COLEGIO DE POSTGRADUADOS

INSTITUCIÓN DE ENSEÑANZA E INVESTIGACIÓN EN CIENCIAS AGRÍCOLAS

CAMPUS MONTECILLO

POSTGRADO EN HIDROCIENCIAS

"AUTOMATIZACIÓN DEL RIEGO CON BASE EN DISPOSITIVOS MÓVILES"

OZIEL LUGO ESPINOSA

TESIS

PRESENTADA COMO REQUISITO PARCIAL PARA OBTENER EL GRADO DE:

DOCTOR EN CIENCIAS

MONTECILLO, TEXCOCO, EDO. DE MÉXICO

2010

La presente tesis, titulada: **"Automatización del riego con base en dispositivos móviles",** realizada por el alumno: **Oziel Lugo Espinosa**, bajo la dirección del Consejo Particular indicado, ha sido aprobada por el mismo y aceptada como requisito parcial para obtener el grado de:

DOCTOR EN CIENCIAS

HIDROCIENCIAS

CONSEJO PARTICULAR

CONSEJERO Y DIRECTOR DE TESIS:

DR. ABEL QUEVEDO NOLASCO

ASESOR:

DR. JUAN R. BAUER MENGELBERG

ASESOR:

DR. ENRIQUE PALACIOS VÉLEZ

ASESOR:

DR. DAVID HEBERT DEL VALLE PANIAGUA

ASESOR:

DR. MARTINIANO CASTRO POPOCA

Montecillo, Texcoco, Edo. De México; Junio de 2010

RESUMEN

El riego proporciona a la planta el agua imprescindible para su crecimiento y desarrollo. Para garantizar el riego adecuado, pero aprovechando al máximo los caudales de agua disponibles, se pueden utilizar sistemas de riego con programación de control automático, para asegurar el suministro cuando y en la cantidad que la planta necesite, para evitar el desperdicio de agua. Se trata de sistemas que activan el riego espontáneamente, en función de la valoración continua de uno o varios parámetros o variables de control. Este trabajo presenta un prototipo para automatizar el riego en una zona con distintos cultivos, donde cada uno de ellos recibe la cantidad de agua necesaria en el momento que se requiera para un óptimo crecimiento. El prototipo consiste en una integración de tres tecnologías, computación, comunicaciones y electrónica. El área de influencia es una zona geográfica con condiciones meteorológicas similares, dado que el sistema usa las variables meteorológicas temperatura del aire, humedad relativa, velocidad del viento, precipitación y radiación global. También se le proporciona la información del suelo (porcentaje contenido de arena, arcilla, materia orgánica y densidad aparente): y de los cultivos (tipo, duración, funciones de crecimiento radicular y Kc). Con esta información se elabora un balance hídrico, el cual resulta en la decisión de regar o no alguno de los cultivos. Cuando, como resultado de la función de abatimiento del agua en el suelo en conjunción con el balance hídrico, se determina que se debe regar, el software envía señales digitales de salida por el puerto serial RS232 de una computadora hacia un dispositivo electrónico, para accionar los dispositivos físicos de riego y complementarios (v.gr. electroválvulas, ventiladores, lámparas.) Se integró un subsistema de comunicación bidireccional PC-Modem GSM-Celular que permite encender/apagar un dispositivo mediante comandos remotos desde un teléfono celular, como también el envío de alarmas y avisos que indican

el estado actual del sistema por medio de mensajes de texto vía celular y correos electrónicos. El sistema de automatización se diseñó en módulos, lo que permite que se adapte a diversas aplicaciones que sean susceptibles de automatización y control. Se describen los dispositivos y programas que componen el sistema, y las entradas, salidas y proceso que controlan el riego automatizado.

Palabras clave: automatización, riego, balance hídrico, multicultivo.

ABSTRACT

Irrigation is vital for the plants growth and development. To ensure adequate irrigation, but contemplating the most efficient use of available water, an automated irrigation system can be used. It consists of a programmed automatic control which will furnish the plant with the necessary quantity of water, at the appropriate times, so that the waste of water is minimized. A system will spontaneously activate the irrigation devices as a function of the continuous stream of information regarding one or more control parameters or variables. This work presents a prototype to automatize the irrigation at a zone with several cultivations, where each one of them receives the quantity of necessary water the moment that it take to an optimal growth. The prototype consists in an integration of three technologies, computation, communications and electronics. Since the system uses prevailing meteorological variables, including temperature, relative humidity, wind velocity, precipitation and global radiation, the area of influence must be a geographic zone with similar climate conditions. Other variables used by the model are information about soils (percentages of sand, clay, organic materials content, and the apparent density) as well as information about the crops, consisting in type, duration, functions of root growth and Kc.) A hydric balance computed with all this information will result in the decision to water or not some of the crops. Whenever the result of the water abatement in the ground, in conjunction with the hydric balance, determines that water should be provided, the software sends a digital signal via a computer's RS232 serial port to an electronic device, to activate the suitable physical irrigation devices (e.g. solenoid valves, fans and lamps.) A bidirectional communication system from a PC-Modem to a GSM-cell phone can turn a device on or off, through remote commands sent from a cell phone, as well as text messages or emails conveying an alarm or notice describing the current state of the system.

The system was designed in modules, so that it may be applied to different situations which involve automatization and control. The devices and programs, as well as inputs and outputs of the system, are described in detail.

Keywords: automatization, irrigation, hydric balance, multi-crop.

DEDICATORIA

Dedico esta tesis:

A mi esposa:

Samantha A. Díaz Luna que comparte a mi lado cada triunfo y tropiezo (laboral y personal) y que gracias al apoyo y comprensión que me brinda, día a día, los triunfos van aumentando…

A mi hija:

Nicole Lugo Díaz que es mi orgullo e inspiración. En conjunto con mi esposa, son todo para mi.

A mis padres y hermanas:

Que son parte importante de mi vida y que siempre están presentes y dispuestos a ayudarme cuando lo necesito.

A mi nueva familia, compañeros y amigos

mil gracias

AGRADECIMIENTOS

Deseo expresar mi más sincero agradecimiento al Consejo Nacional de Ciencia y Tecnología y al Colegio de postgraduados por haberme dado la oportunidad de continuar con mi formación académica.

Al programa de Hidrociencias, personal académico y administrativo por el apoyo y facilidades otorgadas durante mi permanencia en el posgrado.

Al Dr. Abel Quevedo Nolasco por su dirección y persistencia para la realización de este trabajo. Hago incapié en la calidad de ser humano que posee el Dr. al hacer suyos los problemas prefesionales y personales del alumno ofreciendo su ayuda y apoyo incondiconal.

Al Dr. Juan R. Bauer Mangelberg que ofrece ayuda incondiconal al alumno haciendo gala de su experiencia y capacidad de abstracción de los problemas para brindar la mejor solución al problema que se plantea.

Al Dr. David H. del Valle Paniagua por su disposición, interés y apoyo para el desarrollo del presente trabajo de investigación.

Al Dr. Enrique Palacios Vélez por su atención, disposición y valiosos comentarios para la realización de la investigación.

Al Dr. Francisco Miguel Águila Marín que me apoyó al inicio de la investigación dando ideas y comentarios para definir el rumbo del proyecto.

Al Dr. Martiniano Castro Popoca por su valioso apoyo para comprender de manera inicial la magnitud del problema y soluciones que fueron llevadas a cabo en este trabajo de investigación.

A Samantha Díaz, Gilberto Lugo, Luis Carlos Miranda Trujillo, Ricardo Luna, Ricardo Campos, Arturo Gamalier, Pedro Ríos, entre otros, que ayudaron de forma directa para la realización del prototipo de invernadero para mostrar el concepto de la investigación, así también como el desarrollo de las interfaces electrónicas para la comunicación de la computadora con los diferentes dispositivos de riego.

CONTENIDO

ARTÍCULO 1.

PROTOTIPO PARA AUTOMATIZAR UN SISTEMA DE RIEGO MULTICULTIVO…….... 13

ARTÍCULO 2.

TECNOLOGÍA GSM COMO MÓDULO DE COMUNICACIÓN EN UN PROTOTIPO DE AUTOMATIZACIÓN DE RIEGO ……37

ARTÍCULO 3.

