ANEXO 2.
CÓDIGO FUENTE DE WASPMONITOR.

En este anexo están los archivos de código fuente necesarios para compilar el programa Waspmonitor haciendo uso de la herramienta Qt Creator.

A2.1. Projects.

Waspmonitor.pro

```
#-------------------------------------------------
#
# Project created by QtCreator 2012-02-25T20:18:13
#
#-------------------------------------------------

QT       += core gui

# Se incluyen las DLL de "Qwt"
CONFIG += qwt

TARGET = WaspMonitor
TEMPLATE = app


SOURCES += main.cpp\
        waspmonitor.cpp \
    sleeper.cpp \
    showinfo.cpp \
    showmonitor.cpp \
    logger.cpp \
    waspflags.cpp \
    waspfilenames.cpp \
    valuebar.cpp \
    graficobarrasnodo.cpp \
    graficobarrasradio.cpp \
    historico.cpp \
    panelalarmasnodo.cpp \
    panelalarmasradio.cpp \
    hilosniffer.cpp \
    configdialog.cpp

HEADERS  += waspmonitor.h \
    sleeper.h \
    showinfo.h \
    showmonitor.h \
    logger.h \
    waspflags.h \
    waspfilenames.h \
    valuebar.h \
    graficobarrasnodo.h \
    graficobarrasradio.h \
    historico.h \
    panelalarmasnodo.h \
    panelalarmasradio.h \
    hilosniffer.h \
    configdialog.h

FORMS    += waspmonitor.ui \
    showinfo.ui \
    showmonitor.ui \
    configdialog.ui

RESOURCES += \
    waspmonitor.qrc

# Path de las DLL de "Qextserialport"
LIBS += -LC:/qextserialport-1.2win-alpha/build
# Nombre de lor archivos DLL de "Qextserialport"
LIBS += -lqextserialport
LIBS += -lqextserialportd
# Path de los archivos ".h" de cabecera de "Qextserialport"
INCLUDEPATH += C:/qextserialport-1.2win-alpha
```

A2.2. Cabeceras.

Configdialog.h

```
#ifndef CONFIGDIALOG_H
#define CONFIGDIALOG_H

#include <QDialog>

namespace Ui {
class configDialog;
}

class configDialog : public QDialog
{
    Q_OBJECT

public:
    explicit configDialog(QWidget *parent = 0);
    ~configDialog();

public:
    Ui::configDialog *ui;
};

#endif // CONFIGDIALOG_H
```

Graficobarrasnodo.h

```
#ifndef GRAFICOBARRASNODO_H
#define GRAFICOBARRASNODO_H

#include <QWidget>
#include <QVBoxLayout>
#include "valuebar.h"

class GraficoBarrasNodo : public QWidget
{
    Q_OBJECT
public:
    // Constructor que inicializa el widget de graficas de barras
    // de los nodos
    GraficoBarrasNodo(QWidget *parent);

    // Barras medidoras
    ValueBar *barraTemp;
    ValueBar *barraBatt;
    ValueBar *barraDC;

    // Funciones para actualizar las barras medidoras
    void actualizaBarraTemp(double temp);
    void actualizaBarraBatt(double batt);
    void actualizaBarraDC(double dutyC);

signals:

public slots:

};

#endif // GRAFICOBARRASNODO_H
```

Graficobarrasradio.h

```
#ifndef GRAFICOBARRASRADIO_H
#define GRAFICOBARRASRADIO_H

#include <QWidget>
#include <QVBoxLayout>
#include "valuebar.h"

class GraficoBarrasRadio : public QWidget
{
    Q_OBJECT
public:
```

```
    // Constructor que inicializa el widget de graficas de barras
    // de los nodos
    GraficoBarrasRadio(QWidget *parent);

    // Barras medidoras
    ValueBar *barraRSSI;
    ValueBar *barraThroughPut;

    // Funciones para actualizar las barras medidoras
    void actualizaBarraRSSI(double rssi);
    void actualizaBarraThroughPut(double throughPut);

signals:

public slots:

};

#endif // GRAFICOBARRASRADIO_H
```

Hilosniffer.h

```
#ifndef HILOSNIFFER_H
#define HILOSNIFFER_H

#include <QThread>
#include "qextserialport.h"
#include <QDebug>

class HiloSniffer : public QThread
{
    Q_OBJECT
public:
    HiloSniffer(QObject *parent = 0);

    QextSerialPort *puertoSerie;
    QByteArray bytesReceived;
    QByteArray mensajeRecibido;
    QByteArray auxHexArray;
    QString nombrePuerto;

    // Flag que se pone a 1 si ha habido un error
    // al intentar abrir el puerto serie
    int errorAbrirPuerto;


    // Esto sería la función "main" del hilo
    void run();
    // Finaliza la ejecucion del hilo
    void stop();

    void abrirPuertoSerie();
    void cerrarPuertoSerie();

signals:
    // Esta señal se emitirá cuando se haya leído desde
    // el puerto serie un mensaje completo
    void mensajeListo();

public slots:
    void onReadyRead();

};

#endif // HILOSNIFFER_H
```

Historico.h

```
#ifndef HISTORICO_H
#define HISTORICO_H

#include <QWidget>
#include <QVBoxLayout>
#include <QString>
```

```
#include "qwt_plot.h"
#include "qwt_plot_curve.h"
#include "qwt_plot_grid.h"

#ifndef TAM
   #define TAM 120 // Nº de muestras que se mostrarán en la gráfica
#endif

class Historico : public QWidget
{
    Q_OBJECT
public:
    Historico(QWidget *parent, QString tipoGraf,
              double xSclMin, double xSclMax,
              double ySclMin, double ySclMax,
              QString xLbl, QString yLbl);

    // Objetos Qwt para el renderizado de la gráfica
    QwtPlot *grafica;
    QwtPlotCurve *curva;
    QwtPlotGrid *rejilla;

    // Datos para el trazado de la curva
    // Se representarán 120 muestras (si llegan muestras
    // cada 30 segundos, esto supondria un histórico de 60 minutos)

    double x[TAM];
    double y[TAM];

    // Configuración de las escalas
    double xScaleMin;
    double xScaleMax;
    double yScaleMin;
    double yScaleMax;

    // Etiquetas de los ejes
    QString xLabel; // "Muestras"
    QString yLabel; // "RSSI [dBm]"

    // Tipo de Grafica
    QString tipoGrafica;

    // Funciones
    void inicializarValores();
    void actualizarGrafica(double nuevoValor);

signals:

public slots:

};

#endif // HISTORICO_H
```

Logger.h

```
#ifndef LOGGER_H
#define LOGGER_H

#include <QWidget>
#include <QProcess>
#include <QFile>

#include <QPlainTextEdit>

#include "ui_showmonitor.h"

class Logger : public QWidget
{
    Q_OBJECT
public:
    explicit Logger(QWidget *parent = 0);
    QPlainTextEdit *plainTextEdit;

signals:

public slots:
```

```
private:

};

#endif // LOGGER_H
```

Panelalarmasnodo.h

```
#ifndef PANELALARMASNODO_H
#define PANELALARMASNODO_H

#include <QWidget>
#include <QFrame>
#include <QLabel>
#include <QPushButton>
#include <QGridLayout>
#include <QHBoxLayout>
#include <QPixmap>

class PanelAlarmasNodo : public QWidget
{
    Q_OBJECT
public:
    PanelAlarmasNodo(QWidget *parent = 0);

    // Marcos
    QFrame *frameNodoCaido;
    QFrame *frameBatBaja;
    QFrame *frameDCAlto;
    QFrame *frameTempAlta;

    // Botones
    QPushButton *buttonNodoCaido;
    QPushButton *buttonBatBaja;
    QPushButton *buttonDCAlto;
    QPushButton *buttonTempAlta;

    // Etiquetas de texto
    QLabel *labelNodoCaido;
    QLabel *labelBatBaja;
    QLabel *labelDCAlto;
    QLabel *labelTempAlta;

    // Etiquetas de iconos
    QLabel *labelIconNodoCaido;
    QLabel *labelIconBatBaja;
    QLabel *labelIconDCAlto;
    QLabel *labelIconTempAlta;

    // Iconos
    QPixmap iconoAlarmaON;
    QPixmap iconoAlarmaOFF;

signals:

public slots:

};

#endif // PANELALARMASNODO_H
```

Panelalarmasradio.h

```
#ifndef PANELALARMASRADIO_H
#define PANELALARMASRADIO_H

#include <QWidget>
#include <QFrame>
#include <QLabel>
#include <QPushButton>
#include <QGridLayout>
#include <QPixmap>

class PanelAlarmasRadio : public QWidget
{
    Q_OBJECT
```

```
public:
    PanelAlarmasRadio(QWidget *parent = 0);

    // Marco
    QFrame *frameEnlaceCaido;

    // Botones
    QPushButton *buttonEnlaceCaido;

    // Etiquetas de texto
    QLabel *labelEnlaceCaido;

    // Etiquetas de iconos
    QLabel *labelIconEnlaceCaido;

    // Iconos
    QPixmap iconoAlarmaON;
    QPixmap iconoAlarmaOFF;

signals:

public slots:

};

#endif // PANELALARMASRADIO_H
```

Showinfo.h

```
#ifndef SHOWINFO_H
#define SHOWINFO_H

#include <QWidget>

namespace Ui {
class showInfo;
}

class showInfo : public QWidget
{
    Q_OBJECT

public:
    explicit showInfo(QWidget *parent = 0);
    ~showInfo();

private:
    Ui::showInfo *ui;
};

#endif // SHOWINFO_H
```

Showmonitor.h

```
#ifndef SHOWMONITOR_H
#define SHOWMONITOR_H

#include <QWidget>
#include <QLineEdit>
#include <QMovie>
#include "logger.h"
#include "graficobarrasnodo.h"
#include "graficobarrasradio.h"
#include "historico.h"
#include "panelalarmasnodo.h"
#include "panelalarmasradio.h"

namespace Ui {
class showMonitor;
}

class showMonitor : public QWidget
{
    Q_OBJECT

public:
    explicit showMonitor(QWidget *parent = 0);
    ~showMonitor();
```

```
// Configuracion
void setupTreeActions();

Logger *logger;
void setMap(int res);
void setTempTags(int res);
void setMapIcons(int res);

// Funciones para colocar los widgets de monitorización
void setGrafBarNodos();
void setGrafBarRadio();
void setHistoricos();
void setPanelAlarmasNodos();
void setPanelAlarmasRadio();

// Icono de Gateway Capturando tramas
QLabel *tagSignal;
QMovie *iconSignal;

// Etiquetas de temperatura

QLineEdit *tag0000;
QLineEdit *tag0041;
QLineEdit *tag0042;
QLineEdit *tag0043;
QLineEdit *tag0044;

// Paletas de color de las etiquetas de temperatura
QPalette paleta0041;
QPalette paleta0042;
QPalette paleta0043;
QPalette paleta0000;
QPalette paleta0044;

// Etiquetas de leds de estado de los radioenlaces
QLabel *tagLed0041_0042;
QLabel *tagLed0042_0043;
QLabel *tagLed0043_0000;
QLabel *tagLed0044_0000;

QLabel *tagLed0041_0043;
QLabel *tagLed0041_0000;
QLabel *tagLed0042_0000;

// Pixmaps de leds de estado de los radioenlaces
QPixmap Led0041_0042;
QPixmap Led0042_0043;
QPixmap Led0043_0000;
QPixmap Led0044_0000;

QPixmap Led0041_0043;
QPixmap Led0041_0000;
QPixmap Led0042_0000;

// Etiquetas donde van insertados los iconos de alarma del mapa

QLabel *tagIconBatBaja0000;

// --- Nodos ---
QLabel *tagIconBatBaja0041;
QLabel *tagIconBatBaja0042;
QLabel *tagIconBatBaja0043;
QLabel *tagIconBatBaja0044;

QLabel *tagIconDCAlto0000;

QLabel *tagIconDCAlto0041;
QLabel *tagIconDCAlto0042;
QLabel *tagIconDCAlto0043;
QLabel *tagIconDCAlto0044;

QLabel *tagTempAlta0000;

QLabel *tagTempAlta0041;
QLabel *tagTempAlta0042;
QLabel *tagTempAlta0043;
QLabel *tagTempAlta0044;

// --- Radioenlaces ---
QLabel *tagRadioOff0041_0042;
QLabel *tagRadioOff0042_0043;
QLabel *tagRadioOff0043_0000;
QLabel *tagRadioOff0044_0000;
```

```
QLabel *tagRadioOff0041_0043;
QLabel *tagRadioOff0041_0000;
QLabel *tagRadioOff0042_0000;

// Pixmaps de los iconos de alarma del mapa

QPixmap iconBatBaja0000;

// --- Nodos ---
QPixmap iconBatBaja0041;
QPixmap iconBatBaja0042;
QPixmap iconBatBaja0043;
QPixmap iconBatBaja0044;

QPixmap iconDCAlto0000;

QPixmap iconDCAlto0041;
QPixmap iconDCAlto0042;
QPixmap iconDCAlto0043;
QPixmap iconDCAlto0044;

QPixmap iconTempAlta0000;

QPixmap iconTempAlta0041;
QPixmap iconTempAlta0042;
QPixmap iconTempAlta0043;
QPixmap iconTempAlta0044;

// --- Radioenlaces ---
QPixmap iconRadioOff0041_0042;
QPixmap iconRadioOff0042_0043;
QPixmap iconRadioOff0043_0000;
QPixmap iconRadioOff0044_0000;

QPixmap iconRadioOff0041_0043;
QPixmap iconRadioOff0041_0000;
QPixmap iconRadioOff0042_0000;

// Widgets de Gráfico de barras de nodos

GraficoBarrasNodo *grafBar0000;

GraficoBarrasNodo *grafBar0041;
GraficoBarrasNodo *grafBar0042;
GraficoBarrasNodo *grafBar0043;
GraficoBarrasNodo *grafBar0044;

// Widgets de Gráfico de barras de radioenlaces primarios
GraficoBarrasRadio *grafBar0041_0042;
GraficoBarrasRadio *grafBar0042_0043;
GraficoBarrasRadio *grafBar0043_0000;
GraficoBarrasRadio *grafBar0044_0000;

// Widgets de Gráfico de barras de radioenlaces secundarios
GraficoBarrasRadio *grafBar0041_0043;
GraficoBarrasRadio *grafBar0041_0000;
GraficoBarrasRadio *grafBar0042_0000;

// Widgets de Históricos de Temperatura de Nodos

Historico *hist0000;

Historico *hist0041;
Historico *hist0042;
Historico *hist0043;
Historico *hist0044;

// Widgets de Históricos de RSSI de radioenlaces primarios
Historico *hist0041_0042;
Historico *hist0042_0043;
Historico *hist0043_0000;
Historico *hist0044_0000;

// Widgets de Históricos de RSSI de radioenlaces secundarios
Historico *hist0041_0043;
Historico *hist0041_0000;
Historico *hist0042_0000;

// Widgets de Panel de Alarmas de nodos

PanelAlarmasNodo *panel0000;
PanelAlarmasNodo *panel0041;
PanelAlarmasNodo *panel0042;
PanelAlarmasNodo *panel0043;
```

```
    PanelAlarmasNodo *panel0044;

    // Widgets de Panel de Alarmas de radioenlaces
    PanelAlarmasRadio *panel0041_0042;
    PanelAlarmasRadio *panel0042_0043;
    PanelAlarmasRadio *panel0043_0000;
    PanelAlarmasRadio *panel0044_0000;

    PanelAlarmasRadio *panel0041_0043;
    PanelAlarmasRadio *panel0041_0000;
    PanelAlarmasRadio *panel0042_0000;



public slots:
    void mostrarGraficos(QTreeWidgetItem* item, int columna);

private:
    Ui::showMonitor *ui;
};

#endif // SHOWMONITOR_H
```

Sleeper.h

```
#ifndef SLEEPER_H
#define SLEEPER_H

#include <QThread>

class Sleeper : public QThread
{
public:
    void usleep(unsigned long usecs);
    void msleep(unsigned long msecs);
    void sleep(unsigned long secs);
};

#endif // SLEEPER_H
```

Valuebar.h

```
#ifndef VALUEBAR_H
#define VALUEBAR_H

#include <QWidget>
#include <QLabel>
#include <qwt_thermo.h>
#include <QVBoxLayout>

class ValueBar : public QWidget
{
    Q_OBJECT
public:
    ValueBar(const QString &text, QWidget *parent =0, double value = 0.0,
             double scaleMin = 0.0, double scaleMax = 0.0, const QBrush &pincel = Qt::green);
    void setValue(double value);

    QLabel *d_label;
    QwtThermo *d_thermo;

signals:

public slots:

private:


};

#endif // VALUEBAR_H
```

Waspfilenames.h

```
#ifndef WASPFILENAMES_H
#define WASPFILENAMES_H
```

```
#include <QObject>
#include <QString>

class WaspFileNames : public QObject
{
    Q_OBJECT
public:
    explicit WaspFileNames(QObject *parent = 0);

    // Nombres de los archivos de log
    QString logHTMLName;
    QString logCSVName;
    QString logName;

signals:

public slots:

};

#endif // WASPFILENAMES_H
```

Waspflags.h

```
#ifndef WASPFLAGS_H
#define WASPFLAGS_H

#include <QObject>

class WaspFlags: public QObject
{
    Q_OBJECT
public:
    explicit WaspFlags(QObject *parent = 0);

    // Flags de archivos
    int saveFileHTML; // Si está a 1, graba el log en HTML
    int saveFileCSV;  // Si está a 1, graba el log en CSV

    int firstTimeOpenHTML; // Está a 1 si se ha abierto por primera vez el HTML
                           // Evita que se ponga el título en el log más de una vez

    // Flags de habilitación/deshabilitación de la monitorización de nodos.
    // Se ponen a 1 si la monitorización del nodo está a ON.
    // Se usan para que Waspmonitor no salte alarmas si no recibe nada de los nodos
    // al sobrepasarse los contadores de tiempo.
    int nodo0041_ON;
    int nodo0042_ON;
    int nodo0043_ON;
    int nodo0044_ON;

    // Flags de contador
    int capturaEnCurso;   // Si está a 1 quiere decir que actualmente se están
                          // capturando datos
};

#endif // WASPFLAGS_H
```

Waspmonitor.h

```
#ifndef WASPMONITOR_H
#define WASPMONITOR_H

#include <QMainWindow>
#include <QProcess>
#include <QString>
#include <QTime>
#include "waspflags.h"
#include "waspfilenames.h"
#include "hilosniffer.h"
#include "configdialog.h"

namespace Ui {
class WaspMonitor;
}

class WaspMonitor : public QMainWindow
{
```

```
    Q_OBJECT

public:
    explicit WaspMonitor(QWidget *parent = 0);
    ~WaspMonitor();

    void closeEvent(QCloseEvent *evento);

protected:
    void setupActions();
    void setupConfig();
    void setupContador();

    void setupConfigDialogBox();
    void setupHiloSniffer();
    void actualizarLog();
    void escribirLogHTML(QString nombre, QString linea);
    void escribirLogCSV(QString nombre, QString linea);
    void cambiarFormatoFecha();
    void procesarMensaje();

    void actualizarContadoresNodos();
    void actualizarContadoresEnlaces();
    void actualizarGraficas();
    void actualizarEtiquetas(QString NAnodo, QString tempGrados);
    void actualizarBarras(QStringList campoMensaje);
    void actualizarHistoricos(QStringList campoMensaje);
    void actualizarIconosEstado(QStringList campoMensaje);
    void actualizarPanelesAlarma(QStringList campoMensaje);
    void actualizarIconosAlarma(QStringList campoMensaje);

private slots:

protected slots:
    void checkEstadoNodos();
    void checkEstadoEnlaces();
    void mostrarInfo();
    void mostrarConfig();
    void limpiarAlarmas();
    void mostrarMonitorizacion();
    void iniciarCaptura();
    void detenerCaptura();
    void salir();
    void updateOutput();

    // Slot para elegir el puerto COM
    void elegirPuerto(QString nombrePuerto);

    // Slots de habilitación/deshabilitación de nodos
    // El nodo 0000 (no se deshabilita nunca... ya que si este está
    // deshabilitado, no se recibe nada en el gateway
    void setNodeON_or_OFF_0041();
    void setNodeON_or_OFF_0042();
    void setNodeON_or_OFF_0043();
    void setNodeON_or_OFF_0044();

    // Slots de habilitación/deshabilitación de logs
    void setLogCSV_ON_or_OFF();
    void setLogHTML_ON_or_OFF();

    // Slots de limpieza de iconos de alarma
    void limpiarNC_0000();
    void limpiarBB_0000();
    void limpiarTA_0000();
    void limpiarDC_0000();

    void limpiarNC_0041();
    void limpiarBB_0041();
    void limpiarTA_0041();
    void limpiarDC_0041();

    void limpiarNC_0042();
    void limpiarBB_0042();
    void limpiarTA_0042();
    void limpiarDC_0042();

    void limpiarNC_0043();
    void limpiarBB_0043();
    void limpiarTA_0043();
    void limpiarDC_0043();

    void limpiarNC_0044();
    void limpiarBB_0044();
    void limpiarTA_0044();
```

```
    void limpiarDC_0044();

    void limpiarRC_0041_0042();
    void limpiarRC_0042_0043();
    void limpiarRC_0043_0000();
    void limpiarRC_0044_0000();

    void limpiarRC_0041_0043();
    void limpiarRC_0041_0000();
    void limpiarRC_0042_0000();


private:
    Ui::WaspMonitor *ui;

    // QDialog de configuración
    configDialog *configDialogBox;

    // Contador general: Cada 30 segundos, llama al slot que comprueba
    // si los nodos siguen vivos "checkEstadoNodos()".
    QTimer *contador;

    // Contadores para cada nodo: Cuando llega un KEEPALIVE, se reinician.
    // En el slot de "checkEstadoNodos()", si alguno de estos nodos está caído,
    // se ordena activar las alarmas correspondientes.
    QTime time_0041;
    QTime time_0042;
    QTime time_0043;
    QTime time_0044;

    // Contadores para cada radioenlace: Cuando llega un ESTADO_ENLACE, se reinician.
    // En el slot de "checkEstadoEnlaces()", se comprueba si siguen vivos y si no,
    // se ponen los leds en rojo.
    QTime time_0041_0042;
    QTime time_0042_0043;
    QTime time_0043_0000;
    QTime time_0044_0000;

    QTime time_0041_0043;
    QTime time_0041_0000;
    QTime time_0042_0000;

    HiloSniffer *hiloSniffer;
    WaspFlags *waspFlags;
    WaspFileNames *waspFileNames;

    // Mensaje en formato RAW (Cadena de caracteres en ASCII)
    QByteArray mensajeRAW;
    // Mensaje en formato QString
    QString mensajeStr;
    // Campos del mensaje agrupados en una lista de QStrings
    QStringList campoMensaje;

};

#endif // WASPMONITOR_H
```

A2.3. Código cpp.

Configdialog.cpp

```
#include "configdialog.h"
#include "ui_configdialog.h"

configDialog::configDialog(QWidget *parent) :
    QDialog(parent),
    ui(new Ui::configDialog)
{
    ui->setupUi(this);
}

configDialog::~configDialog()
{
    delete ui;
}
```

Graficobarrasnodo.cpp

```
#include "graficobarrasnodo.h"

/* ------------------------------------------------------------
 * El constructor crea las barras y las formatea metiendo los
 * titulos y las escalas.
 * ------------------------------------------------------------
 */

GraficoBarrasNodo::GraficoBarrasNodo(QWidget *parent = 0) :
    QWidget(parent)
{
    // Creamos tres widgets de tipo ValueBar (Barras de medida)
    barraTemp = new ValueBar("Temperatura [ºC]",this,-20.0,-20.0,80.0,Qt::cyan);
    barraBatt = new ValueBar("Batería restante [%]",this,0.0,0.0,100.0,Qt::red);
    barraDC = new ValueBar("Duty Cycle usado [%]",this,0.0,0.0,100.0,Qt::green);

    // Las añadimos a un layout vertical, para que se apilen una encima de otra
    QVBoxLayout *layoutGrafNodo = new QVBoxLayout( this );

    layoutGrafNodo->addWidget(barraTemp);
    layoutGrafNodo->addWidget(barraBatt);
    layoutGrafNodo->addWidget(barraDC);
    layoutGrafNodo->setSpacing(0); // Esto es para que no haya distancia entre las barras

}

/* ------------------------------------------------------
 * Función que sirve para actualizar el valor de la barra de
 * temperatura y darle el color adecuado segun la misma
 * ------------------------------------------------------
*/

void GraficoBarrasNodo::actualizaBarraTemp(double temp)
{
    if(temp <= 0.0)
    {
        barraTemp->d_thermo->setValue(temp);
        barraTemp->d_thermo->setFillBrush(Qt::cyan);

    }else if ((temp > 0.0) && (temp <= 45.0))
    {
        barraTemp->d_thermo->setValue(temp);
        barraTemp->d_thermo->setFillBrush(Qt::green);

    }else if ((temp > 45.0) && (temp <= 55.0))
    {
        barraTemp->d_thermo->setValue(temp);
        barraTemp->d_thermo->setFillBrush(Qt::yellow);
    }
    else
    {
        barraTemp->d_thermo->setValue(temp);
        barraTemp->d_thermo->setFillBrush(Qt::red);
    }

}

/* ------------------------------------------------------
 * Función que sirve para actualizar el valor de la barra de
 * nivel de bateria restante y darle el color adecuado segun la misma
 * ------------------------------------------------------
*/

void GraficoBarrasNodo::actualizaBarraBatt(double batt)
{
    if(batt <= 10.0)
    {
        barraBatt->d_thermo->setValue(batt);
        barraBatt->d_thermo->setFillBrush(Qt::red);

    }else if ((batt > 10.0) && (batt <= 20.0))
    {
        barraBatt->d_thermo->setValue(batt);
        barraBatt->d_thermo->setFillBrush(Qt::yellow);
    }
    else
    {
        barraBatt->d_thermo->setValue(batt);
        barraBatt->d_thermo->setFillBrush(Qt::green);
    }
```

```
}

/* ------------------------------------------------------
 * Función que sirve para actualizar el valor de la barra de
 * nivel de duty cycle usado y darle el color adecuado segun la misma
 * ------------------------------------------------------
*/

void GraficoBarrasNodo::actualizaBarraDC(double dutyC)
{
    if(dutyC < 80.0)
    {
        barraDC->d_thermo->setValue(dutyC);
        barraDC->d_thermo->setFillBrush(Qt::green);

    }else if ((dutyC >= 80.0) && (dutyC < 90.0))
    {
        barraDC->d_thermo->setValue(dutyC);
        barraDC->d_thermo->setFillBrush(Qt::yellow);
    }
    else
    {
        barraDC->d_thermo->setValue(dutyC);
        barraDC->d_thermo->setFillBrush(Qt::red);
    }
}
```

Graficobarrasradio.cpp

```
#include "graficobarrasradio.h"

/* ------------------------------------------------------------
 * El constructor crea las barras y las formatea metiendo los
 * titulos y las escalas.
 * ------------------------------------------------------------
 */

GraficoBarrasRadio::GraficoBarrasRadio(QWidget *parent = 0) :
    QWidget(parent)
{
    // Creamos tres widgets de tipo ValueBar (Barras de medida)
    barraRSSI = new ValueBar("RSSI [dBm]",this,-120.0,-120.0,0.0,Qt::red);
    barraThroughPut = new ValueBar("Throughput [%]",this,0.0,0.0,100.0,Qt::red);

    // Las añadimos a un layout vertical, para que se apilen una encima de otra
    QVBoxLayout *layoutGrafRadio = new QVBoxLayout( this );

    layoutGrafRadio->addWidget(barraRSSI);
    layoutGrafRadio->addWidget(barraThroughPut);
    layoutGrafRadio->setSpacing(0); // Esto es para que no haya distancia entre las barras
}

/* ------------------------------------------------------
 * Función que sirve para actualizar el valor de la barra de
 * RSSI y darle el color adecuado segun la misma
 * ------------------------------------------------------
*/

void GraficoBarrasRadio::actualizaBarraRSSI(double rssi)
{
    if(rssi <= -112.0)
    {
        barraRSSI->d_thermo->setValue(rssi);
        barraRSSI->d_thermo->setFillBrush(Qt::red);
    }
    else if ((rssi > -112.0) && (rssi <= -80.0))
    {
        barraRSSI->d_thermo->setValue(rssi);
        barraRSSI->d_thermo->setFillBrush(Qt::yellow);
    }
    else
    {
        barraRSSI->d_thermo->setValue(rssi);
        barraRSSI->d_thermo->setFillBrush(Qt::green);
    }
}

/* ------------------------------------------------------
 * Función que sirve para actualizar el valor de la barra de
 * Throughput y darle el color adecuado segun la misma
 * ------------------------------------------------------
*/
```

```
void GraficoBarrasRadio::actualizaBarraThroughPut(double throughPut)
{
    if(throughPut <= 20.0)
    {
        barraThroughPut->d_thermo->setValue(throughPut);
        barraThroughPut->d_thermo->setFillBrush(Qt::red);
    }
    else if ((throughPut > 20.0) && (throughPut < 50.0))
    {
        barraThroughPut->d_thermo->setValue(throughPut);
        barraThroughPut->d_thermo->setFillBrush(Qt::yellow);
    }
    else
    {
        barraThroughPut->d_thermo->setValue(throughPut);
        barraThroughPut->d_thermo->setFillBrush(Qt::green);
    }
}
```

Hilosniffer.cpp

```
#include "hilosniffer.h"

/* ------------------------------------------------------------------
 *  Constructor: Lo que hace es configurar los parámetros de
 *  la conexión con el puerto serie y el número de puerto
 * ------------------------------------------------------------------
 */

HiloSniffer::HiloSniffer(QObject *parent) :
    QThread(parent)
{
    // Creamos un objeto "Puerto Serie"
    puertoSerie = new QextSerialPort();

    errorAbrirPuerto = 0;
}

/* ------------------------------------------------------------------
 *  Función "main()" del hilo
 *  Se ejecutará cuando se haga un hiloSniffer->start();
 * ------------------------------------------------------------------
 */

void HiloSniffer::run()
{
    // Elegimos el número de puerto
    puertoSerie->setPortName(nombrePuerto);
    // Configuramos el modo de funcionamiento "Event Driven"
    puertoSerie->setQueryMode(QextSerialPort::EventDriven);

    // Abrimos el puerto serie
    abrirPuertoSerie();

    // Iniciamos el "event loop"
    exec();

}

/* ------------------------------------------------------------------
 *  Función que finaliza la ejecución del hilo
 *  ------------------------------------------------------------------
 */

void HiloSniffer::stop()
{
    // Cerramos el puerto serie
    cerrarPuertoSerie();
    // Detenemos el "event loop"
    exit();
}

/* ------------------------------------------------------------------
 *  Abre el puerto serie.
 * ------------------------------------------------------------------
 */

void HiloSniffer::abrirPuertoSerie()
{
    // Abrimos el puerto en modo "Sólo Lectura"
```

```
    if (puertoSerie->open(QextSerialPort::ReadOnly))
    {
        // Configuramos la conexión
        puertoSerie->setBaudRate(BAUD38400);
        puertoSerie->setDataBits(DATA_8);
        puertoSerie->setStopBits(STOP_1);
        puertoSerie->setParity(PAR_NONE);
        puertoSerie->setFlowControl(FLOW_HARDWARE);

        connect(puertoSerie, SIGNAL(readyRead()), this, SLOT(onReadyRead()));

        errorAbrirPuerto = 0;
    }
    else
    {
        errorAbrirPuerto = 1;
    }
}

/* ------------------------------------------------------------------
 *  Cierra el puerto serie
 * ------------------------------------------------------------------
 */

void HiloSniffer::cerrarPuertoSerie()
{
    puertoSerie->close();
}

/* ------------------------------------------------------------------
 * SLOT: Cuando se recibe la señal "readyRead" desde el puerto serie,
 *       se ejecuta esta función
 * ------------------------------------------------------------------
 */

void HiloSniffer::onReadyRead()
{
    QByteArray bytes;
    int a = puertoSerie->bytesAvailable();
    bytes.resize(a);
    puertoSerie->read(bytes.data(), bytes.size());

    /** El mensaje no lo lee del tiron Waspmonitor, sino que lo lee
        en varios trozos. Esto lo he podido comprobar haciendo uso
        de Free Serial Port Monitor. Por tanto, el procedimiento a
        seguir será ir añadiendo con "append" los trozos hasta
        que se reciba el BYTE en HEX de fin de trama API */
    bytesReceived.append(bytes);

    /** Se usa un QByteArray auxiliar en el que se guardan los caracteres
        recibidos en formato hexadecimal*/
    auxHexArray = bytesReceived.toHex();

    /** 0x23232323 es #### en hexadecimal al mirar en
        el programa Free Serial Port Monitor. */
    if(auxHexArray.contains("23232323"))
    {
        /** Eliminamos la cabecera de la API. El número 44 lo he
            hallado con Free Serial Port Monitor.
            IMPORTANTE: Si no se elimina la cabecera API, WaspMonitor
            no puede extraer el mensaje. */
        auxHexArray.remove(0,44);

        /** Esto sirve para quitar el byte '0x97' no imprimible final de
            la cabecera API que nos va a dar problemas si no lo eliminamos */
        int tam = auxHexArray.size();
        auxHexArray.remove(tam-2, 2);

        /** Guardamos en el QByteArray 'mensajeRecibido'
        el mensaje entero sin la cabecera API que nos molestaba.*/
        mensajeRecibido = QByteArray::fromHex(auxHexArray);

        auxHexArray.clear();

        // Limpiamos el QByteArray que usamos como "buffer"
        bytesReceived.clear();

        // Cuando el mensaje esté recibido por completo...
        // emitir SIGNAL que recogerá el SLOT "Waspmonitor::updateOutput()"
        emit mensajeListo();
    }
}
```

