                    g.drawLine(xo,370,xo,yo-dy);
                    g.drawLine(xo,yo-dy,450,yo-dy);
                 }
                 else if(yo-dy<=50-dy||yo>=370+dy) {
                  }
             }
             else if(xo+anchoPing>=(450+2*dx)){

             }
             xo=xo+2*dx;
             yo=yo;
         }
     }//end for

    g.setColor((Color.green).darker());
    g.drawLine(border,border,gb,border);
    g.drawLine(border,ygridSize+border,gb,ygridSize+border);
    g.drawLine(border,border,border,ygridSize+border);
    g.drawLine(gb,border,gb,ygridSize+border);
}
```

De esta parte del código comentar ademas que, si se ha pulsado el botón solución el ping cambia de color, de forma que, la parte en naranja es la correspondiente al preambulo, la parte en rojo es el sfd, la parte en azul son los datos y la parte en rosa el crc.

- Método `public void actionPerformed(ActionEvent evt){}`

A este metodo solo se accede cuando se produce un evento asociado a un ActionListener.

Los elementos que pueden capturar un evento de este tipo son: *plusVolt*, *minusVolt*, *minusTime*, *plusTime*, *solucion* y *nosolucion*.

```
public void actionPerformed(ActionEvent evt){

        if(evt.getSource() == plusVolt){
            if(vPointer!=0){
                vPointer--;
                escalaY=escalaY/2;
            }
            tVolt.setText(volt[vPointer]+ " V");
            repaint();
        }
        if(evt.getSource() == minusVolt){
               if(vPointer < volt.length-1){
                    vPointer++;
                    escalaY=escalaY*2;
                }
                tVolt.setText(volt[vPointer]+ " V");
                repaint();
        }
        if(evt.getSource() == minusTime){
                if(indiceXslider!=resolXslider.length-1){
                    indiceXslider++;
                }
                if(tPointer!=0){
                    tPointer--;
                    escalaX=escalaX*2;
                    xslider.setAutoscrolls(true);
                    value=xslider.getX();
                    xslider.setMinimum(-resolXslider[indiceXslider]);
                    xslider.setMaximum(resolXslider[indiceXslider]);
                    xslider.setExtent(value);
                }
                tTime.setText(time[tPointer]+ " uS");
                repaint();

            }
        if(evt.getSource() == plusTime){
                if(indiceXslider!=0){
                    indiceXslider--;
                }
                if(tPointer < time.length-1){
                    tPointer++;
                    escalaX=escalaX/2;
                    value=xslider.getX();
                    xslider.setMinimum(-resolXslider[indiceXslider]);
                    xslider.setMaximum(resolXslider[indiceXslider]);
                    xslider.setExtent(value);
                }
                tTime.setText(time[tPointer]+ " uS");
                repaint();
        }
```

```
      if(evt.getSource() == solucion){
           solucion.setVisible(false);
           nosolucion.setVisible(true);
           sol=true;
           preambulo.setVisible(true);
           sfd.setVisible(true);
           datos.setVisible(true);
           crc.setVisible(true);
           repaint();
       }
       if(evt.getSource() == nosolucion){
           solucion.setVisible(true);
           nosolucion.setVisible(false);
           sol=false;
           preambulo.setVisible(false);
           sfd.setVisible(false);
           datos.setVisible(false);
           crc.setVisible(false);
           repaint();
       }
       repaint();
  }
```

Este metodo es el encargado de variar los voltios/division y el tiempo/division. Ademas de mostrar y ocultar la solucion.

- Metodo `private String getVolt(){}`

```
private String getVolt(){
        return volt[vPointer] + " V";
    }
```

Este metodo lo que hace es de volver el voltage/division a quien lo llame en funcion del valor del array volt que indique vPointer.

- Método `private String getTime(){}`

```
private String getTime(){
        return time[tPointer]+ " uS";
    }
```

Este metodo lo que hace es de volver el tiempo/division a quien lo llame en funcion del valor del array volt que indique tPointer.

Capítulo 6.

6. Uso academico de la aplicación

6.1. Par Trenzado[1]

6.1.1. Historia del par trenzado

Los primeros teléfonos utilizaban líneas telegráficas, o alambres abiertos de un solo conductor de circuitos de conexión a tierra. En la década de 1880-1890 fueron instalados tranvías eléctricos en muchas ciudades de Estados Unidos, lo que indujo ruido en estos circuitos. Al ser inútiles las demandas por este asunto, las compañías telefónicas pasaron a los sistemas de circuitos balanceados, que tenían el beneficio adicional de reducir la atenuación, y por lo tanto, cada vez mayor alcance.

Como la distribución de energía eléctrica se hizo cada vez más común, esta medida resultó insuficiente. Dos cables, colgados a ambos lados de las barras cruzadas en los postes de alumbrado público, compartían la ruta con las líneas de energía eléctrica. En pocos años, el creciente uso de la electricidad trajo de nuevo un aumento de la interferencia, por lo que los ingenieros idearon un método llamado transposición de conductores, para cancelar la interferencia. En este método, los conductores intercambiaban su posición una vez por cada varios postes. De esta manera, los dos cables recibirían similares interferencias electromagnéticas de las líneas eléctricas. Esto representó una rápida implementación del trenzado, a razón de unos cuatro trenzados por kilómetro, o seis por milla. Estas líneas balanceadas de alambre abierto con transposiciones periódicas aún subsisten, hoy en día, en algunas zonas rurales de Estados Unidos.

Los cables de par trenzado fueron inventados por Alexander Graham Bell en 1881. En 1900, el conjunto de la red estadounidense de la línea telefónica era o de par trenzado o hilo abierto con la transposición a la protección contra interferencias. Hoy en día, la mayoría de los millones de kilómetros de pares trenzados en el mundo está fija en instalaciones aéreas, propiedad de las compañías telefónicas, y se utiliza para el servicio de voz, y sólo son manejados o incluso vistos por los trabajadores telefónicos.

6.1.2. Descripcion, tipos y categorias de cable de par trenzado

El entrelazado de los cables disminuye la interferencia debido a que el área de bucle entre los cables, la cual determina el acoplamiento eléctrico en la señal, se ve aumentada. En la operación de balanceado de pares, los dos cables suelen llevar señales paralelas y adyacentes (modo diferencial), las cuales son combinadas mediante sustracción en el destino. La tasa de trenzado, usualmente definida en vueltas por kilómetro, forma parte de las especificaciones de un tipo concreto de cable. Cuanto mayor es el número de vueltas, menor es la atenuación de la diafonía. Donde los pares no están trenzados, como en la mayoría de las conexiones telefónicas residenciales, un miembro del par puede estar más cercano a la fuente que el otro y, por tanto, expuesto a niveles ligeramente distintos de interferencias electromagnéticas.

- Tipos de cable:

  - *Unshielded twisted pair* o par trenzado sin blindaje: son cables de pares trenzados sin blindar que se utilizan para diferentes tecnologías de redes locales. Son de bajo costo y de fácil uso, pero producen más errores que otros tipos de cable y tienen limitaciones para trabajar a grandes distancias sin regeneración de la señal.

  - *Shielded twisted pair o par trenzado blindado: se trata de cables de cobre aislados dentro de una cubierta protectora, con un número específico de trenzas por pie. STP se refiere a la cantidad de aislamiento alrededor de un conjunto de cables y, por lo tanto, a su inmunidad al ruido. Se utiliza en redes de ordenadores como Ethernet o Token Ring. Es más caro que la versión sin blindaje.*

  - *Foiled twisted pair o par trenzado con blindaje global: son unos cables de pares que poseen una pantalla conductora global en forma trenzada. Mejora la protección frente a interferencias y su impedancia es de 12 ohmios.*

- Categoria:

Categoría	Ancho de banda (MHz)	Aplicaciones	Notas
Categoría 1	0,4 MHz	Líneas telefónicas y módem de banda ancha.	No descrito en las recomendaciones del EIA/TIA. No es adecuado para sistemas modernos.
Categoría 2	4 MHz	Cable para conexión de antiguos terminales como el IBM 3270.	No descrito en las recomendaciones del EIA/TIA. No es adecuado para sistemas modernos.