CONTROL ELECTRÓNICO DE DISPOSITIVOS DE RIEGO POR MEDIO DE UNA COMPUTADORA PERSONAL……...64

CONCLUSIONES Y RECOMENDACIONES……..….86

LITETURA CITADA……88

ANEXOS

A: CÓDIGO FUENTE SISTEMA RAI……..… 93

INDICE DE FIGURAS

ARTÍCULO 1

Figura 1. Diagrama general del sistema de riego automático inteligente 26
Figura 2. Diálogo principal de riego automático inteligente (RAI) 27
Figura 3. Diálogo de captura de información de cultivo y suelo 28
Figura 4. Dialógos para mostrar la información meteorológica 29
Figura 4a. Datos actuales 29
Figura 4b. Datos históricos 29
Figura 4c. Consulta de variables meteorológicas 29
Figura 5. Prototipo desarrollado 31
Figura 5a. Interface de comunicación de salida de la PC 31
Figura 5b. Interface de potencia comunicación RAI,Modem-celular y celular 31
Figura 6. Pantallas de usuario en el celular 32
Figura 6a. Diálogo Principal de RAI en el celular bajo Windows Móvil 32
Figura 6b. Configuración de RAI 32
Figura 6c. Cosultas de campos o de puertos 32
Figura 6d. Envío de comandos 32
Figura 6e. Alarmas 32
Figura 6f. Avisos del celular a una cuenta de correo electrónico 32

ARTÍCULO 2

Figura 1. Envío de SMS desde la hyper-terminal de Windows 52
Figura 2. Pantallas de usuario en el celular 59
Figura 2a. Diálogo Principal de RAI en el celular bajo Windows Móvil 59
Figura 2b. Configuración de RAI 59
Figura 2c. Cosultas de campos o de puertos 59

Figura 2d. Envío de comandos 59
Figura 2e. Alarmas 59
Figura 2f. Avisos del celular a una cuenta de correo electrónico 59

ARTÍCULO 3

Figura 1. Diagrama de bloques de un sistema de control 68
Figura 2. Diagrama de bloques de una planta 68
Figura 3a. Sistema de control en lazo abierto 68
Figura 3b. Sistema de control en lazo cerrado 68
Figura 4. Sistema de control automático de temperatura 69
Figura 5. Circuito electrónico 75
Figura 6. Interface de potencia y optoacoplador 76
Figura 7. Ventana de ejemplo del software de grabación PIC 600 77
Figura 8. Grabador de microcontroladores PIC 600 77
Figura 9. PIC 16F88 78
Figura 10. Regulador de corriente 78
Figura 11. Max232. Convertidor de voltaje para el puerto serial RS232 79
Figura 12. PIC 16F88 79
Figura 13. Diagrama de flujo del programa en el PIC 83

Agricultura Técnica en México

REVISTA AGRICULTURA TÉCNICA EN MÉXICO
INSTITUTO NACIONAL DE INVESTIGACIONES FORESTALES, AGRÍCOLAS Y PECUARIAS
km 18.5 Carr. Los Reyes-Lechería 56230 Texcoco, Edo. de Méx.
Apdo. Postal 10. 56230 Chapingo, Texcoco, Edo. de Méx.
e-mail: revista_atm@yahoo.com.mx
Tel. y Fax: 01 595 95 4 29 64

inifap

Chapingo, Edo. de México, 20 de enero de 2010

M.C. OZIEL LUGO ESPINOSA
COLEGIO DE POSTGRADUADOS
EN CIENCIAS AGRÍCOLAS
PRESENTE:

Por este medio agradezco y acuso de recibo su artículo: **"AUTOMATIZACIÓN DEL RIEGO EN TIEMPO REAL PARA SISTEMA MULTICULTIVO",** que fue enviado para su posible publicación a la revista Agricultura Técnica en México. Le notificamos que el autor principal y los coautores, no podrán alterarse y quedaran como se envía en esta versión.

Asimismo, me permito informarle que su contribución será sometida a revisión técnica por los árbitros que se designen en caso de ser aceptados, se le notificará sobre las observaciones correspondientes.

Agradezco su colaboración y le envío un cordial saludo

Atentamente

DR. JORGE A. ACOSTA GALLEGOS
EDITOR EN JEFE DE LA REVISTA
AGRICULTURA TÉCNICA EN MÉXICO

c.c.p. * Archivo
JAG/DMSJ/mdpg

PROTOTIPO PARA AUTOMATIZAR UN SISTEMA DE RIEGO MULTICULTIVO

PROTOTYPE TO AUTOMATIZE A MULTI CROP IRRIGATION SYSTEM

Oziel Lugo Espinosa[1], Abel Quevedo Nolasco[2], Juan R. Bauer Mengelberg[3], David Hebert del Valle Paniagua[4], Enrique Palacios Vélez[5] y Miguel Águila Marín[6]

Colegio de Postgraduados, México

Postgrado de Hidrociencias, Km. 36.5 México-Texcoco, Texcoco, Edo. de México. CP. 56230.
1. oziel@colpos.mx, (01595) 9523488; 2. anolasco@colpos.mx, 58045900 Ext. 1072 y 1383; 3. jbauer@colpos.mx, 58045900 Ext. 1462; 4.dhvallep@colpos.mx, 58045900 Ext. 1465; 5. epalacios@colpos.mx, 58045900 Ext. 1174; 6. fmaguila@yahoo.com.

RESUMEN

El riego proporciona a la planta el agua imprescindible para su crecimiento y desarrollo. Para garantizar el riego adecuado, pero aprovechando al máximo los caudales de agua disponibles, se pueden utilizar sistemas de riego con programación de control automático, para asegurar el suministro cuando y en la cantidad que la planta necesite, para evitar el desperdicio de agua. Se trata de sistemas que activan el riego espontáneamente, en función de la valoración continua de uno o varios parámetros o variables de control. Este trabajo presenta un prototipo para automatizar el riego en una zona con distintos cultivos, donde cada uno de ellos recibe la cantidad de agua necesaria en el momento que se requiera para un óptimo crecimiento. El prototipo consiste en una integración de tres tecnologías, computación, comunicaciones y electrónica. El área de influencia es una zona geográfica con condiciones meteorológicas similares, dado que el sistema usa las variables meteorológicas temperatura del aire, humedad relativa, velocidad del viento, precipitación y radiación global. También se le proporciona la información del suelo (porcentaje

contenido de arena, arcilla, materia orgánica y densidad aparente): y de los cultivos (tipo, duración, funciones de crecimiento radicular y Kc). Con esta información se elabora un balance hídrico, el cual resulta en la decisión de regar o no alguno de los cultivos. Cuando, como resultado de la función de abatimiento del agua en el suelo en conjunción con el balance hídrico, se determina que se debe regar, el software envía señales digitales de salida por el puerto serial RS232 de una computadora hacia un dispositivo electrónico, para accionar los dispositivos físicos de riego y complementarios (v.gr. electroválvulas, ventiladores, lámparas.) Se integró un subsistema de comunicación bidireccional PC-Modem GSM-Celular que permite encender/apagar un dispositivo mediante comandos remotos desde un teléfono celular, como también el envío de alarmas y avisos que indican el estado actual del sistema por medio de mensajes de texto vía celular y correos electrónicos. El sistema de automatización se diseñó en módulos, lo que permite que se adapte a diversas aplicaciones que sean susceptibles de automatización y control. Se describen los dispositivos y programas que componen el sistema, y las entradas, salidas y proceso que controlan el riego automatizado.

Palabras clave: automatización, riego, balance hídrico, multicultivo.

ABSTRACT

Irrigation is vital for the plants growth and development. To ensure adequate irrigation, but contemplating the most efficient use of available water, an automated irrigation system can be used. It consists of a programmed automatic control which will furnish the plant with the necessary quantity of water, at the appropriate times, so that the waste of water is minimized. A system will spontaneously activate the irrigation devices as a function of the continuous stream of information regarding one or more control parameters or variables.

This work presents a prototype to automatize the irrigation at a zone with several cultivations, where each one of them receives the quantity of necessary water the moment that it take to an optimal growth. The prototype consists in an integration of three technologies, computation, communications and electronics. Since the system uses prevailing meteorological variables, including temperature, relative humidity, wind velocity, precipitation and global radiation, the area of influence must be a geographic zone with similar climate conditions. Other variables used by the model are information about soils (percentages of sand, clay, organic materials content, and the apparent density) as well as information about the crops, consisting in type, duration, functions of root growth and Kc.) A hydric balance computed with all this information will result in the decision to water or not some of the crops. Whenever the result of the water abatement in the ground, in conjunction with the hydric balance, determines that water should be provided, the software sends a digital signal via a computer's RS232 serial port to an electronic device, to activate the suitable physical irrigation devices (e.g. solenoid valves, fans and lamps.) A bidirectional communication system from a PC-Modem to a GSM-cell phone can turn a device on or off, through remote commands sent from a cell phone, as well as text messages or emails conveying an alarm or notice describing the current state of the system. The system was designed in modules, so that it may be applied to different situations which involve automatization and control. The devices and programs, as well as inputs and outputs of the system, are described in detail.

Keywords: automatization, irrigation, hydric balance, multi-crop.