Historico.cpp

```
#include "historico.h"

#include <QDebug>

/* ---------------------------------------------------------------------------
*   Constructor de la gráfica de Historico.
*   Inicializa los objetos Qwt (Gráfica, Curva, Rejilla, Estilo, Etiquetas...)
*   Inicializa los valores de la gráfica según el tipo.
* ---------------------------------------------------------------------------*/

Historico::Historico(QWidget *parent, QString tipoGraf,
                     double xSclMin, double xSclMax, double ySclMin,
                     double ySclMax, QString xLbl,
                     QString yLbl):
                     QWidget(parent)
{
    /* Inicializamos variables */
    tipoGrafica = tipoGraf;
    xScaleMin = xSclMin;
    xScaleMax = xSclMax;
    yScaleMin = ySclMin;
    yScaleMax = ySclMax;
    xLabel = xLbl;
    yLabel = yLbl;

    // Creamos un layout para introducir los objetos Qwt
    QVBoxLayout *layoutHist = new QVBoxLayout(this);

    /*---  Creamos los objetos Qwt ---*/

    // Título de la gráfica
    QwtText titulo("Histórico");
    // Objeto QwtPlot
    grafica = new QwtPlot(titulo, this);
    // Objeto QwtPlotCurve
    curva = new QwtPlotCurve();

    /*---  Añadimos la gráfica al Layout ---*/

    layoutHist->addWidget(grafica);

    /*---  Damos formato la gráfica ---*/

    // Cambiamos el color de fondo de la gráfica
    QBrush fondoBrush(Qt::white);
    grafica->setCanvasBackground(fondoBrush);

    // Configuramos el eje de ordenadas
    grafica->setAxisTitle(QwtPlot::xBottom, xLabel);
    grafica->setAxisScale(QwtPlot::xBottom, xScaleMin, xScaleMax);

    // Configuramos el eje de abcisas
    grafica->setAxisTitle(QwtPlot::yLeft, yLabel);
    grafica->setAxisScale(QwtPlot::yLeft, yScaleMin, yScaleMax);

    // Inicializamos los valores (tipoGraf == Temp o tipoGraf == RSSI)
    inicializarValores();

    // Añadimos las muestras (arrays de datos x e y)
    curva->setSamples(x,y,TAM);

    // Adjuntamos el objeto de tipo "QwtPlotCurve" al objeto de tipo "QwtPlot"
    curva->attach(grafica);

    // Cambiamos el color y el tipo de trazo de la curva (Temp. Rojo, Radioenl. Azul)
    if(tipoGrafica == "Temp")
        curva->setPen(QPen(QColor(Qt::red)));
    else
        curva->setPen(QPen(QColor(Qt::blue)));

    curva->setStyle(QwtPlotCurve::Lines);

    // Metemos antialiasing a la curva para que no se vea pixelizada
    curva->setRenderHint(QwtPlotCurve::RenderAntialiased);

    // Tipo de trazado de la rejilla (grid)
    QPen penRejilla(Qt::DotLine);
    // Configuramos el grid
    rejilla = new QwtPlotGrid();
    rejilla->setPen(penRejilla);
```

```
    rejilla->enableX(false);

    // Adjuntamos el objeto de tipo "QwtPlotGrid" al objeto de tipo "QwtPlot"
    rejilla->attach(grafica);

    // Damos la orden de redibujar la gráfica
    // (En este caso dibujar por primera vez la gráfica)
    grafica->replot();

}

/* ------------------------------------------------------------------------
*   Inicializa los valores
* ------------------------------------------------------------------------*/

void Historico::inicializarValores()
{
    if (tipoGrafica == "Temp")
    {
        for (int i=0; i<TAM; i++)
        {
            x[i] = double(i);
            y[i] = 0.0;
        }
    }
    else if (tipoGrafica == "RSSI")
    {
        for (int i=0; i<TAM; i++)
        {
            x[i] = double(i);
            y[i] = -112.0;
        }
    }
}

/* ------------------------------------------------------------------------
*   Actualizar Gráfica
* ------------------------------------------------------------------------*/

void Historico::actualizarGrafica(double nuevoValor)
{
    int i;

    // Actualizamos los valores
    for (i=0; i<(TAM-1); i++)
        {
            y[i] = y[i+1];
        }

    y[i] = nuevoValor;

    // Asignamos los nuevos valores a la curva
    curva->setSamples(x,y,TAM);
    // Redibujamos la gráfica
    grafica->replot();

}
```

Logger.cpp

```
#include "logger.h"

Logger::Logger(QWidget *parent) :
    QWidget(parent)
{
    // plainTextEdit es un widget hijo del widget logger
    plainTextEdit = new QPlainTextEdit(this);

    plainTextEdit->setSizePolicy(QSizePolicy::Preferred,QSizePolicy::Preferred);
    plainTextEdit->setReadOnly(true);
    plainTextEdit->setMaximumBlockCount(200); // Número maximo de lineas que se muestran en la ventana = 200
                                              // Las demás, se eliminan para liberar memoria.
    QVBoxLayout *vboxLog2 = new QVBoxLayout;

    vboxLog2->setContentsMargins(0,0,0,0);
    vboxLog2->addWidget(plainTextEdit);
    this->setLayout(vboxLog2);
    this->setSizePolicy(QSizePolicy::Preferred,QSizePolicy::Preferred);
}
```

Main.cpp

```
#include <QtGui/QApplication>
#include "waspmonitor.h"
#include <QSplashScreen>

#include "sleeper.h"


#include <QDesktopWidget>
//#include <QDebug>

int main(int argc, char *argv[])
{
    QApplication a(argc, argv);

    // Obtenemos la resolución de la pantalla
    int screenWidth = QApplication::desktop()->width();
    int screenHeight = QApplication::desktop()->height();

    // Configura el pantalla de inicio "splash screen"
    QSplashScreen *splash = new QSplashScreen;
    splash->setPixmap(QPixmap(":/Splash.png"));
    splash->show();

    Sleeper *dormir = new Sleeper;
    dormir->msleep(1000);

    Qt::Alignment topRight = Qt::AlignRight | Qt::AlignTop; //*
    splash->showMessage(QObject::tr("Cargando..."), topRight, Qt::white); //*

    dormir->msleep(2500);
    delete dormir;

    //Inicia la ventana principal
    WaspMonitor w;

    // Automáticamente se cambia el tamaño de la ventana principal
    // según la resolución de la pantalla
    w.resize(screenWidth-100, screenHeight-150);
    w.show();

    splash->finish(&w);
    delete splash;

    return a.exec();
}
```

Panelalarmasnodo.cpp

```
#include "panelalarmasnodo.h"

/* -----------------------------------------------------------------------------
*   Constructor: Crea el widget "Panel de Alarma de Nodos"
* -----------------------------------------------------------------------------*/

PanelAlarmasNodo::PanelAlarmasNodo(QWidget *parent) :
    QWidget(parent)
{
    // Iconos de alarma "ON" y "OFF"
    iconoAlarmaON.load(":/alarma_on_32.png");
    iconoAlarmaOFF.load(":/alarma_off_32.png");

    // Tipo de letra de las etiquetas
    QFont fuenteEtiq("Tahoma", 10, QFont::Bold);

    /* -- Inicializamos el QFrame de alarmas de "Nodo Caído" -- */
    frameNodoCaido = new QFrame(this);
    frameNodoCaido->setFrameShape(QFrame::WinPanel);
    frameNodoCaido->setFrameShadow(QFrame::Raised);
    frameNodoCaido->setLineWidth(2);
    frameNodoCaido->setSizePolicy(QSizePolicy::Fixed,QSizePolicy::Fixed);

    // Insertamos los widgets dentro del marco
    buttonNodoCaido = new QPushButton("Limpiar",this);
    //buttonNodoCaido->setSizePolicy(QSizePolicy::Fixed,QSizePolicy::Fixed);
    labelNodoCaido = new QLabel("  Nodo caído  ",this);
    labelNodoCaido->setFont(fuenteEtiq);
    labelNodoCaido->setAlignment(Qt::AlignHCenter | Qt::AlignVCenter);
    labelIconNodoCaido = new QLabel(this->frameNodoCaido);
    labelIconNodoCaido->setPixmap(iconoAlarmaOFF);
```

```
//labelIconNodoCaido->setAlignment(Qt::AlignHCenter);

// Ordenamos los componentes del marco
QHBoxLayout *layoutNodoCaido = new QHBoxLayout(this);
layoutNodoCaido->addWidget(buttonNodoCaido);
layoutNodoCaido->addWidget(labelNodoCaido);
layoutNodoCaido->addWidget(labelIconNodoCaido);
frameNodoCaido->setLayout(layoutNodoCaido);

/* -- Inicializamos el QFrame de alarmas de "Batería Baja" -- */
frameBatBaja = new QFrame(this);
frameBatBaja->setFrameShape(QFrame::WinPanel);
frameBatBaja->setFrameShadow(QFrame::Raised);
frameBatBaja->setLineWidth(2);
frameBatBaja->setSizePolicy(QSizePolicy::Fixed,QSizePolicy::Fixed);

// Insertamos los widgets dentro del marco
buttonBatBaja = new QPushButton("Limpiar",this);
labelBatBaja = new QLabel("  Batería Baja  ",this);
labelBatBaja->setFont(fuenteEtiq);
labelBatBaja->setAlignment(Qt::AlignHCenter | Qt::AlignVCenter);
labelIconBatBaja = new QLabel(this);
labelIconBatBaja->setPixmap(iconoAlarmaOFF);

// Ordenamos los componentes del marco
QHBoxLayout *layoutBatBaja = new QHBoxLayout(this);
layoutBatBaja->addWidget(buttonBatBaja);
layoutBatBaja->addWidget(labelBatBaja);
layoutBatBaja->addWidget(labelIconBatBaja);
frameBatBaja->setLayout(layoutBatBaja);

/* -- Inicializamos el QFrame de alarmas de "Duty Cycle" -- */
frameDCAlto = new QFrame(this);
frameDCAlto->setFrameShape(QFrame::WinPanel);
frameDCAlto->setFrameShadow(QFrame::Raised);
frameDCAlto->setLineWidth(2);
frameDCAlto->setSizePolicy(QSizePolicy::Fixed,QSizePolicy::Fixed);

// Insertamos los widgets dentro del marco
buttonDCAlto = new QPushButton("Limpiar",this);
labelDCAlto = new QLabel("  Duty Cycle  ",this);
labelDCAlto->setFont(fuenteEtiq);
labelDCAlto->setAlignment(Qt::AlignHCenter | Qt::AlignVCenter);
labelIconDCAlto = new QLabel(this);
labelIconDCAlto->setPixmap(iconoAlarmaOFF);

// Ordenamos los componentes del marco
QHBoxLayout *layoutDCAlto = new QHBoxLayout(this);
layoutDCAlto->addWidget(buttonDCAlto);
layoutDCAlto->addWidget(labelDCAlto);
layoutDCAlto->addWidget(labelIconDCAlto);
frameDCAlto->setLayout(layoutDCAlto);

/* -- Inicializamos el QFrame de alarmas de "Temp. Alta" -- */
frameTempAlta = new QFrame(this);
frameTempAlta->setFrameShape(QFrame::WinPanel);
frameTempAlta->setFrameShadow(QFrame::Raised);
frameTempAlta->setLineWidth(2);
frameTempAlta->setSizePolicy(QSizePolicy::Fixed,QSizePolicy::Fixed);

// Insertamos los widgets dentro del marco
buttonTempAlta = new QPushButton("Limpiar",this);
labelTempAlta = new QLabel("  Temp. Alta  ",this);
labelTempAlta->setFont(fuenteEtiq);
labelTempAlta->setAlignment(Qt::AlignHCenter | Qt::AlignVCenter);
labelIconTempAlta = new QLabel(this);
labelIconTempAlta->setPixmap(iconoAlarmaOFF);

// Ordenamos los componentes del marco
QHBoxLayout *layoutTempAlta = new QHBoxLayout(this);
layoutTempAlta->addWidget(buttonTempAlta);
layoutTempAlta->addWidget(labelTempAlta);
layoutTempAlta->addWidget(labelIconTempAlta);
frameTempAlta->setLayout(layoutTempAlta);

// Ordenamos todos los marcos con un Grid Layout

QGridLayout *layoutGridBase = new QGridLayout(this);

layoutGridBase->addWidget(frameNodoCaido,0,0);
layoutGridBase->addWidget(frameBatBaja,0,1);
layoutGridBase->addWidget(frameDCAlto,1,0);
layoutGridBase->addWidget(frameTempAlta,1,1);
```

```
    this->setLayout(layoutGridBase);
}
```

Panelalarmasradio.cpp

```
#include "panelalarmasradio.h"

/* ------------------------------------------------------------------------
*   Constructor: Crea el widget "Panel de Alarma de Radioenlaces"
* ------------------------------------------------------------------------*/

PanelAlarmasRadio::PanelAlarmasRadio(QWidget *parent) :
    QWidget(parent)
{
    // Iconos de alarma "ON" y "OFF"
    iconoAlarmaON.load(":/alarma_on_32.png");
    iconoAlarmaOFF.load(":/alarma_off_32.png");

    // Tipo de letra de las etiquetas
    QFont fuenteEtiq("Tahoma", 10, QFont::Bold);

    /* -- Inicializamos el QFrame de alarmas de "Radioenlace Caído" -- */
    frameEnlaceCaido = new QFrame(this);
    frameEnlaceCaido->setFrameShape(QFrame::WinPanel);
    frameEnlaceCaido->setFrameShadow(QFrame::Raised);
    frameEnlaceCaido->setLineWidth(2);
    frameEnlaceCaido->setSizePolicy(QSizePolicy::Fixed,QSizePolicy::Fixed);

    // Insertamos los widgets dentro del marco
    buttonEnlaceCaido = new QPushButton("Limpiar",this);
    //buttonEnlaceCaido->setSizePolicy(QSizePolicy::Fixed,QSizePolicy::Fixed);
    labelEnlaceCaido = new QLabel("  Radioenlace caído  ",this);
    labelEnlaceCaido->setFont(fuenteEtiq);
    labelEnlaceCaido->setAlignment(Qt::AlignHCenter | Qt::AlignVCenter);
    labelIconEnlaceCaido = new QLabel(this->frameEnlaceCaido);
    labelIconEnlaceCaido->setPixmap(iconoAlarmaOFF);

    // Ordenamos los componentes del marco
    QHBoxLayout *layoutEnlaceCaido = new QHBoxLayout(this);
    layoutEnlaceCaido->addWidget(buttonEnlaceCaido);
    layoutEnlaceCaido->addWidget(labelEnlaceCaido);
    layoutEnlaceCaido->addWidget(labelIconEnlaceCaido);
    frameEnlaceCaido->setLayout(layoutEnlaceCaido);

    // Ordenamos el marco con un Grid Layout

    QGridLayout *layoutGridBase = new QGridLayout(this);

    layoutGridBase->addWidget(frameEnlaceCaido,0,0);

    this->setLayout(layoutGridBase);
}
```

Showinfo.cpp

```
#include "showinfo.h"
#include "ui_showinfo.h"

showInfo::showInfo(QWidget *parent) :
    QWidget(parent),
    ui(new Ui::showInfo)
{
    ui->setupUi(this);
}

showInfo::~showInfo()
{
    delete ui;
}
```