Categoría 3	16 MHz	10BASE-T and 100BASE-T4 Ethernet	Descrito en la norma EIA/TIA-568. No es adecuado para transmisión de datos mayor a 16 Mbit/s.
Categoría 4	20 MHz	16 Mbit/s Token Ring	
Categoría 5	100 MHz	100BASE-TX y 1000BASE-T Ethernet	
Categoría 5e	100 MHz	100BASE-TX y 1000BASE-T Ethernet	Mejora del cable de Categoría 5. En la práctica es como la categoría anterior pero con mejores normas de prueba. Es adecuado para Gigabit Ethernet
Categoría 6	250 MHz	1000BASE-T Ethernet	Cable más comúnmente instalado en Finlandia según la norma SFS-EN 50173-1.
Categoría 6e	250 MHz (500MHz según otras fuentes)	10GBASE-T Ethernet (en desarrollo)	No es estandarizado. Lleva el sello del fabricante.
Categoría 7	600 MHz	En desarrollo. Aún sin aplicaciones.	Cable U/FTP (sin blindaje) de 4 pares.
Categoría 7a	1200 MHz	Para servicios de telefonía, Televisión por cable y Ethernet 1000BASE-T en el mismo cable.	Cable S/FTP (pares blindados, cable blindado trenzado) de 4 pares. Norma en desarrollo.
Categoría 8	1200 MHz	Norma en desarrollo. Aún sin aplicaciones.	Cable S/FTP (pares blindados, cable blindado trenzado) de 4 pares.

6.1.3. Caracteristicas de la transmision por par trenzado

Está limitado en distancia, ancho de banda y tasa de datos. También destacar que la atenuación es una función fuertemente dependiente de la frecuencia. La interferencia y el ruido externo también son factores importantes, por eso se utilizan coberturas externas y el trenzado. Para señales analógicas se requieren amplificadores cada 5 o 6 kilómetros, para señales digitales cada 2 ó 3. En transmisiones de señales analógicas punto a punto, el ancho de banda puede llegar hasta 250 kHz. En transmisión de señales digitales a larga distancia, el data rate no es demasiado grande, no es muy efectivo para estas aplicaciones.

En redes locales que soportan ordenadores locales, el data rate puede llegar a 10 Mbps (Ethernet) y 100 Mbps (Fast-Ethernet).

En el cable par trenzado de cuatro pares, normalmente solo se utilizan dos pares de conductores, uno para recibir (cables 3 y 6) y otro para transmitir (cables 1 y 2),

aunque no se pueden hacer las dos cosas a la vez, teniendo una trasmisión half-dúplex. Si se utilizan los cuatro pares de conductores la transmisión es full-dúplex.

- **Ventajas:**

  - Bajo costo en su contratación.
  - Alto número de estaciones de trabajo por segmento.
  - Facilidad para el rendimiento y la solución de problemas.
  - Puede estar previamente cableado en un lugar o en cualquier parte.

- **Desventajas:**

  - Altas tasas de error a altas velocidades.
  - Ancho de banda limitado.
  - Baja inmunidad al ruido.
  - Baja inmunidad al efecto crosstalk (diafonía)
  - Alto costo de los equipos.
  - Distancia limitada (100 metros por segmento).

6.2. Nivel físico y de enlace[2]

6.2.1. Nivel físico

El nivel físico o capa física se refiere a las transformaciones que se hacen a la secuencia de bits para trasmitirlos de un lugar a otro. Siempre los bits se manejan dentro del PC como niveles eléctricos. Por ejemplo, puede decirse que en un punto o cable existe un 1 cuando está en una cierta cantidad de voltios y un cero cuando su nivel es de 0 voltios. Cuando se trasmiten los bits siempre se transforman en otro tipo de señales de tal manera que en el punto receptor puede recuperarse la secuencia de bits originales.

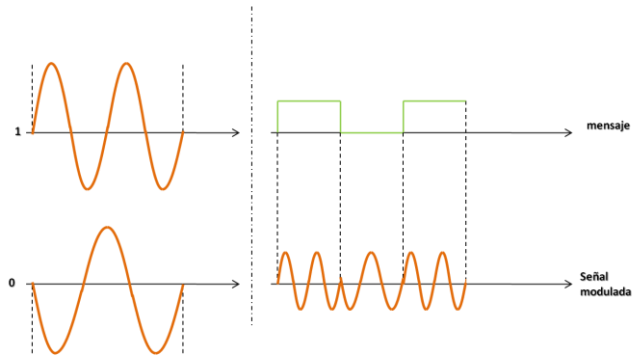


Figura 16. Codificación eléctrica.

La capa física es la capa de red más básica, proporcionando únicamente los medios para transmitir bit a bit sobre un enlace de datos físico conectado a nodos de red. Consecuentemente, la capa física, no añade cabeceras de paquete a los datos. Las cadenas de bits pueden ser agrupadas en palabras codificadas o símbolos, y convertidas a señales físicas, que son transmitidas sobre un medio de transmisión físico.

La capa física proporciona una interfaz eléctrica, mecánico y procedimental para el medio de transmisión.

- **Entramado**

La capa física le proporciona servicios a la capa de enlaces de datos con el objetivo que esta le proporcione servicios a la capa de red. La capa física recibe un flujo de bits e intenta enviarlo a destino, no siendo su responsabilidad entregarlos libre de errores. La capa de enlace de datos es la encargada de detectar y corregir los errores. Los errores pueden consistir en una mayor o menor cantidad de bits recibidos o diferencias en los valores que se emitieron y en los que se recibieron.

Un método común de detección de errores es que la capa de enlace de datos separe el flujo en tramas separadas y que realice la suma de verificación de cada trama. Cuando una trama llega a su destino se recalcula la suma de verificación. Si es distinta de la contenida en la trama es porque ha ocurrido un error y la capa de enlace debe solucionarlo.

- **Funciones y servicios de la capa**

Las principales funciones y servicios realizados por la capa física son:

- Envío bit a bit entre nodos
- Proporcionar una interfaz estandarizada para los medios de transmisión físicos, incluyendo:
    - Especificaciones mecánicas de los conectores eléctricos y cables, por ejemplo longitud máxima del cable
    - Especificación eléctrica de la línea de transmisión, nivel de señal e impedancia
    - Interfaz radio, incluyendo el espectro electromagnético, asignación de frecuencia y especificación de la potencia de señal, ancho de banda analógico, etc.
    - Especificaciones para IR sobre fibra óptica o una conexión de comunicación wireless mediante IR
- Modulación
- Codificación de línea
- Sincronización de bits en comunicación serie síncrona
- Delimitación de inicio y final, y control de flujo en comunicación serie asíncrona
- Multiplexación de Conmutación de circuitos
- Detección de portadora y detección de colisión utilizada por algunos protocolos de acceso múltiple del nivel 2
- Ecualización, filtrado, secuencias de prueba, forma de onda y otros procesados de señales de las señales físicas

La capa física se ocupa también de:

- Configuración de la línea punto a punto, multipunto o punto a multipunto
- Topología física de la red, por ejemplo en bus, anillo, malla o estrella
- Comunicación serie o paralela
- Modo de transmisión Simplex, half duplex o full duplex

6.2.2. Nivel de enlace

El nivel de enlace de datos (en inglés data link level) o capa de enlace de datos es la segunda capa del modelo OSI, el cual es responsable de la transferencia fiable de información a través de un circuito de transmisión de datos. Recibe peticiones de la capa de red y utiliza los servicios de la capa física.

El objetivo de la capa de enlace es conseguir que la información fluya, libre de errores, entre dos máquinas que estén conectadas directamente (servicio orientado a conexión).

Para lograr este objetivo tiene que montar bloques de información (llamados tramas en esta capa), dotarles de una dirección de capa de enlace (Dirección MAC), gestionar la detección o corrección de errores, y ocuparse del control de flujo entre equipos (para evitar que un equipo más rápido desborde a uno más lento).

Cuando el medio de comunicación está compartido entre más de dos equipos es necesario arbitrar el uso del mismo. Esta tarea se realiza en la subcapa de control de acceso al medio.