INTRODUCCIÓN

El riego proporciona a la planta el agua necesaria para su crecimiento y desarrollo. Dada la escasez de agua, es conveniente para la planta, pero también para la protección del medio ambiente, que el riego se aplique con la mayor eficiencia. Una de las alternativas para lograr este objetivo es la utilización de sistemas de riego con programación de autocontrol: Se trata de sistemas que establecen la ejecución automática de riegos mediante la valoración continua de uno o varios parámetros o variables de control. Los factores de control pueden ser edáficos (como la humedad), indicadores compuestos, que relacionan variables meteorológicas y el cultivo (como la demanda evapotranspirativa), y en general, variables que permitan determinar, en forma continua, el momento y cantidad de agua necesaria para un cultivo, de tal forma que el sistema tome decisiones con el apoyo en estos indicadores en tiempo real. Es importante resaltar que la aplicación del agua, en términos de cantidad y oportunidad, se debe realizar con precisión en tiempo real.

Con base en lo anterior, el objetivo del presente artículo es describir el desarrollo de un sistema-prototipo de riego automático, que integra: el componente de entrada, consistente en la información meteorológica que se obtiene de una estación meteorológica comercial (Campbell Scientific Inc.); el componente de control (de software, para la toma de decisiones); y el componente de salida, compuesto por dispositivos electrónicos que encienden/apagan periféricos. A esto se agrega una interface de potencia, para la cual los dispositivos pueden ser electroválvulas para la aplicación del riego, entre otros. El control se lleva a cabo por medio de un balance hídrico a partir de las variables meteorológicas, además de la información de los cultivos y suelos. Como parte del sistema se acopló un

subsistema de comunicación bidireccional por medio de un modem-celular, que permite tanto activar/desactivar algún dispositivo como el envío de mensajes hacia algún celular, o una cuenta de correo electrónico, para informar alguna acción llevada a cabo por el sistema.

La presentación se hace en forma secuencial, es decir, se indica cómo se inicio el desarrollo el sistema de control y las tecnologías de comunicación que se usaron para cumplir el propósito. Una introducción sobre los sistemas existentes de control de riego precede a dicha descripción.

ALGUNOS MODELOS SOBRE EL CONTROL DEL RIEGO

En la búsqueda de la eficiencia en el uso de agua, Bralts *et al*., (1986) desarrollaron un programa de computo (SCS-Scheduler) para el manejo del riego. Los datos de entrada se programaron en un datalogger. El programa contempla estados del sistema, secciones de riego, control y encendido y apagado de dispositivos. En el mismo sentido, Wessels *et al*., (1995) desarrollaron un sistema automático para el riego controlado por computadora, en el cual se usa la información meteorológica para calcular la evapotranspiración. Xin *et al*., (1995) desarrollaron un prototipo de sistema experto (CIMS) para el manejo del riego en tiempo real, la protección de heladas y el control de la fertirrigación en cítricos. Usaron como datos de entrada, información del contenido de humedad del suelo, además de datos obtenidos de una estación meteorológica automatizada.

Moreno *et al*.,(1996) desarrollaron un programa de computo (AUTRI Versión 1.0) para la automatización de un sistema de riego localizado. El programa implementa estrategias para

determinar el momento del riego mediante el balance hídrico, calculado a partir de variables meteorológicas, datos de suelo y cultivo y mediante el monitoreo de la humedad del suelo a través de un electrotensiómetro. Con el mismo propósito, Carrillo (1999) desarrolló un programa de cómputo con fines de control, programación y aplicación del fertirriego en tiempo real y control de temperatura al interior del invernadero.

Por su parte Águila (2003) desarrolló un sistema automatizado para el manejo del riego en tiempo real. En el programa se utilizan varios algoritmos que procesan la información meteorológica, del suelo y cultivo, y resultan en diferentes estrategias de riego. Todos estos datos se procesan por medio de un datalogger de la compañía Campbell Scientific Inc., que a través de sus puertos de control, activa el sistema de distribución del agua, hasta que se cubren los requerimientos del cultivo.

Por otra parte, Castro (2008) implementó un sistema de automatización en tiempo real, con la verificación del riego por medio de las tecnologías de información (internet y dispositivos móviles). Comparó diferentes estrategias de control de riego (balance hídrico, micro lisímetro y el sensor directo de la humedad del suelo mediante un TDR). A partir de los datos obtenidos, el datalogger automatiza el proceso de lectura de datos y acción del control de dispositivos.

La computadora y los sistemas de comunicación electrónica

Las computadoras electrónicas, que permiten realizar los cálculos matemáticos complejos que se requieren para tomar decisiones, funcionan con bajos niveles de energía eléctrica, lo

que hace que no tengan la capacidad para activar directamente alguna interface que encienda o apague un dispositivo de control. Es por ello que se necesita una interface (electrónica de potencia) que eleve los niveles de energía a los necesarios para accionar los mencionados dispositivos. Las diferentes partes de los sistemas de control deben transmitir información (que puede ser de mediciones de sensores, valores de referencia, señales de activación de salidas y comunicación) entre los elementos internos del sistema (Morais y Boaventura, 2000). Aquí la comunicación de datos implica que la información es digital tanto en la fuente como en el destino, aunque la transmisión puede ser en forma digital o analógica. Esto quiere decir que se pueden utilizar canales analógicos (como la modulación en amplitud o frecuencia) para transmitir información digital (Tomasi, 2003). Noergaard (2005) comenta que todos los sistemas de comunicaciones electrónicas tienen que lidiar con problemas de ruido eléctrico (cualquier energía eléctrica indeseable que cae dentro de la banda útil de la señal de interés), por lo que se debe considerar el aislamiento para el circuito, lo que evitará comportamientos extraños del sistema.

Comunicación Celular-Módem-Computadora

Para el desarrollo se usó la comunicación Celular-Modem-PC, donde un módem GSM es un dispositivo que se conecta a la red GSM para enviar/recibir información. La red GSM es una red digital, por lo que no se necesita un módem analógico (adaptador que realiza una conversión analógico-digital MOdulador-DEModulador): basta usar un adaptador que se ajusta al flujo de datos provenientes del PC, al flujo de datos que se utiliza en el enlace digital entre el teléfono y la red GSM.

El modelo de comunicación PC a MODEM se estableció mediante el puerto serial de la PC. El módem funciona mediante un conjunto de instrucciones que se denominan comandos AT, donde el software principal controla el envío y recepción de estos comandos y posterior tratamiento de las respuestas.

Servicio de Mensajes Cortos (SMS)

Es un sistema para enviar mensajes de texto a, y recibir mensajes de, teléfonos móviles. El SMS fue creado como una parte del estándar de telefonía móvil GSM fase 1 en 1992. Dentro del SMS hay varias características que adhieren al estándar referido: a) un mensaje corto puede tener una longitud de hasta 160 caracteres, que consisten de palabras, números o una combinación alfanumérica, aunque también se pueden utilizar mensajes cortos basados en No-texto (por ejemplo, en formato binario); b) los mensajes cortos no se envían directamente del remitente al receptor, sino que se transmiten a través de un centro de SMS; c) los mensajes cortos se pueden enviar y recibir simultáneamente con voz, datos y llamadas del fax. Esto se debe a que se usa un canal de radio dedicado durante la llamada: los mensajes cortos viajan sobre un canal dedicado a señalización independiente de los de tráfico.

Celular con Sistema Operativo Windows Mobile

Para algunas fases del desarrollo se usó Windows Mobile® que es un sistema operativo compacto, con una suite de aplicaciones básicas para dispositivos móviles con base en la librería de programación (API) Win32 de Microsoft®. Los dispositivos que llevan

Windows Mobile® son Pocket PC, Smartphones y Media Center portátil, que se han diseñado para ser similares a las versiones de escritorio de Windows®. Un sistema operativo en un celular permite realizar interfaces de usuario en el propio celular, lo que simplifica al usuario los procesos de acción. Por ejemplo, se puede construir un mensaje de texto con apretar un solo botón, o mostrar información de forma gráfica.

MATERIALES Y MÉTODOS

El desarrollo del prototipo de riego implicó el uso de software, dispositivos electrónicos (para el control de los dispositivos de salida y de comunicación bidireccional), información meteorológica (de entrada), e información del suelo(s) y cultivo (s).

La herramienta de desarrollo de software fue el entorno "NetBeans IDE 6.7.1" (plataforma de desarrollo de aplicaciones Java, Marca registrada de Sun Microsystems), que permitió la creación del software Riego Inteligente Automático (RAI) para la comunicación y el control del riego. Además se usó la suite Visual Studio 2008® (Microsoft Corporation) para desarrollar el software del dispositivo móvil (celular), además de una computadora portátil para el desarrollo e instalación del RAI. Para la componente de comunicación bidireccional de salida, se desarrolló un dispositivo electrónico que usa el puerto serial RS232 de una computadora) y un modem-celular (de uso comercial) que se instaló por USB.