Showmonitor.cpp

```
#include "showmonitor.h"
#include "ui_showmonitor.h"
#include <QBoxLayout>
#include "logger.h"
#include <QDesktopWidget>
#include <QLabel>
```

```
#include <QPixmap>

/* ------------------------------------------------------------------------
*   Constructor de ShowMonitor.
*   Genera el panel de Monitorización y configura todos los widgets.
* ------------------------------------------------------------------------*/

showMonitor::showMonitor(QWidget *parent) :
    QWidget(parent),
    ui(new Ui::showMonitor)
{
    ui->setupUi(this);

    // Configuramos los treeWidgets para que cuando se pulsen se cambien las gráficas
    setupTreeActions();

    // Configuramos la ventana de log
    logger = new Logger(ui->groupBox_2);

    QVBoxLayout *vboxLog = new QVBoxLayout;
    vboxLog->addWidget(logger);
    ui->groupBox_2->setLayout(vboxLog);
    logger->show();

    // Añadimos el mapa de la red y el canal
    setMap(QApplication::desktop()->height());

    // Añadimos las etiquetas de temperatura
    setTempTags(QApplication::desktop()->height());

    // Añadimos los iconos del mapa
    setMapIcons(QApplication::desktop()->height());

    // Añadimos los widgets de Grafico de barras
    setGrafBarNodos();
    setGrafBarRadio();

    // Añadimos los widgets de Históricos
    setHistoricos();

    // Añadimos los widgets de Panel de Alarmas
    setPanelAlarmasNodos();
    setPanelAlarmasRadio();
}

/* ------------------------------------------------------------------------
*   Destructor de showMonitor. Destruye el widget y libera la memoria.
* ------------------------------------------------------------------------*/

showMonitor::~showMonitor()
{
    delete ui;
}

/* ------------------------------------------------------------------------
*   Añade el mapa de un tamaño u otro segun la resolución de la pantalla.
* ------------------------------------------------------------------------*/

void showMonitor::setMap(int res)
{
     QPixmap pixmapMapa;

     if (res > 800)
     {
         pixmapMapa.load(":/mapa1920x1080.bmp");
     }
     else
     {
         pixmapMapa.load(":/mapa1280x800.bmp");
     }

     QLabel *labelMapa = new QLabel;
     labelMapa->setPixmap(pixmapMapa);
     QVBoxLayout *vboxMap = new QVBoxLayout;
     vboxMap->addWidget(labelMapa);
     ui->groupBox->setLayout(vboxMap);
     labelMapa->show();

}

/* ------------------------------------------------------------------------
*   Añade las etiquetas de temperatura en la posición correspondiente
*   del mapa según la resolución de pantalla.
* ------------------------------------------------------------------------*/
```

```
void showMonitor::setTempTags(int res)
{
    // Resolución Alta (Para monitor grande)
    if (res > 800)
    {
        tag0041 = new QLineEdit(ui->groupBox);
        tag0041->move(453,111);
        tag0041->resize(48,21);
        tag0041->setAlignment(Qt::AlignHCenter | Qt::AlignVCenter);
        tag0041->setReadOnly(true);

        tag0042 = new QLineEdit(ui->groupBox);
        tag0042->move(564,274);
        tag0042->resize(48,21);
        tag0042->setAlignment(Qt::AlignHCenter | Qt::AlignVCenter);
        tag0042->setReadOnly(true);

        tag0043 = new QLineEdit(ui->groupBox);
        tag0043->move(579,353);
        tag0043->resize(48,21);
        tag0043->setAlignment(Qt::AlignHCenter | Qt::AlignVCenter);
        tag0043->setReadOnly(true);

        tag0000 = new QLineEdit(ui->groupBox);
        tag0000->move(510,500);
        tag0000->resize(48,21);
        tag0000->setAlignment(Qt::AlignHCenter | Qt::AlignVCenter);
        tag0000->setReadOnly(true);

        tag0044 = new QLineEdit(ui->groupBox);
        tag0044->move(365,554);
        tag0044->resize(48,21);
        tag0044->setAlignment(Qt::AlignHCenter | Qt::AlignVCenter);
        tag0044->setReadOnly(true);

    }
    // Resolución Baja (Para portátil)
    else
    {
        tag0041 = new QLineEdit(ui->groupBox);
        tag0041->move(274,69);
        tag0041->resize(48,21);
        tag0041->setAlignment(Qt::AlignHCenter | Qt::AlignVCenter);
        tag0041->setReadOnly(true);

        tag0042 = new QLineEdit(ui->groupBox);
        tag0042->move(345,170);
        tag0042->resize(48,21);
        tag0042->setAlignment(Qt::AlignHCenter | Qt::AlignVCenter);
        tag0042->setReadOnly(true);

        tag0043 = new QLineEdit(ui->groupBox);
        tag0043->move(353,260);
        tag0043->resize(48,21);
        tag0043->setAlignment(Qt::AlignHCenter | Qt::AlignVCenter);
        tag0043->setReadOnly(true);

        tag0000 = new QLineEdit(ui->groupBox);
        tag0000->move(313,312);
        tag0000->resize(48,21);
        tag0000->setAlignment(Qt::AlignHCenter | Qt::AlignVCenter);
        tag0000->setReadOnly(true);

        tag0044 = new QLineEdit(ui->groupBox);
        tag0044->move(219,344);
        tag0044->resize(48,21);
        tag0044->setAlignment(Qt::AlignHCenter | Qt::AlignVCenter);
        tag0044->setReadOnly(true);
    }

    // Paletas de color de las etiquetas de temperatura (en principio, fondo blanco)
    paleta0041.setColor(QPalette::Base, Qt::white);
    tag0041->setPalette(paleta0041);

    paleta0042.setColor(QPalette::Base, Qt::white);
    tag0042->setPalette(paleta0042);

    paleta0043.setColor(QPalette::Base, Qt::white);
    tag0043->setPalette(paleta0043);

    paleta0000.setColor(QPalette::Base, Qt::white);
    tag0000->setPalette(paleta0000);
```

```
    paleta0044.setColor(QPalette::Base, Qt::white);
    tag0044->setPalette(paleta0044);
}

/* ------------------------------------------------------------------------
*   Añade los iconos de estado y alarma en el mapa
* ------------------------------------------------------------------------*/

void showMonitor::setMapIcons(int res)
{
    // Icono de Gateway capturando tramas
    iconSignal = new QMovie(":/signal_peque.gif");
    tagSignal = new QLabel(ui->groupBox);
    tagSignal->setFrameStyle(QFrame::Panel | QFrame::Raised);
    tagSignal->setLineWidth(2);
    tagSignal->resize(45,45);
    tagSignal->setMovie(iconSignal);
    iconSignal->jumpToFrame(0);
    iconSignal->stop();

    if (res > 800) // Resolución alta
    {
        // Situamos el icono de captura de tramas en la posición adecuada
        tagSignal->move(800,60);

        // ------------- Iconos de Estado (Radioenlaces) ---------

        // Radioenlace 0041_0042
        Led0041_0042.load(":/red_light_16.png");
        tagLed0041_0042 = new QLabel(ui->groupBox);
        tagLed0041_0042->move(472,206);
        tagLed0041_0042->setPixmap(Led0041_0042);

        // Radioenlace 0042_0043
        Led0042_0043.load(":/red_light_16.png");
        tagLed0042_0043 = new QLabel(ui->groupBox);
        tagLed0042_0043->move(531,323);
        tagLed0042_0043->setPixmap(Led0042_0043);

        // Radioenlace 0043_0000
        Led0043_0000.load(":/red_light_16.png");
        tagLed0043_0000 = new QLabel(ui->groupBox);
        tagLed0043_0000->move(504,413);
        tagLed0043_0000->setPixmap(Led0043_0000);

        // Radioenlace 0044_0000
        Led0044_0000.load(":/red_light_16.png");
        tagLed0044_0000 = new QLabel(ui->groupBox);
        tagLed0044_0000->move(451,522);
        tagLed0044_0000->setPixmap(Led0044_0000);

        // Radioenlace 0041_0043
        Led0041_0043.load(":/red_light_16.png");
        tagLed0041_0043 = new QLabel(ui->groupBox);
        tagLed0041_0043->move(468,252);
        tagLed0041_0043->setPixmap(Led0041_0043);

        // Radioenlace 0041_0000
        Led0041_0000.load(":/red_light_16.png");
        tagLed0041_0000 = new QLabel(ui->groupBox);
        tagLed0041_0000->move(433,310);
        tagLed0041_0000->setPixmap(Led0041_0000);

        // Radioenlace 0042_0000
        Led0042_0000.load(":/red_light_16.png");
        tagLed0042_0000 = new QLabel(ui->groupBox);
        tagLed0042_0000->move(488,371);
        tagLed0042_0000->setPixmap(Led0042_0000);


        // ------------- Iconos de Alarma (Nodos) ----------------

        // Nodo 0041
        iconBatBaja0041.load(":/battery_24.png");
        tagIconBatBaja0041 = new QLabel(ui->groupBox);
        tagIconBatBaja0041->move(540,134);
        tagIconBatBaja0041->setPixmap(iconBatBaja0041);
        tagIconBatBaja0041->hide();

        iconTempAlta0041.load(":/termo_24.png");
        tagTempAlta0041 = new QLabel(ui->groupBox);
        tagTempAlta0041->move(540,104);
        tagTempAlta0041->setPixmap(iconTempAlta0041);
        tagTempAlta0041->hide();
```

```
iconDCAlto0041.load(":/dc_24.png");
tagIconDCAlto0041 = new QLabel(ui->groupBox);
tagIconDCAlto0041->move(510,104);
tagIconDCAlto0041->setPixmap(iconDCAlto0041);
tagIconDCAlto0041->hide();

// Nodo 0042
iconBatBaja0042.load(":/battery_24.png");
tagIconBatBaja0042 = new QLabel(ui->groupBox);
tagIconBatBaja0042->move(654,294);
tagIconBatBaja0042->setPixmap(iconBatBaja0042);
tagIconBatBaja0042->hide();

iconTempAlta0042.load(":/termo_24.png");
tagTempAlta0042 = new QLabel(ui->groupBox);
tagTempAlta0042->move(654,264);
tagTempAlta0042->setPixmap(iconTempAlta0042);
tagTempAlta0042->hide();

iconDCAlto0042.load(":/dc_24.png");
tagIconDCAlto0042 = new QLabel(ui->groupBox);
tagIconDCAlto0042->move(624,264);
tagIconDCAlto0042->setPixmap(iconDCAlto0042);
tagIconDCAlto0042->hide();

// Nodo 0043
iconBatBaja0043.load(":/battery_24.png");
tagIconBatBaja0043 = new QLabel(ui->groupBox);
tagIconBatBaja0043->move(672,375);
tagIconBatBaja0043->setPixmap(iconBatBaja0043);
tagIconBatBaja0043->hide();

iconTempAlta0043.load(":/termo_24.png");
tagTempAlta0043 = new QLabel(ui->groupBox);
tagTempAlta0043->move(672,345);
tagTempAlta0043->setPixmap(iconTempAlta0043);
tagTempAlta0043->hide();

iconDCAlto0043.load(":/dc_24.png");
tagIconDCAlto0043 = new QLabel(ui->groupBox);
tagIconDCAlto0043->move(642,345);
tagIconDCAlto0043->setPixmap(iconDCAlto0043);
tagIconDCAlto0043->hide();

// Nodo 0000
iconBatBaja0000.load(":/battery_24.png");
tagIconBatBaja0000 = new QLabel(ui->groupBox);
tagIconBatBaja0000->move(595,530);
tagIconBatBaja0000->setPixmap(iconBatBaja0000);
tagIconBatBaja0000->hide();

iconTempAlta0000.load(":/termo_24.png");
tagTempAlta0000 = new QLabel(ui->groupBox);
tagTempAlta0000->move(595,500);
tagTempAlta0000->setPixmap(iconTempAlta0000);
tagTempAlta0000->hide();

iconDCAlto0000.load(":/dc_24.png");
tagIconDCAlto0000 = new QLabel(ui->groupBox);
tagIconDCAlto0000->move(565,500);
tagIconDCAlto0000->setPixmap(iconDCAlto0000);
tagIconDCAlto0000->hide();

// Nodo 0044
iconBatBaja0044.load(":/battery_24.png");
tagIconBatBaja0044 = new QLabel(ui->groupBox);
tagIconBatBaja0044->move(293,580);
tagIconBatBaja0044->setPixmap(iconBatBaja0044);
tagIconBatBaja0044->hide();

iconTempAlta0044.load(":/termo_24.png");
tagTempAlta0044 = new QLabel(ui->groupBox);
tagTempAlta0044->move(293,550);
tagTempAlta0044->setPixmap(iconTempAlta0044);
tagTempAlta0044->hide();

iconDCAlto0044.load(":/dc_24.png");
tagIconDCAlto0044 = new QLabel(ui->groupBox);
tagIconDCAlto0044->move(323,550);
tagIconDCAlto0044->setPixmap(iconDCAlto0044);
tagIconDCAlto0044->hide();

// ------------- Iconos de Alarma (Radioenlaces) ----------------
```

```
    // ------- Primarios --------

    // Radioenlace 0041-0042
    iconRadioOff0041_0042.load(":/warning_24.png");
    tagRadioOff0041_0042 = new QLabel(ui->groupBox);
    tagRadioOff0041_0042->move(471,204);
    tagRadioOff0041_0042->setPixmap(iconRadioOff0041_0042);
    tagRadioOff0041_0042->hide();

    // Radioenlace 0042-0043
    iconRadioOff0042_0043.load(":/warning_24.png");
    tagRadioOff0042_0043 = new QLabel(ui->groupBox);
    tagRadioOff0042_0043->move(528,320);
    tagRadioOff0042_0043->setPixmap(iconRadioOff0042_0043);
    tagRadioOff0042_0043->hide();

    // Radioenlace 0043-0000
    iconRadioOff0043_0000.load(":/warning_24.png");
    tagRadioOff0043_0000 = new QLabel(ui->groupBox);
    tagRadioOff0043_0000->move(499,413);
    tagRadioOff0043_0000->setPixmap(iconRadioOff0043_0000);
    tagRadioOff0043_0000->hide();

    // Radioenlace 0044-0000
    iconRadioOff0044_0000.load(":/warning_24.png");
    tagRadioOff0044_0000 = new QLabel(ui->groupBox);
    tagRadioOff0044_0000->move(446,520);
    tagRadioOff0044_0000->setPixmap(iconRadioOff0044_0000);
    tagRadioOff0044_0000->hide();

    // ------- Secundarios --------

    // Radioenlace 0041-0043
    iconRadioOff0041_0043.load(":/warning_24.png");
    tagRadioOff0041_0043 = new QLabel(ui->groupBox);
    tagRadioOff0041_0043->move(466,252);
    tagRadioOff0041_0043->setPixmap(iconRadioOff0041_0043);
    tagRadioOff0041_0043->hide();

    // Radioenlace 0041-0000
    iconRadioOff0041_0000.load(":/warning_24.png");
    tagRadioOff0041_0000 = new QLabel(ui->groupBox);
    tagRadioOff0041_0000->move(430,310);
    tagRadioOff0041_0000->setPixmap(iconRadioOff0041_0000);
    tagRadioOff0041_0000->hide();

    // Radioenlace 0042-0000
    iconRadioOff0042_0000.load(":/warning_24.png");
    tagRadioOff0042_0000 = new QLabel(ui->groupBox);
    tagRadioOff0042_0000->move(486,369);
    tagRadioOff0042_0000->setPixmap(iconRadioOff0042_0000);
    tagRadioOff0042_0000->hide();
}
// Resolución baja
else
{
    // Situamos el icono de captura de tramas en la posición adecuada
    tagSignal->move(490,40);

    // ------------- Iconos de Estado (Radioenlaces) ---------

    // Radioenlace 0041_0042
    Led0041_0042.load(":/red_light_16.png");
    tagLed0041_0042 = new QLabel(ui->groupBox);
    tagLed0041_0042->move(294,133);
    tagLed0041_0042->setPixmap(Led0041_0042);

    // Radioenlace 0042_0043
    Led0042_0043.load(":/red_light_16.png");
    tagLed0042_0043 = new QLabel(ui->groupBox);
    tagLed0042_0043->move(335,203);
    tagLed0042_0043->setPixmap(Led0042_0043);

    // Radioenlace 0043_0000
    Led0043_0000.load(":/red_light_16.png");
    tagLed0043_0000 = new QLabel(ui->groupBox);
    tagLed0043_0000->move(315,260);
    tagLed0043_0000->setPixmap(Led0043_0000);

    // Radioenlace 0044_0000
    Led0044_0000.load(":/red_light_16.png");
    tagLed0044_0000 = new QLabel(ui->groupBox);
    tagLed0044_0000->move(279,329);
```

```
tagLed0044_0000->setPixmap(Led0044_0000);

// Radioenlace 0041_0043
Led0041_0043.load(":/red_light_16.png");
tagLed0041_0043 = new QLabel(ui->groupBox);
tagLed0041_0043->move(293,165);
tagLed0041_0043->setPixmap(Led0041_0043);

// Radioenlace 0041_0000
Led0041_0000.load(":/red_light_16.png");
tagLed0041_0000 = new QLabel(ui->groupBox);
tagLed0041_0000->move(273,211);
tagLed0041_0000->setPixmap(Led0041_0000);

// Radioenlace 0042_0000
Led0042_0000.load(":/red_light_16.png");
tagLed0042_0000 = new QLabel(ui->groupBox);
tagLed0042_0000->move(304,227);
tagLed0042_0000->setPixmap(Led0042_0000);

// ------------- Iconos de Alarma (Nodos) ----------------

// Nodo 0041
iconBatBaja0041.load(":/battery_24.png");
tagIconBatBaja0041 = new QLabel(ui->groupBox);
tagIconBatBaja0041->move(357,84);
tagIconBatBaja0041->setPixmap(iconBatBaja0041);
tagIconBatBaja0041->hide();

iconTempAlta0041.load(":/termo_24.png");
tagTempAlta0041 = new QLabel(ui->groupBox);
tagTempAlta0041->move(357,54);
tagTempAlta0041->setPixmap(iconTempAlta0041);
tagTempAlta0041->hide();

iconDCAlto0041.load(":/dc_24.png");
tagIconDCAlto0041 = new QLabel(ui->groupBox);
tagIconDCAlto0041->move(327,54);
tagIconDCAlto0041->setPixmap(iconDCAlto0041);
tagIconDCAlto0041->hide();

// Nodo 0042
iconBatBaja0042.load(":/battery_24.png");
tagIconBatBaja0042 = new QLabel(ui->groupBox);
tagIconBatBaja0042->move(428,185);
tagIconBatBaja0042->setPixmap(iconBatBaja0042);
tagIconBatBaja0042->hide();

iconTempAlta0042.load(":/termo_24.png");
tagTempAlta0042 = new QLabel(ui->groupBox);
tagTempAlta0042->move(428,155);
tagTempAlta0042->setPixmap(iconTempAlta0042);
tagTempAlta0042->hide();

iconDCAlto0042.load(":/dc_24.png");
tagIconDCAlto0042 = new QLabel(ui->groupBox);
tagIconDCAlto0042->move(398,155);
tagIconDCAlto0042->setPixmap(iconDCAlto0042);
tagIconDCAlto0042->hide();

// Nodo 0043
iconBatBaja0043.load(":/battery_24.png");
tagIconBatBaja0043 = new QLabel(ui->groupBox);
tagIconBatBaja0043->move(436,256);
tagIconBatBaja0043->setPixmap(iconBatBaja0043);
tagIconBatBaja0043->hide();

iconTempAlta0043.load(":/termo_24.png");
tagTempAlta0043 = new QLabel(ui->groupBox);
tagTempAlta0043->move(436,226);
tagTempAlta0043->setPixmap(iconTempAlta0043);
tagTempAlta0043->hide();

iconDCAlto0043.load(":/dc_24.png");
tagIconDCAlto0043 = new QLabel(ui->groupBox);
tagIconDCAlto0043->move(406,226);
tagIconDCAlto0043->setPixmap(iconDCAlto0043);
tagIconDCAlto0043->hide();

// Nodo 0000
iconBatBaja0000.load(":/battery_24.png");
tagIconBatBaja0000 = new QLabel(ui->groupBox);
tagIconBatBaja0000->move(396,327);
tagIconBatBaja0000->setPixmap(iconBatBaja0000);
```

```
tagIconBatBaja0000->hide();

iconTempAlta0000.load(":/termo_24.png");
tagTempAlta0000 = new QLabel(ui->groupBox);
tagTempAlta0000->move(396,297);
tagTempAlta0000->setPixmap(iconTempAlta0000);
tagTempAlta0000->hide();

iconDCAlto0000.load(":/dc_24.png");
tagIconDCAlto0000 = new QLabel(ui->groupBox);
tagIconDCAlto0000->move(366,297);
tagIconDCAlto0000->setPixmap(iconDCAlto0000);
tagIconDCAlto0000->hide();

// Nodo 0044
iconBatBaja0044.load(":/battery_24.png");
tagIconBatBaja0044 = new QLabel(ui->groupBox);
tagIconBatBaja0044->move(159,359);
tagIconBatBaja0044->setPixmap(iconBatBaja0044);
tagIconBatBaja0044->hide();

iconTempAlta0044.load(":/termo_24.png");
tagTempAlta0044 = new QLabel(ui->groupBox);
tagTempAlta0044->move(159,329);
tagTempAlta0044->setPixmap(iconTempAlta0044);
tagTempAlta0044->hide();

iconDCAlto0044.load(":/dc_24.png");
tagIconDCAlto0044 = new QLabel(ui->groupBox);
tagIconDCAlto0044->move(189,329);
tagIconDCAlto0044->setPixmap(iconDCAlto0044);
tagIconDCAlto0044->hide();

// ------------- Iconos de Alarma (Radioenlaces) ----------------

// ------- Primarios --------

// Radioenlace 0041-0042
iconRadioOff0041_0042.load(":/warning_24.png");
tagRadioOff0041_0042 = new QLabel(ui->groupBox);
tagRadioOff0041_0042->move(290,133);
tagRadioOff0041_0042->setPixmap(iconRadioOff0041_0042);
tagRadioOff0041_0042->hide();

// Radioenlace 0042-0043
iconRadioOff0042_0043.load(":/warning_24.png");
tagRadioOff0042_0043 = new QLabel(ui->groupBox);
tagRadioOff0042_0043->move(331,203);
tagRadioOff0042_0043->setPixmap(iconRadioOff0042_0043);
tagRadioOff0042_0043->hide();

// Radioenlace 0043-0000
iconRadioOff0043_0000.load(":/warning_24.png");
tagRadioOff0043_0000 = new QLabel(ui->groupBox);
tagRadioOff0043_0000->move(311,260);
tagRadioOff0043_0000->setPixmap(iconRadioOff0043_0000);
tagRadioOff0043_0000->hide();

// Radioenlace 0044-0000
iconRadioOff0044_0000.load(":/warning_24.png");
tagRadioOff0044_0000 = new QLabel(ui->groupBox);
tagRadioOff0044_0000->move(275,329);
tagRadioOff0044_0000->setPixmap(iconRadioOff0044_0000);
tagRadioOff0044_0000->hide();

// ------- Secundarios --------

// Radioenlace 0041-0043
iconRadioOff0041_0043.load(":/warning_24.png");
tagRadioOff0041_0043 = new QLabel(ui->groupBox);
tagRadioOff0041_0043->move(289,165);
tagRadioOff0041_0043->setPixmap(iconRadioOff0041_0043);
tagRadioOff0041_0043->hide();

// Radioenlace 0041-0000
iconRadioOff0041_0000.load(":/warning_24.png");
tagRadioOff0041_0000 = new QLabel(ui->groupBox);
tagRadioOff0041_0000->move(269,211);
tagRadioOff0041_0000->setPixmap(iconRadioOff0041_0000);
tagRadioOff0041_0000->hide();

// Radioenlace 0042-0000
iconRadioOff0042_0000.load(":/warning_24.png");
tagRadioOff0042_0000 = new QLabel(ui->groupBox);
```

```
        tagRadioOff0042_0000->move(300,227);
        tagRadioOff0042_0000->setPixmap(iconRadioOff0042_0000);
        tagRadioOff0042_0000->hide();
    }
}

/* ------------------------------------------------------------------------
*   Configura los widgets de Grafica de Barras de los Nodos
* ------------------------------------------------------------------------*/

void showMonitor::setGrafBarNodos()
{
    // Nodo Ctra. Utrera
    grafBar0041 = new GraficoBarrasNodo(ui->page_18);

    QVBoxLayout *vboxBarras0041 = new QVBoxLayout(this);
    vboxBarras0041->addWidget(grafBar0041);
    ui->page_18->setLayout(vboxBarras0041);

    // Nodo C. Maribáñez
    grafBar0042 = new GraficoBarrasNodo(ui->page_19);

    QVBoxLayout *vboxBarras0042 = new QVBoxLayout(this);
    vboxBarras0042->addWidget(grafBar0042);
    ui->page_19->setLayout(vboxBarras0042);

    // Nodo San Fernando
    grafBar0043 = new GraficoBarrasNodo(ui->page_20);

    QVBoxLayout *vboxBarras0043 = new QVBoxLayout(this);
    vboxBarras0043->addWidget(grafBar0043);
    ui->page_20->setLayout(vboxBarras0043);

    // Nodo Oficina
    grafBar0000 = new GraficoBarrasNodo(ui->page_21);

    QVBoxLayout *vboxBarras0000 = new QVBoxLayout(this);
    vboxBarras0000->addWidget(grafBar0000);
    ui->page_21->setLayout(vboxBarras0000);

    // Nodo Alcantarillas
    grafBar0044 = new GraficoBarrasNodo(ui->page_22);

    QVBoxLayout *vboxBarras0044 = new QVBoxLayout(this);
    vboxBarras0044->addWidget(grafBar0044);
    ui->page_22->setLayout(vboxBarras0044);

}

/* ------------------------------------------------------------------------
*   Configura los widgets de Grafica de Barras de los Nodos
* ------------------------------------------------------------------------*/

void showMonitor::setGrafBarRadio()
{
    // Radioenlaces PRIMARIOS

    // Radioenlace Ctra. Utrera - C. Maribáñez
    grafBar0041_0042 = new GraficoBarrasRadio(ui->page_23);

    QVBoxLayout *vboxBarras0041_0042 = new QVBoxLayout(this);
    vboxBarras0041_0042->addWidget(grafBar0041_0042);
    ui->page_23->setLayout(vboxBarras0041_0042);

    // Radioenlace C. Maribáñez - San Fernando
    grafBar0042_0043 = new GraficoBarrasRadio(ui->page_24);

    QVBoxLayout *vboxBarras0042_0043 = new QVBoxLayout(this);
    vboxBarras0042_0043->addWidget(grafBar0042_0043);
    ui->page_24->setLayout(vboxBarras0042_0043);

    // Radioenlace San Fernando - Oficina
    grafBar0043_0000 = new GraficoBarrasRadio(ui->page_25);

    QVBoxLayout *vboxBarras0043_0000 = new QVBoxLayout(this);
    vboxBarras0043_0000->addWidget(grafBar0043_0000);
    ui->page_25->setLayout(vboxBarras0043_0000);

    // Radioenlace Alcantarillas - Oficina
    grafBar0044_0000 = new GraficoBarrasRadio(ui->page_26);

    QVBoxLayout *vboxBarras0044_0000 = new QVBoxLayout(this);
    vboxBarras0044_0000->addWidget(grafBar0044_0000);
    ui->page_26->setLayout(vboxBarras0044_0000);
```

```
    // Radioenlaces SECUNDARIOS

    // Radioenlace Ctra. Utrera - San Fernando
    grafBar0041_0043 = new GraficoBarrasRadio(ui->page_27);

    QVBoxLayout *vboxBarras0041_0043 = new QVBoxLayout(this);
    vboxBarras0041_0043->addWidget(grafBar0041_0043);
    ui->page_27->setLayout(vboxBarras0041_0043);

    // Radioenlace Ctra. Utrera - Oficina
    grafBar0041_0000 = new GraficoBarrasRadio(ui->page_28);

    QVBoxLayout *vboxBarras0041_0000 = new QVBoxLayout(this);
    vboxBarras0041_0000->addWidget(grafBar0041_0000);
    ui->page_28->setLayout(vboxBarras0041_0000);

    // Radioenlace C. Maribáñez - Oficina
    grafBar0042_0000 = new GraficoBarrasRadio(ui->page_29);

    QVBoxLayout *vboxBarras0042_0000 = new QVBoxLayout(this);
    vboxBarras0042_0000->addWidget(grafBar0042_0000);
    ui->page_29->setLayout(vboxBarras0042_0000);
}

/* ------------------------------------------------------------------------
*  Configura los widgets de Históricos de temperatura y RSSI
* ------------------------------------------------------------------------*/

void showMonitor::setHistoricos()
{
    /* -- Históricos de los nodos -- */

    // Nodo Ctra. Utrera
    hist0041 = new Historico(ui->page_1, "Temp", 0.0, 120.0,
                             -20.0, 80.0, "Muestras", "Temperatura [ºC]");
    QVBoxLayout *vboxHist0041 = new QVBoxLayout(this);
    vboxHist0041->addWidget(hist0041);
    ui->page_1->setLayout(vboxHist0041);

    // Nodo C. Maribáñez
    hist0042 = new Historico(ui->page_2, "Temp", 0.0, 120.0,
                             -20.0, 80.0, "Muestras", "Temperatura [ºC]");
    QVBoxLayout *vboxHist0042 = new QVBoxLayout(this);
    vboxHist0042->addWidget(hist0042);
    ui->page_2->setLayout(vboxHist0042);

    // Nodo San Fernando
    hist0043 = new Historico(ui->page_3, "Temp", 0.0, 120.0,
                             -20.0, 80.0, "Muestras", "Temperatura [ºC]");
    QVBoxLayout *vboxHist0043 = new QVBoxLayout(this);
    vboxHist0043->addWidget(hist0043);
    ui->page_3->setLayout(vboxHist0043);

    // Nodo Oficina
    hist0000 = new Historico(ui->page_4, "Temp", 0.0, 120.0,
                             -20.0, 80.0, "Muestras", "Temperatura [ºC]");
    QVBoxLayout *vboxHist0000 = new QVBoxLayout(this);
    vboxHist0000->addWidget(hist0000);
    ui->page_4->setLayout(vboxHist0000);

    // Nodo Alcantarillas
    hist0044 = new Historico(ui->page_5, "Temp", 0.0, 120.0,
                             -20.0, 80.0, "Muestras", "Temperatura [ºC]");
    QVBoxLayout *vboxHist0044 = new QVBoxLayout(this);
    vboxHist0044->addWidget(hist0044);
    ui->page_5->setLayout(vboxHist0044);

    /* -- Históricos de los radioenlaces primarios -- */

    // Radioenlace Ctra. Utrera - C. Maribáñez
    hist0041_0042 = new Historico(ui->page_6, "RSSI", 0.0, 120.0,
                             -120.0, 0.0, "Muestras", "RSSI [dBm]");
    QVBoxLayout *vboxHist0041_0042 = new QVBoxLayout(this);
    vboxHist0041_0042->addWidget(hist0041_0042);
    ui->page_6->setLayout(vboxHist0041_0042);

    // Radioenlace C. Maribáñez - San Fernando
    hist0042_0043 = new Historico(ui->page_7, "RSSI", 0.0, 120.0,
                             -120.0, 0.0, "Muestras", "RSSI [dBm]");
    QVBoxLayout *vboxHist0042_0043 = new QVBoxLayout(this);
    vboxHist0042_0043->addWidget(hist0042_0043);
    ui->page_7->setLayout(vboxHist0042_0043);
```

```
    // Radioenlace San Fernando - Oficina
    hist0043_0000 = new Historico(ui->page_8, "RSSI", 0.0, 120.0,
                            -120.0, 0.0, "Muestras", "RSSI [dBm]");
    QVBoxLayout *vboxHist0043_0000 = new QVBoxLayout(this);
    vboxHist0043_0000->addWidget(hist0043_0000);
    ui->page_8->setLayout(vboxHist0043_0000);

    // Radioenlace Alcantarillas - Oficina
    hist0044_0000 = new Historico(ui->page_9, "RSSI", 0.0, 120.0,
                            -120.0, 0.0, "Muestras", "RSSI [dBm]");
    QVBoxLayout *vboxHist0044_0000 = new QVBoxLayout(this);
    vboxHist0044_0000->addWidget(hist0044_0000);
    ui->page_9->setLayout(vboxHist0044_0000);

    /* -- Históricos de los radioenlaces secundarios -- */

    // Radioenlace Ctra. Utrera - C. Maribáñez
    hist0041_0043 = new Historico(ui->page_10, "RSSI", 0.0, 120.0,
                            -120.0, 0.0, "Muestras", "RSSI [dBm]");
    QVBoxLayout *vboxHist0041_0043 = new QVBoxLayout(this);
    vboxHist0041_0043->addWidget(hist0041_0043);
    ui->page_10->setLayout(vboxHist0041_0043);

    // Radioenlace C. Maribáñez - San Fernando
    hist0041_0000 = new Historico(ui->page_11, "RSSI", 0.0, 120.0,
                            -120.0, 0.0, "Muestras", "RSSI [dBm]");
    QVBoxLayout *vboxHist0041_0000 = new QVBoxLayout(this);
    vboxHist0041_0000->addWidget(hist0041_0000);
    ui->page_11->setLayout(vboxHist0041_0000);

    // Radioenlace San Fernando - Oficina
    hist0042_0000 = new Historico(ui->page_12, "RSSI", 0.0, 120.0,
                            -120.0, 0.0, "Muestras", "RSSI [dBm]");
    QVBoxLayout *vboxHist0042_0000 = new QVBoxLayout(this);
    vboxHist0042_0000->addWidget(hist0042_0000);
    ui->page_12->setLayout(vboxHist0042_0000);

}

/* ------------------------------------------------------------------------
*   Configura los widgets de Panel de Alarmas de los Nodos
* ------------------------------------------------------------------------*/

void showMonitor::setPanelAlarmasNodos()
{
    /* -- Paneles de alarmas de los nodos -- */

    // Nodo Ctra. Utrera
    panel0041 = new PanelAlarmasNodo(ui->page_30);

    QVBoxLayout *gridpanel0041 = new QVBoxLayout(this);
    gridpanel0041->addWidget(panel0041);
    ui->page_30->setLayout(gridpanel0041);

    // Nodo C. Maribáñez
    panel0042 = new PanelAlarmasNodo(ui->page_31);

    QVBoxLayout *gridpanel0042 = new QVBoxLayout(this);
    gridpanel0042->addWidget(panel0042);
    ui->page_31->setLayout(gridpanel0042);

    // Nodo San Fernando
    panel0043 = new PanelAlarmasNodo(ui->page_32);

    QVBoxLayout *gridpanel0043 = new QVBoxLayout(this);
    gridpanel0043->addWidget(panel0043);
    ui->page_32->setLayout(gridpanel0043);

    // Nodo Oficina
    panel0000 = new PanelAlarmasNodo(ui->page_33);

    QVBoxLayout *gridpanel0000 = new QVBoxLayout(this);
    gridpanel0000->addWidget(panel0000);
    ui->page_33->setLayout(gridpanel0000);

    // Nodo Alcantarillas
    panel0044 = new PanelAlarmasNodo(ui->page_34);

    QVBoxLayout *gridpanel0044 = new QVBoxLayout(this);
    gridpanel0044->addWidget(panel0044);
    ui->page_34->setLayout(gridpanel0044);
}

/* ------------------------------------------------------------------------
```

```
*   Configura los widgets de Panel de Alarmas de los Radioenlaces
* ------------------------------------------------------------------------------*/

void showMonitor::setPanelAlarmasRadio()
{
    /* -- Paneles de alarmas de los radioenlaces -- */

    // Radioenlaces primarios

    // Radioenlace Ctra. Utrera - C. Maribáñez
    panel0041_0042 = new PanelAlarmasRadio(ui->page_35);

    QVBoxLayout *gridpanel0041_0042 = new QVBoxLayout(this);
    gridpanel0041_0042->addWidget(panel0041_0042);
    ui->page_35->setLayout(gridpanel0041_0042);

    // Radioenlace C. Maribáñez - San Fernando
    panel0042_0043 = new PanelAlarmasRadio(ui->page_36);

    QVBoxLayout *gridpanel0042_0043 = new QVBoxLayout(this);
    gridpanel0042_0043->addWidget(panel0042_0043);
    ui->page_36->setLayout(gridpanel0042_0043);

    // Radioenlace San Fernando - Oficina
    panel0043_0000 = new PanelAlarmasRadio(ui->page_37);

    QVBoxLayout *gridpanel0043_0000 = new QVBoxLayout(this);
    gridpanel0043_0000->addWidget(panel0043_0000);
    ui->page_37->setLayout(gridpanel0043_0000);

    // Radioenlace Alcantarillas - Oficina
    panel0044_0000 = new PanelAlarmasRadio(ui->page_38);

    QVBoxLayout *gridpanel0044_0000 = new QVBoxLayout(this);
    gridpanel0044_0000->addWidget(panel0044_0000);
    ui->page_38->setLayout(gridpanel0044_0000);

    // Radioenlaces secundarios

    // Radioenlace Ctra. Utrera - C. Maribáñez
    panel0041_0043 = new PanelAlarmasRadio(ui->page_39);

    QVBoxLayout *gridpanel0041_0043 = new QVBoxLayout(this);
    gridpanel0041_0043->addWidget(panel0041_0043);
    ui->page_39->setLayout(gridpanel0041_0043);

    // Radioenlace C. Maribáñez - San Fernando
    panel0041_0000 = new PanelAlarmasRadio(ui->page_40);

    QVBoxLayout *gridpanel0041_0000 = new QVBoxLayout(this);
    gridpanel0041_0000->addWidget(panel0041_0000);
    ui->page_40->setLayout(gridpanel0041_0000);

    // Radioenlace San Fernando - Oficina
    panel0042_0000 = new PanelAlarmasRadio(ui->page_41);

    QVBoxLayout *gridpanel0042_0000 = new QVBoxLayout(this);
    gridpanel0042_0000->addWidget(panel0042_0000);
    ui->page_41->setLayout(gridpanel0042_0000);
}

/* ------------------------------------------------------------------------------
*   Conexiones de las pulsaciones de los items del arbol
*   con la muestra de gráficas
* ------------------------------------------------------------------------------*/

void showMonitor::setupTreeActions()
{
    // Conectamos la señal "item pulsado" al slot "mostrarGraficos"
    // El mismo slot sirve para los dos árboles de ítems

    // Conexión para ítems del arbol 1 (Nodos)
    connect(ui->treeWidget,SIGNAL(itemClicked(QTreeWidgetItem*,int)),
            this,SLOT(mostrarGraficos(QTreeWidgetItem*,int)));

    // Conexión para ítems del arbol 2 (Radioenlaces)
    connect(ui->treeWidget_2,SIGNAL(itemClicked(QTreeWidgetItem*,int)),
            this,SLOT(mostrarGraficos(QTreeWidgetItem*,int)));

}

/* ------------------------------------------------------------------------------
*   SLOT en respuesta a la pulsación de un ítem del árbol de nodos o
*   del árbol de radioenlaces.
```

```
*   Actualiza los widgets de Gráficas de los stackedWidgets de la derecha.
* ----------------------------------------------------------------------------*/

void showMonitor::mostrarGraficos(QTreeWidgetItem *item, int columna)
{
    // Árbol de NODOS
    if (item->text(columna) == "Ctra. Utrera")
    {
        // Títulos de los GroupBoxes
        ui->groupBox_3->setTitle("Niveles en tiempo real  (Nodo Ctra. Utrera)");
        ui->groupBox_4->setTitle("Histórico  (Nodo Ctra. Utrera)");
        ui->groupBox_5->setTitle("Alarmas  (Nodo Ctra. Utrera)");

        // Cambiamos la visibilidad los stackedWidgets
        ui->stackedWidget->setCurrentIndex(0);
        ui->stackedWidget_2->setCurrentIndex(0);
        ui->stackedWidget_3->setCurrentIndex(0);

    }
    else if (item->text(columna) == "C. Maribáñez")
    {
        // Títulos de los GroupBoxes
        ui->groupBox_3->setTitle("Niveles en tiempo real  (Nodo C. Maribáñez)");
        ui->groupBox_4->setTitle("Histórico  (Nodo C. Maribáñez)");
        ui->groupBox_5->setTitle("Alarmas  (Nodo C. Maribáñez)");

        // Cambiamos la visibilidad de los stackedWidgets
        ui->stackedWidget->setCurrentIndex(1);
        ui->stackedWidget_2->setCurrentIndex(1);
        ui->stackedWidget_3->setCurrentIndex(1);


    }
    else if (item->text(columna) == "San Fernando")
    {
        // Títulos de los GroupBoxes
        ui->groupBox_3->setTitle("Niveles en tiempo real  (Nodo San Fernando)");
        ui->groupBox_4->setTitle("Histórico  (Nodo C. San Fernando)");
        ui->groupBox_5->setTitle("Alarmas  (Nodo C. San Fernando)");

        // Cambiamos la visibilidad de los stackedWidgets
        ui->stackedWidget->setCurrentIndex(2);
        ui->stackedWidget_2->setCurrentIndex(2);
        ui->stackedWidget_3->setCurrentIndex(2);

    }
    else if (item->text(columna) == "Oficina")
    {
        // Títulos de los GroupBoxes
        ui->groupBox_3->setTitle("Niveles en tiempo real  (Nodo Oficina)");
        ui->groupBox_4->setTitle("Histórico  (Nodo Oficina)");
        ui->groupBox_5->setTitle("Alarmas  (Nodo C. Oficina)");

        // Cambiamos la visibilidad de los stackedWidgets
        ui->stackedWidget->setCurrentIndex(3);
        ui->stackedWidget_2->setCurrentIndex(3);
        ui->stackedWidget_3->setCurrentIndex(3);

    }
    else if (item->text(columna) == "Alcantarillas")
    {
        // Títulos de los GroupBoxes
        ui->groupBox_3->setTitle("Niveles en tiempo real  (Nodo Alcantarillas)");
        ui->groupBox_4->setTitle("Histórico  (Nodo Alcantarillas)");
        ui->groupBox_5->setTitle("Alarmas  (Nodo Alcantarillas)");

        // Cambiamos la visibilidad de los stackedWidgets
        ui->stackedWidget->setCurrentIndex(4);
        ui->stackedWidget_2->setCurrentIndex(4);
        ui->stackedWidget_3->setCurrentIndex(4);

    } // Árbol de RADIOENLACES
    else if (item->text(columna) == "Ctra. Utrera - C. Maribáñez")
    {
        // Títulos de los GroupBoxes
        ui->groupBox_3->setTitle("Niveles en tiempo real  (Radioenlace Ctra. Utrera - C. Maribáñez)");
        ui->groupBox_4->setTitle("Histórico  (Radioenlace Ctra. Utrera - C. Maribáñez)");
        ui->groupBox_5->setTitle("Alarmas  (Radioenlace Ctra. Utrera - C. Maribáñez)");

        // Cambiamos la visibilidad de los stackedWidgets
        ui->stackedWidget->setCurrentIndex(5);
        ui->stackedWidget_2->setCurrentIndex(5);
        ui->stackedWidget_3->setCurrentIndex(5);
```

```
    }
    else if (item->text(columna) == "C. Maribáñez - San Fernando")
    {
        // Títulos de los GroupBoxes
        ui->groupBox_3->setTitle("Niveles en tiempo real  (Radioenlace C. Maribáñez - San Fernando)");
        ui->groupBox_4->setTitle("Histórico  (Radioenlace C. Maribáñez - San Fernando)");
        ui->groupBox_5->setTitle("Alarmas  (Radioenlace C. Maribáñez - San Fernando)");

        // Cambiamos la visibilidad de los stackedWidgets
        ui->stackedWidget->setCurrentIndex(6);
        ui->stackedWidget_2->setCurrentIndex(6);
        ui->stackedWidget_3->setCurrentIndex(6);

    }
    else if (item->text(columna) == "San Fernando - Oficina")
    {
        // Títulos de los GroupBoxes
        ui->groupBox_3->setTitle("Niveles en tiempo real  (Radioenlace San Fernando - Oficina)");
        ui->groupBox_4->setTitle("Histórico  (Radioenlace San Fernando - Oficina)");
        ui->groupBox_5->setTitle("Alarmas  (Radioenlace San Fernando - Oficina)");

        // Cambiamos la visibilidad de los stackedWidgets
        ui->stackedWidget->setCurrentIndex(7);
        ui->stackedWidget_2->setCurrentIndex(7);
        ui->stackedWidget_3->setCurrentIndex(7);

    }
    else if (item->text(columna) == "Oficina - Alcantarillas")
    {
        ui->groupBox_3->setTitle("Niveles en tiempo real  (Radioenlace Oficina - Alcantarillas)");
        ui->groupBox_4->setTitle("Histórico  (Radioenlace Oficina - Alcantarillas)");
        ui->groupBox_5->setTitle("Alarmas  (Radioenlace Oficina - Alcantarillas)");

        // Cambiamos la visibilidad de los stackedWidgets
        ui->stackedWidget->setCurrentIndex(8);
        ui->stackedWidget_2->setCurrentIndex(8);
        ui->stackedWidget_3->setCurrentIndex(8);

    }
    else if (item->text(columna) == "Ctra. Utrera - San Fernando")
    {
        ui->groupBox_3->setTitle("Niveles en tiempo real  (Radioenlace Ctra. Utrera - San Fernando)");
        ui->groupBox_4->setTitle("Histórico  (Radioenlace Ctra. Utrera - San Fernando)");
        ui->groupBox_5->setTitle("Alarmas  (Radioenlace Ctra. Utrera - San Fernando)");

        // Cambiamos la visibilidad de los stackedWidgets
        ui->stackedWidget->setCurrentIndex(9);
        ui->stackedWidget_2->setCurrentIndex(9);
        ui->stackedWidget_3->setCurrentIndex(9);

    }
    else if (item->text(columna) == "Ctra. Utrera - Oficina")
    {
        ui->groupBox_3->setTitle("Niveles en tiempo real  (Radioenlace Ctra. Utrera - Oficina)");
        ui->groupBox_4->setTitle("Histórico  (Radioenlace Ctra. Utrera - Oficina)");
        ui->groupBox_5->setTitle("Alarmas  (Radioenlace Ctra. Utrera - Oficina)");

        // Cambiamos la visibilidad de los stackedWidgets
        ui->stackedWidget->setCurrentIndex(10);
        ui->stackedWidget_2->setCurrentIndex(10);
        ui->stackedWidget_3->setCurrentIndex(10);

    }
    else if (item->text(columna) == "C. Maribáñez - Oficina")
    {
        ui->groupBox_3->setTitle("Niveles en tiempo real  (Radioenlace C. Maribáñez - Oficina)");
        ui->groupBox_4->setTitle("Histórico  (Radioenlace C. Maribáñez - Oficina)");
        ui->groupBox_5->setTitle("Alarmas  (Radioenlace C. Maribáñez - Oficina)");

        // Cambiamos la visibilidad de los stackedWidgets
        ui->stackedWidget->setCurrentIndex(11);
        ui->stackedWidget_2->setCurrentIndex(11);
        ui->stackedWidget_3->setCurrentIndex(11);

    }
}
```

Sleeper.cpp

```
#include "sleeper.h"

void Sleeper::usleep(unsigned long usecs)
```

```
{
    QThread::usleep(usecs);
}

void Sleeper::msleep(unsigned long msecs)
{
    QThread::msleep(msecs);
}

void Sleeper::sleep(unsigned long secs)
{
    QThread::sleep(secs);
}
```

Valuebar.cpp

```
#include "valuebar.h"

ValueBar::ValueBar(const QString &text, QWidget *parent, double value,
                   double scaleMin, double scaleMax, const QBrush &pincel) :
    QWidget(parent)
{
    // Creamos la QLabel donde va a ir el titulo de la barra
    d_label = new QLabel( text, this );
    d_label->setFont( QFont( "Helvetica", 10 ) );

    // Creamos el termo donde va a ir
    d_thermo = new QwtThermo( this );
    d_thermo->setRange( scaleMin, scaleMax );
    d_thermo->setValue( value );
    d_thermo->setFont( QFont( "Helvetica", 8 ) );
    d_thermo->setPipeWidth( 6 );
    d_thermo->setScaleMaxMajor( 6 ); //Nº Intervalos grandes
    d_thermo->setScaleMaxMinor( 5 ); //Nº Intervalos pequeños
    d_thermo->setFillBrush( pincel );

    QVBoxLayout *layoutVB = new QVBoxLayout( this );
    layoutVB->setMargin( 0 );
    layoutVB->setSpacing( 0 );

    Qt::Orientation orientation = Qt::Horizontal;
    d_label->setAlignment( Qt::AlignCenter );
    d_thermo->setOrientation( orientation, QwtThermo::BottomScale );
    layoutVB->addWidget( d_label );
    layoutVB->addWidget( d_thermo );
}

void ValueBar::setValue(double value)
{
    d_thermo->setValue( value );
}
```