Dentro del grupo de normas IEEE 802, la subcapa de enlace lógico se recoge en la norma IEEE 802.2 y es común para todos los demás tipos de redes (Ethernet o IEEE 802.3, IEEE 802.11 o Wi-Fi, IEEE 802.16 o WiMAX, etc.); todas ellas especifican un subcapa de acceso al medio así como una capa física distinta.

Otro tipo de protocolos de la capa de enlace serían PPP (Point to point protocol o protocolo punto a punto), HDLC (High level data link control o protocolo de enlace de alto nivel), por citar dos.

En la práctica la subcapa de acceso al medio suele formar parte de la propia tarjeta de comunicaciones, mientras que la subcapa de enlace lógico estaría en el programa adaptador de la tarjeta (driver en inglés).

- **Funciones**

La capa de enlace de datos es responsable de la transferencia fiable de información a través de un Circuito eléctrico de transmisión de datos. La transmisión de datos lo realiza mediante tramas que son las unidades de información con sentido lógico para el intercambio de datos en la capa de enlace. También hay que tener en cuenta que en el modelo TCP/IP se corresponde a la segunda capa.

Sus principales funciones son:

- Iniciación, terminación e identificación.
- Segmentación y bloqueo.
- Sincronización de octeto y carácter.
- Delimitación de trama y transparencia.
- Control de errores.
- Control de flujo.
- Recuperación de fallos.
- Gestión y coordinación de la comunicación.

Iniciación, terminación e identificación

La función de iniciación comprende los procesos necesarios para activar el enlace e implica el intercambio de tramas de control con el fin de establecer la disponibilidad de las estaciones para transmitir y recibir información.

Las funciones de terminación son de liberar los recursos ocupados hasta la recepción/envío de la última trama. También de usar tramas de control. La identificación es para saber a que terminal se debe de enviar una trama o para conocer quien envía la trama. Se lleva a cabo mediante la dirección de la capa de enlace.

Segmentación y bloqueo

La segmentación surge por la longitud de las tramas ya que si es muy extensa, se debe de realizar tramas más pequeñas con la información de esa trama excesivamente larga.

Si estas tramas son excesivamente cortas, se ha de implementar unas técnicas de bloque que mejoran la eficiencia y que consiste en concatenar varios mensajes cortos de nivel superior en una única trama de la capa de enlace más larga.

Sincronización de octeto y carácter

En las transferencias de información en la capa de enlace es necesario identificar los bits y saber que posición les corresponde en cada carácter u octeto dentro de una serie de bits recibidos.

Esta función de sincronización comprende los procesos necesarios para adquirir, mantener y recuperar la sincronización de carácter u octeto. Es decir, poner en fase los mecanismos de codificación del emisor con los mecanismos de decodificación del receptor.

Delimitación de trama

La capa de enlace debe ocuparse de la delimitación y sincronización de la trama. Para la sincronización puede usar 3 métodos:

- El primero de ellos es "Principio y fin" (caracteres específicos para identificar el principio o el fin de cada trama).
- También puede usar "Principio y cuenta" (Utiliza un carácter para indicar comienzo y seguido por un contador que indica su longitud).
- Por último puede usar el "Guion" (se emplea una agrupación especifica de bits para identificar el principio y fin mediante banderas/flags).

La transparencia se realiza mediante la inserción de bits. Consta de ir contando los unos consecutivos y cuando se encuentra con 5 unos seguidos y consecutivos introduce el bit 0 después del quinto uno. Ejemplo: Las banderas/flag suelen ser 01111110, y al aplicar la transparencia pasa a ser 011111010.

Control de errores

Proporciona detección y corrección de errores en el envío de tramas entre computadores, y provee el control de la capa física. Sus funciones, en general, son:

- Identificar Trama de datos
- Códigos detectores y correctores de error
- Control de flujo
- Gestión y coordinación de la comunicación.

Correctores de error : Es opcional en esta capa, la encargada de realizar esta funcion es la capa de transporte , en una WAN es muy problable que la verificacion, la realiza la capa de enlace

Para la Identificación de tramas puede usar distintas técnicas como:

- Contador de caracteres
- Caracteres de inicio y final con caracteres de relleno
- Secuencia de bits indicadora de inicio y final, con bits de relleno

El control de flujo es necesario para no 'agobiar' al receptor. Se realiza normalmente en la capa de transporte, también a veces en la capa de enlace. Utiliza mecanismos de retroalimentación. Suele ir unido a la corrección de errores y no debe limitar la eficiencia del canal.

Los métodos de control de errores son básicamente 2:

- FEC o corrección de errores por anticipado y no tiene control de flujo.
- ARQ: Posee control de flujo mediante parada y espera, o/y ventana deslizante.

Las posibles implementaciones son:

- Parada y espera simple: Emisor envía trama y espera una señal del receptor para enviar la siguiente o la que acaba de enviar en caso de error.
- Envío continuo y rechazo simple: Emisor envía continuamente tramas y el receptor las va validando. Si encuentra una errónea, elimina todas las posteriores y pide al emisor que envíe a partir de la trama errónea.
- Envío continuo y rechazo selectivo: transmisión continua salvo que sólo retransmite la trama defectuosa.

La detección de errores la realiza mediante diversos tipos de códigos del que hay que resaltar:

- CRC (control de redundancia cíclica)
- Simple paridad
- Paridad cruzada (Paridad horizontal y vertical)
- Suma de verificación

La corrección de errores están basados en Código Hamming, por repetición, verificación de paridad cruzada, Reed-Solomon y de goyle.

Control de flujo

El control de flujo es necesario para no saturar al receptor de uno a más emisores. Se realiza normalmente en la capa de transporte, también a veces en la capa de enlace. Utiliza mecanismos de retroalimentación. Suele ir unido a la corrección de errores y no debe limitar la eficiencia del canal. El control de flujo conlleva dos acciones importantísimas que son la detección de errores y la corrección de errores.

La detección de errores se utiliza para detectar errores a la hora de enviar tramas al receptor e intentar solucionarlos. Se realiza mediante diversos tipos de códigos del que hay que resaltar el CRC (códigos de redundancia cíclica), simple paridad (puede ser par, números de 1´s par, o impar) paridad cruzada (Paridad horizontal y vertical) y Suma de verificación.

La corrección de errores surge a partir de la detección para corregir errores detectados y necesitan añadir a la información útil un número de bits redundantes bastante superior al necesario para detectar y retransmitir. Sus técnicas son variadas. El Código Hamming, Repetición, que cada bit se repite 3 veces y en caso de fallo se toma el bit que más se repite; También puede hacerse mediante verificación de paridad cruzada, Reed-Solomon y de goyle.

También cabe destacar los protocolos HDLC que es un control de enlace de datos a alto nivel, orientado a bit y obedece a una ARQ de ventana deslizante o continuo. También existen protocolos orientados a carácter.

Recuperación de fallos

Se refiere a los procedimientos para detectar situaciones y recuperar al nivel de situaciones anómalas como la ausencia de respuesta, recepción de tramas inválidas, etc. Las situaciones más típicas son la pérdida de tramas, aparición de tramas duplicadas y llegada de tramas fuera de secuencia.

Si no se tratasen correctamente estos eventos se perderá información y se aceptarán datos erróneos como si fuesen correctos. Generalmente se suelen utilizar contadores para limitar el número de errores o reintentos de los procesos y procedimientos. También se pueden usar temporizadores para establecer plazos de espera (timeout) de los sucesos.

Gestión y coordinación de la comunicación

La gestión atiende a 2 tipos:

- El primero de ellos es un sistema centralizado donde existe una máquina maestra y varias esclavas. Estas conexiones se pueden realizar punto a punto o multipunto.
- El segundo de ellos es el distribuido, donde no existe máquina maestra y todas compiten por el control del sistema de comunicación.

La coordinación se puede realizar mediante selección o contienda:

- La selección se puede implementar mediante sondeo/selección, donde el maestro recoge un mensaje de una secundaria y se la entrega a quien seleccione. También es posible asignando un testigo a una máquina que es la que puede emitir mensajes/tramas. Son típicas las configuraciones Token Ring y Token Bus.
- La contienda se basa en que cada ordenador emite su trama/mensaje cuando le apetece. Todos los componentes de la red son tanto emisores como receptores. Son típicos los sistemas ALOHA y CSMA/CD. Hay que tener cuidado con las colisiones.