A continuación se describe la integración de todos los componentes, tras presentar las partes del prototipo del sistema de control. La parte de control se realizó con base en el balance hídrico a partir de la información meteorológica, de suelo y del cultivo. El balance se realiza entre las salidas y entradas de agua al sistema, donde se compensan de una forma

eficiente las pérdidas de agua en el sistema a partir de una función de abatimiento de agua en el suelo. El balance hídrico fue la herramienta para la toma de decisiones en la verificación de la disponibilidad de agua en el sistema cultivo-suelo. Es un método ampliamente usado en los estudios de zonificación agrícola, influencia de la deficiencia hídrica en la productividad de los cultivos (Calvache *et al.,* 1997), y en el diseño e implementación y monitoreo de sistemas de riego y drenaje. Silva (2001) señala que el balance hídrico se puede utilizar para establecer las comparaciones entre las condiciones hídricas de localidades distintas. Incluso, se aplica a diferentes escalas de tiempo en función de la disponibilidad de información. El balance del contenido de humedad del suelo se determina con la ecuación:

$W_i = W_{i-1} + Pe_i - ETr_i$ …….(1)

Donde:

w_i Humedad del suelo en la hora i, mm.

w_{i-1} Humedad del suelo en la hora i-1, mm.

Pe_i Precipitación efectiva, mm.

ETr_i Evapotranspiración real del cultivo, mm

La evapotranspiración real del cultivo se estima con la siguiente ecuación.

$ETr = Eto\ kc$ ……..…….(2)

Donde:

ETr Evapotranspiración real del cultivo, mm /hora

Eto Evapotranspiración de referencia, mm / hora

kc Coeficiente de desarrollo del cultivo

Para estimar la ETo se usó el método de Penman-Monteith (1990) que se describe con la ecuación:

$$ETo = \frac{\Delta(R_n - G)}{\lambda(\Delta + \gamma^*)} + \frac{\gamma^* M_W (e_a - e_d)}{R\Theta r_v (\Delta + \gamma^*)} \quad \text{...……(3)}$$

Donde:

ETo Evapotranspiración de referencia (mm h^{-1})

Δ Gradiente de saturación de presión de vapor (Pa $°C^{-1}$)

R_n Radiación neta (kW m^{-2})

λ Calor latente de vaporización del agua (2450 kJ kg^{-1})

G Flujo de calor del suelo (kW m^{-2})

γ^* Constante psicrométrica aparente (Pa $°C^{-1}$)

M_W Masa molecular del agua (0.018 Kg mol^{-1})

$e_a - e_d$ Déficit de presión de vapor del aire (kPa)

R Constante de gas ideal (8.31 x 10^{-3} kJ mol^{-1} K^{-1})

Θ Temperatura en grados Kelvin (293 °K)

r_v Resistencia del área foliar del cultivo (s m^{-1})

La precipitación efectiva se ajustó con el siguiente modelo (Palacios, 1989):

$$Pe = Lluvia_{total} - (0.05\ Lluvia_{total})^{2.} \quad \text{……...…….(4)}$$

Donde Pe es la precipitación efectiva (mm)

Para determinar las propiedades del suelo, estimación de capacidad de campo (CC) y punto de marchitez permanente (PMP), se usaron las siguientes ecuaciones (Rawls, *et al*., 1983):

$$PMP = 0.0854 - 0.0004(X) + 0.0044(Y) + 0.0122\ (Z) - 0.0182\ (D) \quad \text{……..….(5)}$$

$$CC = 0.3486 - 0.0018(X) + 0.0039(Y) + 0.0228\ (Z) - 0.0738\ (D) \quad \text{…… (6)}$$

Donde:

X, Y, Z, son los porcentajes de arena, arcilla y materia orgánica respectivamente. D es la densidad aparente del suelo en [g/cm^3].

El cálculo de la humedad total fácilmente aprovechable dentro de la zona radicular (HuFaAp, ecuación 7), necesita la profundidad de las raíces (ProRaiz_m, ecuación 8) y un factor de abatimiento de la humedad permisible (FaAbHuPer, ecuación 9). Los modelos de profundidad radicular y del factor de abatimiento para el cultivo de calabaza zucchini grey (*cucúrbita spp*.), fueron propuestos por Castro (2008).

HuFaAp = HA x ProRaiz x FaAbHuPer …….……..…(7)

Donde:

HuFaAp	humedad fácilmente aprovechable total dentro de la zona radicular (escalar)
HA	humedad aprovechable
ProRaiz	profundidad de la raíz
FaAbHuPer	factor de abatimiento de la humedad permisible

Función de desarrollo radicular

Allen *et al*., (1998) mencionan para calabaza zucchini grey (*cucúrbita spp*.) una profundidad máxima de raíces de 0.6 a 1.0 m. Sin embargo, sugiere para la planeación del riego una profundidad de 0.6m. Como valor inicial de profundidad de raíces se tomó un valor de 0.1m, que corresponde a la longitud de las raíces de las plántulas a la hora de la plantación; los valores subsecuentes se calculan con una curva semejante al desarrollo vegetativo de la planta representado por la curva KC (ecuación 10).

ProRaiz (m) = 0.020395275 + 0.0027690429 * X + 0.00020960909 * X^2 -

0.0000034621581 * X^3 + 0.000000014608712 * X^4…….....…(8)

Donde X el día juliano del total de la duración del cultivo en días (cero – último día de duración).

El factor de abatimiento está dado por la ecuación:

FaAbHuPer (escalar)= 0.350546158672458 - 0.001495941458749 * X + 0.000165169025107 * X^2 - 0.000006529893935 * X^3 + 0.000000075165292 * X^4 - 0.000000000266580 * X^5……..…(9)

Donde X el día juliano del total de la duración del cultivo en días (cero – último día de duración).

Función de Kc (Allen, *et al.,* 1998)

En el FAO-paper No 56 "Crop Evapotranspiration" se presentan valores KC para tres etapas de desarrollo, designadas como KCini, KCmed y KCfin. Para calabaza zucchini grey (*cucúrbita spp*.), se tomaron valores de 0.52, 0.90, 0.90 y 0.70 para las etapas B,C,D,y E que se muestran en la figura 1. Para estimar Kc se usó el modelo propuesto por Castro (2008), como:

Kc = 0.41470218 - 0.00034576721 * X + 0.00065725214 * X^2 - 0.000011333138 * X^3 + 0.000000049443828 * X_4……...……(10)

Aquí X representa la duración del cultivo en días (se consideraron 90 días para la duración de todo el ciclo del cultivo) y Kc representa el coeficiente del cultivo.

RESULTADOS Y DISCUSIÓN

Se describe primero el Sistema de Automatización de Riego, que se integra en el software que se denominó Riego Automático Inteligente (R.A.I). El sistema de subdividió en tres partes: la entrada, el control, y finamente, la salida. Éstas se complementan con la comunicación. En la Figura 1, se indica la integración de estos elementos, y la interacción entre los mismos. En la Figura 2 se indica el cuadro de diálogo principal del software.

Información de entrada

El sistema usa la información meteorológica que se recupera de una estación meteorológica Campbell con los siguientes sensores: temperatura del ambiente, radiación global, precipitación, velocidad del viento y humedad relativa, en intervalos de cada hora.