Waspfilenames.cpp

```
#include "waspfilenames.h"

WaspFileNames::WaspFileNames(QObject *parent) :
    QObject(parent)
{
    logName = "log";
}
```

Waspflags.cpp

```
#include "waspflags.h"

WaspFlags::WaspFlags(QObject *parent) :
    QObject(parent)
{
    saveFileHTML = 1;
    saveFileCSV = 1;
    firstTimeOpenHTML = 1;

    nodo0041_ON = 1;
    nodo0042_ON = 1;
    nodo0043_ON = 1;
    nodo0044_ON = 1;

    capturaEnCurso = 0;
```

```
}
```

Waspmonitor.cpp

```
#include <QtCore>
#include <QtGui>
#include "waspmonitor.h"
#include "ui_waspmonitor.h"
#include "ui_showinfo.h"
#include "ui_configdialog.h"
// Esto se usa para dar tiempo al proceso de apertura
// del puerto serie
#include "sleeper.h"

/** ***************************************************** **/
/**                     VERSIÓN 3.2.4                     **/
/** ***************************************************** **/

/* ------------------------------------------------------------------------
*   Constructor de la Ventana Principal de WaspMonitor.
* ------------------------------------------------------------------------*/

WaspMonitor::WaspMonitor(QWidget *parent) :
    QMainWindow(parent),
    ui(new Ui::WaspMonitor)
{
    ui->setupUi(this);

    /** Inicializamos el menú de configuración */
    configDialogBox = new configDialog(this);

    setupConfigDialogBox();
    setupConfig();
    setupActions();
    setupContador();
    setupHiloSniffer();
}

/* ------------------------------------------------------------------------
*   Destructor de la Ventana Principal de WaspMonitor.
* ------------------------------------------------------------------------*/

WaspMonitor::~WaspMonitor()
{
    delete ui;
}

/* ------------------------------------------------------------------------
*   Configuración del programa mediante los WaspFlags
* ------------------------------------------------------------------------*/

void WaspMonitor::setupConfig()
{
    // Configuramos los flags
    waspFlags = new WaspFlags(this);

    // Configuramos los nombres de archivo
    waspFileNames = new WaspFileNames(this);
}

/* ------------------------------------------------------------------------
*   Configuración del contador que servirá para comprobar cada 2 minutos si
*   los nodos siguen vivos y sus correspondientes radioenlaces
* ------------------------------------------------------------------------*/

void WaspMonitor::setupContador()
{
    // Creamos el contador general
    contador = new QTimer(this);
    // Lo conectamos con su SLOT correspondiente que se activará con la señal "timeout"
    connect(contador, SIGNAL(timeout()), this, SLOT(checkEstadoNodos()));
    connect(contador, SIGNAL(timeout()), this, SLOT(checkEstadoEnlaces()));

    // Los contadores de cada nodo y de cada enlace se inician al crear el objeto WaspMonitor
}

/* ------------------------------------------------------------------------
*   Configura las acciones a realizar cuando se pulsen las casillas del
*   QDialogBox de Configuración (nodos y logs) y los ítems del QComboBox (puertos COM).
*   También configura el botón Aceptar para que se guarden los cambios.
* ------------------------------------------------------------------------*/
```

```
void WaspMonitor::setupConfigDialogBox()
{
    // QCheckBoxes de habilitación/deshabilitación de nodos
    connect(configDialogBox->ui->checkBox_0041, SIGNAL(clicked()),
            this, SLOT(setNodeON_or_OFF_0041()));
    connect(configDialogBox->ui->checkBox_0042, SIGNAL(clicked()),
            this, SLOT(setNodeON_or_OFF_0042()));
    connect(configDialogBox->ui->checkBox_0043, SIGNAL(clicked()),
            this, SLOT(setNodeON_or_OFF_0043()));
    connect(configDialogBox->ui->checkBox_0044, SIGNAL(clicked()),
            this, SLOT(setNodeON_or_OFF_0044()));

    // QCheckBoxes de habilitación/deshabilitación de logs
    connect(configDialogBox->ui->checkBox_logCSV, SIGNAL(clicked()),
            this, SLOT(setLogCSV_ON_or_OFF()));
    connect(configDialogBox->ui->checkBox_logHTML, SIGNAL(clicked()),
            this, SLOT(setLogHTML_ON_or_OFF()));

    // QComboBox de puertos COM
    connect(configDialogBox->ui->comboBox, SIGNAL(activated(QString)),
            this, SLOT(elegirPuerto(QString)));

    // Botón de aceptar
    connect(configDialogBox->ui->pushButton, SIGNAL(clicked()),
            configDialogBox, SLOT(hide()));
}

/* ------------------------------------------------------------------------
*   Configura las acciones a realizar cuando se pulsen los botones de la ToolBar,
*   y los botones de los paneles de alarma
* ------------------------------------------------------------------------*/

void WaspMonitor::setupActions()
{
    // Botones de la ToolBar
    connect(ui->actionInfo, SIGNAL(triggered(bool)),
            this, SLOT(mostrarInfo()));

    connect(ui->actionConfiguracion, SIGNAL(triggered(bool)),
            this, SLOT(mostrarConfig()));

    connect(ui->actionLimpiarAlarmas, SIGNAL(triggered(bool)),
            this, SLOT(limpiarAlarmas()));

    connect(ui->actionMonitorizacion, SIGNAL(triggered(bool)),
            this, SLOT(mostrarMonitorizacion()));

    connect(ui->actionIniciarCaptura, SIGNAL(triggered(bool)),
            this, SLOT(iniciarCaptura()));

    connect(ui->actionDetenerCaptura, SIGNAL(triggered(bool)),
            this, SLOT(detenerCaptura()));

    connect(ui->actionSalir, SIGNAL(triggered(bool)),
            this, SLOT(salir()));

    // Botones de los paneles de alarma

    /* --- Nodo 0000 ---*/
    connect(ui->page3->showMonitor::panel0000->buttonNodoCaido, SIGNAL(clicked()),
            this, SLOT(limpiarNC_0000()));
    connect(ui->page3->showMonitor::panel0000->buttonBatBaja, SIGNAL(clicked()),
            this, SLOT(limpiarBB_0000()));
    connect(ui->page3->showMonitor::panel0000->buttonTempAlta, SIGNAL(clicked()),
            this, SLOT(limpiarTA_0000()));
    connect(ui->page3->showMonitor::panel0000->buttonDCAlto, SIGNAL(clicked()),
            this, SLOT(limpiarDC_0000()));

    /* --- Nodo 0041 ---*/
    connect(ui->page3->showMonitor::panel0041->buttonNodoCaido, SIGNAL(clicked()),
            this, SLOT(limpiarNC_0041()));
    connect(ui->page3->showMonitor::panel0041->buttonBatBaja, SIGNAL(clicked()),
            this, SLOT(limpiarBB_0041()));
    connect(ui->page3->showMonitor::panel0041->buttonTempAlta, SIGNAL(clicked()),
            this, SLOT(limpiarTA_0041()));
    connect(ui->page3->showMonitor::panel0041->buttonDCAlto, SIGNAL(clicked()),
            this, SLOT(limpiarDC_0041()));

    /* --- Nodo 0042 ---*/
    connect(ui->page3->showMonitor::panel0042->buttonNodoCaido, SIGNAL(clicked()),
            this, SLOT(limpiarNC_0042()));
    connect(ui->page3->showMonitor::panel0042->buttonBatBaja, SIGNAL(clicked()),
            this, SLOT(limpiarBB_0042()));
    connect(ui->page3->showMonitor::panel0042->buttonTempAlta, SIGNAL(clicked()),
```

```
            this, SLOT(limpiarTA_0042()));
    connect(ui->page3->showMonitor::panel0042->buttonDCAlto, SIGNAL(clicked()),
            this, SLOT(limpiarDC_0042()));

    /* --- Nodo 0043 ---*/
    connect(ui->page3->showMonitor::panel0043->buttonNodoCaido, SIGNAL(clicked()),
            this, SLOT(limpiarNC_0043()));
    connect(ui->page3->showMonitor::panel0043->buttonBatBaja, SIGNAL(clicked()),
            this, SLOT(limpiarBB_0043()));
    connect(ui->page3->showMonitor::panel0043->buttonTempAlta, SIGNAL(clicked()),
            this, SLOT(limpiarTA_0043()));
    connect(ui->page3->showMonitor::panel0043->buttonDCAlto, SIGNAL(clicked()),
            this, SLOT(limpiarDC_0043()));

    /* --- Nodo 0044 ---*/
    connect(ui->page3->showMonitor::panel0044->buttonNodoCaido, SIGNAL(clicked()),
            this, SLOT(limpiarNC_0044()));
    connect(ui->page3->showMonitor::panel0044->buttonBatBaja, SIGNAL(clicked()),
            this, SLOT(limpiarBB_0044()));
    connect(ui->page3->showMonitor::panel0044->buttonTempAlta, SIGNAL(clicked()),
            this, SLOT(limpiarTA_0044()));
    connect(ui->page3->showMonitor::panel0044->buttonDCAlto, SIGNAL(clicked()),
            this, SLOT(limpiarDC_0044()));

    /* --- Radioenlaces primarios ---*/
    connect(ui->page3->showMonitor::panel0041_0042->buttonEnlaceCaido, SIGNAL(clicked()),
            this, SLOT(limpiarRC_0041_0042()));
    connect(ui->page3->showMonitor::panel0042_0043->buttonEnlaceCaido, SIGNAL(clicked()),
            this, SLOT(limpiarRC_0042_0043()));
    connect(ui->page3->showMonitor::panel0043_0000->buttonEnlaceCaido, SIGNAL(clicked()),
            this, SLOT(limpiarRC_0043_0000()));
    connect(ui->page3->showMonitor::panel0044_0000->buttonEnlaceCaido, SIGNAL(clicked()),
            this, SLOT(limpiarRC_0044_0000()));

    /* --- Radioenlaces secundarios ---*/
    connect(ui->page3->showMonitor::panel0041_0043->buttonEnlaceCaido, SIGNAL(clicked()),
            this, SLOT(limpiarRC_0041_0043()));
    connect(ui->page3->showMonitor::panel0041_0000->buttonEnlaceCaido, SIGNAL(clicked()),
            this, SLOT(limpiarRC_0041_0000()));
    connect(ui->page3->showMonitor::panel0042_0000->buttonEnlaceCaido, SIGNAL(clicked()),
            this, SLOT(limpiarRC_0042_0000()));

}

/* ------------------------------------------------------------------------
*   Configura el hilo Sniffer y realiza las conexiones de las signals
*   con los slots apropiados para el manejo del proceso de captura.
* ------------------------------------------------------------------------*/

void WaspMonitor::setupHiloSniffer()
{
    // Creamos el objeto e invocamos al constructor donde se configurarán
    // los parámetros del puerto serie
    hiloSniffer = new HiloSniffer(this);

    // Cuando haya un mensaje listo... actualizar las gráficas, log, etc.
    connect(hiloSniffer, SIGNAL(mensajeListo()), this, SLOT(updateOutput()));

}

/* ------------------------------------------------------------------------
*   SLOT: Actualiza la salida del hiloSniffer.
*         Cuando se recibe una nueva línea desde hiloSniffer:
*         - Se ordena actualizar el Log (Ventana, archivo CSV y HTML)
*         - Se ordena actualizar las gráficas de monitorización.
* ------------------------------------------------------------------------*/

void WaspMonitor::updateOutput()
{
    // Grabamos linea a linea en un QByteArray (RAW) los mensajes recibidos en ASCII
    // desde el puerto serie
    mensajeRAW = hiloSniffer->mensajeRecibido;

    // Convertimos el QByteArray en Qstring para poder operar con la cadena
    mensajeStr = mensajeStr.fromAscii(mensajeRAW);

    // Procesamos el mensaje (Se traducen los días de la semana,
    // y se eliminan espacios en las fechas que dan errores en tiempo de ejecución)
    procesarMensaje();

    // Actualización de contadores de estado de nodos, graficas de monitorización
    // ventana de log, etc.

    actualizarContadoresNodos();
```

```
    actualizarContadoresEnlaces();
    actualizarLog();
    actualizarGraficas();

}

/* ------------------------------------------------------------------------
*   SLOT: Elige el puerto COM a utilizar
* ------------------------------------------------------------------------*/

void WaspMonitor::elegirPuerto(QString nombrePuerto)
{
    // Si la captura no esta en curso, se puede cambiar de puerto,
    // si no, no se puede.
    if (waspFlags->capturaEnCurso == 0)
    {
        hiloSniffer->nombrePuerto = nombrePuerto;
    }
    else
    {
        QMessageBox warnCannotChangePort;
        warnCannotChangePort.warning(this,"Atención","La captura de datos está en curso.\nSi desea cambiar el puerto COM del
Waspmote Gateway,\n detenga antes la captura.");
    }
}

/* ------------------------------------------------------------------------
*   SLOT: Habilita o deshabilita al nodo 0041
* ------------------------------------------------------------------------*/

void WaspMonitor::setNodeON_or_OFF_0041()
{

    if(configDialogBox->ui->checkBox_0041->isChecked())
    {
        waspFlags->nodo0041_ON = 1;

        // Ponemos la etiqueta de temperatura en blanco
        ui->page3->showMonitor::paleta0041.setColor(QPalette::Base, Qt::white);
        ui->page3->showMonitor::tag0041->setPalette(ui->page3->showMonitor::paleta0041);

        // Ponemos los radioenlaces que dependen del nodo en rojo
        ui->page3->showMonitor::tagLed0041_0042->show();
        ui->page3->showMonitor::Led0041_0042.load(":/red_light_16.png");
        ui->page3->showMonitor::tagLed0041_0042->setPixmap(ui->page3->showMonitor::Led0041_0042);

        ui->page3->showMonitor::tagLed0041_0043->show();
        ui->page3->showMonitor::Led0041_0043.load(":/red_light_16.png");
        ui->page3->showMonitor::tagLed0041_0043->setPixmap(ui->page3->showMonitor::Led0041_0043);

        ui->page3->showMonitor::tagLed0041_0000->show();
        ui->page3->showMonitor::Led0041_0000.load(":/red_light_16.png");
        ui->page3->showMonitor::tagLed0041_0000->setPixmap(ui->page3->showMonitor::Led0041_0000);

        // StatusBar: Monitorización Nodo0041 habilitada
        QString msg0041_ON = "Monitorización del Nodo 0041 habilitada";
        ui->statusBar->showMessage(msg0041_ON, 4000);
    }
    else
    {
        waspFlags->nodo0041_ON = 0;

        // Ponemos la etiqueta de temperatura en amarillo claro
        ui->page3->showMonitor::paleta0041.setColor(QPalette::Base, QColor(255,230,0));
        ui->page3->showMonitor::tag0041->setPalette(ui->page3->showMonitor::paleta0041);

        // Ponemos los radioenlaces que dependen del nodo en amarillo
        ui->page3->showMonitor::tagLed0041_0042->show();
        ui->page3->showMonitor::Led0041_0042.load(":/yellow_light_16.png");
        ui->page3->showMonitor::tagLed0041_0042->setPixmap(ui->page3->showMonitor::Led0041_0042);

        ui->page3->showMonitor::tagLed0041_0043->show();
        ui->page3->showMonitor::Led0041_0043.load(":/yellow_light_16.png");
        ui->page3->showMonitor::tagLed0041_0043->setPixmap(ui->page3->showMonitor::Led0041_0043);

        ui->page3->showMonitor::tagLed0041_0000->show();
        ui->page3->showMonitor::Led0041_0000.load(":/yellow_light_16.png");
        ui->page3->showMonitor::tagLed0041_0000->setPixmap(ui->page3->showMonitor::Led0041_0000);

        // StatusBar: Monitorización Nodo0041 deshabilitada
        QString msg0041_OFF = "Monitorización del Nodo 0041 deshabilitada";
        ui->statusBar->showMessage(msg0041_OFF, 4000);
    }
}
```

```
/* ------------------------------------------------------------------------
*   SLOT: Habilita o deshabilita al nodo 0042
* ------------------------------------------------------------------------*/

void WaspMonitor::setNodeON_or_OFF_0042()
{

    if(configDialogBox->ui->checkBox_0042->isChecked())
    {
        waspFlags->nodo0042_ON = 1;

        // Ponemos la etiqueta de temperatura en blanco
        ui->page3->showMonitor::paleta0042.setColor(QPalette::Base, Qt::white);
        ui->page3->showMonitor::tag0042->setPalette(ui->page3->showMonitor::paleta0042);

        // Ponemos los radioenlaces que dependen del nodo en rojo
        ui->page3->showMonitor::tagLed0041_0042->show();
        ui->page3->showMonitor::Led0041_0042.load(":/red_light_16.png");
        ui->page3->showMonitor::tagLed0041_0042->setPixmap(ui->page3->showMonitor::Led0041_0042);

        ui->page3->showMonitor::tagLed0042_0043->show();
        ui->page3->showMonitor::Led0042_0043.load(":/red_light_16.png");
        ui->page3->showMonitor::tagLed0042_0043->setPixmap(ui->page3->showMonitor::Led0042_0043);

        ui->page3->showMonitor::tagLed0042_0000->show();
        ui->page3->showMonitor::Led0042_0000.load(":/red_light_16.png");
        ui->page3->showMonitor::tagLed0042_0000->setPixmap(ui->page3->showMonitor::Led0042_0000);

        // StatusBar: Monitorización Nodo0042 habilitada
        QString msg0042_ON = "Monitorización del Nodo 0042 habilitada";
        ui->statusBar->showMessage(msg0042_ON, 4000);
    }
    else
    {
        waspFlags->nodo0042_ON = 0;

        // Ponemos la etiqueta de temperatura en amarillo claro
        ui->page3->showMonitor::paleta0042.setColor(QPalette::Base, QColor(255,230,0));
        ui->page3->showMonitor::tag0042->setPalette(ui->page3->showMonitor::paleta0042);

        // Ponemos los radioenlaces que dependen del nodo en amarillo
        ui->page3->showMonitor::tagLed0041_0042->show();
        ui->page3->showMonitor::Led0041_0042.load(":/yellow_light_16.png");
        ui->page3->showMonitor::tagLed0041_0042->setPixmap(ui->page3->showMonitor::Led0041_0042);

        ui->page3->showMonitor::tagLed0042_0043->show();
        ui->page3->showMonitor::Led0042_0043.load(":/yellow_light_16.png");
        ui->page3->showMonitor::tagLed0042_0043->setPixmap(ui->page3->showMonitor::Led0042_0043);

        ui->page3->showMonitor::tagLed0042_0000->show();
        ui->page3->showMonitor::Led0042_0000.load(":/yellow_light_16.png");
        ui->page3->showMonitor::tagLed0042_0000->setPixmap(ui->page3->showMonitor::Led0042_0000);

        // StatusBar: Monitorización Nodo0042 deshabilitada
        QString msg0042_OFF = "Monitorización del Nodo 0042 deshabilitada";
        ui->statusBar->showMessage(msg0042_OFF, 4000);
    }
}

/* ------------------------------------------------------------------------
*   SLOT: Habilita o deshabilita al nodo 0043
* ------------------------------------------------------------------------*/

void WaspMonitor::setNodeON_or_OFF_0043()
{

    if(configDialogBox->ui->checkBox_0043->isChecked())
    {
        waspFlags->nodo0043_ON = 1;

        // Ponemos la etiqueta de temperatura en blanco
        ui->page3->showMonitor::paleta0043.setColor(QPalette::Base, Qt::white);
        ui->page3->showMonitor::tag0043->setPalette(ui->page3->showMonitor::paleta0043);

        // Ponemos los radioenlaces que dependen del nodo en rojo
        ui->page3->showMonitor::tagLed0041_0043->show();
        ui->page3->showMonitor::Led0041_0043.load(":/red_light_16.png");
        ui->page3->showMonitor::tagLed0041_0043->setPixmap(ui->page3->showMonitor::Led0041_0043);

        ui->page3->showMonitor::tagLed0042_0043->show();
        ui->page3->showMonitor::Led0042_0043.load(":/red_light_16.png");
        ui->page3->showMonitor::tagLed0042_0043->setPixmap(ui->page3->showMonitor::Led0042_0043);
```

```
        ui->page3->showMonitor::tagLed0043_0000->show();
        ui->page3->showMonitor::Led0043_0000.load(":/red_light_16.png");
        ui->page3->showMonitor::tagLed0043_0000->setPixmap(ui->page3->showMonitor::Led0043_0000);

        // StatusBar: Monitorización Nodo0043 habilitada
        QString msg0043_ON = "Monitorización del Nodo 0043 habilitada";
        ui->statusBar->showMessage(msg0043_ON, 4000);
    }
    else
    {
        waspFlags->nodo0043_ON = 0;

        // Ponemos la etiqueta de temperatura en amarillo claro
        ui->page3->showMonitor::paleta0043.setColor(QPalette::Base, QColor(255,230,0));
        ui->page3->showMonitor::tag0043->setPalette(ui->page3->showMonitor::paleta0043);

        // Ponemos los radioenlaces que dependen del nodo en amarillo
        ui->page3->showMonitor::tagLed0041_0043->show();
        ui->page3->showMonitor::Led0041_0043.load(":/yellow_light_16.png");
        ui->page3->showMonitor::tagLed0041_0043->setPixmap(ui->page3->showMonitor::Led0041_0043);

        ui->page3->showMonitor::tagLed0042_0043->show();
        ui->page3->showMonitor::Led0042_0043.load(":/yellow_light_16.png");
        ui->page3->showMonitor::tagLed0042_0043->setPixmap(ui->page3->showMonitor::Led0042_0043);

        ui->page3->showMonitor::tagLed0043_0000->show();
        ui->page3->showMonitor::Led0043_0000.load(":/yellow_light_16.png");
        ui->page3->showMonitor::tagLed0043_0000->setPixmap(ui->page3->showMonitor::Led0043_0000);

        // StatusBar: Monitorización Nodo0043 deshabilitada
        QString msg0043_OFF = "Monitorización del Nodo 0043 deshabilitada";
        ui->statusBar->showMessage(msg0043_OFF, 4000);
    }
}

/* ------------------------------------------------------------------------
*   SLOT: Habilita o deshabilita al nodo 0044
* ------------------------------------------------------------------------*/

void WaspMonitor::setNodeON_or_OFF_0044()
{
    if(configDialogBox->ui->checkBox_0044->isChecked())
    {
        waspFlags->nodo0044_ON = 1;

        // Ponemos la etiqueta de temperatura en blanco
        ui->page3->showMonitor::paleta0044.setColor(QPalette::Base, Qt::white);
        ui->page3->showMonitor::tag0044->setPalette(ui->page3->showMonitor::paleta0044);

        // Ponemos los radioenlaces que dependen del nodo en rojo
        ui->page3->showMonitor::tagLed0044_0000->show();
        ui->page3->showMonitor::Led0044_0000.load(":/red_light_16.png");
        ui->page3->showMonitor::tagLed0044_0000->setPixmap(ui->page3->showMonitor::Led0044_0000);

        // StatusBar: Monitorización Nodo0044 habilitada
        QString msg0044_ON = "Monitorización del Nodo 0044 habilitada";
        ui->statusBar->showMessage(msg0044_ON, 4000);
    }
    else
    {
        waspFlags->nodo0044_ON = 0;

        // Ponemos la etiqueta de temperatura en amarillo claro
        ui->page3->showMonitor::paleta0044.setColor(QPalette::Base, QColor(255,230,0));
        ui->page3->showMonitor::tag0044->setPalette(ui->page3->showMonitor::paleta0044);

        // Ponemos los radioenlaces que dependen del nodo en amarillo
        ui->page3->showMonitor::tagLed0044_0000->show();
        ui->page3->showMonitor::Led0044_0000.load(":/yellow_light_16.png");
        ui->page3->showMonitor::tagLed0044_0000->setPixmap(ui->page3->showMonitor::Led0044_0000);

        // StatusBar: Monitorización Nodo0044 deshabilitada
        QString msg0044_OFF = "Monitorización del Nodo 0044 deshabilitada";
        ui->statusBar->showMessage(msg0044_OFF, 4000);
    }
}

/* ------------------------------------------------------------------------
*   SLOT: Habilita o deshabilita el log CSV
* ------------------------------------------------------------------------*/

void WaspMonitor::setLogCSV_ON_or_OFF()
{
    if(configDialogBox->ui->checkBox_logCSV->isChecked())
```

```
    {
        waspFlags->saveFileCSV = 1;

        // StatusBar: Log en CSV habilitado
        QString msgLogCSV_ON = "Log en CSV habilitado";
        ui->statusBar->showMessage(msgLogCSV_ON, 4000);
    }
    else
    {
        waspFlags->saveFileCSV = 0;

        // StatusBar: Log en CSV deshabilitado
        QString msgLogCSV_OFF = "Log en CSV deshabilitado";
        ui->statusBar->showMessage(msgLogCSV_OFF, 4000);
    }
}

/* ------------------------------------------------------------------------
*   SLOT: Habilita o deshabilita el log HTML
* ------------------------------------------------------------------------*/

void WaspMonitor::setLogHTML_ON_or_OFF()
{
    if(configDialogBox->ui->checkBox_logHTML->isChecked())
    {
        waspFlags->saveFileHTML = 1;

        // StatusBar: Log en HTML habilitado
        QString msgLogHTML_ON = "Log en HTML habilitado";
        ui->statusBar->showMessage(msgLogHTML_ON, 4000);
    }
    else
    {
        waspFlags->saveFileHTML = 0;

        // StatusBar: Log en HTML deshabilitado
        QString msgLogHTML_OFF = "Log en HTML deshabilitado";
        ui->statusBar->showMessage(msgLogHTML_OFF, 4000);
    }
}

/* ------------------------------------------------------------------------
*   SLOTS: Limpian los iconos de alarma del mapa y paneles
*   asociados al nodo 0000
* ------------------------------------------------------------------------*/

void WaspMonitor::limpiarNC_0000()
{
    ui->page3->showMonitor::panel0000->labelIconNodoCaido->setPixmap(ui->page3->showMonitor::panel0000->iconoAlarmaOFF);
}

void WaspMonitor::limpiarBB_0000()
{
    ui->page3->showMonitor::tagIconBatBaja0000->hide();
    ui->page3->showMonitor::panel0000->labelIconBatBaja->setPixmap(ui->page3->showMonitor::panel0000->iconoAlarmaOFF);
}

void WaspMonitor::limpiarTA_0000()
{
    ui->page3->showMonitor::tagTempAlta0000->hide();
    ui->page3->showMonitor::panel0000->labelIconTempAlta->setPixmap(ui->page3->showMonitor::panel0000->iconoAlarmaOFF);
}

void WaspMonitor::limpiarDC_0000()
{
    ui->page3->showMonitor::tagIconDCAlto0000->hide();
    ui->page3->showMonitor::panel0000->labelIconDCAlto->setPixmap(ui->page3->showMonitor::panel0000->iconoAlarmaOFF);
}

/* ------------------------------------------------------------------------
*   SLOTS: Limpian los iconos de alarma del mapa y paneles
*   asociados al nodo 0041
* ------------------------------------------------------------------------*/

void WaspMonitor::limpiarNC_0041()
{
    ui->page3->showMonitor::panel0041->labelIconNodoCaido->setPixmap(ui->page3->showMonitor::panel0041->iconoAlarmaOFF);
}

void WaspMonitor::limpiarBB_0041()
{
    ui->page3->showMonitor::tagIconBatBaja0041->hide();
    ui->page3->showMonitor::panel0041->labelIconBatBaja->setPixmap(ui->page3->showMonitor::panel0041->iconoAlarmaOFF);
}
```

```
void WaspMonitor::limpiarTA_0041()
{
    ui->page3->showMonitor::tagTempAlta0041->hide();
    ui->page3->showMonitor::panel0041->labelIconTempAlta->setPixmap(ui->page3->showMonitor::panel0041->iconoAlarmaOFF);
}

void WaspMonitor::limpiarDC_0041()
{
    ui->page3->showMonitor::tagIconDCAlto0041->hide();
    ui->page3->showMonitor::panel0041->labelIconDCAlto->setPixmap(ui->page3->showMonitor::panel0041->iconoAlarmaOFF);
}

/* ------------------------------------------------------------------------
*   SLOTS: Limpian los iconos de alarma del mapa y paneles
*   asociados al nodo 0042
* ------------------------------------------------------------------------*/

void WaspMonitor::limpiarNC_0042()
{
    ui->page3->showMonitor::panel0042->labelIconNodoCaido->setPixmap(ui->page3->showMonitor::panel0042->iconoAlarmaOFF);
}

void WaspMonitor::limpiarBB_0042()
{
    ui->page3->showMonitor::tagIconBatBaja0042->hide();
    ui->page3->showMonitor::panel0042->labelIconBatBaja->setPixmap(ui->page3->showMonitor::panel0042->iconoAlarmaOFF);
}

void WaspMonitor::limpiarTA_0042()
{
    ui->page3->showMonitor::tagTempAlta0042->hide();
    ui->page3->showMonitor::panel0042->labelIconTempAlta->setPixmap(ui->page3->showMonitor::panel0042->iconoAlarmaOFF);
}

void WaspMonitor::limpiarDC_0042()
{
    ui->page3->showMonitor::tagIconDCAlto0042->hide();
    ui->page3->showMonitor::panel0042->labelIconDCAlto->setPixmap(ui->page3->showMonitor::panel0042->iconoAlarmaOFF);
}

/* ------------------------------------------------------------------------
*   SLOTS: Limpian los iconos de alarma del mapa y paneles
*   asociados al nodo 0043
* ------------------------------------------------------------------------*/

void WaspMonitor::limpiarNC_0043()
{
    ui->page3->showMonitor::panel0043->labelIconNodoCaido->setPixmap(ui->page3->showMonitor::panel0043->iconoAlarmaOFF);
}

void WaspMonitor::limpiarBB_0043()
{
    ui->page3->showMonitor::tagIconBatBaja0043->hide();
    ui->page3->showMonitor::panel0043->labelIconBatBaja->setPixmap(ui->page3->showMonitor::panel0043->iconoAlarmaOFF);
}

void WaspMonitor::limpiarTA_0043()
{
    ui->page3->showMonitor::tagTempAlta0043->hide();
    ui->page3->showMonitor::panel0043->labelIconTempAlta->setPixmap(ui->page3->showMonitor::panel0043->iconoAlarmaOFF);
}

void WaspMonitor::limpiarDC_0043()
{
    ui->page3->showMonitor::tagIconDCAlto0043->hide();
    ui->page3->showMonitor::panel0043->labelIconDCAlto->setPixmap(ui->page3->showMonitor::panel0043->iconoAlarmaOFF);
}

/* ------------------------------------------------------------------------
*   SLOTS: Limpian los iconos de alarma del mapa y paneles
*   asociados al nodo 0044
* ------------------------------------------------------------------------*/

void WaspMonitor::limpiarNC_0044()
{
    ui->page3->showMonitor::panel0044->labelIconNodoCaido->setPixmap(ui->page3->showMonitor::panel0044->iconoAlarmaOFF);
}

void WaspMonitor::limpiarBB_0044()
{
    ui->page3->showMonitor::tagIconBatBaja0044->hide();
    ui->page3->showMonitor::panel0044->labelIconBatBaja->setPixmap(ui->page3->showMonitor::panel0044->iconoAlarmaOFF);
}
```

```
void WaspMonitor::limpiarTA_0044()
{
    ui->page3->showMonitor::tagTempAlta0044->hide();
    ui->page3->showMonitor::panel0044->labelIconTempAlta->setPixmap(ui->page3->showMonitor::panel0044->iconoAlarmaOFF);
}

void WaspMonitor::limpiarDC_0044()
{
    ui->page3->showMonitor::tagIconDCAlto0044->hide();
    ui->page3->showMonitor::panel0044->labelIconDCAlto->setPixmap(ui->page3->showMonitor::panel0044->iconoAlarmaOFF);
}


/* ------------------------------------------------------------------------
*   SLOTS: Limpian los iconos de alarma del mapa y paneles
*   asociados a los radioenlaces primarios
* ------------------------------------------------------------------------*/

void WaspMonitor::limpiarRC_0041_0042()
{
    ui->page3->showMonitor::tagRadioOff0041_0042->hide();
    ui->page3->showMonitor::panel0041_0042->labelIconEnlaceCaido->setPixmap(ui->page3->showMonitor::panel0041_0042-
>iconoAlarmaOFF);
}

void WaspMonitor::limpiarRC_0042_0043()
{
    ui->page3->showMonitor::tagRadioOff0042_0043->hide();
    ui->page3->showMonitor::panel0042_0043->labelIconEnlaceCaido->setPixmap(ui->page3->showMonitor::panel0042_0043-
>iconoAlarmaOFF);
}

void WaspMonitor::limpiarRC_0043_0000()
{
    ui->page3->showMonitor::tagRadioOff0043_0000->hide();
    ui->page3->showMonitor::panel0043_0000->labelIconEnlaceCaido->setPixmap(ui->page3->showMonitor::panel0043_0000-
>iconoAlarmaOFF);
}

void WaspMonitor::limpiarRC_0044_0000()
{
    ui->page3->showMonitor::tagRadioOff0044_0000->hide();
    ui->page3->showMonitor::panel0044_0000->labelIconEnlaceCaido->setPixmap(ui->page3->showMonitor::panel0044_0000-
>iconoAlarmaOFF);
}

/* ------------------------------------------------------------------------
*   SLOTS: Limpian los iconos de alarma del mapa y paneles
*   asociados a los radioenlaces secundarios
* ------------------------------------------------------------------------*/

void WaspMonitor::limpiarRC_0041_0043()
{
    ui->page3->showMonitor::tagRadioOff0041_0043->hide();
    ui->page3->showMonitor::panel0041_0043->labelIconEnlaceCaido->setPixmap(ui->page3->showMonitor::panel0041_0043-
>iconoAlarmaOFF);
}

void WaspMonitor::limpiarRC_0041_0000()
{
    ui->page3->showMonitor::tagRadioOff0041_0000->hide();
    ui->page3->showMonitor::panel0041_0000->labelIconEnlaceCaido->setPixmap(ui->page3->showMonitor::panel0041_0000-
>iconoAlarmaOFF);
}

void WaspMonitor::limpiarRC_0042_0000()
{
    ui->page3->showMonitor::tagRadioOff0042_0000->hide();
    ui->page3->showMonitor::panel0042_0000->labelIconEnlaceCaido->setPixmap(ui->page3->showMonitor::panel0042_0000-
>iconoAlarmaOFF);
}

/*-------------------------------------------------------
 *  Esta función actualiza la ventana de Log y
 *  crea los archivos de log tanto Html como CSV, dependiendo
 *  de si los flags de configuración están activados.
 *-------------------------------------------------------*/

void WaspMonitor::actualizarLog()
{
    // Si llega un mensaje erroneo se pone a 1
    int error_mensaje = 0;
```

```
/** Esta QStringList es una copia temporal de la QStringList de la clase
    que se introduce para hacer modificaciones sobre los campos del mensaje
    añadiendo las etiquetas HTML y por tanto pudiendo representar bien el mismo.*/
QStringList campoMensaje2 = campoMensaje;