1: Definición según: http://es.wikipedia.org/wiki/Cable_de_par_trenzado

2: Definicion según: http://es.wikipedia.org/wiki/Capa_f%C3%ADsica y http://es.wikipedia.org/wiki/Capa_de_enlace_de_datos

6.3. Codificación de los bits en una red Ethernet

La codificación de los bits en la red que estudiamos en el laboratorio utiliza el código Manchester, que consiste en lo siguiente:

Código Manchester[3]: La codificación Manchester, también denominada codificación bifase-L, es un método de codificación eléctrica de una señal binaria en el que en cada tiempo de bit hay una transición entre dos niveles de señal. Es una codificación autosincronizada, ya que en cada bit se puede obtener la señal de reloj, lo que hace posible una sincronización precisa del flujo de datos. Una desventaja es que consume el doble de ancho de banda que una transmisión asíncrona. Hoy en día hay numerosas codificaciones (8b/10b) que logran el mismo resultado pero consumiendo menor ancho de banda que la codificación Manchester.

En la siguiente imagen se muestra las dos codificaciones más conocidas:

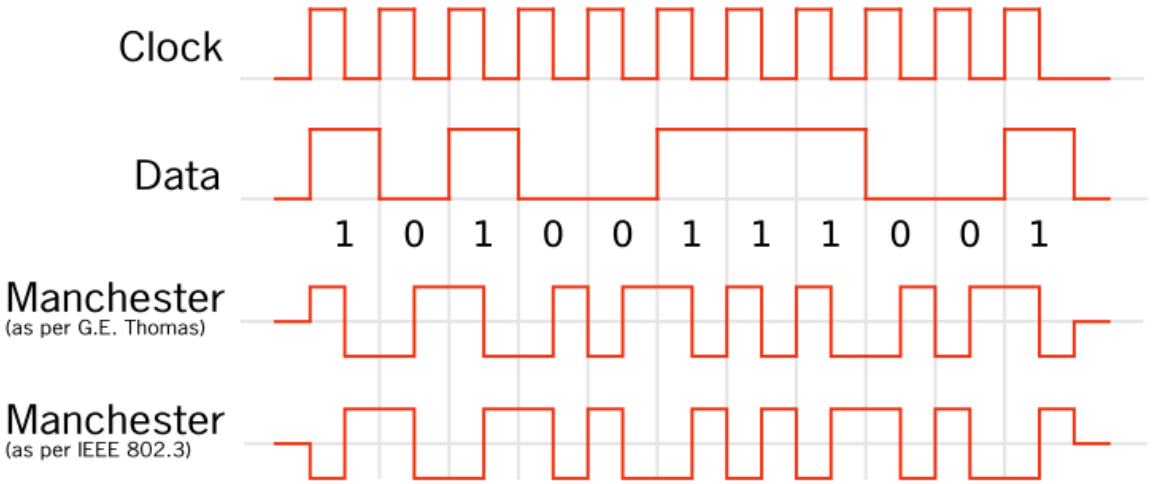


Figura 17. Codificación Manchester

El utilizado para comunicaciones Ethernet es el segundo (IEEE 802.3), donde un uno se representa como un flanco ascendente y un cero como un flanco descendente.

3: Definición según: http://es.wikipedia.org/wiki/Codificaci%C3%B3n_Manchester

Los datos se organizan según el siguiente formato:

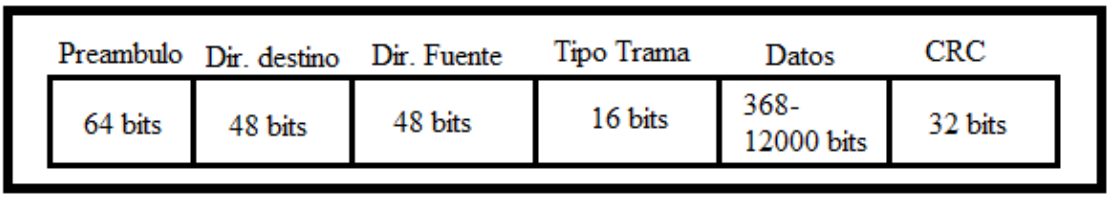


Figura 18. Formato trama Ethernet

Los bits del Preambulo junto con el SFD se envían con su orden natural. Los campos de la dirección destino, de la dirección de la fuente, del tipo de trama, de datos y el CRC se envían los bytes en orden natural, y los bits de cada byte se envían de forma que el primero que se transmite es el de menor peso.

Capítulo 7.

7. Conclusiones y líneas futuras

La principal conclusión es que en este proyecto fin de carrera se han logrado todos los objetivos que inicialmente estaban expuestos al principio respecto de la práctica 3 de la asignatura Fundamentos de Telemática.

A modo de resumen, los objetivos alcanzados son:

- Se ha construido un simulador para poder entender el funcionamiento del Cablemeter así como dar a conocer las distintas averias que se puenden dar en una red Ethernet. También ayuda a comprender como se representan los datos de una trama Ethernet a nivel físico y de enalce.

- Hemos creado una interfaz gráfica para poder realizar las prácticas desde casa, ya que no todo el mundo tiene en su poder un osciloscopio, y un testeador de cable.

- La interfaz se ha tratado de hacer lo más real posible, para que se parezcan, tanto las prácticas realizadas en el laboratorio, como las simuladas en el PC.

- El código fuente se deja abierto y disponible, para así dejar el programa abierto para posibles mejoras futuras, y a la depuración de posibles errores.

- Se ha conseguido el objetivo de la portabilidad, ya que el programa al estar hecho en JAVA, se puede ejecutar en cualquier plataforma (Windows, Linux,Solaris…).

- Uno de los objetivos principales del proyecto que también se han cumplido, es que el APPLET es muy intuitivo a la hora de realizar las prácticas.

Las líneas futuras para este programa dependerán de las nuevas necesidades que se encuentren a los largo de las prácticas donde se utilice este simulador. En prácticas sucesivas de los cursos posteriores a la asignatura de Fundamentos de Telemática, tanto profesores como alumnos podrían ver que mejoras se pueden realizar en función de las necesidades didácticas, completando el ciclo de vida de este software. La práctica del cablemeter se podría implementar en varios idiomas y se podrían añadir distincas unidades de medida para la longitud de los cables, así como la posibilidad de configurar el dispositivo para ello.

La especificación de este Proyecto Final de Carrera consistió en una serie de objetivos claramente definidos y que se han conseguido en su totalidad. El definir unas líneas futuras para él va a depender de las necesidades didácticas que se puedan ir encontrando durante la utilización de la herramienta a lo largo de los cursos académicos. Se tiene que producir un proceso de retroalimentación en el que los usuarios (alumnos y profesores), dependiendo de hipotéticas necesidades futuras que el simulador pudiera ofrecer. Esta retroalimentación permitiría modificaciones en el programa para ajustarlo a necesidades futuras.

Capítulo 8.

8. Pasos para realizar un archivo ".jar"

Los archivos ".java" que contienen el código, al compilarlos generan los archivos ".class" que son los que hacen funcionar la aplicación. Para tener juntas estas clases y las imágenes (si las hubiera dentro del proyecto) del programa, lo mejor es empaquetarlo todo en un archivo ".jar".

Los pasos para crear el archivo ".jar" son los siguientes:

i. Entrar en las propiedades de "Mi PC", en "Opciones Avanzadas, Variables de entorno", y aquí creamos una variable llamada Path, donde su valor será el directorio bin del JDK que tengamos instalado.

ii. Editamos el archivo MANIFEST.MF, y aquí indicamos cuál es la clase principal (main class) del programa.

iii. Editar el archivo Comandos-jar.bat e indicamos cómo se llamará el archivo ".jar" que se desea construir, los archivos ".class" y todas las imágenes si hubiese inmágenes insertadas en el proyecto. La forma de editarlo es la siguiente:

jar cmvf manifest.mf AppletPráctica3Telematica.jar*.class

El asterisco en .class quiere decir que se incluirán todos los archivos que contengan esta extensión.

iv. Finalmente, hacemos doble click en el archivo Comandos-jar.bat. Debe estar absolutamente todo en una misma carpeta, sino el archivo .bat no lo incluirá.