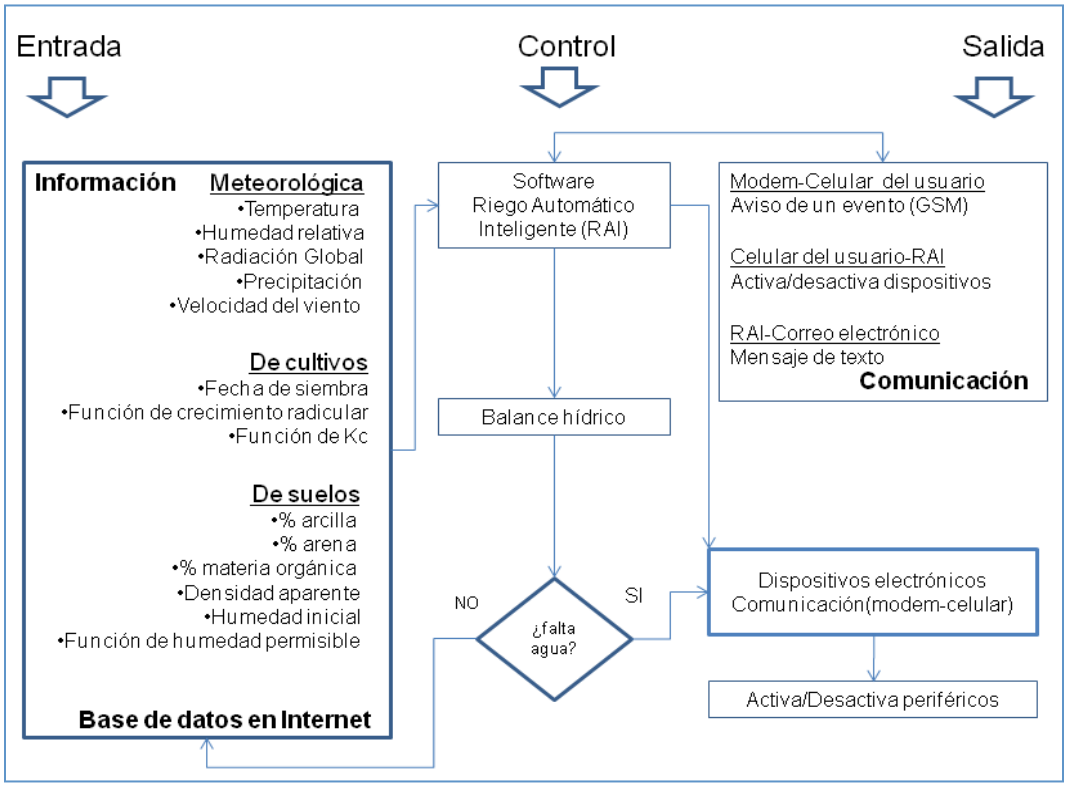


Figura 1. Diagrama general del sistema de riego automático inteligente.

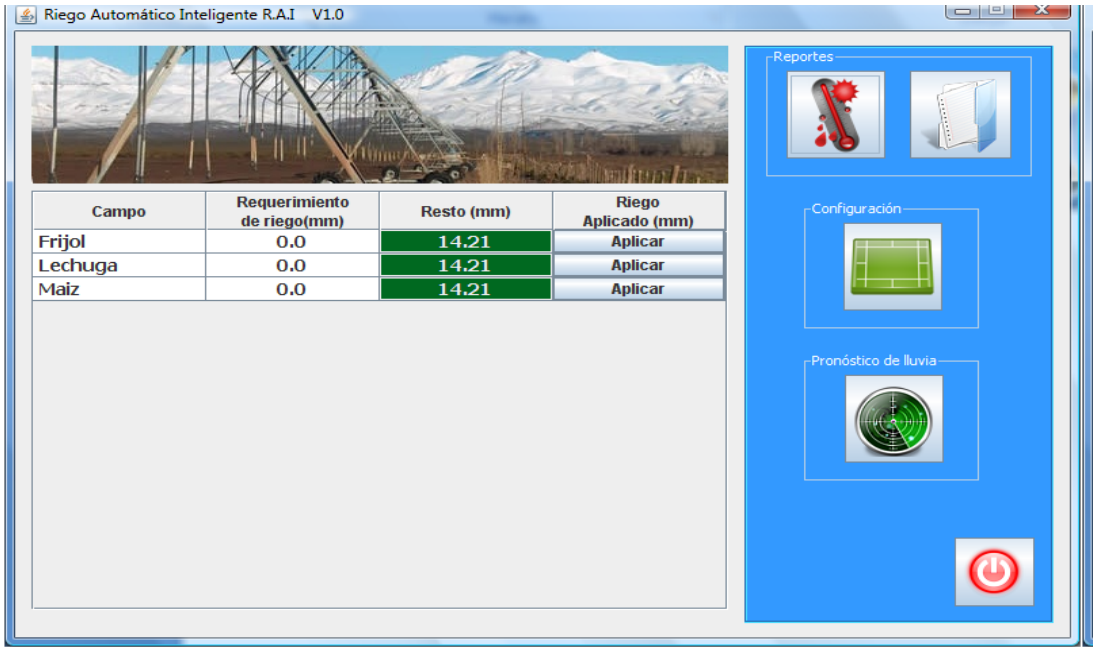


Figura 2. Diálogo principal de riego automático inteligente (RAI).

A partir de esta información se determina la evapotranspiración. Por parte de los cultivos se obtiene el tipo de cultivo, y tres funciones: el coeficiente del cultivo Kc, de desarrollo radicular y la duración del cultivo en días. En cuanto a las características del suelo, se necesitan conocer los porcentajes de arcilla, arena, materia orgánica y densidad aparente. Todos estos datos (meteorológicos, propiedades del suelo e información de los cultivos) se almacenan en una base de datos. En la figura 3 se indica el cuadro diálogo con el que se introducen todos los datos que necesita el sistema.

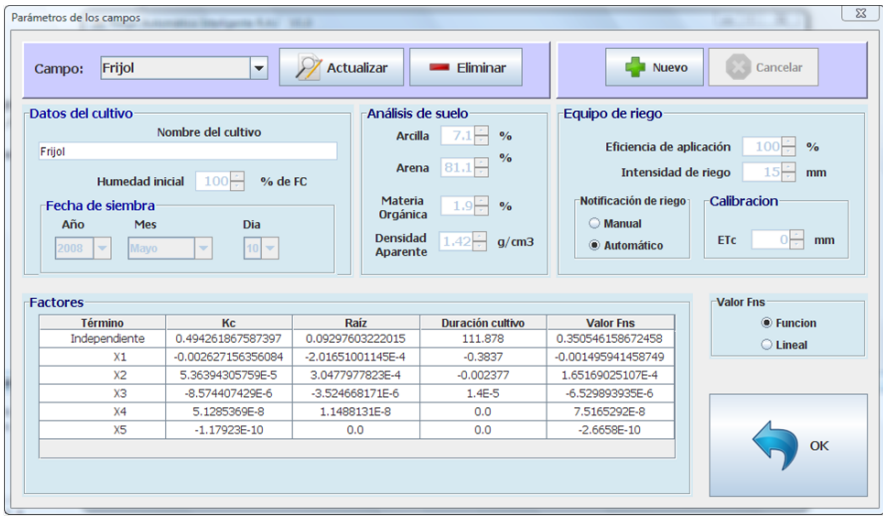
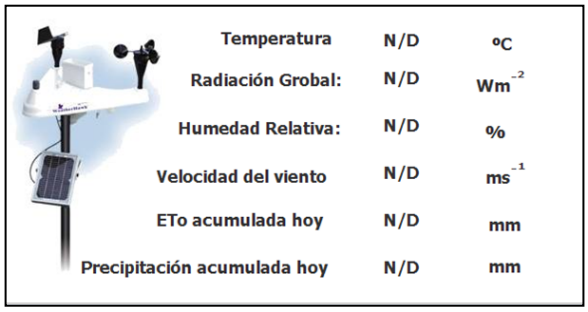


Figura 3. Diálogo de captura de información de cultivo y suelo.

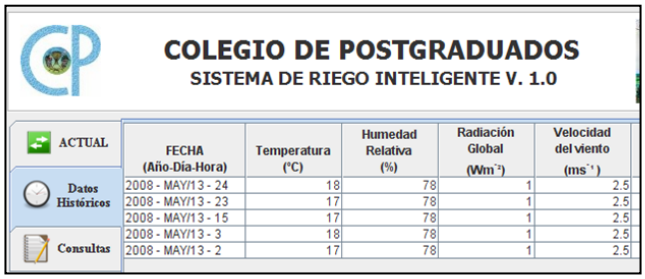
Sistema de control: Riego Automático Inteligente (RAI)

A partir de la información meteorológica, de los cultivos y de los suelos, se hace un balance hídrico. Para un cultivo, se parte de la información del suelo: se calculan la capacidad de campo y el punto de marchitez permanente, a partir de los cuales se determina la humedad aprovechable, usando la densidad aparente y la profundidad (como una función del crecimiento radicular del cultivo). Para el cálculo del balance hídrico se parte de la humedad inicial del suelo, haciendo uso de la ecuación (1). Se activa el riego si el contenido de humedad resulta menor que un nivel de humedad definido previamente por el usuario (que dentro del software se define por la función de abatimiento permisible). Periódicamente, en intervalos definidos en el sistema, se realiza la recuperación de datos, almacenamiento, cálculos necesarios para cada cultivo y la toma de decisiones para regar, aplacando los criterios citados.

Los datos almacenados en la base de datos de información meteorológica permiten revisar los datos actuales (Figura 4a), los históricos (Figura 4b) o bien hacer una consulta de alguna variable meteorológica (Figura 4c). La consulta comienza por la selección de una variable, y a continuación se especifican los criterios de búsqueda (con los operadores mayor, menor, o igual que un valor), en un período determinado.