/** Esta QString también es un objeto temporal que se usa para crear cada línea
    del log en formato HTML */
QString mensajeHTML = mensajeStr;

// Si hay más de un elemento en la QStringList, suponemos
// que es un mensaje válido
if( (campoMensaje2.size()) > 1)
{
    QString tipoMensaje = campoMensaje2[1];

if(tipoMensaje.contains("KEEP",Qt::CaseInsensitive))
{
    error_mensaje = 0;

    campoMensaje2[0].prepend("<b><i>");
    campoMensaje2[0].append("</i></b>");
    campoMensaje2[1].prepend("Mensaje: <b>");
    campoMensaje2[1].append("</b>");
    campoMensaje2[2].prepend("NA Origen: <b>");
    campoMensaje2[2].append("</b>");
    campoMensaje2[3].prepend("NA Destino: <b>");
    campoMensaje2[3].append("</b>");
    campoMensaje2[4].prepend("Temperatura: <b>");
    campoMensaje2[4].append(" ºC</b>");
    campoMensaje2[5].prepend("Batería: <b>");
    campoMensaje2[5].append(" %</b>");
    campoMensaje2[6].prepend("DC Usado: <b>");
    campoMensaje2[6].append(" %</b>");
    campoMensaje2[7].prepend("Nº Saltos: <b>");
    campoMensaje2[7].append("</b>");

    mensajeHTML = campoMensaje2.join(", ");

    mensajeHTML.prepend("<font color=\"GREEN\">");
    mensajeHTML.append("</font>");

}
else if(tipoMensaje.contains("ALARM",Qt::CaseInsensitive))
{
    error_mensaje = 0;

    QString tipoAlarma = campoMensaje2[4];

    campoMensaje2[0].prepend("<b><i>");
    campoMensaje2[0].append("</i></b>");
    campoMensaje2[1].prepend("Mensaje: <b>");
    campoMensaje2[1].append("</b>");
    campoMensaje2[2].prepend("NA Origen: <b>");
    campoMensaje2[2].append("</b>");
    campoMensaje2[3].prepend("NA Destino: <b>");
    campoMensaje2[3].append("</b>");
    campoMensaje2[4].prepend("Código Alarma: <b>");
    campoMensaje2[4].append("</b>");

    if (tipoAlarma.contains("CAIDA",Qt::CaseInsensitive))
    {
        campoMensaje2[5].prepend("Radioenlace: <b>");
        campoMensaje2[6].append("</b>");
    }

    mensajeHTML = campoMensaje2.join(", ");

    mensajeHTML.prepend("<font color=\"RED\">");
    mensajeHTML.append("</font>");

}
else if(tipoMensaje.contains("ENLACE",Qt::CaseInsensitive))
{
    error_mensaje = 0;

    campoMensaje2[0].prepend("<b><i>");
    campoMensaje2[0].append("</i></b>");
    campoMensaje2[1].prepend("Mensaje: <b>");
    campoMensaje2[1].append("</b>");
    campoMensaje2[2].prepend("NA Origen: <b>");
    campoMensaje2[2].append("</b>");
    campoMensaje2[3].prepend("NA Destino: <b>");
    campoMensaje2[3].append("</b>");
    campoMensaje2[4].prepend("RSSI:<b> -");
```

```
        campoMensaje2[4].append(" dBm</b>");
        campoMensaje2[5].prepend("Nº reintentos: <b>");
        campoMensaje2[5].append("</b>");
        campoMensaje2[6].prepend("Radioenlace: <b>");
        campoMensaje2[7].append("</b>");

        mensajeHTML = campoMensaje2.join(", ");

        mensajeHTML.prepend("<font color=\"BLACK\">");
        mensajeHTML.append("</font>");

    }
    else // Esto seria un mensaje con campo erroneo
    {
        error_mensaje = 1;

        mensajeHTML.prepend("<font color=\"GRAY\">");
        mensajeHTML.append("</font>");
    }

    if (error_mensaje == 0)
    {

        // Opcionalmente, se graban los archivos de log
        // Esto se configura en la Configuración de Waspmonitor.

        if (waspFlags->saveFileHTML != 0)
        {
            escribirLogHTML(waspFileNames->logHTMLName, mensajeHTML);
        }

        if (waspFlags->saveFileCSV != 0)
        {
            escribirLogCSV(waspFileNames->logCSVName, mensajeStr);
        }

        // Se añade el mensaje como línea nueva en la ventana de log

        ui->page3->showMonitor::logger->plainTextEdit->appendHtml(mensajeHTML);

        QTextCursor cursor(ui->page3->showMonitor::logger->plainTextEdit->textCursor());
        cursor.movePosition(QTextCursor::End);
        ui->page3->showMonitor::logger->plainTextEdit->setTextCursor(cursor);

    }

    }
    // En caso contrario, ignorar la trama inválida
}

/* ------------------------------------------------------------------------
*   Escribe el archivo de log en HTML
* ------------------------------------------------------------------------*/

void WaspMonitor::escribirLogHTML(QString nombre, QString linea)
{
    QFile logHTMLFile(waspFileNames->logHTMLName);

    if(!logHTMLFile.open(QFile::ReadWrite | QFile::Text | QIODevice::Append))
    {
        QMessageBox warnCannotOpenHTML;
        warnCannotOpenHTML.warning(this,"Atención","Se ha producido un error al intentar abrir el archivo de log HTML.");
        return;
    }

    QTextStream aux(&logHTMLFile);

    if(waspFlags->firstTimeOpenHTML!=0)
    {
        linea.prepend("<h1>Log de mensajes recibidos desde los Waspmotes</h1>");
        waspFlags->firstTimeOpenHTML=0;
    }
    linea.append("<br>");
    aux << linea;

    logHTMLFile.flush();
    logHTMLFile.close();
}

/* ------------------------------------------------------------------------
*   Escribe el archivo de log en CSV
* ------------------------------------------------------------------------*/

void WaspMonitor::escribirLogCSV(QString nombre, QString linea)
```

```
{
    QFile logCSVFile(waspFileNames->logCSVName);

    if(!logCSVFile.open(QFile::ReadWrite | QFile::Text | QIODevice::Append))
    {
        QMessageBox warnCannotOpenCSV;
        warnCannotOpenCSV.warning(this,"Atención","Se ha producido un error al intentar abrir el archivo de log CSV.");
        return;
    }

    QTextStream aux(&logCSVFile);
    aux << linea;

    logCSVFile.flush();
    logCSVFile.close();
}

/* ------------------------------------------------------------------------
*   Esta función cambia el formato de la fecha para
*   que no den errores las funciones de representación de gráficas
* ------------------------------------------------------------------------*/

void WaspMonitor::cambiarFormatoFecha()
{
    if(mensajeStr.contains("Sunday",Qt::CaseInsensitive))
    {
        mensajeStr.replace("Sunday,","Domingo.");
    }
    else if(mensajeStr.contains("Monday",Qt::CaseInsensitive))
    {
        mensajeStr.replace("Monday,","Lunes.");
    }
    else if(mensajeStr.contains("Tuesday",Qt::CaseInsensitive))
    {
        mensajeStr.replace("Tuesday,","Martes.");
    }
    else if(mensajeStr.contains("Wednesday",Qt::CaseInsensitive))
    {
        mensajeStr.replace("Wednesday,","Miercoles.");
    }
    else if(mensajeStr.contains("Thursday",Qt::CaseInsensitive))
    {
        mensajeStr.replace("Thursday,","Jueves.");
    }
    else if(mensajeStr.contains("Friday",Qt::CaseInsensitive))
    {
        mensajeStr.replace("Friday,","Viernes.");
    }
    else if(mensajeStr.contains("Saturday",Qt::CaseInsensitive))
    {
        mensajeStr.replace("Saturday,","Sabado.");
    }
}

/* ------------------------------------------------------------------------
*   Esta función procesa el mensaje válido recibido en formato QString y hace lo siguiente:
*   1) Crea una QStringList con los campos del mensaje separados por las comas.
*   2) Traduce los días de la semana al castellano.
* ------------------------------------------------------------------------*/

void WaspMonitor::procesarMensaje()
{
    // Quitamos los caracteres '#' de la trama y le ponemos '\n' al final
    mensajeStr.remove(QChar('#'));
    mensajeStr.append(QChar('\n'));

    // Cambiamos el formato del mensaje para que no de problemas en las funciones siguientes
    cambiarFormatoFecha();
    // Separamos los campos del mensaje útil separados por comas para que sean procesados
    campoMensaje = mensajeStr.split(',');
}

/* ------------------------------------------------------------------------
*   Actualiza los contadores de estado de los nodos.
*   Si se recibe un KEEPALIVE de un determinado nodo, se reinicia el contador
* ------------------------------------------------------------------------*/

void WaspMonitor::actualizarContadoresNodos()
{
    // Si hay más de un elemento en la QStringList, suponemos
    // que es un mensaje válido
    if( (campoMensaje.size()) > 1)
    {
        if (campoMensaje[1]=="KEEPALIVE")
```

```
        {
            if(campoMensaje[2]=="0041")
            {
                time_0041.restart();
            }
            else if(campoMensaje[2]=="0042")
            {
                time_0042.restart();
            }
            else if(campoMensaje[2]=="0043")
            {
                time_0043.restart();
            }
            else if(campoMensaje[2]=="0044")
            {
                time_0044.restart();
            }
        }
    }
    // En caso contrario, la trama recibida es corrupta y se ignora
}

/* ------------------------------------------------------------------------
*   Actualiza los contadores de estado de los enlaces.
*   Si se recibe un ESTADO_ENLACE de un determinado enlace, se reinicia el contador
* ------------------------------------------------------------------------*/

void WaspMonitor::actualizarContadoresEnlaces()
{
    // Si hay más de un elemento en la QStringList, suponemos
    // que es un mensaje válido
    if( (campoMensaje.size()) > 1)
    {
        QString tipoMensaje = campoMensaje[1];

        if (tipoMensaje.contains("ENLACE"))
        {
            QString RE_Origen = campoMensaje[6];
            QString RE_Destino = campoMensaje[7];

            if(RE_Origen.contains("0041") && RE_Destino.contains("0042"))
            {
                time_0041_0042.restart();
            }
            else if(RE_Origen.contains("0042") && RE_Destino.contains("0043"))
            {
                time_0042_0043.restart();
            }
            else if(RE_Origen.contains("0043") && RE_Destino.contains("0000"))
            {
                time_0043_0000.restart();
            }
            else if(RE_Origen.contains("0044") && RE_Destino.contains("0000"))
            {
                time_0044_0000.restart();
            }
            else if(RE_Origen.contains("0041") && RE_Destino.contains("0043"))
            {
                time_0041_0043.restart();
            }
            else if(RE_Origen.contains("0041") && RE_Destino.contains("0000"))
            {
                time_0041_0000.restart();
            }
            else if(RE_Origen.contains("0042") && RE_Destino.contains("0000"))
            {
                time_0042_0000.restart();
            }
        }
    }
    // En caso contrario, la trama recibida es corrupta y se ignora
}

/* ------------------------------------------------------------------------
*   Actualiza las gráficas de monitorización, las etiquetas de temperatura,
*   los iconos de estado y de alarma y los paneles de alarma a
*   partir del mensaje nuevo de datos llegado desde hiloSniffer
* ------------------------------------------------------------------------*/

void WaspMonitor::actualizarGraficas()
{
    // Si hay más de un elemento en la QStringList, suponemos
    // que es un mensaje válido
    if( (campoMensaje.size()) > 1)
```

```
    {
        // Actualizamos las etiquetas de temperatura del mapa
        if(campoMensaje[1]=="KEEPALIVE")
        {
            actualizarEtiquetas(campoMensaje[2],campoMensaje[4]);
        }

        // Actualizamos los Iconos de estado del mapa
        actualizarIconosEstado(campoMensaje);

        // Actualizamos las barras de medida
        actualizarBarras(campoMensaje);

        // Actualizamos las gráficas de históricos
        actualizarHistoricos(campoMensaje);

        // Actualizamos los paneles de alarma
        actualizarPanelesAlarma(campoMensaje);

        // Actualizamos los iconos de alarma del mapa
        actualizarIconosAlarma(campoMensaje);

    }
    // En caso contrario, la trama recibida es corrupta y se ignora

}

/*-----------------------------------------------------------------
 * Actualiza las etiquetas de temperatura del mapa.
 *-----------------------------------------------------------------*/

void WaspMonitor::actualizarEtiquetas(QString NAnodo, QString tempGrados)
{
    tempGrados.append(" ºC");

    if(NAnodo == "0041")
    {
        ui->page3->tag0041->clear();
        ui->page3->tag0041->insert(tempGrados);
    }
    else if(NAnodo == "0042")
    {
        ui->page3->tag0042->clear();
        ui->page3->tag0042->insert(tempGrados);
    }
    else if(NAnodo == "0043")
    {
        ui->page3->tag0043->clear();
        ui->page3->tag0043->insert(tempGrados);
    }
    else if(NAnodo == "0000")
    {
        ui->page3->tag0000->clear();
        ui->page3->tag0000->insert(tempGrados);
    }
    else if(NAnodo == "0044")
    {
        ui->page3->tag0044->clear();
        ui->page3->tag0044->insert(tempGrados);
    }
}

/*-----------------------------------------------------------------
 * Actualiza los gráficos de barras.
 *-----------------------------------------------------------------*/

void WaspMonitor::actualizarBarras(QStringList campoMensaje)
{
    if(campoMensaje[1]=="KEEPALIVE")
    {
        QString temp = campoMensaje[4];
        QString bat = campoMensaje[5];
        QString dc = campoMensaje[6];

        // Nodo Ctra. Utrera
        if(campoMensaje[2] == "0041")
        {
            ui->page3->showMonitor::grafBar0041->actualizaBarraTemp(temp.toDouble());
            ui->page3->showMonitor::grafBar0041->actualizaBarraBatt(bat.toDouble());
            ui->page3->showMonitor::grafBar0041->actualizaBarraDC(dc.toDouble());
        }
        // Nodo C. Maribáñez
        else if(campoMensaje[2] == "0042")
        {
```

```
        ui->page3->showMonitor::grafBar0042->actualizaBarraTemp(temp.toDouble());
        ui->page3->showMonitor::grafBar0042->actualizaBarraBatt(bat.toDouble());
        ui->page3->showMonitor::grafBar0042->actualizaBarraDC(dc.toDouble());
    }
    // Nodo San Fernando
    else if(campoMensaje[2] == "0043")
    {
        ui->page3->showMonitor::grafBar0043->actualizaBarraTemp(temp.toDouble());
        ui->page3->showMonitor::grafBar0043->actualizaBarraBatt(bat.toDouble());
        ui->page3->showMonitor::grafBar0043->actualizaBarraDC(dc.toDouble());
    }
    // Nodo Oficina
    else if(campoMensaje[2] == "0000")
    {
        ui->page3->showMonitor::grafBar0000->actualizaBarraTemp(temp.toDouble());
        ui->page3->showMonitor::grafBar0000->actualizaBarraBatt(bat.toDouble());
        ui->page3->showMonitor::grafBar0000->actualizaBarraDC(dc.toDouble());
    }
    // Nodo Alcantarillas
    else if(campoMensaje[2] == "0044")
    {
        ui->page3->showMonitor::grafBar0044->actualizaBarraTemp(temp.toDouble());
        ui->page3->showMonitor::grafBar0044->actualizaBarraBatt(bat.toDouble());
        ui->page3->showMonitor::grafBar0044->actualizaBarraDC(dc.toDouble());
    }
}
else if(campoMensaje[1]=="ESTADO_ENLACE")
{
    QString rssi = campoMensaje[4];
    rssi.prepend("-"); // Se añade un signo menos para poder representarlo,
                       // ya que el RSSI recibido desde waspsniffer es en valor absoluto

    /** El Throughput = 1 trama recibida / (1 trama txtida + n tramas reenviadas)

        El throughput será del 100% si por cada trama recibida se emplea 1 envio y 0 reenvios
      */

    QString reenvios = campoMensaje[5];

    double throughPut = 100.0 * ( 1 / ( 1 + reenvios.toDouble()));

    QString RE_Origen = campoMensaje[6];
    QString RE_Destino = campoMensaje[7];

    /** Radioenlaces Primarios **/
    // Radioenlace Ctra. Utrera - C. Maribáñez
    if( (RE_Origen.contains("0041")) && (RE_Destino.contains("0042")) )
    {
        ui->page3->showMonitor::grafBar0041_0042->actualizaBarraRSSI(rssi.toDouble());
        ui->page3->showMonitor::grafBar0041_0042->actualizaBarraThroughPut(throughPut);
    }
    // Radioenlace C. Maribáñez - San Fernando
    else if( (RE_Origen.contains("0042")) && (RE_Destino.contains("0043")) )
    {
        ui->page3->showMonitor::grafBar0042_0043->actualizaBarraRSSI(rssi.toDouble());
        ui->page3->showMonitor::grafBar0042_0043->actualizaBarraThroughPut(throughPut);
    }
    // Radioenlace San Fernando - Oficina
    else if( (RE_Origen.contains("0043")) && (RE_Destino.contains("0000")) )
    {
        ui->page3->showMonitor::grafBar0043_0000->actualizaBarraRSSI(rssi.toDouble());
        ui->page3->showMonitor::grafBar0043_0000->actualizaBarraThroughPut(throughPut);
    }
    // Radioenlace Alcantarillas - Oficina
    else if( (RE_Origen.contains("0044")) && (RE_Destino.contains("0000")) )
    {
        ui->page3->showMonitor::grafBar0044_0000->actualizaBarraRSSI(rssi.toDouble());
        ui->page3->showMonitor::grafBar0044_0000->actualizaBarraThroughPut(throughPut);
    }
    /** Radioenlaces Secundarios **/
    // Radioenlace Ctra. Utrera - San Fernando
    else if( (RE_Origen.contains("0041")) && (RE_Destino.contains("0043")) )
    {
        ui->page3->showMonitor::grafBar0041_0043->actualizaBarraRSSI(rssi.toDouble());
        ui->page3->showMonitor::grafBar0041_0043->actualizaBarraThroughPut(throughPut);
    }
    // Radioenlace Ctra. Utrera - Oficina
    else if( (RE_Origen.contains("0041")) && (RE_Destino.contains("0000")) )
    {
        ui->page3->showMonitor::grafBar0041_0000->actualizaBarraRSSI(rssi.toDouble());
        ui->page3->showMonitor::grafBar0041_0000->actualizaBarraThroughPut(throughPut);
    }
    // Radioenlace C. Maribáñez - Oficina
    else if( (RE_Origen.contains("0042")) && (RE_Destino.contains("0000")) )
```

```
        {
            ui->page3->showMonitor::grafBar0042_0000->actualizaBarraRSSI(rssi.toDouble());
            ui->page3->showMonitor::grafBar0042_0000->actualizaBarraThroughPut(throughPut);
        }
    }
}

/*-----------------------------------------------------------------
 * Esta función actualiza los históricos.
 *-----------------------------------------------------------------*/

void WaspMonitor::actualizarHistoricos(QStringList campoMensaje)
{
    QString tipoMensaje = campoMensaje[1];
    QString tipoAlarma = campoMensaje[4];

    /** Si llega un mensaje de ESTADO DE ENLACE normal -> Actualizar Temperatura **/
    if(tipoMensaje.contains("KEEPALIVE",Qt::CaseInsensitive))
    {
        QString temp = campoMensaje[4];

        // Nodo Ctra. Utrera
        if(campoMensaje[2] == "0041")
        {
            ui->page3->showMonitor::hist0041->actualizarGrafica(temp.toDouble());
        }
        // Nodo C. Maribáñez
        else if(campoMensaje[2] == "0042")
        {
            ui->page3->showMonitor::hist0042->actualizarGrafica(temp.toDouble());
        }
        // Nodo San Fernando
        else if(campoMensaje[2] == "0043")
        {
            ui->page3->showMonitor::hist0043->actualizarGrafica(temp.toDouble());
        }
        // Nodo Oficina
        else if(campoMensaje[2] == "0000")
        {
            ui->page3->showMonitor::hist0000->actualizarGrafica(temp.toDouble());
        }
        // Nodo Alcantarillas
        else if(campoMensaje[2] == "0044")
        {
            ui->page3->showMonitor::hist0044->actualizarGrafica(temp.toDouble());
        }
    } /** Si llega un mensaje de ESTADO DE ENLACE normal -> Actualizar valor RSSI **/
    else if(tipoMensaje.contains("ESTADO",Qt::CaseInsensitive))
    {
        QString rssi = campoMensaje[4];
        rssi.prepend("-"); // Se añade un signo menos para poder representarlo,
                           // ya que el RSSI recibido desde waspsniffer es en valor absoluto

        QString RE_Origen = campoMensaje[6];
        QString RE_Destino = campoMensaje[7];

        // Radioenlace Ctra. Utrera - C. Maribáñez
        if( (RE_Origen.contains("0041")) && (RE_Destino.contains("0042")) )
        {
            ui->page3->showMonitor::hist0041_0042->actualizarGrafica(rssi.toDouble());
        }
        // Radioenlace C. Maribáñez - San Fernando
        else if( (RE_Origen.contains("0042")) && (RE_Destino.contains("0043")) )
        {
            ui->page3->showMonitor::hist0042_0043->actualizarGrafica(rssi.toDouble());
        }
        // Radioenlace San Fernando - Oficina
        else if( (RE_Origen.contains("0043")) && (RE_Destino.contains("0000")) )
        {
            ui->page3->showMonitor::hist0043_0000->actualizarGrafica(rssi.toDouble());
        }
        // Radioenlace Alcantarillas - Oficina
        else if( (RE_Origen.contains("0044")) && (RE_Destino.contains("0000")) )
        {
            ui->page3->showMonitor::hist0044_0000->actualizarGrafica(rssi.toDouble());
        }
        // Radioenlace Ctra. Utrera - San Fernando
        else if( (RE_Origen.contains("0041")) && (RE_Destino.contains("0043")) )
        {
            ui->page3->showMonitor::hist0041_0043->actualizarGrafica(rssi.toDouble());
        }
        // Radioenlace Ctra. Utrera - Oficina
        else if( (RE_Origen.contains("0041")) && (RE_Destino.contains("0000")) )
        {
```

```
            ui->page3->showMonitor::hist0041_0000->actualizarGrafica(rssi.toDouble());
        }
        // Radioenlace C. Maribáñez - Oficina
        else if( (RE_Origen.contains("0042")) && (RE_Destino.contains("0000")) )
        {
            ui->page3->showMonitor::hist0042_0000->actualizarGrafica(rssi.toDouble());
        }
    }
    /** Si hay una ALARMA por CAIDA DE ENLACE -> Poner la gráfica de RSSI a -112.0 dBm**/
    else if( tipoMensaje.contains("ALARMA",Qt::CaseInsensitive) &&
             tipoAlarma.contains("CAIDA",Qt::CaseInsensitive))
    {

        QString RE_ALM_Origen = campoMensaje[5];
        QString RE_ALM_Destino = campoMensaje[6];

        // Radioenlace Ctra. Utrera - C. Maribáñez
        if( (RE_ALM_Origen.contains("0041")) && (RE_ALM_Destino.contains("0042")) )
        {
            ui->page3->showMonitor::hist0041_0042->actualizarGrafica(-112.0);
        }
        // Radioenlace C. Maribáñez - San Fernando
        else if( (RE_ALM_Origen.contains("0042")) && (RE_ALM_Destino.contains("0043")) )
        {
            ui->page3->showMonitor::hist0042_0043->actualizarGrafica(-112.0);
        }
        // Radioenlace San Fernando - Oficina
        else if( (RE_ALM_Origen.contains("0043")) && (RE_ALM_Destino.contains("0000")) )
        {
            ui->page3->showMonitor::hist0043_0000->actualizarGrafica(-112.0);
        }
        // Radioenlace San Fernando - Oficina
        else if( (RE_ALM_Origen.contains("0043")) && (RE_ALM_Destino.contains("0000")) )
        {
            ui->page3->showMonitor::hist0043_0000->actualizarGrafica(-112.0);
        }
        // Radioenlace Alcantarillas - Oficina
        else if( (RE_ALM_Origen.contains("0044")) && (RE_ALM_Destino.contains("0000")) )
        {
            ui->page3->showMonitor::hist0044_0000->actualizarGrafica(-112.0);
        }
        // Radioenlace Ctra. Utrera - San Fernando
        else if( (RE_ALM_Origen.contains("0041")) && (RE_ALM_Destino.contains("0043")) )
        {
            ui->page3->showMonitor::hist0041_0043->actualizarGrafica(-112.0);
        }
        // Radioenlace Ctra. Utrera - Oficina
        else if( (RE_ALM_Origen.contains("0041")) && (RE_ALM_Destino.contains("0000")) )
        {
            ui->page3->showMonitor::hist0041_0000->actualizarGrafica(-112.0);
        }
        // Radioenlace C. Maribáñez - Oficina
        else if( (RE_ALM_Origen.contains("0042")) && (RE_ALM_Destino.contains("0000")) )
        {
            ui->page3->showMonitor::hist0042_0000->actualizarGrafica(-112.0);
        }
    }
}

/*-------------------------------------------------------------
 * Actualiza el color de las etiquetas de temperatura
 * Blanco: Nodo activado (monitorización activada)
 * Amarillo: Nodo desactivado (monitorización desactivada)
 * Verde: Mensaje recibido
 * Rojo: Nodo caido
 *-------------------------------------------------------------*/

void WaspMonitor::actualizarIconosEstado(QStringList campoMensaje)
{
    QString tipoMensaje = campoMensaje[1];
    QString tipoAlarma = campoMensaje[4];

    /** Si es un KEEPALIVE -> Poner etiqueta del nodo en verde **/
    if(tipoMensaje.contains("KEEPALIVE",Qt::CaseInsensitive))
    {
        if(campoMensaje[2]=="0041")
        {
            ui->page3->showMonitor::paleta0041.setColor(QPalette::Base, QColor(204,255,204)); // Verde claro
            ui->page3->showMonitor::tag0041->setPalette(ui->page3->showMonitor::paleta0041);
        }
        else if(campoMensaje[2]=="0042")
        {
            ui->page3->showMonitor::paleta0042.setColor(QPalette::Base, QColor(204,255,204)); // Verde claro
            ui->page3->showMonitor::tag0042->setPalette(ui->page3->showMonitor::paleta0042);
```

```
        }
        else if(campoMensaje[2]=="0043")
        {
            ui->page3->showMonitor::paleta0043.setColor(QPalette::Base, QColor(204,255,204)); // Verde claro
            ui->page3->showMonitor::tag0043->setPalette(ui->page3->showMonitor::paleta0043);
        }
        else if(campoMensaje[2]=="0000")
        {
            ui->page3->showMonitor::paleta0000.setColor(QPalette::Base, QColor(204,255,204)); // Verde claro
            ui->page3->showMonitor::tag0000->setPalette(ui->page3->showMonitor::paleta0000);
        }
        else if(campoMensaje[2]=="0044")
        {
            ui->page3->showMonitor::paleta0044.setColor(QPalette::Base, QColor(204,255,204)); // Verde claro
            ui->page3->showMonitor::tag0044->setPalette(ui->page3->showMonitor::paleta0044);
        }
    }
    /** Si es una ALARMA por CAIDA-> Poner LED de NADestino en Rojo **/
    /** NOTA: No se añaden los casos en los que el Destino es 0000, porque
              en ese caso directamente no llegaría ningún mensaje al nodo central... */
    else if(tipoMensaje.contains("ALARMA",Qt::CaseInsensitive) &&
            tipoAlarma.contains("CAIDA",Qt::CaseInsensitive))
    {
        QString RE_ALM_Destino = campoMensaje[6];

        if(RE_ALM_Destino.contains("0042"))
        {
            ui->page3->showMonitor::paleta0042.setColor(QPalette::Base, QColor(255,100,61)); // Rojo claro
            ui->page3->showMonitor::tag0042->setPalette(ui->page3->showMonitor::paleta0042);
        }
        else if(RE_ALM_Destino.contains("0043"))
        {
            ui->page3->showMonitor::paleta0043.setColor(QPalette::Base, QColor(255,100,61)); // Rojo claro
            ui->page3->showMonitor::tag0043->setPalette(ui->page3->showMonitor::paleta0043);
        }
    }
    /** Si es un mensaje de ESTADO_ENLACE, poner Led Radio en Verde */
    else if(tipoMensaje.contains("ENLACE",Qt::CaseInsensitive))
    {
        QString RE_Origen = campoMensaje[6];
        QString RE_Destino = campoMensaje[7];

        // Radioenlaces primarios
        if( (RE_Origen.contains("0041")) && (RE_Destino.contains("0042")) )
        {
            ui->page3->showMonitor::Led0041_0042.load(":/green_light_16.png");
            ui->page3->showMonitor::tagLed0041_0042->setPixmap(ui->page3->showMonitor::Led0041_0042);
        }
        else if( (RE_Origen.contains("0042")) && (RE_Destino.contains("0043")) )
        {
            ui->page3->showMonitor::Led0042_0043.load(":/green_light_16.png");
            ui->page3->showMonitor::tagLed0042_0043->setPixmap(ui->page3->showMonitor::Led0042_0043);
        }
        else if( (RE_Origen.contains("0043")) && (RE_Destino.contains("0000")) )
        {
            ui->page3->showMonitor::Led0043_0000.load(":/green_light_16.png");
            ui->page3->showMonitor::tagLed0043_0000->setPixmap(ui->page3->showMonitor::Led0043_0000);
        }
        else if( (RE_Origen.contains("0044")) && (RE_Destino.contains("0000")) )
        {
            ui->page3->showMonitor::Led0044_0000.load(":/green_light_16.png");
            ui->page3->showMonitor::tagLed0044_0000->setPixmap(ui->page3->showMonitor::Led0044_0000);
        }
        // Radioenlaces secundarios
        else if( (RE_Origen.contains("0041")) && (RE_Destino.contains("0043")) )
        {
            ui->page3->showMonitor::Led0041_0043.load(":/green_light_16.png");
            ui->page3->showMonitor::tagLed0041_0043->setPixmap(ui->page3->showMonitor::Led0041_0043);
        }
        else if( (RE_Origen.contains("0041")) && (RE_Destino.contains("0000")) )
        {
            ui->page3->showMonitor::Led0041_0000.load(":/green_light_16.png");
            ui->page3->showMonitor::tagLed0041_0000->setPixmap(ui->page3->showMonitor::Led0041_0000);
        }
        else if( (RE_Origen.contains("0042")) && (RE_Destino.contains("0000")) )
        {
            ui->page3->showMonitor::Led0042_0000.load(":/green_light_16.png");
            ui->page3->showMonitor::tagLed0042_0000->setPixmap(ui->page3->showMonitor::Led0042_0000);
        }
    }
}

/*-------------------------------------------------------------
 * Actualiza los Paneles de Alarma
```

```
 *-------------------------------------------------------------*/

void WaspMonitor::actualizarPanelesAlarma(QStringList campoMensaje)
{
    QString tipoAlarma = campoMensaje[4];

    if(campoMensaje[1]=="ALARMA")
    {
        /* ---- Muestra alarma por CAIDA de ENLACE en los paneles  ---- */
        if(tipoAlarma.contains("CAIDA",Qt::CaseInsensitive))
        {
            /** Lo que determina si un nodo ha caido ese el NA DESTINO **/

            QString RE_ALM_Origen = campoMensaje[5];
            QString RE_ALM_Destino = campoMensaje[6];

            // Caida del radioenlace 0041-0042
            if((RE_ALM_Origen.contains("0041")) && (RE_ALM_Destino.contains("0042")))
            {
                ui->page3->showMonitor::panel0041_0042->labelIconEnlaceCaido->setPixmap(ui->page3->showMonitor::panel0041_0042-
>iconoAlarmaON);
                // Implica la caída del nodo 0042 -> Activar icono alarma ON en el panel de alarmas
                ui->page3->showMonitor::panel0042->labelIconNodoCaido->setPixmap(ui->page3->showMonitor::panel0042-
>iconoAlarmaON);
                // y también la caída de los radioenlaces dependientes del nodo 0042
                ui->page3->showMonitor::panel0042_0000->labelIconEnlaceCaido->setPixmap(ui->page3->showMonitor::panel0042_0000-
>iconoAlarmaON);
                ui->page3->showMonitor::panel0042_0043->labelIconEnlaceCaido->setPixmap(ui->page3->showMonitor::panel0042_0043-
>iconoAlarmaON);

            }
            // Caida del radioenlace 0042-0043
            else if((RE_ALM_Origen.contains("0042")) && (RE_ALM_Destino.contains("0043")))
            {
                ui->page3->showMonitor::panel0042_0043->labelIconEnlaceCaido->setPixmap(ui->page3->showMonitor::panel0042_0043-
>iconoAlarmaON);
                // Implica la caida del nodo 0043 -> Activar icono alarma ON en el panel de alarmas
                ui->page3->showMonitor::panel0043->labelIconNodoCaido->setPixmap(ui->page3->showMonitor::panel0043-
>iconoAlarmaON);
                ui->page3->showMonitor::panel0043_0000->labelIconEnlaceCaido->setPixmap(ui->page3->showMonitor::panel0043_0000-
>iconoAlarmaON);

            }
            // Caida del radioenlace 0041-0043
            else if((RE_ALM_Origen.contains("0041")) && (RE_ALM_Destino.contains("0043")))
            {
                ui->page3->showMonitor::panel0041_0043->labelIconEnlaceCaido->setPixmap(ui->page3->showMonitor::panel0041_0043-
>iconoAlarmaON);
                // Implica la caida del nodo 0043 -> Activar icono alarma ON en el panel de alarmas
                ui->page3->showMonitor::panel0043->labelIconNodoCaido->setPixmap(ui->page3->showMonitor::panel0043-
>iconoAlarmaON);
                ui->page3->showMonitor::panel0043_0000->labelIconEnlaceCaido->setPixmap(ui->page3->showMonitor::panel0043_0000-
>iconoAlarmaON);

            }