Después de hacer doble click, en el archivo ".jar" se encuentra construido en la misma carpeta donde estamos, listo para funcionar.

Recomendaciones:

Tener instalado el JDK 1.7.0 o superior.

Si en el sistema está instalado el WinRAR, desasociarlo con las extensiones ".jar", ya que si no lo hacemos, el archivo ".jar" en vez de ejecutarse, se abre para explorar lo que hay dentro, es decir, el sistema lo trata como un archivo comprimido a explorar, no como un ejecutable.

Anexos.

Anexo A.
Manual de usuario

A. Introducción

El programa ha sido creado mediante JAVA, por lo que en los ordenadores donde se utilice se deberá tener instalado el JRE(Java Runtime Environment) para que pueda ejecutarse, y éste incluye la Java Virtual Machine (JVM) y la API. Todo está incluido en el J2SE (Java 2 Standard Edition), donde puede instalarse cada cosa por separado.

Si lo que desea es modificar el código fuente del programa, no es suficiente tener instalado el JRE, sino que se necesita el J2SE.

El software está disponible gratuitamente en http://java.sun.com/

La aplicación consiste en un único archivo con extensión ".jar" (AppletPráctica3Telematica.jar), donde se encuentran las clases que componen el programa.

B. Objetivos

El objetivo final de esra práctica es que el alumno entienda el funcionamiento de un testeador de red (Cablemeter),y como funciona una red Ethernet a nivel físico y de enlace.

C. Página principal

El programa se inicia desde una web (http://labit501.upct.es/~jpsaura/), desde aquí tendremos acceso a todo el conjunto de ventanas de las cuales dispone nuestro Applet.

La página tiene el siguiente aspecto:

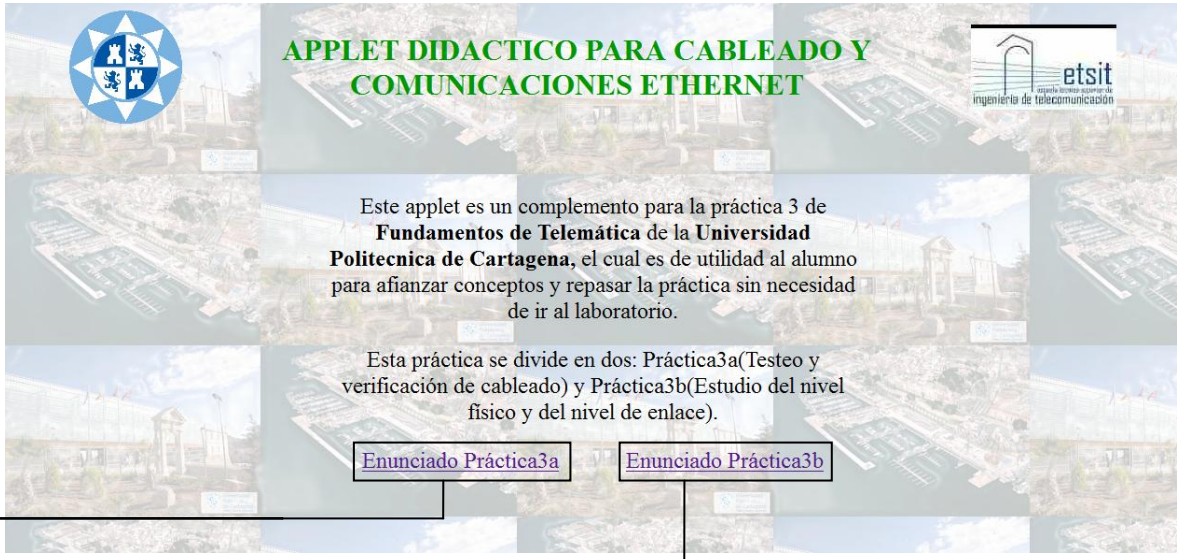


Figura 19

Si se hace click encima de este enlace, se abre una página en la que se muestra el enunciado de la primera parte de la práctica así como un enlace al correspondiente Applet.

Si se hace click encima de este otro enlace, se abre una página en la que se muestra el enunciado de la segunda parte de la práctica así como un enlace al correspondiente Appelt.

D. Página práctica 3A

El aspecto de esta página es el siguiente:

Figura 20

Al hacer click en este enlace se abre una página donde se muestra el Applet de dicha parte de la práctica.

E. Página práctica 3B

El aspecto de esta página es el siguiente:

Figura 21

Al hacer click en este enlace se abre una página donde se muestra el Applet de dicha parte de la práctica.

F. Applet práctica 3A

Interfaz de usuario

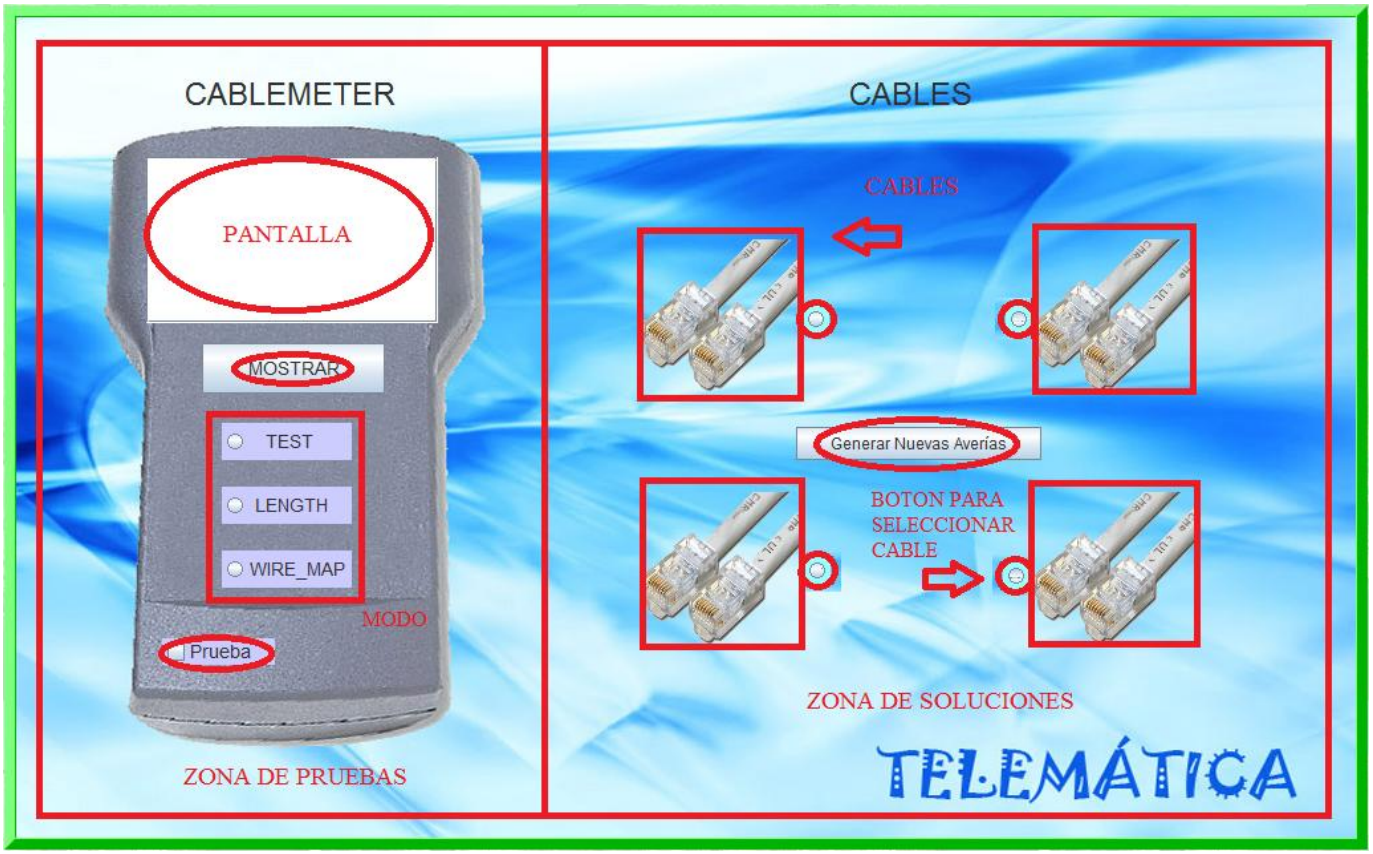


Figura 22

La interfaz de usuario del Applet es la mostrada en la Figura. Como se puede observar está dividida en dos zonas principales: zona de pruebas y zona de soluciones.