4(a)



4(b)

4(c)

Figura 4. Diálogos para mostrar la información meteorológica.

a) Datos actuales, b) Datos históricos, c) Consulta de variables meteorológicas

Sistemas de salida (acciones de los sistemas de control)

Este componente consiste de tres partes: una interface de comunicación electrónica que se vincula con una interface de potencia, y un módem celular. En forma conjunta activan los dispositivos, que pueden ser electroválvulas, ventiladores, calefactores, entre otros.

Comunicación de salida (PC-SERIAL)

La comunicación de salida se realiza con un dispositivo electrónico integrado por circuitos electrónicos, cuya función es enviar información digital (bits) por medio de la comunicación serial. Para la comunicación entre la computadora y el circuito electrónico, se utilizó el microcontrolador PIC18F88 de Microchip®, que se programó con el compilador CCS-C, que permitió codificar en lenguaje C el problema, y que generó los archivos en código máquina que se graban físicamente en el dispositivo (Figura 5a). Dado que la señal que se trasmite no puede encender/apagar un dispositivo, se utilizó una interfaz electrónica de potencia.

Dispositivo de potencia

La función de las interfaces de potencia es proporcionar la corriente eléctrica necesaria para que las señales lógicas generadas por el controlador puedan actuar sobre los elementos físicos. Es decir se usan para transportar *energía,* integrada por optoacopladores y relés, quedando a disposición del usuario la elección de estos últimos, ya que depende del voltaje de los dispositivos que se tengan que controlar (Figura 5b).

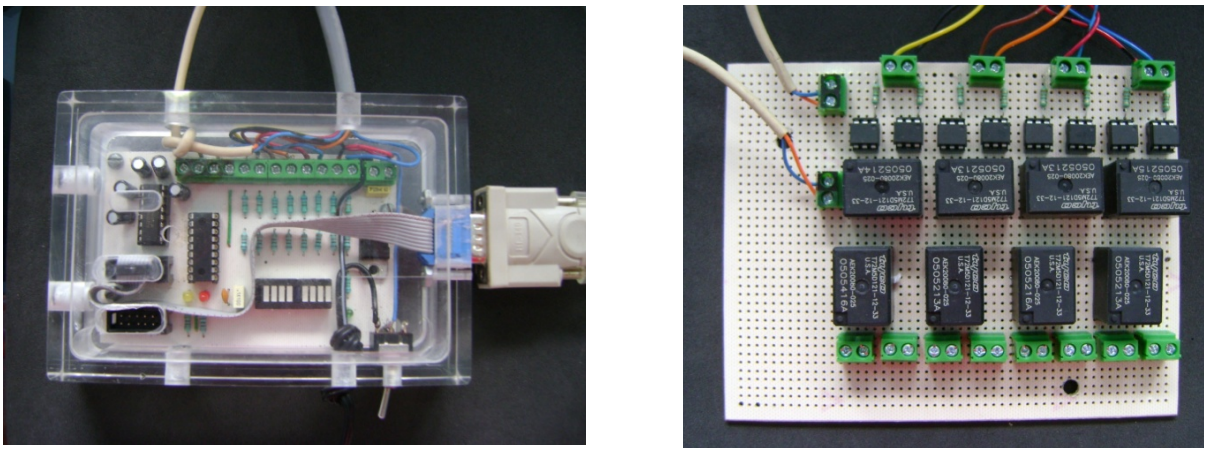


Figura 5. Prototipo desarrollado
(a) Interface de comunicación de salida de la PC. (b) Interface de potencia

Comunicación RAI, Modem-celular y celular

El software R.A.I. se puede configurar para encender/apagar dispositivos a partir del celular. Se parte de configurar el modem-celular que se instala en la computadora. Hecho esto, se pueden activar o desactivar los dispositivos que se conectan a algún puerto. La aplicación que proporciona esta funcionalidad se desarrolló para Windows Móvil®. En la figura 6a se indica el cuadro de diálogo principal, con el que se puede hacer una consulta, enviar comandos para activar los dispositivos, definir alarmas y avisos vía correo electrónico, mensajes SMS y llevar a cabo la configuración del software. Para configurar la aplicación, se especifica el número telefónico del módem celular, y se indica si se desea que se confirme – o no - el mensaje SMS (ver Figura 6b). La opción de consulta ofrece un campo en el que se pueden pedir las condiciones actuales para un cultivo establecido o bien de un puerto, es decir el estado (encendido o apagado) de algún dispositivo de riego de los que controlan el sistema (Figura 6c). La interfaz que permite el envío de comandos para encender/apagar un dispositivo asociado a un puerto se ilustra en la Figura 6d. Con las alarmas (Figura 6e) se visualizan los datos enviados por la computadora a través del

módem GSM, dando los informes de la situación actual del cultivo en particular, y con los correos electrónicos e-mail y mensajes SMS, se puede, a partir del celular, enviar un mensaje a alguna cuenta de correo electrónico (Figura 6f).

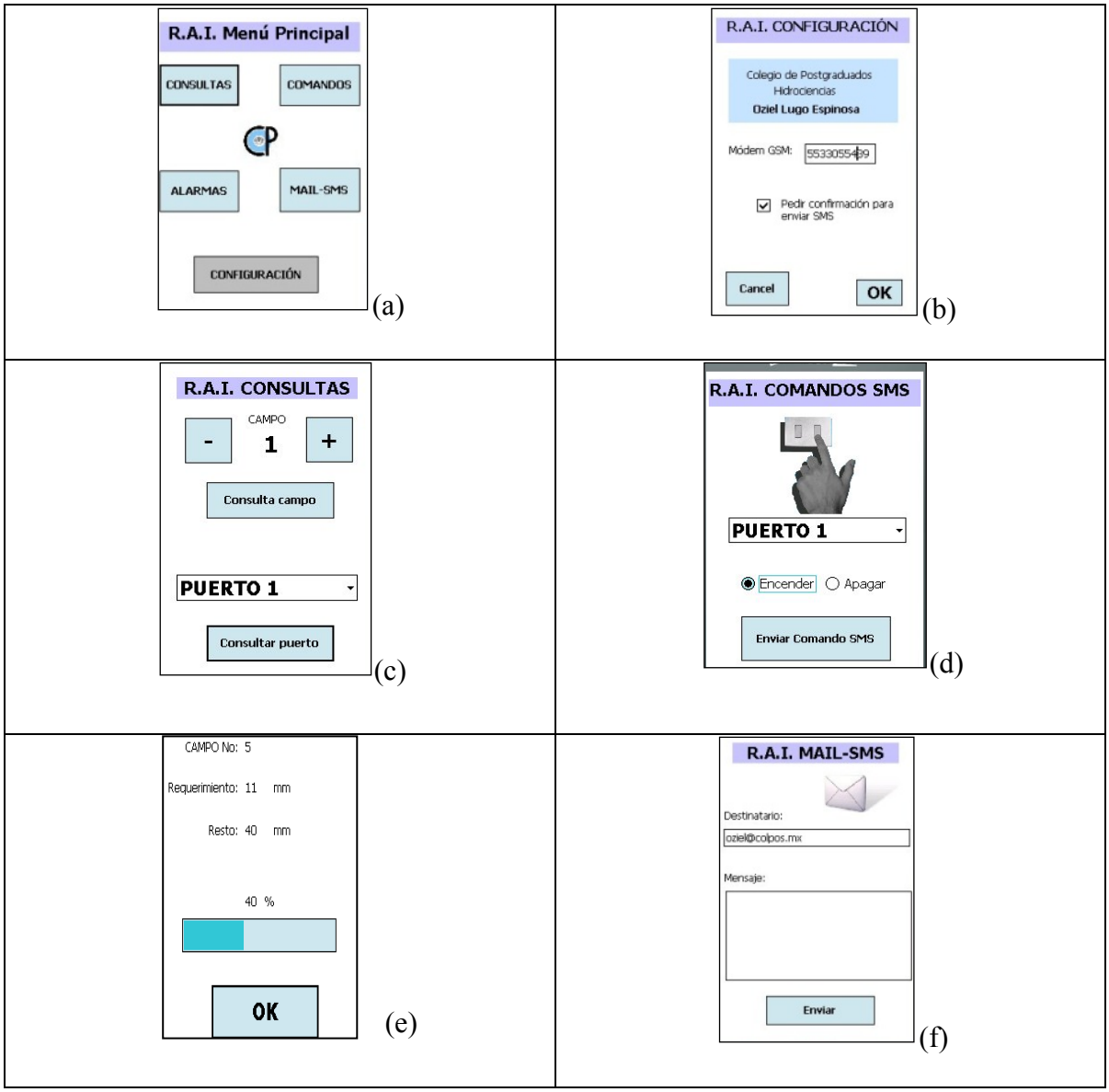


Figura 6. Pantallas de usuario en el celular.