        /* ---- Muestra alarma por DUTY CYCLE ALTO en los paneles  ---- */
        }
        else if(tipoAlarma.contains("DUTY",Qt::CaseInsensitive))
        {
            if(campoMensaje[2]=="0041")
            {
                ui->page3->showMonitor::panel0041->labelIconDCAlto->setPixmap(ui->page3->showMonitor::panel0041->iconoAlarmaON);
            }
            else if(campoMensaje[2]=="0042")
            {
                ui->page3->showMonitor::panel0042->labelIconDCAlto->setPixmap(ui->page3->showMonitor::panel0042->iconoAlarmaON);
            }
            else if(campoMensaje[2]=="0043")
            {
                ui->page3->showMonitor::panel0043->labelIconDCAlto->setPixmap(ui->page3->showMonitor::panel0043->iconoAlarmaON);
            }
            else if(campoMensaje[2]=="0000")
            {
                ui->page3->showMonitor::panel0000->labelIconDCAlto->setPixmap(ui->page3->showMonitor::panel0000->iconoAlarmaON);
            }
            else if(campoMensaje[2]=="0044")
            {
                ui->page3->showMonitor::panel0044->labelIconDCAlto->setPixmap(ui->page3->showMonitor::panel0044->iconoAlarmaON);
            }
        }
        /* ---- Muestra alarma por BATERIA BAJA en los paneles  ---- */
        else if(tipoAlarma.contains("BATERIA",Qt::CaseInsensitive))
        {
            if(campoMensaje[2]=="0041")
```

```
            {
                ui->page3->showMonitor::panel0041->labelIconBatBaja->setPixmap(ui->page3->showMonitor::panel0041-
>iconoAlarmaON);
            }
            else if(campoMensaje[2]=="0042")
            {
                ui->page3->showMonitor::panel0042->labelIconBatBaja->setPixmap(ui->page3->showMonitor::panel0042-
>iconoAlarmaON);
            }
            else if(campoMensaje[2]=="0043")
            {
                ui->page3->showMonitor::panel0043->labelIconBatBaja->setPixmap(ui->page3->showMonitor::panel0043-
>iconoAlarmaON);
            }
            else if(campoMensaje[2]=="0000")
            {
                ui->page3->showMonitor::panel0000->labelIconBatBaja->setPixmap(ui->page3->showMonitor::panel0000-
>iconoAlarmaON);
            }
            else if(campoMensaje[2]=="0044")
            {
                ui->page3->showMonitor::panel0044->labelIconBatBaja->setPixmap(ui->page3->showMonitor::panel0044-
>iconoAlarmaON);
            }
        }
        /* ---- Muestra alarma por TEMPERATURA ALTA en los paneles  ---- */
        else if(tipoAlarma.contains("TEMP",Qt::CaseInsensitive))
        {
            if(campoMensaje[2]=="0041")
            {
                ui->page3->showMonitor::panel0041->labelIconTempAlta->setPixmap(ui->page3->showMonitor::panel0041-
>iconoAlarmaON);
            }
            else if(campoMensaje[2]=="0042")
            {
                ui->page3->showMonitor::panel0042->labelIconTempAlta->setPixmap(ui->page3->showMonitor::panel0042-
>iconoAlarmaON);
            }
            else if(campoMensaje[2]=="0043")
            {
                ui->page3->showMonitor::panel0043->labelIconTempAlta->setPixmap(ui->page3->showMonitor::panel0043-
>iconoAlarmaON);
            }
            else if(campoMensaje[2]=="0000")
            {
                ui->page3->showMonitor::panel0000->labelIconTempAlta->setPixmap(ui->page3->showMonitor::panel0000-
>iconoAlarmaON);
            }
            else if(campoMensaje[2]=="0044")
            {
                ui->page3->showMonitor::panel0044->labelIconTempAlta->setPixmap(ui->page3->showMonitor::panel0044-
>iconoAlarmaON);
            }
        }
    }
}

/*-------------------------------------------------------------
 * Actualiza los Iconos de alarma del mapa (Temp.Alta, Bat Baja,
 * DC Alto, Caida Enlace y TAMBIEN los LEDS de los radioenlaces.
 * Los pone en ROJO.
 * La función que se encarga de ponerlos en verde es
 * "actualizarIconosEstado"
 *-------------------------------------------------------------*/

void WaspMonitor::actualizarIconosAlarma(QStringList campoMensaje)
{
    QString tipoMensaje = campoMensaje[1];
    QString tipoAlarma = campoMensaje[4];

    if(tipoMensaje.contains("ALARMA",Qt::CaseInsensitive))
    {
        /** Si es una ALARMA de TEMP_ALTA -> Mostrar icono "termo" en NAOrigen */
        if(tipoAlarma.contains("TEMP",Qt::CaseInsensitive))
        {
            if(campoMensaje[2]=="0041")
            {
                ui->page3->showMonitor::tagTempAlta0041->show();
            }
            else if(campoMensaje[2]=="0042")
            {
                ui->page3->showMonitor::tagTempAlta0042->show();
            }
            else if(campoMensaje[2]=="0043")
```

```
    {
        ui->page3->showMonitor::tagTempAlta0043->show();
    }
    else if(campoMensaje[2]=="0000")
    {
        ui->page3->showMonitor::tagTempAlta0000->show();
    }
    else if(campoMensaje[2]=="0044")
    {
        ui->page3->showMonitor::tagTempAlta0044->show();
    }
}
/** Si es una ALARMA de BATERIA_BAJA -> Mostrar icono "bat_low" en NAOrigen */
else if(tipoAlarma.contains("BATERIA",Qt::CaseInsensitive))
{
    if(campoMensaje[2]=="0041")
    {
        ui->page3->showMonitor::tagIconBatBaja0041->show();
    }
    else if(campoMensaje[2]=="0042")
    {
        ui->page3->showMonitor::tagIconBatBaja0042->show();
    }
    else if(campoMensaje[2]=="0043")
    {
        ui->page3->showMonitor::tagIconBatBaja0043->show();
    }
    else if(campoMensaje[2]=="0000")
    {
        ui->page3->showMonitor::tagIconBatBaja0000->show();
    }
    else if(campoMensaje[2]=="0044")
    {
        ui->page3->showMonitor::tagIconBatBaja0044->show();
    }
}
/** Si es una ALARMA de DUTY_CYCLE -> Mostrar icono "dc" en NAOrigen */
else if(tipoAlarma.contains("DUTY",Qt::CaseInsensitive))
{
    if(campoMensaje[2]=="0041")
    {
        ui->page3->showMonitor::tagIconDCAlto0041->show();
    }
    else if(campoMensaje[2]=="0042")
    {
        ui->page3->showMonitor::tagIconDCAlto0042->show();
    }
    else if(campoMensaje[2]=="0043")
    {
        ui->page3->showMonitor::tagIconDCAlto0043->show();
    }
    else if(campoMensaje[2]=="0000")
    {
        ui->page3->showMonitor::tagIconDCAlto0000->show();
    }
    else if(campoMensaje[2]=="0044")
    {
        ui->page3->showMonitor::tagIconDCAlto0044->show();
    }
}
/** Si es una ALARMA por CAIDA_ENLACE -> Mostrar icono "warning" al lado del radioenlace
    y poner los Leds de los radioenlaces en Rojo */
else if(tipoAlarma.contains("CAIDA",Qt::CaseInsensitive))
{
    QString RE_ALM_Origen = campoMensaje[5];
    QString RE_ALM_Destino = campoMensaje[6];

    // Radioenlaces primarios
    if((RE_ALM_Origen.contains("0041")) && (RE_ALM_Destino.contains("0042")))
    {
        ui->page3->showMonitor::tagRadioOff0041_0042->show();
        ui->page3->showMonitor::Led0041_0042.load(":/red_light_16.png");
        ui->page3->showMonitor::tagLed0041_0042->setPixmap(ui->page3->showMonitor::Led0041_0042);

        // Además de caer el radioenlace indicado en el mensaje, caen los radioenlaces asociados al nodo destino.
        ui->page3->showMonitor::tagRadioOff0042_0000->show();
        ui->page3->showMonitor::Led0042_0000.load(":/red_light_16.png");
        ui->page3->showMonitor::tagLed0042_0000->setPixmap(ui->page3->showMonitor::Led0042_0000);

        ui->page3->showMonitor::tagRadioOff0042_0043->show();
        ui->page3->showMonitor::Led0042_0043.load(":/red_light_16.png");
        ui->page3->showMonitor::tagLed0042_0043->setPixmap(ui->page3->showMonitor::Led0042_0043);
    }
    else if((RE_ALM_Origen.contains("0042")) && (RE_ALM_Destino.contains("0043")))
```

```
            {
                ui->page3->showMonitor::tagRadioOff0042_0043->show();
                ui->page3->showMonitor::Led0042_0043.load(":/red_light_16.png");
                ui->page3->showMonitor::tagLed0042_0043->setPixmap(ui->page3->showMonitor::Led0042_0043);

                // Además de caer el radioenlace indicado en el mensaje, caen los radioenlaces asociados al nodo destino.
                ui->page3->showMonitor::tagRadioOff0043_0000->show();
                ui->page3->showMonitor::Led0043_0000.load(":/red_light_16.png");
                ui->page3->showMonitor::tagLed0043_0000->setPixmap(ui->page3->showMonitor::Led0043_0000);
            }
            else if((RE_ALM_Origen.contains("0043")) && (RE_ALM_Destino.contains("0000")))
            {
                ui->page3->showMonitor::tagRadioOff0043_0000->show();
                ui->page3->showMonitor::Led0043_0000.load(":/red_light_16.png");
                ui->page3->showMonitor::tagLed0043_0000->setPixmap(ui->page3->showMonitor::Led0043_0000);
            }
            // Radioenlaces secundarios
            else if( (RE_ALM_Origen.contains("0041")) && (RE_ALM_Destino.contains("0043")) )
            {
                ui->page3->showMonitor::tagRadioOff0041_0043->show();
                ui->page3->showMonitor::Led0041_0043.load(":/red_light_16.png");
                ui->page3->showMonitor::tagLed0041_0043->setPixmap(ui->page3->showMonitor::Led0041_0043);

                // Además de caer el radioenlace indicado en el mensaje, caen los radioenlaces asociados al nodo destino.
                ui->page3->showMonitor::tagRadioOff0043_0000->show();
                ui->page3->showMonitor::Led0043_0000.load(":/red_light_16.png");
                ui->page3->showMonitor::tagLed0043_0000->setPixmap(ui->page3->showMonitor::Led0043_0000);
            }
        }
    }
}

/*-------------------------------------------------------------
 * SLOT: Comprueba si los nodos están vivos y en caso negativo,
 *       activa las alarmas correspondientes
 *-------------------------------------------------------------*/

void WaspMonitor::checkEstadoNodos()
{
    if (waspFlags->capturaEnCurso == 1)
    {
        // Si pasan más de 90 segundos sin recibir un KEEPALIVE y la monitorización del nodo
        // esta habilitada se considera que el nodo ha caído.
        if ( (time_0041.elapsed() > 90000) && (waspFlags->nodo0041_ON == 1))
        {
            // Ponemos la etiqueta del nodo 0041 en rojo
            ui->page3->showMonitor::paleta0041.setColor(QPalette::Base, QColor(255,100,61));
            ui->page3->showMonitor::tag0041->setPalette(ui->page3->showMonitor::paleta0041);

            // Ponemos el LED del enlace 0041_0000 en rojo
            ui->page3->showMonitor::tagLed0041_0000->show();
            ui->page3->showMonitor::Led0041_0000.load(":/red_light_16.png");
            ui->page3->showMonitor::tagLed0041_0000->setPixmap(ui->page3->showMonitor::Led0041_0000);

            // Se supone enlace caído sólo si esta activada la monitorización en el enlace destino
            if(waspFlags->nodo0042_ON == 1)
            {
                // Ponemos el LED del enlace 0041_0042 en rojo
                ui->page3->showMonitor::tagLed0041_0042->show();
                ui->page3->showMonitor::Led0041_0042.load(":/red_light_16.png");
                ui->page3->showMonitor::tagLed0041_0042->setPixmap(ui->page3->showMonitor::Led0041_0042);
                // Activamos en icono de enlace caído en 0041_0042
                ui->page3->showMonitor::tagRadioOff0041_0042->show();
                // Panel de Alarma: Radioenlace 0041-0042 Caído
                ui->page3->showMonitor::panel0041_0042->labelIconEnlaceCaido->setPixmap(ui->page3->showMonitor::panel0041_0042-
>iconoAlarmaON);
            }

            // Se supone enlace caído sólo si esta activada la monitorización en el enlace destino
            if(waspFlags->nodo0043_ON == 1)
            {
                // Ponemos el LED del enlace 0041_0043 en rojo
                ui->page3->showMonitor::tagLed0041_0043->show();
                ui->page3->showMonitor::Led0041_0043.load(":/red_light_16.png");
                ui->page3->showMonitor::tagLed0041_0043->setPixmap(ui->page3->showMonitor::Led0041_0043);
                // Activamos en icono de enlace caído en 0041_0043
                ui->page3->showMonitor::tagRadioOff0041_0043->show();
                // Panel de Alarma: Radioenlace 0041-0043 Caído
                ui->page3->showMonitor::panel0041_0043->labelIconEnlaceCaido->setPixmap(ui->page3->showMonitor::panel0041_0043-
>iconoAlarmaON);
            }

            // Activamos en icono de enlace caído en 0041_0000
            ui->page3->showMonitor::tagRadioOff0041_0000->show();
```

```
            // Panel de Alarma: Nodo 0041 Caído
            ui->page3->showMonitor::panel0041->labelIconNodoCaido->setPixmap(ui->page3->showMonitor::panel0041->iconoAlarmaON);

            // Panel de Alarma: Radioenlace 0041-0000 Caído
            ui->page3->showMonitor::panel0041_0000->labelIconEnlaceCaido->setPixmap(ui->page3->showMonitor::panel0041_0000-
>iconoAlarmaON);

        }
        if( (time_0042.elapsed() > 90000) && (waspFlags->nodo0042_ON == 1))
        {
            // Ponemos la etiqueta del nodo 0042 en rojo
            ui->page3->showMonitor::paleta0042.setColor(QPalette::Base, QColor(255,100,61));
            ui->page3->showMonitor::tag0042->setPalette(ui->page3->showMonitor::paleta0042);

            // Ponemos el LED del enlace 0042_0000 en rojo
            ui->page3->showMonitor::tagLed0042_0000->show();
            ui->page3->showMonitor::Led0042_0000.load(":/red_light_16.png");
            ui->page3->showMonitor::tagLed0042_0000->setPixmap(ui->page3->showMonitor::Led0042_0000);

            // Se supone enlace caído sólo si esta activada la monitorización en el enlace destino
            if(waspFlags->nodo0043_ON == 1)
            {
                // Ponemos el LED del enlace 0042_0043 en rojo
                ui->page3->showMonitor::tagLed0042_0043->show();
                ui->page3->showMonitor::Led0042_0043.load(":/red_light_16.png");
                ui->page3->showMonitor::tagLed0042_0043->setPixmap(ui->page3->showMonitor::Led0042_0043);
                // Activamos en icono de enlace caído en 0042_0043
                ui->page3->showMonitor::tagRadioOff0042_0043->show();
                // Panel de Alarma: Radioenlace 0042-0043 Caído
                ui->page3->showMonitor::panel0042_0043->labelIconEnlaceCaido->setPixmap(ui->page3->showMonitor::panel0042_0043-
>iconoAlarmaON);
            }

            // También se tiene en cuenta el radioenlace anterior (0041-0042)
            if(waspFlags->nodo0041_ON == 1)
            {
                // Ponemos el LED del enlace 0041_0042 en rojo
                ui->page3->showMonitor::tagLed0041_0042->show();
                ui->page3->showMonitor::Led0041_0042.load(":/red_light_16.png");
                ui->page3->showMonitor::tagLed0041_0042->setPixmap(ui->page3->showMonitor::Led0041_0042);
                // Activamos en icono de enlace caído en 0041_0042
                ui->page3->showMonitor::tagRadioOff0041_0042->show();
                // Panel de Alarma: Radioenlace 0041-0042 Caído
                ui->page3->showMonitor::panel0041_0042->labelIconEnlaceCaido->setPixmap(ui->page3->showMonitor::panel0041_0042-
>iconoAlarmaON);
            }

            // Activamos en icono de enlace caído en 0042_0000
            ui->page3->showMonitor::tagRadioOff0042_0000->show();

            // Panel de Alarma: Nodo 0042 Caído
            ui->page3->showMonitor::panel0042->labelIconNodoCaido->setPixmap(ui->page3->showMonitor::panel0042->iconoAlarmaON);

            // Panel de Alarma: Radioenlace 0042-0000 Caído
            ui->page3->showMonitor::panel0042_0000->labelIconEnlaceCaido->setPixmap(ui->page3->showMonitor::panel0042_0000-
>iconoAlarmaON);

        }
        if( (time_0043.elapsed() > 90000) && (waspFlags->nodo0043_ON == 1) )
        {
            // Ponemos la etiqueta del nodo 0043 en rojo
            ui->page3->showMonitor::paleta0043.setColor(QPalette::Base, QColor(255,100,61));
            ui->page3->showMonitor::tag0043->setPalette(ui->page3->showMonitor::paleta0043);

            // Ponemos el LED del enlace 0043_0000 en rojo
            ui->page3->showMonitor::tagLed0043_0000->show();
            ui->page3->showMonitor::Led0043_0000.load(":/red_light_16.png");
            ui->page3->showMonitor::tagLed0043_0000->setPixmap(ui->page3->showMonitor::Led0043_0000);

            // También se tiene en cuenta el radioenlace anterior (0041-0043)
            if(waspFlags->nodo0041_ON == 1)
            {
                // Ponemos el LED del enlace 0041_0043 en rojo
                ui->page3->showMonitor::tagLed0041_0043->show();
                ui->page3->showMonitor::Led0041_0043.load(":/red_light_16.png");
                ui->page3->showMonitor::tagLed0041_0043->setPixmap(ui->page3->showMonitor::Led0041_0043);
                // Activamos en icono de enlace caído en 0041_0043
                ui->page3->showMonitor::tagRadioOff0041_0043->show();
                // Panel de Alarma: Radioenlace 0041-0043 Caído
                ui->page3->showMonitor::panel0041_0043->labelIconEnlaceCaido->setPixmap(ui->page3->showMonitor::panel0041_0043-
>iconoAlarmaON);
            }
```

```
            // También se tiene en cuenta el radioenlace anterior (0042-0043)
            if(waspFlags->nodo0042_ON == 1)
            {
                // Ponemos el LED del enlace 0042_0043 en rojo
                ui->page3->showMonitor::tagLed0042_0043->show();
                ui->page3->showMonitor::Led0042_0043.load(":/red_light_16.png");
                ui->page3->showMonitor::tagLed0042_0043->setPixmap(ui->page3->showMonitor::Led0042_0043);
                // Activamos en icono de enlace caído en 0042_0043
                ui->page3->showMonitor::tagRadioOff0042_0043->show();
                // Panel de Alarma: Radioenlace 0042-0043 Caído
                ui->page3->showMonitor::panel0042_0043->labelIconEnlaceCaido->setPixmap(ui->page3->showMonitor::panel0042_0043-
>iconoAlarmaON);
            }

            // Activamos en icono de enlace caído en 0043_0000
            ui->page3->showMonitor::tagRadioOff0043_0000->show();

            // Panel de Alarma: Nodo 0043 Caído
            ui->page3->showMonitor::panel0043->labelIconNodoCaido->setPixmap(ui->page3->showMonitor::panel0043->iconoAlarmaON);
            // Panel de Alarma: Radioenlace 0043-0000 Caído
            ui->page3->showMonitor::panel0043_0000->labelIconEnlaceCaido->setPixmap(ui->page3->showMonitor::panel0043_0000-
>iconoAlarmaON);

        }
        if( (time_0044.elapsed() > 90000) && (waspFlags->nodo0044_ON == 1) )
        {
            // Ponemos la etiqueta del nodo 0044 en rojo
            ui->page3->showMonitor::paleta0044.setColor(QPalette::Base, QColor(255,100,61));
            ui->page3->showMonitor::tag0044->setPalette(ui->page3->showMonitor::paleta0044);

            // Ponemos el LED del enlace 0044_0000 en rojo
            ui->page3->showMonitor::tagLed0044_0000->show();
            ui->page3->showMonitor::Led0044_0000.load(":/red_light_16.png");
            ui->page3->showMonitor::tagLed0044_0000->setPixmap(ui->page3->showMonitor::Led0044_0000);

            // Activamos en icono de enlace caído en 0044_0000
            ui->page3->showMonitor::tagRadioOff0044_0000->show();

            // Panel de Alarma: Nodo 0044 Caído
            ui->page3->showMonitor::panel0044->labelIconNodoCaido->setPixmap(ui->page3->showMonitor::panel0044->iconoAlarmaON);
            // Panel de Alarma: Radioenlace 0044-0000 Caído
            ui->page3->showMonitor::panel0044_0000->labelIconEnlaceCaido->setPixmap(ui->page3->showMonitor::panel0044_0000-
>iconoAlarmaON);
        }
    }

}

/*-----------------------------------------------------------------
 * SLOT: Comprueba si los enlaces están activos y en caso
 *       contrario, pone los LEDs en rojo
 *-----------------------------------------------------------------*/

void WaspMonitor::checkEstadoEnlaces()
{
    if (waspFlags->capturaEnCurso == 1)
    {
        // Si pasan más de 90 segundos sin recibir un ESTADO_ENLACE y la monitorización de los extremos del radioenlace
        // está habilitada se considera que el enlace ha caído.
        if ((waspFlags->nodo0041_ON == 1) && (waspFlags->nodo0042_ON == 1) && (time_0041_0042.elapsed() > 90000))
        {
            // Ponemos el LED del enlace 0041_0042 en rojo
            ui->page3->showMonitor::tagLed0041_0042->show();
            ui->page3->showMonitor::Led0041_0042.load(":/red_light_16.png");
            ui->page3->showMonitor::tagLed0041_0042->setPixmap(ui->page3->showMonitor::Led0041_0042);

            qDebug() << "ms transcurridos 0041_0042: " << time_0041_0042.elapsed();
        }
        if ((waspFlags->nodo0042_ON == 1) && (waspFlags->nodo0043_ON == 1) && (time_0042_0043.elapsed() > 90000))
        {
            // Ponemos el LED del enlace 0042_0043 en rojo
            ui->page3->showMonitor::tagLed0042_0043->show();
            ui->page3->showMonitor::Led0042_0043.load(":/red_light_16.png");
            ui->page3->showMonitor::tagLed0042_0043->setPixmap(ui->page3->showMonitor::Led0042_0043);

            qDebug() << "ms transcurridos 0042_0043: " << time_0042_0043.elapsed();
        }
        if ((waspFlags->nodo0043_ON == 1) && (time_0043_0000.elapsed() > 90000))
        {
            // Ponemos el LED del enlace 0043_0000 en rojo
            ui->page3->showMonitor::tagLed0043_0000->show();
            ui->page3->showMonitor::Led0043_0000.load(":/red_light_16.png");
            ui->page3->showMonitor::tagLed0043_0000->setPixmap(ui->page3->showMonitor::Led0043_0000);
```

```
            qDebug() << "ms transcurridos 0043_0000: " << time_0043_0000.elapsed();
        }
        if ((waspFlags->nodo0044_ON == 1) && (time_0044_0000.elapsed() > 90000))
        {
            // Ponemos el LED del enlace 0044_0000 en rojo
            ui->page3->showMonitor::tagLed0044_0000->show();
            ui->page3->showMonitor::Led0044_0000.load(":/red_light_16.png");
            ui->page3->showMonitor::tagLed0044_0000->setPixmap(ui->page3->showMonitor::Led0044_0000);

            qDebug() << "ms transcurridos 0044_0000: " << time_0044_0000.elapsed();
        }
        if ((waspFlags->nodo0041_ON == 1) && (waspFlags->nodo0043_ON == 1) && (time_0041_0043.elapsed() > 90000))
        {
            // Ponemos el LED del enlace 0041_0043 en rojo
            ui->page3->showMonitor::tagLed0041_0043->show();
            ui->page3->showMonitor::Led0041_0043.load(":/red_light_16.png");
            ui->page3->showMonitor::tagLed0041_0043->setPixmap(ui->page3->showMonitor::Led0041_0043);

            qDebug() << "ms transcurridos 0041_0043: " << time_0041_0043.elapsed();
        }
        if ((waspFlags->nodo0041_ON == 1) && (time_0041_0000.elapsed() > 90000))
        {
            // Ponemos el LED del enlace 0041_0000 en rojo
            ui->page3->showMonitor::tagLed0041_0000->show();
            ui->page3->showMonitor::Led0041_0000.load(":/red_light_16.png");
            ui->page3->showMonitor::tagLed0041_0000->setPixmap(ui->page3->showMonitor::Led0041_0000);

            qDebug() << "ms transcurridos 0041_0000: " << time_0041_0000.elapsed();
        }
        if ((waspFlags->nodo0042_ON == 1) && (time_0042_0000.elapsed() > 90000))
        {
            // Ponemos el LED del enlace 0042_0000 en rojo
            ui->page3->showMonitor::tagLed0042_0000->show();
            ui->page3->showMonitor::Led0042_0000.load(":/red_light_16.png");
            ui->page3->showMonitor::tagLed0042_0000->setPixmap(ui->page3->showMonitor::Led0042_0000);

            qDebug() << "ms transcurridos 0042_0000: " << time_0042_0000.elapsed();
        }
    }
}

/*-----------------------------------------------------------------
 * SLOT: Hace visible el widget de Información
 *-----------------------------------------------------------------*/

void WaspMonitor::mostrarInfo()
{
    ui->stackedWidget->setCurrentIndex(0);
}

/*-----------------------------------------------------------------
 * SLOT: Muestra el Qdialog de Configuración de Waspmonitor
 *-----------------------------------------------------------------*/

void WaspMonitor::mostrarConfig()
{
    configDialogBox->show();
}

/*-----------------------------------------------------------------
 * SLOT: Limpia todos los:
 *       - Iconos de alarma del mapa.
 *       - Iconos de alarma del panel de alarmas.
 *-----------------------------------------------------------------*/

void WaspMonitor::limpiarAlarmas()
{
    /* --- Nodo 0000 ---*/
    limpiarNC_0000();
    limpiarBB_0000();
    limpiarTA_0000();
    limpiarDC_0000();

    /* --- Nodo 0041 ---*/
    limpiarNC_0041();
    limpiarBB_0041();
    limpiarTA_0041();
    limpiarDC_0041();

    /* --- Nodo 0042 ---*/
    limpiarNC_0042();
    limpiarBB_0042();
    limpiarTA_0042();
    limpiarDC_0042();
```

```
    /* --- Nodo 0043 ---*/
    limpiarNC_0043();
    limpiarBB_0043();
    limpiarTA_0043();
    limpiarDC_0043();

    /* --- Nodo 0044 ---*/
    limpiarNC_0044();
    limpiarBB_0044();
    limpiarTA_0044();
    limpiarDC_0044();

    /* --- Radioenlaces primarios ---*/
    limpiarRC_0041_0042();
    limpiarRC_0042_0043();
    limpiarRC_0043_0000();
    limpiarRC_0044_0000();

    /* --- Radioenlaces secundarios ---*/
    limpiarRC_0041_0043();
    limpiarRC_0041_0000();
    limpiarRC_0042_0000();
}

/*-----------------------------------------------------------------
 * SLOT: Hace visible el widget de Monitorización
 *-----------------------------------------------------------------*/

void WaspMonitor::mostrarMonitorizacion()
{
    ui->stackedWidget->setCurrentIndex(1);
}

/*-----------------------------------------------------------------
 * SLOT: Inicia el hilo de captura "hiloSniffer"
 *-----------------------------------------------------------------*/

void WaspMonitor::iniciarCaptura()
{
    // Si la captura de datos NO esta en curso...
    if(waspFlags->capturaEnCurso == 0)
    {
        // Cambiamos el nombre de los archivos de Log

        waspFileNames->logHTMLName = waspFileNames->logName + "_" + QDate::currentDate().toString() + "_" +
QTime::currentTime().toString() + ".html";
        waspFileNames->logHTMLName.replace(':',"_");
        waspFileNames->logHTMLName.replace(' ',"_");

        waspFileNames->logCSVName = waspFileNames->logName + "_" + QDate::currentDate().toString() + "_" +
QTime::currentTime().toString() + ".csv";
        waspFileNames->logCSVName.replace(':',"_");
        waspFileNames->logCSVName.replace(' ',"_");

        Sleeper sleeper;

        // Se inicia el hilo de captura y se abre el puerto serie
        hiloSniffer->start();

        // Esperamos un instante (50 ms) a que le de tiempo al otro
        // hilo ("hiloSniffer"), a intentar abrir el puerto
        sleeper.msleep(50);

        if(hiloSniffer->errorAbrirPuerto == 0)
        {
            waspFlags->capturaEnCurso = 1;

            // Ponemos el icono de captura de tramas en animación
            ui->page3->showMonitor::iconSignal->start();

            // Se muestra un mensaje en la Barra de Estado informando de que se ha abierto bien el puerto.
            QString msgPuertoAbierto = "Puerto " + hiloSniffer->nombrePuerto + " abierto correctamente. Capturando tramas";
            ui->statusBar->showMessage(msgPuertoAbierto, 4000);

            // Iniciamos el contador inicial para que emita una señal cada 30 segundos
            // y ponemos a cero los contadores de cada nodo y de cada enlace
            contador->start(30000);
            time_0041.restart();
            time_0042.restart();
            time_0043.restart();
            time_0044.restart();

            time_0041_0042.restart();
```

```
            time_0042_0043.restart();
            time_0043_0000.restart();
            time_0044_0000.restart();

            time_0041_0043.restart();
            time_0041_0000.restart();
            time_0042_0000.restart();
        }
        else
        {
            // Se muestra un mensaje en la Barra de estado informando de que ha habido un error al abrir el puerto.
            QString msgPuertoAbierto = "Error al abrir el puerto " + hiloSniffer->nombrePuerto;
            ui->statusBar->showMessage(msgPuertoAbierto, 4000);

            // Se muestra un Message Box indicando el error.
            QMessageBox criticalErrorAlAbrirPuerto;
            criticalErrorAlAbrirPuerto.critical(this,"Error",
                                                "Se ha producido un error al intentar abrir el puerto serie.\n\nLa captura de datos
no se ha iniciado.\n\nCompruebe que el cable USB está conectado al Waspmote,\néste está encendido y el puerto COM es el
correcto.");
         }
    }
    else
    {
        QMessageBox warnYaIniciado;
        warnYaIniciado.warning(this,"Atención","El proceso de captura ya ha está ejecutándose.\n\nSi desea detenerlo pulse
'Detener Captura'.");
    }
}

/*-----------------------------------------------------------------
 * SLOT: Detiene el hilo de captura "hiloSniffer"
 *-----------------------------------------------------------------*/

void WaspMonitor::detenerCaptura()
{
    if (waspFlags->capturaEnCurso == 1)
    {
        // Detenemos la animación del icono de captura de tramas.
        ui->page3->showMonitor::iconSignal->jumpToFrame(0);
        ui->page3->showMonitor::iconSignal->stop();

        // Se muestra un mensaje en la Barra de Estado informando de que se ha abierto bien el puerto.
        QString msgCapturaDetenida = "Captura de tramas detenida";
        ui->statusBar->showMessage(msgCapturaDetenida, 4000);
    }

    waspFlags->capturaEnCurso = 0;

    hiloSniffer->stop();

}

/*-----------------------------------------------------------------
 * SLOT: Sale de la aplicación (En respuesta a Acción de Botón)
 *-----------------------------------------------------------------*/

void WaspMonitor::salir()
{
     QPixmap IconoSalir;
     IconoSalir.load(":/Salida.png");

     QMessageBox msgBoxSalir;
     msgBoxSalir.setWindowTitle("Salir de Waspmonitor");
     msgBoxSalir.setText("¿Desea salir del programa?\n\nSe detendrá la captura de datos.");

     QPushButton *SiButton = msgBoxSalir.addButton(tr("Sí"), QMessageBox::YesRole);
     QPushButton *NoButton = msgBoxSalir.addButton(tr("No"), QMessageBox::NoRole);

     msgBoxSalir.setIconPixmap(IconoSalir);
     msgBoxSalir.exec();

     if(msgBoxSalir.clickedButton()==SiButton)
      {
         // Finalizamos el proceso de captura y cerramos el puerto serie
         hiloSniffer->stop();

         qApp->quit();
      }
}

/*-----------------------------------------------------------------
 * EVENT: Sale de la aplicación (En respuesta a Evento de Cierre, o sea,
 * pulsar la X de la ventana)
```

```
 *-------------------------------------------------------------*/

void WaspMonitor::closeEvent(QCloseEvent *evento)
{
    QPixmap IconoSalir;

    IconoSalir.load(":/Salida.png");

    QMessageBox msgBoxSalir;
    msgBoxSalir.setWindowTitle("Salir de Waspmonitor");
    msgBoxSalir.setText("¿Desea salir del programa?\n\nSe detendrá la captura de datos.");

    QPushButton *SiButton = msgBoxSalir.addButton(tr("Sí"), QMessageBox::YesRole);
    QPushButton *NoButton = msgBoxSalir.addButton(tr("No"), QMessageBox::NoRole);

    msgBoxSalir.setIconPixmap(IconoSalir);

    msgBoxSalir.exec();

    if(msgBoxSalir.clickedButton()==SiButton)
     {
        // Finalizamos el proceso de captura y cerramos el puerto serie
        hiloSniffer->stop();

        evento->accept();
     }
    else
     {
        evento->ignore();
     }
}
```