1. En la zona de pruebas es donde se encuentra el Cablemeter, es la zona de la cual tenemos que interpretar los resultados.

Esta zona está constituida por: una pantalla, botón "MOSTRAR", tres botones de MODO("TEST","LENGTH" y "WIRE_MAP") y el botón de "Prueba".

- Pantalla: Es donde se visualizan los resultados.
- Mostrar: Es el botón que hay que pulsar para que se muestren los resultados en la pantalla.
- Modo: En modo hay tres botones de los cuales solo se puede seleccionar uno en función de lo que se quiera analizar.

- Prueba: El botón de prueba indica si estamos haciendo las mediciones con la prueba conectada o no. Si está pulsado, las mediciones sí se están haciendo con la prueba, si no lo está no.

2. La zona de soluciones es donde se comprueba si se han interpretado bien los mensajes del Cablemeter.

Esta zona está constituida por: 4 cables y el botón " GenerarNuevasAverias".

- Cables: Al hacer click encima de la imagen de un cable se muestran sus propiedades. Si seleccionamos uno de los botones que hay al lado de los cables, lo que hacemos es seleccionar el cable que queremos que compruebe el Cablemeter.

- GenerarNuevasAverias: Este botón lo que hace es cambiar las averías de los cables, de forma que, cada vez que sea pulsado las averías cambiaran de forma aleatoria.

Uso del Applet

Se selecciona un cable, un modo y el botón de prueba en el caso de que se quieran hacer las mediciones con ella. Una vez hecho esto se procede a pulsar el botón "MOSTRAR" del Cablemeter y los resultados serán visualizados en la pantalla de este.

El usuario tiene que interpretar los resultados que salen en la pantalla del Cablemeter; y cuando tenga una respuesta, puede comprobar si es correcta haciendo click en la imagen del cable que había comprobado. Cuando se hayan comprobado todos los cables, se puede proceder a generar nuevas averías pulsando el botón "GenerarNuevasAverias".

NOTA: Las medidas en las que no es utilizada la Prueba no son fiables. Para realizar una buena medición es necesario que la Prueba esté conectada.

G. Applet práctica 3B

Interfaz de usuario

Figura 23

La interfaz de usuario del Applet es la mostrada en la Figura. Consta de dos botones (botón “Enviar Ping”, botón “Mostrar Captura”) y de un osciloscopio.

- Enviar Ping: Con este botón lo que se hace es enviar el ping, el cual pasará por el osciloscopio y llegará al PC destino.

- Mostrar Captura: Este botón aparece en el PC destino cuando el ping se ha enviado, y lo que hace es mostrarlo como lo haría “wireshark”.

- Osciloscopio: Al hacer click en la imagen se abre una ventana en la cual podemos observar la forma de onda ideal del ping mostrada por un osciloscopio.

Uso del Applet

Lo primero que hay que hacer es pulsar el botón “Enviar Ping”, para que el ping se envíe y pueda ser analizado. Una vez hecho esto, se puede pasar a visualizar el ping capturado por el osciloscopio o por “wireshark”.

Se deben de interpretar los datos de la captura de “wireshark”, viendo los distintos campos del ping, cabecera, dirección IP origen-destino, tamaño, etc. De la misma forma se deben interpretar los pulsos mostrados por el osciloscopio, intentando diferenciar las cuatro partes más importantes (Preambulo,sdf,datos,crc).

H. Osciloscopio

Interfaz grafica del osciloscopio

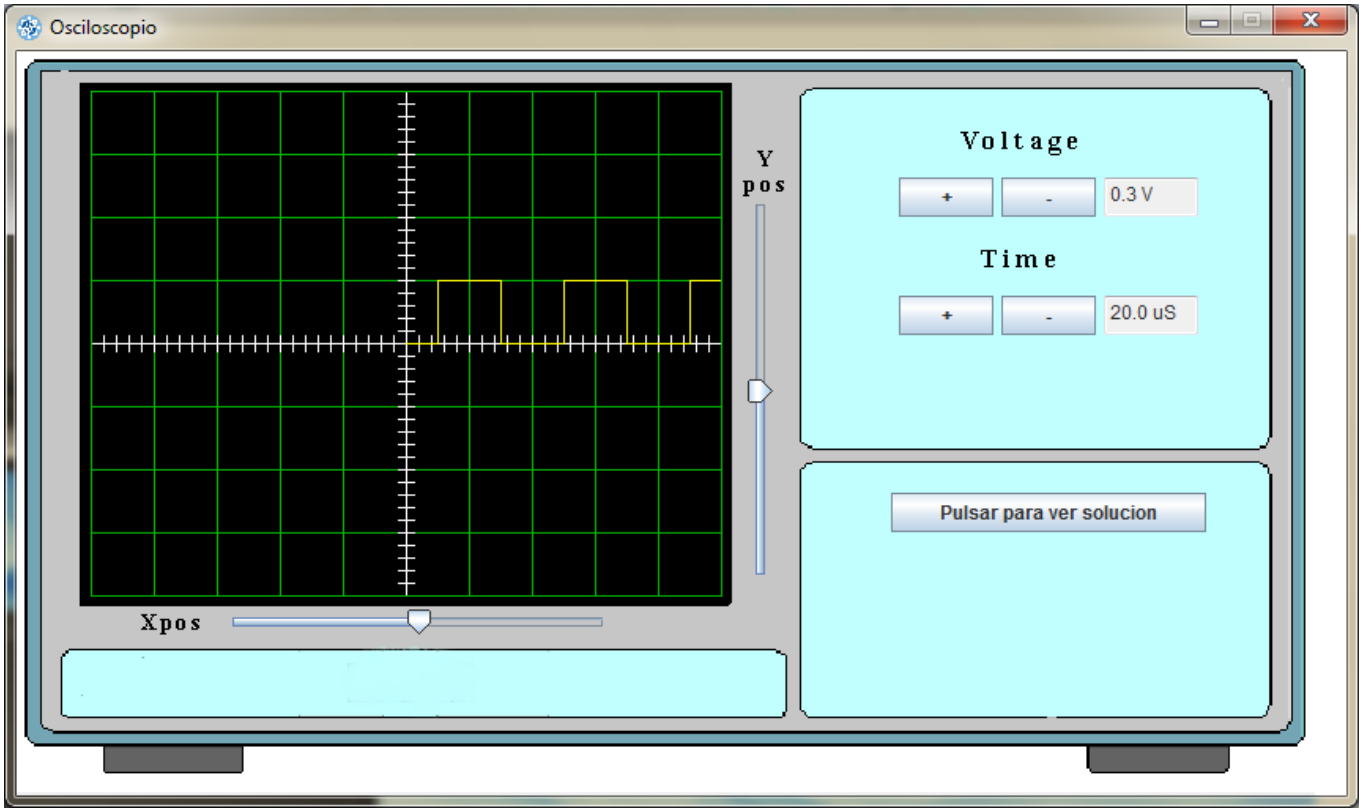


Figura 24

- Botones “Voltage”: Cuando pulsamos estos botones lo que hacemos es cambiar los voltios por división (ejey) del osciloscopio, de forma que la señal varia su amplitud.

- Botones “Time”: Cuando pulsamos estos botones lo que hacemos es cambiar el tiempo por división (ejex) del osciloscopio, de forma que la señal varia su anchura.

- Barra “Xpos”: Cuando desplazamos esta barra movemos la señal por el ejex.

- Barra “Ypos”: Cuando desplazamos esta barra movemos la señal por el ejey.

- Boton “Pulsar para ver solucion”: Cuando pulsamos este botón la señal cambia de color, diferenciándose así las cuatro partes más importante por las que está formado el ping: preámbulo,sdf,datos,crc.

Anexo B.

Trabajo a realizar por el alumno.