Diálogo principal de RAI en el celular bajo Windows Móvil (a), configuración de RAI(b), consultas de campos o de puertos (c), envió de comandos (d), Alarmas (e) Avisos del celular a una cuenta de correo electrónico (f) .

CONCLUSIONES

Se logró el objetivo planteado: el desarrollo del prototipo que permite accionar los dispositivos de salida a partir de un sistema de control - donde se toma la decisión a partir de un balance hídrico - de manera autorregulada. Como parte complementaria se logró encender/apagar algún dispositivo por medio de teléfonos celulares, lo que a su vez posibilita la automatización del riego, en tiempo real, a partir de variables de suelo, clima y cultivo, con el fin de alcanzar una mayor eficiencia en la aplicación del limitado recurso agua.

Se utilizó los valores del cultivo de calabaza zucchini grey (*cucúrbita spp*.) para probar el sistema y comparar resultados, obteniendo los mismos datos que Castro (2008), con su proyecto de automatización de riego con base en tres estrategias distintas de riego.

La utilización de tecnologías de comunicación como teléfonos celulares, facilita un seguimiento e incluso el control del sistema en tiempo real, al obtener información del estado del sistema y enviar comandos u órdenes de ejecución, lo que se traduce en beneficios inmediatos en cuanto a la aplicación del riego.

Para que el prototipo realice los cálculos adecuados (cantidad de agua y momento adecuado) para cada cultivo, es necesario introducir en el sistema los parámetros del suelo donde se establece cada cultivo y los datos característicos de cada cultivo (Kc).

La programación orientada a objetos, junto con las diferentes tecnologías utilizadas, facilitó la inserción necesarios en el programa principal de diversos algoritmos y la alimentación de información (meteorológicos, cultivo, suelo, geográficos), indispensables para la estimación objetiva del riego.

RECOMENDACIONES

Se debe tener en cuenta el ruido eléctrico, es decir, la energía eléctrica indeseable generan por la computadora y los dispositivos físicos que se desean controlar, por ello es importante tener un acoplamiento óptico entre los dispositivos de salida y la computadora, lo que evita el ruido y resulta en un funcionamiento adecuado del sistema. Para contar con el desarrollo completo del control de riego en un sistema abierto, falta el desarrollo de un sistema de adquisición automatizada de datos meteorológicos.

AGRADECIMIENTOS

Se agradece a las siguientes personas su apoyo para el desarrollo del prototipo: Luis Carlos Miranda Trujillo, Ricardo Luna, Pedro Ríos González, Ricardo Campos, Arturo Gamalier, Gilberto Lugo, entre otros.

REFERENCIAS BIBLIOGRÁFICAS

Águila, M. F. Entwicklung eines vollautomatischen Bewässerungsregelungssystems für den Freilandgemüsebau. Editorial Verlag Grauer, Beuren – Stuttgart, Alemania. ISBN 3 – 86186–434–7. 2003.

Bralts, V. F., M. A. Driscoll and F. Kelly S. Microcomputer based irrigation management and control system. ASAE, Paper No. 86-1223. St Joseph, MI.USA,1986.

Castro P., M,. Sistema de riego automatizado en tiempo real con balance hídrico, medición de humedad del suelo y lisímetro, 2008, Agricultura técnica en México. México. ISBN 0568-2517.

CIMIS. CIMIS Agricultural Resources Book. California Irrigation Management information system CIMIS, California Department of Water Resources. 2005.

Doorenbos J. y Pruitt, W. Las necesidades de agua de los cultivos. Estudio. FAO: Riego y drenaje N° 24. Roma, FAO, 1984,181p.

Calvache, M.; Reichard, K.; Bacchi, O. Efecto de épocas de deficiencia hídrica en evapotranspiración actual de cultivo. Congreso Brasileiro de agrometeorología, 1997, p.668-670.

IMTA, Instituto Mexicano de Tecnología del Agua. [en línea].Morelos, México. [Citado el 29 de agosto de 2009]. Disponible para World Wide Web: http://www.imta.gob.mx/

INIFAP, Instituto Nacional de Investigaciones Forestales, Agrícolas y Pecuarias, Red Nacional De Estaciones Estatales Agroclimatológicas, [en línea], México. [Citado el 27 de agosto de 2009]. Disponible para World Wide Web: http://clima.inifap.gob.mx/redclima/

Mesonet [en línea]. Oklahoma, EU. [Citado el 10 de septiembre de 2009]. Disponible para World Wide Web: http://www.mesonet.org/

Morais, R. y Boaventura, J. 2000. Agritronics: A Distributed Data Aquisition and Control Network For Agriculture Environments. Acta Horticulturae 534: 319-325.

Moreno A., S., L. Tijerina Ch., R. Acosta H., V.M. Ruiz C., F.S. Zazueta R., y G. Crespo P. Automatización de un sistema de riego localizado aplicado a una plantación de durazno. Agrociencia, Vol. 33, 1996,2: 191-197

Noergaard, T. Embedded Systems Architecture. A Comprehensive Guide For Engineers And Programmers. Elsevier. USA. 2005. 642 p.

Palacios V. E., & A. Excebio. Introducción a la teoría de la operación de distritos de riego. Segunda reimpresión corregida. Centro de Hidrociencias. Colegio de Postgraduados, México. 1989.

Silva V.M.A. Meteorología e Climatología. Brasilia: INMET, Gráfica y Editora Pax, 2001. 532 p.

Tomasi, W., Sistemas de comunicaciones electrónicas. Pearson Educación. México, 2003, 976 p.

United Nations. Water for people water for life. Executive Summary of the UN Worl Water Development Report. Educational Scientific and Cultural Organuzation (UNESCO), Paris, France, 2003, 604 p.

Wessels, W.P.J., W.H. Steyn and J.H. Moolman. Automatic microirrigation and salt injection system for research and commercial applications. Proceeding of the fifth international microirrigation Congress. Orlando Fl., USA. ASAE,1995, p. 116-122.

TECNOLOGÍA GSM COMO MÓDULO DE COMUNICACIÓN EN UN PROTOTIPO DE AUTOMATIZACIÓN DE RIEGO

GSM TECHNOLOGY AS A MODULE OF COMMUNICATION IN AN IRRIGATION AUTOMATIZATION PROTOTYPE

Oziel Lugo Espinosa[1], Abel Quevedo Nolasco[2], Juan R. Bauer Mengelberg[3], David Hebert del Valle Paniagua[4], Enrique Palacios Vélez[5] y Miguel Águila Marín[6]

Colegio de Postgraduados, México

Postgrado de Hidrociencias, Km. 36.5 México-Texcoco, Texcoco, Edo. de México. CP. 56230.
1. oziel@colpos.mx, (01595) 9523488; 2. anolasco@colpos.mx, 58045900 Ext. 1072 y 1383; 3. jbauer@colpos.mx, 58045900 Ext. 1462; 4.dhvallep@colpos.mx, 58045900 Ext. 1465; 5. epalacios@colpos.mx, 58045900 Ext. 1174;
6. fmaguila@yahoo.com.

RESUMEN

La automatización permite liberar al hombre de tareas repetitivas que fácilmente puede realizar un dispositivo electrónico o computarizado. Esto ahorra tiempo porque en algunos casos, los dispositivos son más rápidos y confiables que la mano de obra humana, lo cual puede incrementar la calidad del producto y reducir costos de producción. El uso del teléfono celular en las actividades agrícolas cada vez adquiere mayor auge a nivel mundial, debido a las prestaciones y ventajas que ofrece en la obtencion de datos e información cuando las distancias son un factor que limitan a los sistemas de información para su correcto funcionamiento. Existen varias estrategias que automatizan el proceso de riego mediante el uso de computadoras. El presente trabajo describe el uso de un módem GSM que interactúa con un software, que se basa en el balance hidrico para calcular la cantidad de agua necesaria y el momento idoneo para una serie de cultivos. Los dos objetivos de esta interacción módem-PC son el envío de alarmas - vía mensajes de texto - al celular de un

usuario, y la decodificación de comandos remotos enviados por el usuario desde su teléfono celular. Se describe el desarrollo de software para plataformas móviles, cuya importancia radica en la creacion de una interfaz de usuario en el celular para simplificar el envío de comandos remotos y facilitar la lectura de las alarmas mediante la sustitución de textos por elementos gráficos.

Palabras clave: automatización, control remoto, celular, módem GSM

ABSTRACT

Automation liberates man from repetitive tasks that can easily make by an electronic device or computer. This saves time because, in some cases, the devices are faster and more reliable than human labor, which can increase product quality and reduce production costs. Cell phone use in agricultural activities is having a growing global boom, due to the benefits and advantages in obtaining data and information when the distance is a limiting factor for information systems for normal operation. There are several strategies that automate the process of irrigation through the use of computers. This paper describes the use of a GSM modem that interacts with a software program, which is based on the hydric balance to calculate the amount of water needed and the ideal time for a several number of crops. The two objectives of this interaction PC-modem is send alerts – through text messages – to a cell phone, and decoding of remote commands sent by the user from his cell phone. It describes the development of software for mobile platforms, the importance lies in the creation of a user interface in the cell to simplify sending remote commands and readability of the alarms by replacing graphics texts.