A2.4. Forms.

Configdialog.ui

```
<?xml version="1.0" encoding="UTF-8"?>
<ui version="4.0">
 <class>configDialog</class>
 <widget class="QDialog" name="configDialog">
  <property name="geometry">
   <rect>
    <x>0</x>
    <y>0</y>
    <width>499</width>
    <height>480</height>
   </rect>
  </property>
  <property name="sizePolicy">
   <sizepolicy hsizetype="Fixed" vsizetype="Fixed">
    <horstretch>0</horstretch>
    <verstretch>0</verstretch>
   </sizepolicy>
  </property>
  <property name="windowTitle">
   <string>Configuración de Waspmonitor</string>
  </property>
  <widget class="QFrame" name="frame_2">
   <property name="geometry">
    <rect>
     <x>9</x>
     <y>9</y>
     <width>481</width>
     <height>462</height>
    </rect>
   </property>
   <property name="sizePolicy">
    <sizepolicy hsizetype="Fixed" vsizetype="Fixed">
     <horstretch>0</horstretch>
     <verstretch>0</verstretch>
    </sizepolicy>
   </property>
   <property name="frameShape">
    <enum>QFrame::StyledPanel</enum>
   </property>
   <property name="frameShadow">
    <enum>QFrame::Raised</enum>
   </property>
```

```
<widget class="QFrame" name="frame">
 <property name="geometry">
  <rect>
   <x>40</x>
   <y>130</y>
   <width>191</width>
   <height>151</height>
  </rect>
 </property>
 <property name="frameShape">
  <enum>QFrame::WinPanel</enum>
 </property>
 <property name="frameShadow">
  <enum>QFrame::Raised</enum>
 </property>
 <property name="lineWidth">
  <number>2</number>
 </property>
 <layout class="QVBoxLayout" name="verticalLayout">
  <item>
   <widget class="QCheckBox" name="checkBox_0041">
    <property name="text">
     <string>Nodo 0041 (Ctra. Utrera) </string>
    </property>
    <property name="checked">
     <bool>true</bool>
    </property>
   </widget>
  </item>
  <item>
   <widget class="QCheckBox" name="checkBox_0042">
    <property name="text">
     <string>Nodo 0042 (C. Maribáñez)</string>
    </property>
    <property name="checked">
     <bool>true</bool>
    </property>
   </widget>
  </item>
  <item>
   <widget class="QCheckBox" name="checkBox_0043">
    <property name="text">
     <string>Nodo 0043 (San Fernando)</string>
    </property>
    <property name="checked">
     <bool>true</bool>
    </property>
   </widget>
  </item>
  <item>
   <widget class="QCheckBox" name="checkBox_0044">
    <property name="text">
     <string>Nodo 0044 (Alcantarillas)</string>
    </property>
    <property name="checked">
     <bool>true</bool>
    </property>
   </widget>
  </item>
 </layout>
</widget>
<widget class="QLabel" name="label">
 <property name="geometry">
  <rect>
   <x>40</x>
   <y>70</y>
   <width>181</width>
   <height>31</height>
  </rect>
 </property>
 <property name="font">
  <font>
   <pointsize>9</pointsize>
   <weight>75</weight>
   <bold>true</bold>
  </font>
 </property>
 <property name="text">
  <string>Habilitar o deshabilitar</string>
 </property>
 <property name="alignment">
  <set>Qt::AlignCenter</set>
 </property>
</widget>
```

```
<widget class="QLabel" name="label_2">
 <property name="geometry">
  <rect>
   <x>40</x>
   <y>300</y>
   <width>191</width>
   <height>41</height>
  </rect>
 </property>
 <property name="font">
  <font>
   <pointsize>9</pointsize>
   <weight>75</weight>
   <bold>true</bold>
  </font>
 </property>
 <property name="text">
  <string>Puerto COM para el Gateway</string>
 </property>
 <property name="alignment">
  <set>Qt::AlignCenter</set>
 </property>
</widget>
<widget class="QLabel" name="label_3">
 <property name="geometry">
  <rect>
   <x>320</x>
   <y>290</y>
   <width>91</width>
   <height>91</height>
  </rect>
 </property>
 <property name="text">
  <string/>
 </property>
 <property name="pixmap">
  <pixmap resource="waspmonitor.qrc">:/settings.png</pixmap>
 </property>
</widget>
<widget class="QLabel" name="label_4">
 <property name="geometry">
  <rect>
   <x>50</x>
   <y>20</y>
   <width>181</width>
   <height>41</height>
  </rect>
 </property>
 <property name="font">
  <font>
   <pointsize>19</pointsize>
   <weight>75</weight>
   <bold>true</bold>
  </font>
 </property>
 <property name="text">
  <string>Configuración</string>
 </property>
 <property name="alignment">
  <set>Qt::AlignCenter</set>
 </property>
</widget>
<widget class="QPushButton" name="pushButton">
 <property name="geometry">
  <rect>
   <x>320</x>
   <y>420</y>
   <width>101</width>
   <height>31</height>
  </rect>
 </property>
 <property name="text">
  <string>Aceptar</string>
 </property>
</widget>
<widget class="QLabel" name="label_5">
 <property name="geometry">
  <rect>
   <x>260</x>
   <y>90</y>
   <width>181</width>
   <height>31</height>
  </rect>
 </property>
```

```
 <property name="font">
  <font>
   <pointsize>9</pointsize>
   <weight>75</weight>
   <bold>true</bold>
  </font>
 </property>
 <property name="text">
  <string>Habilitar o deshabilitar logs</string>
 </property>
 <property name="alignment">
  <set>Qt::AlignCenter</set>
 </property>
</widget>
<widget class="QFrame" name="frame_3">
 <property name="geometry">
  <rect>
   <x>270</x>
   <y>130</y>
   <width>171</width>
   <height>111</height>
  </rect>
 </property>
 <property name="frameShape">
  <enum>QFrame::WinPanel</enum>
 </property>
 <property name="frameShadow">
  <enum>QFrame::Raised</enum>
 </property>
 <property name="lineWidth">
  <number>2</number>
 </property>
 <layout class="QVBoxLayout" name="verticalLayout_2">
  <item>
   <widget class="QCheckBox" name="checkBox_logCSV">
    <property name="text">
     <string>Log en CSV</string>
    </property>
    <property name="checked">
     <bool>true</bool>
    </property>
   </widget>
  </item>
  <item>
   <widget class="QCheckBox" name="checkBox_logHTML">
    <property name="text">
     <string>Log en HTML</string>
    </property>
    <property name="checked">
     <bool>true</bool>
    </property>
   </widget>
  </item>
 </layout>
</widget>
<widget class="QFrame" name="frame_4">
 <property name="geometry">
  <rect>
   <x>40</x>
   <y>340</y>
   <width>191</width>
   <height>61</height>
  </rect>
 </property>
 <property name="frameShape">
  <enum>QFrame::WinPanel</enum>
 </property>
 <property name="frameShadow">
  <enum>QFrame::Raised</enum>
 </property>
 <property name="lineWidth">
  <number>2</number>
 </property>
 <widget class="QComboBox" name="comboBox">
  <property name="geometry">
   <rect>
    <x>50</x>
    <y>20</y>
    <width>91</width>
    <height>21</height>
   </rect>
  </property>
  <item>
   <property name="text">
```

```
  <string>COM1</string>
 </property>
</item>
<item>
 <property name="text">
  <string>COM2</string>
 </property>
</item>
<item>
 <property name="text">
  <string>COM3</string>
 </property>
</item>
<item>
 <property name="text">
  <string>COM4</string>
 </property>
</item>
<item>
 <property name="text">
  <string>COM5</string>
 </property>
</item>
<item>
 <property name="text">
  <string>COM6</string>
 </property>
</item>
<item>
 <property name="text">
  <string>COM7</string>
 </property>
</item>
<item>
 <property name="text">
  <string>COM8</string>
 </property>
</item>
<item>
 <property name="text">
  <string>COM9</string>
 </property>
</item>
<item>
 <property name="text">
  <string>COM10</string>
 </property>
</item>
<item>
 <property name="text">
  <string>COM11</string>
 </property>
</item>
<item>
 <property name="text">
  <string>COM12</string>
 </property>
</item>
<item>
 <property name="text">
  <string>COM13</string>
 </property>
</item>
<item>
 <property name="text">
  <string>COM14</string>
 </property>
</item>
<item>
 <property name="text">
  <string>COM15</string>
 </property>
</item>
<item>
 <property name="text">
  <string>COM16</string>
 </property>
</item>
<item>
 <property name="text">
  <string>COM17</string>
 </property>
</item>
<item>
```

```
      <property name="text">
       <string>COM18</string>
      </property>
     </item>
     <item>
      <property name="text">
       <string>COM19</string>
      </property>
     </item>
     <item>
      <property name="text">
       <string>COM20</string>
      </property>
     </item>
     <item>
      <property name="text">
       <string>COM21</string>
      </property>
     </item>
     <item>
      <property name="text">
       <string>COM22</string>
      </property>
     </item>
     <item>
      <property name="text">
       <string>COM23</string>
      </property>
     </item>
     <item>
      <property name="text">
       <string>COM24</string>
      </property>
     </item>
     <item>
      <property name="text">
       <string>COM25</string>
      </property>
     </item>
    </widget>
   </widget>
   <widget class="QLabel" name="label_6">
    <property name="geometry">
     <rect>
      <x>40</x>
      <y>90</y>
      <width>191</width>
      <height>31</height>
     </rect>
    </property>
    <property name="font">
     <font>
      <pointsize>9</pointsize>
      <weight>75</weight>
      <bold>true</bold>
     </font>
    </property>
    <property name="text">
     <string>monitorización de nodos</string>
    </property>
    <property name="alignment">
     <set>Qt::AlignCenter</set>
    </property>
   </widget>
  </widget>
 </widget>
 <resources>
  <include location="waspmonitor.qrc"/>
 </resources>
 <connections/>
</ui>
```

Showinfo.ui

```
<?xml version="1.0" encoding="UTF-8"?>
<ui version="4.0">
 <class>showInfo</class>
 <widget class="QWidget" name="showInfo">
  <property name="geometry">
   <rect>
    <x>0</x>
    <y>0</y>
```

```
    <width>686</width>
    <height>463</height>
   </rect>
  </property>
  <property name="windowTitle">
   <string>Form</string>
  </property>
  <layout class="QVBoxLayout" name="verticalLayout">
   <property name="margin">
    <number>9</number>
   </property>
   <item>
    <widget class="QTextBrowser" name="textBrowser">
     <property name="html">
      <string>&lt;!DOCTYPE HTML PUBLIC &quot;-//W3C//DTD HTML 4.0//EN&quot; &quot;http://www.w3.org/TR/REC-
html40/strict.dtd&quot;&gt;
&lt;html&gt;&lt;head&gt;&lt;meta name=&quot;qrichtext&quot; content=&quot;1&quot; /&gt;&lt;title&gt;Ayuda de
Waspmonitor&lt;/title&gt;&lt;style type=&quot;text/css&quot;&gt;
p, li { white-space: pre-wrap; }
&lt;/style&gt;&lt;/head&gt;&lt;body style=&quot; font-family:'MS Shell Dlg 2'; font-size:8.25pt; font-weight:400; font-
style:normal;&quot;&gt;
&lt;p align=&quot;center&quot; style=&quot; margin-top:12px; margin-bottom:12px; margin-left:0px; margin-right:0px; -qt-block-
indent:0; text-indent:0px;&quot;&gt;&lt;img src=&quot;index_archivos/image001.png&quot; /&gt; &lt;/p&gt;
&lt;p style=&quot; margin-top:12px; margin-bottom:12px; margin-left:0px; margin-right:0px; -qt-block-indent:0; text-
indent:0px;&quot;&gt;&lt;span style=&quot; font-size:14pt; font-weight:600;&quot;&gt; &lt;/span&gt; &lt;/p&gt;
&lt;hr width=&quot;100%&quot;/&gt;
&lt;p style=&quot; margin-top:12px; margin-bottom:0px; margin-left:0px; margin-right:0px; -qt-block-indent:0; text-
indent:0px;&quot;&gt;  &lt;/p&gt;
&lt;p style=&quot; margin-top:0px; margin-bottom:0px; margin-left:0px; margin-right:0px; -qt-block-indent:0; text-
indent:0px;&quot;&gt;&lt;a name=&quot;Indice&quot;&gt;&lt;/a&gt;&lt;span style=&quot; font-size:14pt; font-
weight:600;&quot;&gt;T&lt;/span&gt;&lt;span style=&quot; font-size:14pt; font-weight:600;&quot;&gt;abla de contenidos.&lt;br
/&gt;&lt;br /&gt;&lt;/span&gt;&lt;a href=&quot;#Barra_de_herramientas&quot;&gt;&lt;span style=&quot; font-size:14pt; font-
weight:600; text-decoration: underline; color:#0000ff;&quot;&gt;1. Barra de herramientas.&lt;/span&gt;&lt;/a&gt;&lt;span
style=&quot; font-size:14pt; font-weight:600;&quot;&gt;&lt;br /&gt;&lt;/span&gt;&lt;a
href=&quot;#Lista_de_nodos&quot;&gt;&lt;span style=&quot; font-size:14pt; font-weight:600; text-decoration: underline;
color:#0000ff;&quot;&gt;2. Lista de nodos y radioenlaces.&lt;/span&gt;&lt;/a&gt;&lt;span style=&quot; font-size:14pt; font-
weight:600;&quot;&gt;&lt;br /&gt;&lt;/span&gt;&lt;a href=&quot;#Mapa_del_canal&quot;&gt;&lt;span style=&quot; font-size:14pt;
font-weight:600; text-decoration: underline; color:#0000ff;&quot;&gt;3. Mapa del canal.&lt;/span&gt;&lt;/a&gt;&lt;br /&gt;&lt;a
href=&quot;#Ventana_de_log&quot;&gt;&lt;span style=&quot; font-size:14pt; font-weight:600; text-decoration: underline;
color:#0000ff;&quot;&gt;4. Ventana de log.&lt;/span&gt;&lt;/a&gt;&lt;br /&gt;&lt;a
href=&quot;#Barras_de_medida&quot;&gt;&lt;span style=&quot; font-size:14pt; font-weight:600; text-decoration: underline;
color:#0000ff;&quot;&gt;5. Barras de medida.&lt;/span&gt;&lt;/a&gt;&lt;br /&gt;&lt;a href=&quot;#Historicos&quot;&gt;&lt;span
style=&quot; font-size:14pt; font-weight:600; text-decoration: underline; color:#0000ff;&quot;&gt;6. Históricos de
medidas.&lt;/span&gt;&lt;/a&gt;&lt;span style=&quot; font-size:14pt; font-weight:600;&quot;&gt;&lt;br /&gt;&lt;/span&gt;&lt;a
href=&quot;#Paneles_de_alarma&quot;&gt;&lt;span style=&quot; font-size:14pt; font-weight:600; text-decoration: underline;
color:#0000ff;&quot;&gt;7. Paneles de alarma.&lt;/span&gt;&lt;/a&gt;&lt;br /&gt;&lt;a
href=&quot;#Configuracion&quot;&gt;&lt;span style=&quot; font-size:14pt; font-weight:600; text-decoration: underline;
color:#0000ff;&quot;&gt;8. Configuración de Waspmonitor.&lt;/span&gt;&lt;/a&gt;&lt;br /&gt;&lt;a
href=&quot;#Cuadros_de_dialogo&quot;&gt;&lt;span style=&quot; font-size:14pt; font-weight:600; text-decoration: underline;
color:#0000ff;&quot;&gt;9. Cuadros de diálogo.&lt;/span&gt;&lt;/a&gt;&lt;br /&gt;&lt;a
href=&quot;#Barra_de_estado&quot;&gt;&lt;span style=&quot; font-size:14pt; font-weight:600; text-decoration: underline;
color:#0000ff;&quot;&gt;10. Barra de Estado.&lt;/span&gt;&lt;/a&gt;&lt;a href=&quot;#Barra_de_estado&quot;&gt;&lt;span
style=&quot; text-decoration: underline; color:#0000ff;&quot;&gt;&lt;br /&gt;&lt;/span&gt;&lt;/a&gt;&lt;/p&gt;
&lt;hr width=&quot;100%&quot;/&gt;
&lt;p style=&quot; margin-top:0px; margin-bottom:0px; margin-left:0px; margin-right:0px; -qt-block-indent:0; text-
indent:0px;&quot;&gt;&lt;a name=&quot;Barra_de_herramientas&quot;&gt;&lt;/a&gt;&lt;span style=&quot; font-size:14pt; font-
weight:600;&quot;&gt;1&lt;/span&gt;&lt;span style=&quot; font-size:14pt; font-weight:600;&quot;&gt;. Barra de
herramientas.&lt;/span&gt; &lt;/p&gt;
&lt;p align=&quot;center&quot; style=&quot; margin-top:12px; margin-bottom:12px; margin-left:0px; margin-right:0px; -qt-block-
indent:0; text-indent:0px;&quot;&gt;&lt;br /&gt;&lt;a name=&quot;_x0000_i1061&quot;&gt;&lt;/a&gt;&lt;img
src=&quot;index_archivos/image002.png&quot; /&gt; &lt;/p&gt;
&lt;p style=&quot; margin-top:12px; margin-bottom:12px; margin-left:0px; margin-right:0px; -qt-block-indent:0; text-
indent:0px;&quot;&gt;  &lt;/p&gt;
&lt;p align=&quot;center&quot; style=&quot;-qt-paragraph-type:empty; margin-top:0px; margin-bottom:0px; margin-left:0px; margin-
right:0px; -qt-block-indent:0; text-indent:0px;&quot;&gt;&lt;/p&gt;
&lt;table border=&quot;0&quot; style=&quot; margin-top:0px; margin-bottom:0px; margin-left:0px; margin-right:0px;&quot;
align=&quot;center&quot; cellspacing=&quot;0&quot; cellpadding=&quot;0&quot;&gt;
&lt;tr&gt;
&lt;td width=&quot;83&quot; style=&quot; vertical-align:top; padding-left:0; padding-right:0; padding-top:0; padding-
bottom:0;&quot;&gt;
&lt;p style=&quot; margin-top:12px; margin-bottom:0px; margin-left:0px; margin-right:0px; -qt-block-indent:0; text-
indent:0px;&quot;&gt;&lt;a name=&quot;_x0000_i1060&quot;&gt;&lt;/a&gt;&lt;img src=&quot;index_archivos/image003.png&quot;
style=&quot;vertical-align: top;&quot; /&gt; &lt;/p&gt;&lt;/td&gt;
&lt;td width=&quot;493&quot; style=&quot; vertical-align:top; padding-left:0; padding-right:0; padding-top:0; padding-
bottom:0;&quot;&gt;
&lt;p style=&quot; margin-top:12px; margin-bottom:0px; margin-left:0px; margin-right:0px; -qt-block-indent:0; text-
indent:0px;&quot;&gt;&lt;span style=&quot; font-weight:600;&quot;&gt;Salir. &lt;br /&gt;&lt;/span&gt;Sale del programa,
deteniendo la captura de tramas en caso de que esté en curso y cerrando el puerto serie. &lt;/p&gt;&lt;/td&gt;&lt;/tr&gt;
&lt;tr&gt;
&lt;td style=&quot; vertical-align:top; padding-left:0; padding-right:0; padding-top:0; padding-bottom:0;&quot;&gt;
&lt;p style=&quot; margin-top:12px; margin-bottom:0px; margin-left:0px; margin-right:0px; -qt-block-indent:0; text-
indent:0px;&quot;&gt;&lt;a name=&quot;_x0000_i1059&quot;&gt;&lt;/a&gt;&lt;img src=&quot;index_archivos/image004.png&quot;
style=&quot;vertical-align: top;&quot; /&gt; &lt;/p&gt;&lt;/td&gt;
&lt;td style=&quot; vertical-align:top; padding-left:0; padding-right:0; padding-top:0; padding-bottom:0;&quot;&gt;
```

<p style=" margin-top:12px; margin-bottom:0px; margin-left:0px; margin-right:0px; -qt-block-indent:0; text-indent:0px;">Iniciar Captura. </p>
<p style=" margin-top:12px; margin-bottom:0px; margin-left:0px; margin-right:0px; -qt-block-indent:0; text-indent:0px;">Abre el puerto serie e inicia la captura de tramas desde el Waspmote Gateway.
</p></td></tr>
<tr>
<td style=" vertical-align:top; padding-left:0; padding-right:0; padding-top:0; padding-bottom:0;">
<p style=" margin-top:12px; margin-bottom:0px; margin-left:0px; margin-right:0px; -qt-block-indent:0; text-indent:0px;"> </p></td>
<td style=" vertical-align:top; padding-left:0; padding-right:0; padding-top:0; padding-bottom:0;">
<p style=" margin-top:12px; margin-bottom:0px; margin-left:0px; margin-right:0px; -qt-block-indent:0; text-indent:0px;">Detener Captura. </p>
<p style=" margin-top:12px; margin-bottom:0px; margin-left:0px; margin-right:0px; -qt-block-indent:0; text-indent:0px;">Detiene la captura de tramas desde el Waspmote Gateway. </p></td></tr>
<tr>
<td style=" vertical-align:top; padding-left:0; padding-right:0; padding-top:0; padding-bottom:0;">
<p style=" margin-top:12px; margin-bottom:0px; margin-left:0px; margin-right:0px; -qt-block-indent:0; text-indent:0px;"> </p></td>
<td style=" vertical-align:top; padding-left:0; padding-right:0; padding-top:0; padding-bottom:0;">
<p style=" margin-top:12px; margin-bottom:0px; margin-left:0px; margin-right:0px; -qt-block-indent:0; text-indent:0px;">Monitorización. </p>
<p style=" margin-top:12px; margin-bottom:0px; margin-left:0px; margin-right:0px; -qt-block-indent:0; text-indent:0px;">Muestra la ventana principal de monitorización de Waspmonitor. </p></td></tr>
<tr>
<td style=" vertical-align:top; padding-left:0; padding-right:0; padding-top:0; padding-bottom:0;">
<p style=" margin-top:12px; margin-bottom:0px; margin-left:0px; margin-right:0px; -qt-block-indent:0; text-indent:0px;"> </p></td>
<td style=" vertical-align:top; padding-left:0; padding-right:0; padding-top:0; padding-bottom:0;">
<p style=" margin-top:12px; margin-bottom:0px; margin-left:0px; margin-right:0px; -qt-block-indent:0; text-indent:0px;">Limpiar Alarmas. </p>
<p style=" margin-top:12px; margin-bottom:0px; margin-left:0px; margin-right:0px; -qt-block-indent:0; text-indent:0px;">Limpia los iconos de alarma del mapa y de los paneles de alarma de los distintos nodos y radioenlaces.
</p></td></tr>
<tr>
<td style=" vertical-align:top; padding-left:0; padding-right:0; padding-top:0; padding-bottom:0;">
<p style=" margin-top:12px; margin-bottom:0px; margin-left:0px; margin-right:0px; -qt-block-indent:0; text-indent:0px;"> </p></td>
<td style=" vertical-align:top; padding-left:0; padding-right:0; padding-top:0; padding-bottom:0;">
<p style=" margin-top:12px; margin-bottom:0px; margin-left:0px; margin-right:0px; -qt-block-indent:0; text-indent:0px;">Configuración. </p>
<p style=" margin-top:12px; margin-bottom:0px; margin-left:0px; margin-right:0px; -qt-block-indent:0; text-indent:0px;">Abre la ventana de configuración de Waspmonitor. </p></td></tr>
<tr>
<td style=" vertical-align:top; padding-left:0; padding-right:0; padding-top:0; padding-bottom:0;">
<p style=" margin-top:12px; margin-bottom:0px; margin-left:0px; margin-right:0px; -qt-block-indent:0; text-indent:0px;"> </p></td>
<td style=" vertical-align:top; padding-left:0; padding-right:0; padding-top:0; padding-bottom:0;">
<p style=" margin-top:12px; margin-bottom:0px; margin-left:0px; margin-right:0px; -qt-block-indent:0; text-indent:0px;">Info. </p>
<p style=" margin-top:12px; margin-bottom:0px; margin-left:0px; margin-right:0px; -qt-block-indent:0; text-indent:0px;">Muestra la ayuda de Waspmonitor. </p></td></tr></table>
<p align="center" style=" margin-top:12px; margin-bottom:12px; margin-left:0px; margin-right:0px; -qt-block-indent:0; text-indent:0px;"> </p>
<p style=" margin-top:12px; margin-bottom:12px; margin-left:0px; margin-right:0px; -qt-block-indent:0; text-indent:0px;">Volver al índice. </p>
<hr width="100%"/>
<p style=" margin-top:0px; margin-bottom:0px; margin-left:0px; margin-right:0px; -qt-block-indent:0; text-indent:0px;">2. Lista de nodos y radioenlaces. </p>
<p style=" margin-top:12px; margin-bottom:12px; margin-left:0px; margin-right:0px; -qt-block-indent:0; text-indent:0px;">Al pulsar sobre alguno de los ítems de estas listas que se ven en la figura, cambia el panel de monitorización al nodo o radioenlace determinado que sea. En concreto, cambian las barras de medida, el histórico de medidas y el panel de alarmas. </p>
<p align="center" style=" margin-top:12px; margin-bottom:12px; margin-left:0px; margin-right:0px; -qt-block-indent:0; text-indent:0px;"> </p>
<p style=" margin-top:12px; margin-bottom:12px; margin-left:0px; margin-right:0px; -qt-block-indent:0; text-indent:0px;">Volver al índice. </p>
<hr width="100%"/>
<p style=" margin-top:0px; margin-bottom:0px; margin-left:0px; margin-right:0px; -qt-block-indent:0; text-indent:0px;">3. Mapa del canal. </p>
<p align="center" style=" margin-top:12px; margin-bottom:12px; margin-left:0px; margin-right:0px; -qt-block-indent:0; text-indent:0px;"> </p>

<p style=" margin-top:12px; margin-bottom:12px; margin-left:0px; margin-right:0px; -qt-block-indent:0; text-indent:0px;">En el mapa del canal se muestran diversos iconos y etiquetas que sirven para tener una visión global del canal y de la red e identificar de manera clara el estado de los nodos y radioenlaces y ver si ha saltado alguna alarma. </p>
<p style=" margin-top:12px; margin-bottom:12px; margin-left:0px; margin-right:0px; -qt-block-indent:0; text-indent:0px;">Los diferentes iconos que pueden aparecer en el mapa son los siguientes: </p>
<p align="center" style="-qt-paragraph-type:empty; margin-top:0px; margin-bottom:0px; margin-left:0px; margin-right:0px; -qt-block-indent:0; text-indent:0px;"></p>
<table border="0" style=" margin-top:0px; margin-bottom:0px; margin-left:0px; margin-right:0px;" align="center" cellspacing="0" cellpadding="0">
<tr>
<td width="187" style=" vertical-align:top; padding-left:0; padding-right:0; padding-top:0; padding-bottom:0;">
<p align="center" style=" margin-top:12px; margin-bottom:0px; margin-left:0px; margin-right:0px; -qt-block-indent:0; text-indent:0px;"> </p></td>
<td width="389" style=" vertical-align:top; padding-left:0; padding-right:0; padding-top:0; padding-bottom:0;">
<p style=" margin-top:12px; margin-bottom:0px; margin-left:0px; margin-right:0px; -qt-block-indent:0; text-indent:0px;">Iconos de estado. </p>
<p style=" margin-top:12px; margin-bottom:0px; margin-left:0px; margin-right:0px; -qt-block-indent:0; text-indent:0px;">En el caso de los nodos pueden tener 3 estados:</p>