I. Cuestionario Cablemeter Fluke 620

[1] Enumerar los tipos de cables que puede testear el cablemeter.

[2] Calibrar el cable nº 1. ¿Qué longitud indica la calibración?

[3] Hacer un esquema de las averías a simular en el apartado 3.4.

[4] Hacer un análisis de los cables etiquetados con y sin prueba que se entregan en la práctica y determinar que tipo de avería tienen. Realizar su WIRE-MAP.

2	
3	
4	
5	
6	
7	
8	
9	

[5] Determinar las longitudes de los cables etiquetados que se entregan en la práctica realizando las medidas con y sin la prueba, dando los resultados en pies y en metros.

Nº	Metros (con prueba)	Metros (sin Prueba)	Pies (con prueba)	Pies (sin prueba)
2				
3				
4				
5				
6				
7				
8				
9				

[6] Dibujar un mapa de conexiones de los cables etiquetados que se entregan en la práctica.

2	
3	
4	
5	
6	
7	
8	
9	

J. Cuestionario estudio del nivel físico y de enlace de una red Ethernet.

[1] ¿ Qué dirección IP tiene cada uno de los ordenadores del banco de trabajo?

Puesto de trabajo	Dirección IP	DNS	Gateway

[2] ¿Se utilizan latiguillos cruzados o normales? ¿Porqué?

[3] ¿ Cómo se representa el "0" lógico y el "1" lógico? Representarlo con eje vertical en voltios y eje horizontal en nanosegundos.

[4] ¿ Cuál es la velocidad de transmisión (en Mbps) observada al inicio de la trama? ¿Porqué? ¿Y en la parte central de la trama?

[5] ¿Cuál es la dirección física de la tarjeta Ethernet del PC origen? ¿Y del PC destino?

[6] ¿Qué significado tiene este byte? ¿Qué tipo de trama es (EthernetII, 802.3, etc.)?

[7] Seleccionar un paquete de solicitud de "ping" describiendo los niveles de enlace y red, con el máximo detalle en el nivel de enlace.

Bibliografía.

→ Manual de prácticas de Fundamentos de Telemática.

http://labit501.upct.es/~fburrull/docencia/FundamentosTelematica/prácticas/P3a_TesteoYVerificacionDeCableado.pdf

http://labit501.upct.es/~fburrull/docencia/FundamentosTelematica/prácticas/P3b_EstudioDelNivelFisicoYDelNivelDeEnlace.pdf

→ Manuales de clase Java. Clase Graphics:

http://www.javaya.com.ar/detalleconcepto.php?codigo=130&inicio=40

→ Manuales de métodos de JAVA.

http://objetos2.wordpress.com/2010/04/24/clase-window-listener/

→ Manuales de interfaz gráfica de JAVA.

http://es.scribd.com/doc/36907088/Interfaz-Grafica-en-Java

http://doutdex.wordpress.com/2007/06/09/gui-graphical-user-interface-interface-grafica-de-usuario-en-java/

→ Apuntes de clase de la asignatura Fundamentos de Telemática, de 1º curso de Ingeniería Telemática de la Universidad Politécnica de Cartagena.

→ Apuntes de clase de la asignatura de Ingeniería de Protocolos, de 3º curso de Ingeniería de Telemática de la Universidad Polotécnica de Cartagena.

→ Apuntes de clase de la asignatura de Fundamentos de Programación, de 2º curso de Ingeniería Telemática de la Universidad Politécnica de Cartagena.

→ Stallings, W.,"Comunicaciones y Redes de Computadores". Prentice-Hall Iberia, 2000 (6ª Ed.). ISBN: 84-205-2986-9.

→ Halsall, F., "Data Communications, Computer Networks and Open Systems". Ed. Addison-Wesley, 1996 (4ª Ed.).

Universidad Politécnica de Cartagena

Escuela Técnica Superior de Ingeniería de Telecomunicación

FUNDAMENTOS DE TELEMÁTICA

MANUAL DE AYUDA

PRÁCTICA 3A: TESTEO Y VERIFICACIÓN DE CABLEADO

Interfaz de usuario

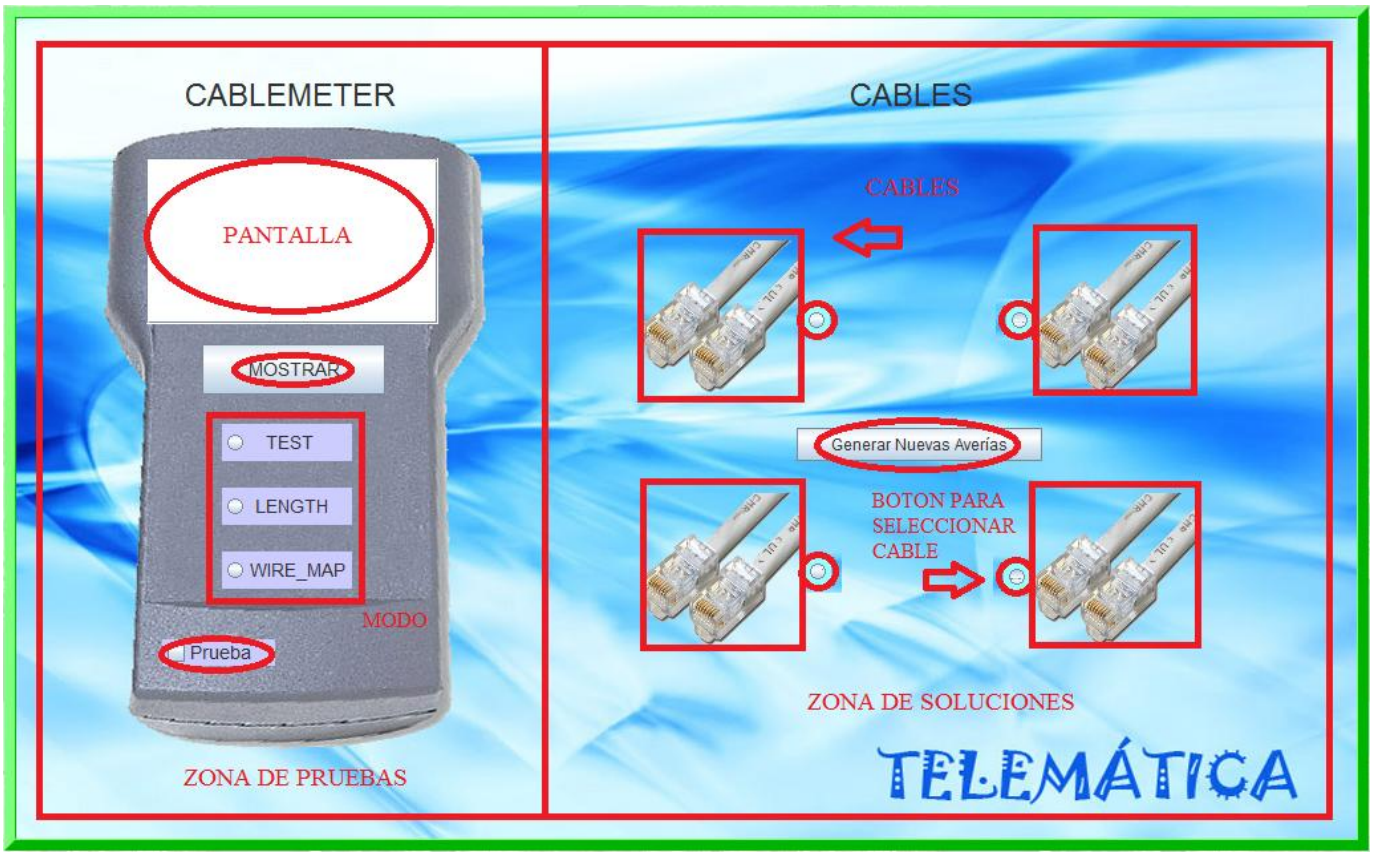


Figura-1

La interfaz de usuario del Applet es la mostrada en la Figura-1. Como se puede observar está dividida en dos zonas principales: zona de pruebas y zona de soluciones.

1. En la zona de pruebas es donde se encuentra el Cablemeter, es la zona de la cual tenemos que interpretar los resultados.

   Esta zona está constituida por: una pantalla, botón “MOSTRAR”, tres botones de MODO(“TEST”,”LENGTH” y “WIRE_MAP”) y el botón de “Prueba”.

   - Pantalla: Es donde se visualizan los resultados.
   - Mostrar: Es el botón que hay que pulsar para que se muestren los resultados en la pantalla.
   - Modo: En modo hay tres botones de los cuales solo se puede seleccionar uno en función de lo que se quiera analizar.
   - Prueba: El botón de prueba indica si estamos haciendo las mediciones con la prueba conectada o no. Si está pulsado, las mediciones sí se están haciendo con la prueba, si no lo está no.

2. La zona de soluciones es donde se comprueba si se han interpretado bien los mensajes del Cablemeter.

   Esta zona está constituida por: 4 cables y el botón " GenerarNuevasAverias".

   - Cables: Al hacer click encima de la imagen de un cable se muestran sus propiedades. Si seleccionamos uno de los botones que hay al lado de los cables, lo que hacemos es seleccionar el cable que queremos que compruebe el Cablemeter.

   - GenerarNuevasAverias: Este botón lo que hace es cambiar las averías de los cables, de forma que, cada vez que sea pulsado las averías cambiaran de forma aleatoria.

Uso del Applet

Se selecciona un cable, un modo y el botón de prueba en el caso de que se quieran hacer las mediciones con ella. Una vez hecho esto se procede a pulsar el botón "MOSTRAR" del Cablemeter y los resultados serán visualizados en la pantalla de este.

El usuario tiene que interpretar los resultados que salen en la pantalla del Cablemeter; y cuando tenga una respuesta, puede comprobar si es correcta haciendo click en la imagen del cable que había comprobado. Cuando se hayan comprobado todos los cables, se puede proceder a generar nuevas averías pulsando el botón "GenerarNuevasAverias".

NOTA: Las medidas en las que no es utilizada la Prueba no son fiables. Para realizar una buena medición es necesario que la Prueba esté conectada.

Universidad Politécnica de Cartagena

Escuela Técnica Superior de Ingeniería de Telecomunicación

FUNDAMENTOS DE TELEMÁTICA

MANUAL DE AYUDA

PRÁCTICA 3B: ESTUDIODEL NIVEL FÍSICO Y DEL NIVEL DE ENLACE.

Interfaz de usuario

Figura-1

La interfaz de usuario del Applet es la mostrada en la Figura-1. Consta de dos botones (botón “Enviar Ping”, botón “Mostrar Captura”) y de un osciloscopio.

- Enviar Ping: Con este botón lo que se hace es enviar el ping, el cual pasará por el osciloscopio y llegará al PC destino.
- Mostrar Captura: Este botón aparece en el PC destino cuando el ping se ha enviado, y lo que hace es mostrarlo como lo haría “wireshark”.
- Osciloscopio: Al hacer click en la imagen se abre una ventana en la cual podemos observar la forma de onda ideal del ping mostrada por un osciloscopio.

Uso del Applet

Lo primero que hay que hacer es pulsar el botón “Enviar Ping”, para que el ping se envíe y pueda ser analizado. Una vez hecho esto, se puede pasar a visualizar el ping capturado por el osciloscopio o por “wireshark”.

Se deben de interpretar los datos de la captura de “wireshark”, viendo los distintos campos del ping, cabecera, dirección IP origen-destino, tamaño, etc. De la misma forma se deben interpretar los pulsos mostrados por el osciloscopio, intentando diferenciar las cuatro partes más importantes (Preambulo,sdf,datos,crc).

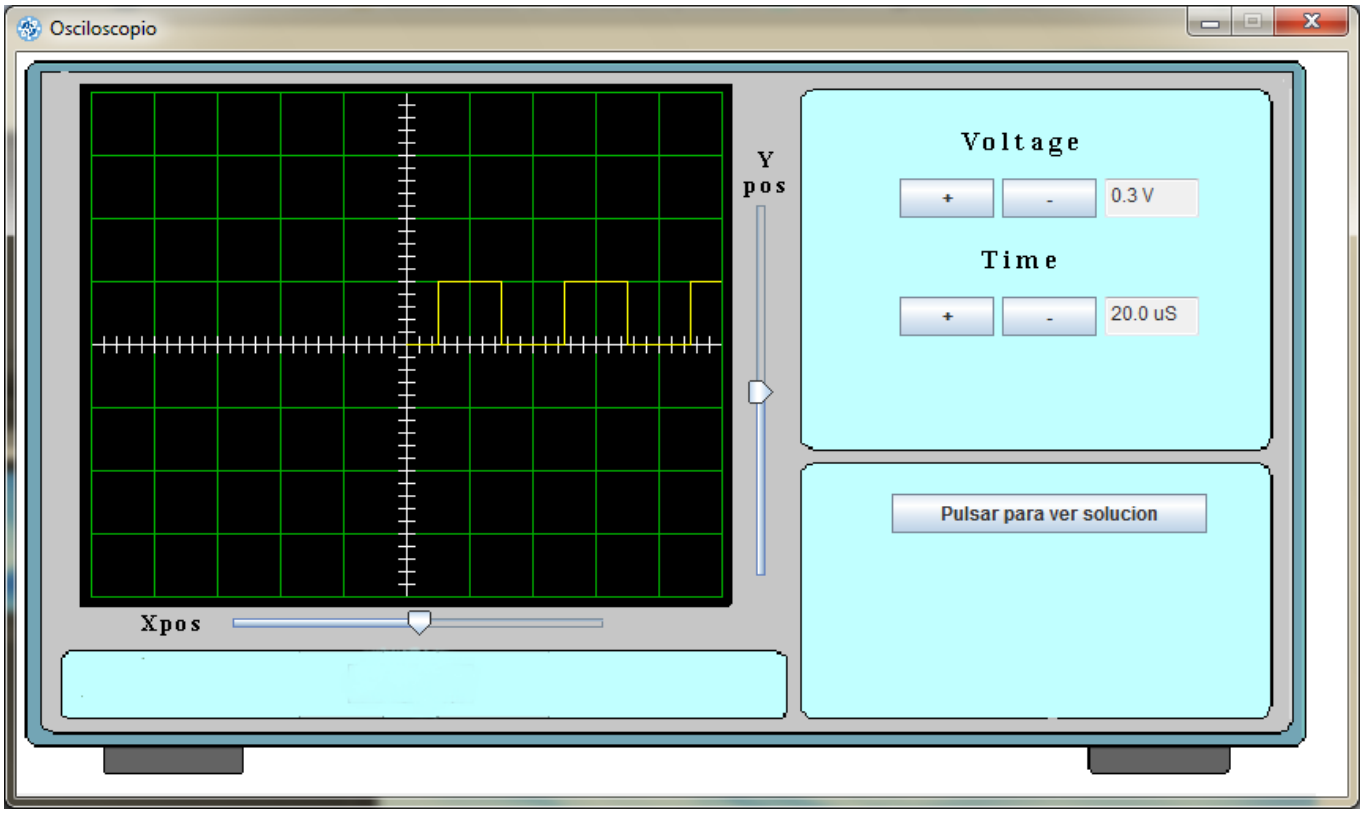
Universidad Politécnica de Cartagena

Escuela Técnica Superior de Ingeniería de Telecomunicación

FUNDAMENTOS DE TELEMÁTICA

MANUAL DE AYUDA OSCILOSCOPIO

Interfaz grafica del osciloscopio



- Botones “Voltage”: Cuando pulsamos estos botones lo que hacemos es cambiar los voltios por división (ejey) del osciloscopio, de forma que la señal varia su amplitud
- Botones “Time”: Cuando pulsamos estos botones lo que hacemos es cambiar el tiempo por división (ejex) del osciloscopio, de forma que la señal varia su anchura.
- Barra “Xpos”: Cuando desplazamos esta barra movemos la señal por el ejex.
- Barra “Ypos”: Cuando desplazamos esta barra movemos la señal por el ejey.
- Boton “Pulsar para ver solucion”: Cuando pulsamos este botón la señal cambia de color, diferenciándose así las cuatro partes más importante por las que está formado el ping: preámbulo,sdf,datos,crc.