<p style=" margin-top:12px; margin-bottom:0px; margin-left:0px; margin-right:0px; -qt-block-indent:0; text-indent:0px;">- Rojo: Nodo caído. </p>
<p style=" margin-top:12px; margin-bottom:0px; margin-left:0px; margin-right:0px; -qt-block-indent:0; text-indent:0px;">- Amarillo: Deshabilitada la monitorización del nodo. </p>
<p style=" margin-top:12px; margin-bottom:0px; margin-left:0px; margin-right:0px; -qt-block-indent:0; text-indent:0px;">- Verde: Nodo funcionando. </p>
<p style=" margin-top:12px; margin-bottom:0px; margin-left:0px; margin-right:0px; -qt-block-indent:0; text-indent:0px;">En el caso de los radioenlaces, sólo pueden ser de color rojo o verde e indican que el radioenlace está funcionando o está caído. </p></td></tr>
<tr>
<td style=" vertical-align:top; padding-left:0; padding-right:0; padding-top:0; padding-bottom:0;">
<p align="center" style=" margin-top:12px; margin-bottom:0px; margin-left:0px; margin-right:0px; -qt-block-indent:0; text-indent:0px;"> </p></td>
<td style=" vertical-align:top; padding-left:0; padding-right:0; padding-top:0; padding-bottom:0;">
<p style=" margin-top:12px; margin-bottom:0px; margin-left:0px; margin-right:0px; -qt-block-indent:0; text-indent:0px;">Iconos de alarma. </p>
<p style=" margin-top:12px; margin-bottom:0px; margin-left:0px; margin-right:0px; -qt-block-indent:0; text-indent:0px;">Están asociados a los nodos e indican: </p>
<p style=" margin-top:12px; margin-bottom:0px; margin-left:0px; margin-right:0px; -qt-block-indent:0; text-indent:0px;">- Batería Baja. </p>
<p style=" margin-top:12px; margin-bottom:0px; margin-left:0px; margin-right:0px; -qt-block-indent:0; text-indent:0px;">- Duty Cycle alto. </p>
<p style=" margin-top:12px; margin-bottom:0px; margin-left:0px; margin-right:0px; -qt-block-indent:0; text-indent:0px;">- Temperatura alta.
</p></td></tr></table>
<p style=" margin-top:12px; margin-bottom:12px; margin-left:0px; margin-right:0px; -qt-block-indent:0; text-indent:0px;">
Volver al índice. </p>
<hr width="100%"/>
<p style=" margin-top:0px; margin-bottom:0px; margin-left:0px; margin-right:0px; -qt-block-indent:0; text-indent:0px;">4. Ventana de log. </p>
<p style=" margin-top:12px; margin-bottom:12px; margin-left:0px; margin-right:0px; -qt-block-indent:0; text-indent:0px;">En la ventana de log, se muestran los mensajes de Keepalive, Estado de Enlace y Alarma, recibidos desde los nodos. </p>
<p align="center" style=" margin-top:12px; margin-bottom:12px; margin-left:0px; margin-right:0px; -qt-block-indent:0; text-indent:0px;"> </p>
<p style=" margin-top:12px; margin-bottom:12px; margin-left:0px; margin-right:0px; -qt-block-indent:0; text-indent:0px;">Volver al índice. </p>
<hr width="100%"/>
<p style=" margin-top:0px; margin-bottom:0px; margin-left:0px; margin-right:0px; -qt-block-indent:0; text-indent:0px;">5. Barras de medida. </p>
<p style=" margin-top:12px; margin-bottom:12px; margin-left:0px; margin-right:0px; -qt-block-indent:0; text-indent:0px;">Hay dos tipos de barra de medida, una para los nodos y otra para los radioenlaces. En el primer caso se muestran los valores de temperatura obtenida por el sensor de temperatura del RTC, el nivel de batería restante en el Waspmote y el Duty Cycle usado por el xBee. Las barras cambian de color, cuando se superan diferentes umbrales, como se puede observar en las figuras. </p>

<p align="center" style=" margin-top:12px; margin-bottom:12px; margin-left:0px; margin-right:0px; -qt-block-indent:0; text-indent:0px;"> </p>
<p align="center" style=" margin-top:12px; margin-bottom:12px; margin-left:0px; margin-right:0px; -qt-block-indent:0; text-indent:0px;"> </p>
<p align="center" style=" margin-top:12px; margin-bottom:12px; margin-left:0px; margin-right:0px; -qt-block-indent:0; text-indent:0px;"> </p>
<p style=" margin-top:12px; margin-bottom:12px; margin-left:0px; margin-right:0px; -qt-block-indent:0; text-indent:0px;"> Volver al índice. </p>
<hr width="100%"/>
<p style=" margin-top:0px; margin-bottom:0px; margin-left:0px; margin-right:0px; -qt-block-indent:0; text-indent:0px;">6. Históricos de medidas. </p>
<p align="center" style=" margin-top:12px; margin-bottom:12px; margin-left:0px; margin-right:0px; -qt-block-indent:0; text-indent:0px;"> </p>
<p style=" margin-top:12px; margin-bottom:12px; margin-left:0px; margin-right:0px; -qt-block-indent:0; text-indent:0px;">Waspmonitor tiene dos históricos de medidas en los que se representan las últimas 120 muestras tomadas cada 15 segundos. Al igual que en el caso de las barras de medida, existe uno para nodos y uno para radioenlaces. En el caso de los nodos, los valores representados son las temperaturas medidas por el RTC y en el caso de los radioenlaces, los RSSI.

Volver al índice. </p>
<hr width="100%"/>
<p style=" margin-top:0px; margin-bottom:0px; margin-left:0px; margin-right:0px; -qt-block-indent:0; text-indent:0px;">7. Paneles de alarma. </p>
<p style=" margin-top:12px; margin-bottom:12px; margin-left:0px; margin-right:0px; -qt-block-indent:0; text-indent:0px;">En el caso de los paneles de alarma, Waspmonitor también tiene paneles distintos para el caso de los nodos y de los radioenlaces. </p>
<p style=" margin-top:12px; margin-bottom:12px; margin-left:0px; margin-right:0px; -qt-block-indent:0; text-indent:0px;"> </p>
<p align="center" style=" margin-top:12px; margin-bottom:12px; margin-left:0px; margin-right:0px; -qt-block-indent:0; text-indent:0px;"> </p>
<p align="center" style=" margin-top:12px; margin-bottom:12px; margin-left:0px; margin-right:0px; -qt-block-indent:0; text-indent:0px;"> </p>
<p style=" margin-top:12px; margin-bottom:12px; margin-left:0px; margin-right:0px; -qt-block-indent:0; text-indent:0px;">Las alarmas de los paneles de alarma son las siguientes: </p>
<p style=" margin-top:12px; margin-bottom:12px; margin-left:0px; margin-right:0px; -qt-block-indent:0; text-indent:0px;">• Nodo caído: Se indica cuando se recibe un mensaje de Alarma por Caída de Enlace. Al caer un enlace, el nodo de destino se supone que está caído. También se produce cuando se sobrepasa un tiempo de 120 segundos sin recibir ningún mensaje de Keepalive desde dicho nodo. </p>
<p style=" margin-top:12px; margin-bottom:12px; margin-left:0px; margin-right:0px; -qt-block-indent:0; text-indent:0px;">• Batería Baja: Se indica cuando se recibe un mensaje de Alarma por Batería Baja. </p>
<p style=" margin-top:12px; margin-bottom:12px; margin-left:0px; margin-right:0px; -qt-block-indent:0; text-indent:0px;">• Duty Cycle: Se indica cuando se recibe un mensaje de Alarma por Duty Cycle alto. </p>
<p style=" margin-top:12px; margin-bottom:12px; margin-left:0px; margin-right:0px; -qt-block-indent:0; text-indent:0px;">• Temperatura alta: Se indica cuando se recibe un mensaje de Alarma por Temperatura alta. </p>
<p style=" margin-top:12px; margin-bottom:12px; margin-left:0px; margin-right:0px; -qt-block-indent:0; text-indent:0px;">• Radioenlace caído: Se indica cuando se recibe un mensaje de Alarma por Radioenlace Caído. También se activa esta alarma cuando transcurren 120 segundos sin recibir ningún mensaje de Keepalive desde el nodo origen del radioenlace (se activará esta alarma en todos los radioenlaces asociados al nodo caído). </p>
<p style=" margin-top:12px; margin-bottom:12px; margin-left:0px; margin-right:0px; -qt-block-indent:0; text-indent:0px;">
Volver al índice. </p>
<hr width="100%"/>
<p style=" margin-top:0px; margin-bottom:0px; margin-left:0px; margin-right:0px; -qt-block-indent:0; text-indent:0px;">8. Configuración de Waspmonitor. </p>
<p style=" margin-top:12px; margin-bottom:12px; margin-left:0px; margin-right:0px; -qt-block-indent:0; text-indent:0px;">Al pulsar en el botón con un icono de llave inglesa se accede al menú de configuración de Waspmonitor. </p>
<p align="center" style=" margin-top:12px; margin-bottom:12px; margin-left:0px; margin-right:0px; -qt-block-indent:0; text-indent:0px;"> </p>
<p style=" margin-top:12px; margin-bottom:12px; margin-left:0px; margin-right:0px; -qt-block-indent:0; text-indent:0px;">Antes de empezar a capturar ningún dato, se recomienda configurar Waspmonitor correctamente. En especial es fundamental seleccionar el puerto COM adecuado para el Waspmote Gateway. En caso de elegir uno erróneo, Waspmonitor indicará un error de apertura de puerto. También se recomienda deshabilitar la monitorización de los nodos que no se vayan a emplear, para evitar que salten alarmas falsas. </p>
<p style=" margin-top:12px; margin-bottom:12px; margin-left:0px; margin-right:0px; -qt-block-indent:0; text-indent:0px;">Las opciones de configuración son las siguientes: </p>
<p style=" margin-top:12px; margin-bottom:12px; margin-left:0px; margin-right:0px; -qt-block-indent:0; text-indent:0px;">• Habilitación/deshabilitación de monitorización de nodos. </p>

<p style=" margin-top:12px; margin-bottom:12px; margin-left:0px; margin-right:0px; -qt-block-indent:0; text-indent:0px;">-
Deshabilitar la monitorización implica que Waspmonitor no hará saltar las alarmas si pasan 120 segundos sin recibir un mensaje de Keepalive desde uno de los nodos. </p>
<p style=" margin-top:12px; margin-bottom:12px; margin-left:0px; margin-right:0px; -qt-block-indent:0; text-indent:0px;">Por defecto están habilitados todos los nodos. </p>
<p style=" margin-top:12px; margin-bottom:12px; margin-left:0px; margin-right:0px; -qt-block-indent:0; text-indent:0px;">• Habiltación/deshabilitación de logs. </p>
<p style=" margin-top:12px; margin-bottom:12px; margin-left:0px; margin-right:0px; -qt-block-indent:0; text-indent:0px;">-
Waspmonitor puede crear archivos de log en formato .CSV (Comma-Separated Values) y HTML donde se guarden todos los mensajes recibidos desde los nodos. </p>
<p style=" margin-top:12px; margin-bottom:12px; margin-left:0px; margin-right:0px; -qt-block-indent:0; text-indent:0px;">Por defecto están habilitados los logs en .CSV y en .HTML. </p>
<p style=" margin-top:12px; margin-bottom:12px; margin-left:0px; margin-right:0px; -qt-block-indent:0; text-indent:0px;">• Selección de puerto COM para el Gateway. </p>
<p style=" margin-top:12px; margin-bottom:12px; margin-left:0px; margin-right:0px; -qt-block-indent:0; text-indent:0px;">-
Para que Waspmonitor pueda comunicarse con el Waspmote Gateway es necesario elegir el puerto serie COM adecuado. </p>
<p style=" margin-top:12px; margin-bottom:12px; margin-left:0px; margin-right:0px; -qt-block-indent:0; text-indent:0px;">Por defecto, el puerto serie elegido es COM1, pero es necesario elegir correctamente el puerto antes de iniciar la monitorización, ya que el puerto COM cambia a menudo al conectar y desconectar el Waspmote Gateway. </p>
<p style=" margin-top:12px; margin-bottom:12px; margin-left:0px; margin-right:0px; -qt-block-indent:0; text-indent:0px;">Volver al índice. </p>
<hr width="100%"/>
<p style=" margin-top:0px; margin-bottom:0px; margin-left:0px; margin-right:0px; -qt-block-indent:0; text-indent:0px;">9. Cuadros de diálogo. </p>
<p style=" margin-top:12px; margin-bottom:12px; margin-left:0px; margin-right:0px; -qt-block-indent:0; text-indent:0px;">Waspmonitor dispone de diversos cuadros de diálogo que informan de errores, dan advertencias, etc. sobre eventos que ocurran durante la ejecución del programa. </p>
<p align="center" style=" margin-top:12px; margin-bottom:12px; margin-left:0px; margin-right:0px; -qt-block-indent:0; text-indent:0px;"> </p>
<p style=" margin-top:12px; margin-bottom:12px; margin-left:0px; margin-right:0px; -qt-block-indent:0; text-indent:0px;">Los diferentes cuadros de diálogo que pueden aparecer en Waspmonitor son estos: </p>
<p style=" margin-top:12px; margin-bottom:12px; margin-left:0px; margin-right:0px; -qt-block-indent:0; text-indent:0px;">• Advertencias: </p>
<p style=" margin-top:12px; margin-bottom:12px; margin-left:0px; margin-right:0px; -qt-block-indent:0; text-indent:0px;">-
Proceso de captura ya iniciado: Aparece si se pulsa más de una vez el botón Iniciar Captura ya que si la captura ya está en curso, no se puede volver a iniciar, hay que detenerla antes con el botón Detener Captura. </p>
<p style=" margin-top:12px; margin-bottom:12px; margin-left:0px; margin-right:0px; -qt-block-indent:0; text-indent:0px;">-
Salir del programa: Al pulsar el botón de Salir o la X de la ventana, aparece este cuadro de diálogo. Se detiene el proceso de captura y se cierra el puerto serie. </p>
<p style=" margin-top:12px; margin-bottom:12px; margin-left:0px; margin-right:0px; -qt-block-indent:0; text-indent:0px;">-
No se puede cambiar el puerto mientras la captura está en curso. Aparece al intentar cambiar el puerto COM en la ventana de configuración cuando la captura está iniciada. </p>
<p style=" margin-top:12px; margin-bottom:12px; margin-left:0px; margin-right:0px; -qt-block-indent:0; text-indent:0px;">-
No se puede abrir el archivo de log CSV o HTML. Aparecen si hay algún tipo de problema al abrir los archivos de log por ejemplo, porque el disco duro esté lleno. </p>
<p style=" margin-top:12px; margin-bottom:12px; margin-left:0px; margin-right:0px; -qt-block-indent:0; text-indent:0px;">• Errores críticos: </p>
<p style=" margin-top:12px; margin-bottom:12px; margin-left:0px; margin-right:0px; -qt-block-indent:0; text-indent:0px;">-
Error al intentar abrir el puerto serie. Este cuadro de diálogo se muestra si el puerto serie elegido en la ventana de configuración es incorrecto, si el Waspmote no está conectado o está conectado pero apagado. También puede aparecer si existe algún otro tipo de error al intentar abrir el puerto serie. </p>
<p style=" margin-top:12px; margin-bottom:12px; margin-left:0px; margin-right:0px; -qt-block-indent:0; text-indent:0px;"> </p>
<p style=" margin-top:12px; margin-bottom:12px; margin-left:0px; margin-right:0px; -qt-block-indent:0; text-indent:0px;">Volver al índice. </p>
<hr width="100%"/>
<p style=" margin-top:0px; margin-bottom:0px; margin-left:0px; margin-right:0px; -qt-block-indent:0; text-indent:0px;">10. Barra de Estado. </p>

```
&lt;p style=&quot; margin-top:12px; margin-bottom:12px; margin-left:0px; margin-right:0px; -qt-block-indent:0; text-
indent:0px;&quot;&gt;La Barra de Estado es la zona inferior de la ventana principal de Waspmonitor en la que se muestran
mensajes de carácter informativo cuando suceden distintos eventos o cambios en el programa. &lt;/p&gt;
&lt;p align=&quot;center&quot; style=&quot; margin-top:12px; margin-bottom:12px; margin-left:0px; margin-right:0px; -qt-block-
indent:0; text-indent:0px;&quot;&gt;&lt;img src=&quot;index_archivos/image027.png&quot; /&gt; &lt;/p&gt;
&lt;p style=&quot; margin-top:12px; margin-bottom:12px; margin-left:0px; margin-right:0px; -qt-block-indent:0; text-
indent:0px;&quot;&gt;  &lt;/p&gt;
&lt;p style=&quot; margin-top:12px; margin-bottom:12px; margin-left:0px; margin-right:0px; -qt-block-indent:0; text-
indent:0px;&quot;&gt;Los diferentes mensajes que se muestran  son: &lt;/p&gt;
&lt;p style=&quot; margin-top:12px; margin-bottom:12px; margin-left:0px; margin-right:0px; -qt-block-indent:0; text-
indent:0px;&quot;&gt;&lt;span style=&quot; font-family:'Symbol';&quot;&gt;•&lt;/span&gt;         &lt;span style=&quot; font-
weight:600;&quot;&gt;Mensajes informativos&lt;/span&gt;. Puerto abierto/cerrado, captura iniciada, etc. &lt;/p&gt;
&lt;p style=&quot; margin-top:12px; margin-bottom:12px; margin-left:0px; margin-right:0px; -qt-block-indent:0; text-
indent:0px;&quot;&gt;&lt;span style=&quot; font-family:'Symbol';&quot;&gt;•&lt;/span&gt;         &lt;span style=&quot; font-
weight:600;&quot;&gt;Mensajes de error&lt;/span&gt;. Error al abrir el puerto, etc. &lt;/p&gt;
&lt;p style=&quot; margin-top:12px; margin-bottom:12px; margin-left:0px; margin-right:0px; -qt-block-indent:0; text-
indent:0px;&quot;&gt;&lt;span style=&quot; font-family:'Symbol';&quot;&gt;•&lt;/span&gt;         &lt;span style=&quot; font-
weight:600;&quot;&gt;Mensajes de cambio de configuración&lt;/span&gt;. Habilitación/deshabilitación de monitorización de nodos,
etc. &lt;/p&gt;
&lt;p style=&quot; margin-top:12px; margin-bottom:12px; margin-left:0px; margin-right:0px; -qt-block-indent:0; text-
indent:0px;&quot;&gt; &lt;a href=&quot;#Indice&quot;&gt;&lt;span style=&quot; font-size:14pt; font-weight:600; text-decoration:
underline; color:#0000ff;&quot;&gt;Volver al índice.&lt;/span&gt;&lt;/a&gt; &lt;/p&gt;&lt;/body&gt;&lt;/html&gt;</string>
     </property>
     <property name="source">
      <url>
       <string>qrc:/ayuda.html</string>
      </url>
     </property>
    </widget>
   </item>
  </layout>
 </widget>
 <resources/>
 <connections/>
</ui>
```

Showmonitor.ui

```
<?xml version="1.0" encoding="UTF-8"?>
<ui version="4.0">
 <class>showMonitor</class>
 <widget class="QWidget" name="showMonitor">
  <property name="geometry">
   <rect>
    <x>0</x>
    <y>0</y>
    <width>1218</width>
    <height>683</height>
   </rect>
  </property>
  <property name="windowTitle">
   <string>Form</string>
  </property>
  <layout class="QVBoxLayout" name="verticalLayout_5">
   <item>
    <widget class="QSplitter" name="splitter">
     <property name="orientation">
      <enum>Qt::Horizontal</enum>
     </property>
     <widget class="QFrame" name="frame">
      <property name="maximumSize">
       <size>
        <width>240</width>
        <height>16777215</height>
       </size>
      </property>
      <property name="frameShape">
       <enum>QFrame::StyledPanel</enum>
      </property>
      <property name="frameShadow">
       <enum>QFrame::Raised</enum>
      </property>
      <layout class="QVBoxLayout" name="verticalLayout_4">
       <property name="spacing">
        <number>2</number>
       </property>
       <property name="margin">
        <number>2</number>
       </property>
       <item>
```

```
 <widget class="QTreeWidget" name="treeWidget">
  <column>
   <property name="text">
    <string>Nodos</string>
   </property>
  </column>
  <item>
   <property name="text">
    <string>Ctra. Utrera</string>
   </property>
  </item>
  <item>
   <property name="text">
    <string>C. Maribáñez</string>
   </property>
  </item>
  <item>
   <property name="text">
    <string>San Fernando</string>
   </property>
  </item>
  <item>
   <property name="text">
    <string>Oficina</string>
   </property>
  </item>
  <item>
   <property name="text">
    <string>Alcantarillas</string>
   </property>
  </item>
 </widget>
</item>
<item>
 <widget class="QTreeWidget" name="treeWidget_2">
  <column>
   <property name="text">
    <string>Radioenlaces</string>
   </property>
  </column>
  <item>
   <property name="text">
    <string>Primarios</string>
   </property>
   <item>
    <property name="text">
     <string>Ctra. Utrera - C. Maribáñez</string>
    </property>
   </item>
   <item>
    <property name="text">
     <string>C. Maribáñez - San Fernando</string>
    </property>
   </item>
   <item>
    <property name="text">
     <string>San Fernando - Oficina</string>
    </property>
   </item>
   <item>
    <property name="text">
     <string>Oficina - Alcantarillas</string>
    </property>
   </item>
  </item>
  <item>
   <property name="text">
    <string>Secundarios</string>
   </property>
   <item>
    <property name="text">
     <string>Ctra. Utrera - San Fernando</string>
    </property>
   </item>
   <item>
    <property name="text">
     <string>Ctra. Utrera - Oficina</string>
    </property>
   </item>
   <item>
    <property name="text">
     <string>C. Maribáñez - Oficina</string>
    </property>
   </item>
```

```
    </item>
   </widget>
  </item>
 </layout>
</widget>
<widget class="QFrame" name="frame_2">
 <property name="frameShape">
  <enum>QFrame::StyledPanel</enum>
 </property>
 <property name="frameShadow">
  <enum>QFrame::Raised</enum>
 </property>
 <layout class="QVBoxLayout" name="verticalLayout">
  <property name="margin">
   <number>2</number>
  </property>
  <item>
   <layout class="QHBoxLayout" name="horizontalLayout">
    <item>
     <widget class="QFrame" name="frame_3">
      <property name="frameShape">
       <enum>QFrame::StyledPanel</enum>
      </property>
      <property name="frameShadow">
       <enum>QFrame::Raised</enum>
      </property>
      <layout class="QVBoxLayout" name="verticalLayout_3">
       <property name="spacing">
        <number>1</number>
       </property>
       <property name="leftMargin">
        <number>2</number>
       </property>
       <property name="topMargin">
        <number>1</number>
       </property>
       <property name="rightMargin">
        <number>2</number>
       </property>
       <property name="bottomMargin">
        <number>1</number>
       </property>
       <item>
        <widget class="QGroupBox" name="groupBox">
         <property name="title">
          <string>Esquema de la red</string>
         </property>
        </widget>
       </item>
       <item>
        <widget class="QGroupBox" name="groupBox_2">
         <property name="title">
          <string>Mensajes recibidos</string>
         </property>
        </widget>
       </item>
      </layout>
     </widget>
    </item>
    <item>
     <widget class="QFrame" name="frame_4">
      <property name="frameShape">
       <enum>QFrame::StyledPanel</enum>
      </property>
      <property name="frameShadow">
       <enum>QFrame::Raised</enum>
      </property>
      <layout class="QVBoxLayout" name="verticalLayout_2">
       <property name="spacing">
        <number>1</number>
       </property>
       <property name="leftMargin">
        <number>2</number>
       </property>
       <property name="topMargin">
        <number>1</number>
       </property>
       <property name="rightMargin">
        <number>2</number>
       </property>
       <property name="bottomMargin">
        <number>1</number>
       </property>
       <item>
```

```
 <widget class="QGroupBox" name="groupBox_3">
  <property name="minimumSize">
   <size>
    <width>464</width>
    <height>217</height>
   </size>
  </property>
  <property name="maximumSize">
   <size>
    <width>16777215</width>
    <height>350</height>
   </size>
  </property>
  <property name="title">
   <string>Niveles en tiempo real  (Nodo Ctra. Utrera)</string>
  </property>
  <layout class="QVBoxLayout" name="verticalLayout_6">
   <property name="spacing">
    <number>0</number>
   </property>
   <property name="margin">
    <number>0</number>
   </property>
   <item>
    <widget class="QStackedWidget" name="stackedWidget">
     <property name="currentIndex">
      <number>0</number>
     </property>
     <widget class="QWidget" name="page_18"/>
     <widget class="QWidget" name="page_19"/>
     <widget class="QWidget" name="page_20"/>
     <widget class="QWidget" name="page_21"/>
     <widget class="QWidget" name="page_22"/>
     <widget class="QWidget" name="page_23"/>
     <widget class="QWidget" name="page_24"/>
     <widget class="QWidget" name="page_25"/>
     <widget class="QWidget" name="page_26"/>
     <widget class="QWidget" name="page_27"/>
     <widget class="QWidget" name="page_28"/>
     <widget class="QWidget" name="page_29"/>
    </widget>
   </item>
  </layout>
 </widget>
</item>
<item>
 <widget class="QGroupBox" name="groupBox_4">
  <property name="minimumSize">
   <size>
    <width>464</width>
    <height>217</height>
   </size>
  </property>
  <property name="title">
   <string>Histórico  (Nodo Ctra. Utrera)</string>
  </property>
  <layout class="QVBoxLayout" name="verticalLayout_7">
   <item>
    <widget class="QStackedWidget" name="stackedWidget_2">
     <property name="currentIndex">
      <number>0</number>
     </property>
     <widget class="QWidget" name="page_1"/>
     <widget class="QWidget" name="page_2"/>
     <widget class="QWidget" name="page_3"/>
     <widget class="QWidget" name="page_4"/>
     <widget class="QWidget" name="page_5"/>
     <widget class="QWidget" name="page_6"/>
     <widget class="QWidget" name="page_7"/>
     <widget class="QWidget" name="page_8"/>
     <widget class="QWidget" name="page_9"/>
     <widget class="QWidget" name="page_10"/>
     <widget class="QWidget" name="page_11"/>
     <widget class="QWidget" name="page_12"/>
    </widget>
   </item>
  </layout>
 </widget>
</item>
<item>
 <widget class="QGroupBox" name="groupBox_5">
  <property name="minimumSize">
   <size>
    <width>464</width>
```

```
                <height>217</height>
               </size>
              </property>
              <property name="title">
               <string>Alarmas  (Nodo Ctra. Utrera)</string>
              </property>
              <layout class="QVBoxLayout" name="verticalLayout_8">
               <item>
                <widget class="QStackedWidget" name="stackedWidget_3">
                 <widget class="QWidget" name="page_30"/>
                 <widget class="QWidget" name="page_31"/>
                 <widget class="QWidget" name="page_32"/>
                 <widget class="QWidget" name="page_33"/>
                 <widget class="QWidget" name="page_34"/>
                 <widget class="QWidget" name="page_35"/>
                 <widget class="QWidget" name="page_36"/>
                 <widget class="QWidget" name="page_37"/>
                 <widget class="QWidget" name="page_38"/>
                 <widget class="QWidget" name="page_39"/>
                 <widget class="QWidget" name="page_40"/>
                 <widget class="QWidget" name="page_41"/>
                </widget>
               </item>
              </layout>
             </widget>
            </item>
           </layout>
          </widget>
         </item>
        </layout>
       </item>
      </layout>
     </widget>
    </widget>
   </item>
  </layout>
 </widget>
 <resources/>
 <connections/>
</ui>
```

Waspmonitor.ui

```
<?xml version="1.0" encoding="UTF-8"?>
<ui version="4.0">
 <class>WaspMonitor</class>
 <widget class="QMainWindow" name="WaspMonitor">
  <property name="geometry">
   <rect>
    <x>0</x>
    <y>0</y>
    <width>1280</width>
    <height>720</height>
   </rect>
  </property>
  <property name="sizePolicy">
   <sizepolicy hsizetype="Preferred" vsizetype="Preferred">
    <horstretch>0</horstretch>
    <verstretch>0</verstretch>
   </sizepolicy>
  </property>
  <property name="windowTitle">
   <string>WaspMonitor</string>
  </property>
  <widget class="QWidget" name="centralWidget">
   <layout class="QVBoxLayout" name="verticalLayout">
    <property name="margin">
     <number>0</number>
    </property>
    <item>
     <widget class="QStackedWidget" name="stackedWidget">
      <property name="currentIndex">
       <number>1</number>
      </property>
      <widget class="showInfo" name="page1"/>
      <widget class="showMonitor" name="page3"/>
     </widget>
    </item>
   </layout>
  </widget>
  <widget class="QToolBar" name="mainToolBar">
```

```
 <property name="movable">
  <bool>false</bool>
 </property>
 <property name="iconSize">
  <size>
   <width>40</width>
   <height>40</height>
  </size>
 </property>
 <property name="toolButtonStyle">
  <enum>Qt::ToolButtonTextUnderIcon</enum>
 </property>
 <attribute name="toolBarArea">
  <enum>TopToolBarArea</enum>
 </attribute>
 <attribute name="toolBarBreak">
  <bool>false</bool>
 </attribute>
 <addaction name="actionSalir"/>
 <addaction name="separator"/>
 <addaction name="actionIniciarCaptura"/>
 <addaction name="actionDetenerCaptura"/>
 <addaction name="separator"/>
 <addaction name="actionMonitorizacion"/>
 <addaction name="actionLimpiarAlarmas"/>
 <addaction name="separator"/>
 <addaction name="actionConfiguracion"/>
 <addaction name="actionInfo"/>
</widget>
<widget class="QStatusBar" name="statusBar"/>
<action name="actionSalir">
 <property name="icon">
  <iconset resource="waspmonitor.qrc">
   <normaloff>:/Salida.png</normaloff>:/Salida.png</iconset>
 </property>
 <property name="text">
  <string>Salir</string>
 </property>
 <property name="toolTip">
  <string>Salir del programa</string>
 </property>
</action>
<action name="actionIniciarCaptura">
 <property name="icon">
  <iconset resource="waspmonitor.qrc">
   <normaloff>:/Start2.png</normaloff>:/Start2.png</iconset>
 </property>
 <property name="text">
  <string>Iniciar Captura</string>
 </property>
 <property name="toolTip">
  <string>Iniciar captura de datos</string>
 </property>
</action>
<action name="actionDetenerCaptura">
 <property name="icon">
  <iconset resource="waspmonitor.qrc">
   <normaloff>:/Stop2.png</normaloff>:/Stop2.png</iconset>
 </property>
 <property name="text">
  <string>Detener Captura</string>
 </property>
 <property name="toolTip">
  <string>Detener captura de datos</string>
 </property>
</action>
<action name="actionMonitorizacion">
 <property name="icon">
  <iconset resource="waspmonitor.qrc">
   <normaloff>:/Activity Monitor.png</normaloff>:/Activity Monitor.png</iconset>
 </property>
 <property name="text">
  <string>Monitorización</string>
 </property>
 <property name="toolTip">
  <string>Monitorizacion en tiempo real</string>
 </property>
</action>
<action name="actionInfo">
 <property name="icon">
  <iconset resource="waspmonitor.qrc">
   <normaloff>:/About.png</normaloff>:/About.png</iconset>
 </property>
 <property name="text">
```

```
    <string>Info</string>
   </property>
   <property name="toolTip">
    <string>Información</string>
   </property>
  </action>
  <action name="actionLimpiarAlarmas">
   <property name="icon">
    <iconset resource="waspmonitor.qrc">
     <normaloff>:/refresh_alarm.png</normaloff>:/refresh_alarm.png</iconset>
   </property>
   <property name="text">
    <string>Limpiar Alarmas</string>
   </property>
   <property name="toolTip">
    <string>Limpia todas las alarmas</string>
   </property>
  </action>
  <action name="actionConfiguracion">
   <property name="icon">
    <iconset resource="waspmonitor.qrc">
     <normaloff>:/settings.png</normaloff>:/settings.png</iconset>
   </property>
   <property name="text">
    <string>Configuración</string>
   </property>
   <property name="toolTip">
    <string>Configuración de Waspmonitor</string>
   </property>
  </action>
 </widget>
 <layoutdefault spacing="6" margin="11"/>
 <customwidgets>
  <customwidget>
   <class>showInfo</class>
   <extends>QWidget</extends>
   <header>showinfo.h</header>
   <container>1</container>
  </customwidget>
  <customwidget>
   <class>showMonitor</class>
   <extends>QWidget</extends>
   <header>showmonitor.h</header>
   <container>1</container>
  </customwidget>
 </customwidgets>
 <resources>
  <include location="waspmonitor.qrc"/>
 </resources>
 <connections/>
</ui>
```