Keywords: automatization, remote control, cell phone, GSM modem

INTRODUCCIÓN

La automatización de ciertas tareas repetitivas que un dispositivo puede realizar, libera al hombre de estas actividades y permite el ahorro de tiempo y dinero. Como se tiene la ventaja de que los dispositivos son más rápidos y precisos que la mano de obra humana, se puede incrementar la calidad del producto y reducir costos de producción.

En la agricultura de riego es frecuente que no se aplique la cantidad necesaria de agua en tiempo y forma, por el contrario, cuando hay excesos resulta en altos consumos de agua, lo que constituye un desperdicio, e incluso puede provocar contaminación en las corrientes de agua superficiales y subterráneas, cuando se aplican los diferentes agroquímicos por medio del riego (IMTA, 1995). Una solución para evitar la ausencia del riego y desperdicios de agua es la automatización del riego con base en alguna estrategia, por ejemplo mediante el cálculo del balance hídrico, medición directa de humedad del suelo o lisímetro. Sin embargo, hasta ahora ninguno de estos sistemas ha tenido una aceptación considerable, y mucho menos comercial. Esto se debe a la alta demanda de tiempo, trabajo y capacitación para operar y alimentar de datos e información a estos sistemas (Águila, 2003).

Una desventaja de los sistemas de control clásicos, es que son tarjetas electrónicas que almacenan datos, y solo muestran la información mínima necesaria para los responsables del riego. Sin embargo, el uso de una computadora permite mostrar en forma gráfica lo que ocurre dentro del invernadero en cada momento con cada una de las variables que se registran. Por ejemplo, se puede mostrar, con mucha precisión e inclusive cada minuto, el contenido de humedad del suelo o sustrato, el comportamiento de la temperatura, la conductividad eléctrica, entre otras, información que puede apoyar en la toma de decisiones.

Los sistemas de automatización en riego permiten conocer por medio de sensores (p.e. TDR, DFR), la disponibilidad de humedad en el suelo y mantener una humedad en condiciones apropiadas para un cultivo. Los nuevos esquemas de control y las tecnologías modernas de comunicaciones permiten controlar el riego a "larga distancia", que puede ser por medio de Internet (ahora disponible en cualquier parte del mundo) o la tecnología GSM, con él mismo enfoque.

Castro (2008) implementó un sistema de automatización en tiempo real, con la verificación del riego por medio de las tecnologías de información (internet y dispositivos móviles). Comparó diferentes estrategias de control de riego (balance hídrico, micro lisímetro y el sensor directo de la humedad del suelo mediante un TDR). A partir de los datos obtenidos, el datalogger automatiza el proceso de lectura de datos y la acción del control de dispositivos de riego. Las diferencias entre las tres estrategias de riego fueron mínimas, lo que indica el éxito de una adecuada automatización del riego depende de la aplicación oportuna del riego de acuerdo a los cálculos obtenidos, sea cual fuere el método utilizado para dichos cálculos.

La integración de un módem GSM con una computadora programada para calcular el momento y la cantidad de agua necesaria para una serie de cultivos, permite el envío de mensajes y alarmas al teléfono celular del agricultor o persona responsable del riego, y así ofrecer una ayuda para la toma de decisiones en este caso, riego, para disminuir algún efecto en la producción y calidad del cultivo. Esta información se ofrece en tiempo real y permite que una persona interactue con el sistema de riego en forma remota el sistema de riego desde cualquier zona geográfica, siempre que exista cobertura de telefonía celular. Esta interacción del usuario con el sistema se traduce en acciones de encencido y apagado

de electroválvulas de forma remota, envío de consultas para conocer el estado del sistema y recepción de alarmas en el celular como parte de algún suceso en sl sistema de riego.

Se presenta a continuación una comparación entre distintas tecnologías disponibles para el envío de datos vía celular. Se describen los componentes de control que permiten calcular la lámina y momento de riego para una serie de cultivos. Y así, informar a los usuarios de los cultivos que requieren del riego mediante el envío de alarmas al usuario y la forma en que se puede interactuar con el sistema mediante el envió de comandos – usando mensajes de texto - desde un teléfono celular.

Tecnología GPRS

La **tecnología GPRS** (**G**eneral **P**acket **R**adio **S**ervice) es diferente a la conexión **CSD** (**C**ircuit **S**witched **D**ata) incluida en el estandard **GSM**. En **CSD**, una conexión de datos establece un circuito virtual y reserva todo el ancho de banda de ese circuito durante toda la conexión, independientemente de si se están enviando datos o no, con el consiguiente desaprovechamiento del **ancho de banda.** En cambio, **GPRS** funciona por conmutación de paquetes, lo que implica que muchos usuarios pueden compartir el mismo canal de transmisión o, equivalentemente, el **ancho de banda** se ocupa con aquellos usuarios que desean enviar datos en un momento dado. Por lo tanto, se aprovecha mucho más el ancho de banda en el caso de los usuarios que transmiten y reciben datos intermitentemente. El correo electrónico y las descargas de datos de sitios web constituyen ejemplos típicos de transferencias intermitentes.

Otra diferencia importante con **CSD**, es que en **CSD** los operadores cobran por tiempo de conexión, en cambio en **GPRS** se cobra por volumen de tráfico enviado y recibido, pues sólo se usa el canal cuando hay transacciones de información. Los múltiples métodos de acceso usados en **GSM** con **GPRS** están con base en la **FDD** (**F**requency **D**ivision **D**uplex) y **FDMA (**Frequency Division Multiple Access**)**.

Durante una sesión **GPRS** se asigna al usuario un par de canales de frecuencia de subida y de bajada. Se combina esta circunstancia con la multiplexación temporal, es decir, la comunicación en modo paquetes, lo cual hace posible que varios dispositivos puedan estar compartiendo en mismo canal de frecuencia. Los paquetes de datos son de longitud fija, correspondiendo al tiempo del slot **GSM**. En el canal de descarga (downlink) el modo que se utiliza es un método PEPS (Primero en Entrar Primero en Salir) o **FIFO** (First Input First Ouput)**, por su nombre en inglés**, es decir, el primer paquete que entra es el primer paquete servido. En el caso del canal de subida (uplink), se usa el mecanismo de acceso con base en un protocolo de reserva como el **SAPR** (**S**lotted **A**LOHA Packet **R**eservation) pero siempre con la política se servicio PEPS.

Clases de dispositivos GPRS

Los dispositivos que disponen de conectividad **GPRS** se pueden dividir en tres clases, en función de la capacidad que tienen para sostener simultáneamente una comunicación **GSM** y **GPRS:**

Clase A. Pueden estar conectados simultáneamente tanto a **GPRS** como a **GSM**.

Clase B. Pueden estar conectados a **GPRS** y **GSM**, pero sólo se puede usar uno de los servicios. Es decir, si tenemos una llamada de voz, se suspenderá el servicio de **GPRS**. Una vez finalizada la llamada de voz, el servicio de **GPRS** se restablecerá automáticamente. La mayor parte de dispositivos móviles **GPRS** son de clase B.

Clase C. Pueden conectarse tanto a servicios **GPRS** como **GSM**, pero se debe escoger manualmente uno o el otro.

Clases Multislot GPRS

En muchas ocasiones se puede "oír" si un dispositivo **GPRS** es de clase 8, de clase 10, de clase 12, etc, esto hace referencia a la velocidad de transmisión de datos sobre **GPRS**. La velocidad de **GPRS** va en función del número de slots temporales **TDMA** asignados, que dependerá tanto de la estación a la cual se conecta, como de la capacidad de nuestro dispositivo **GPRS.**

GPRS coding scheme

La velocidad de transferencia no depende únicamente del número de slots asignados; también se afecta por el método de codificación (**coding scheme**). El método menos robusto, y por consiguiente más rápido, es el **CS-4**, que estará disponible cerca de la estación base (donde la calidad de la señal será más alta y por tanto es menor la incidencia de errores de comunicación); y más robusto (y por tanto, más lento) es el **CS-1,** que será invocado cuando el dispositivo móvil esté lejos de la estación base.

Al utilizar **CS-4** es posible alcanzar velocidades de 20kbps por slot, sin embargo el uso de este sistema hará que la cobertura del dispositivo sea sólo el 25% de lo normal. Con **CS-1**

sólo se pueden conseguir velocidades de 8kbps por slot, pero con una cobertura del 98% de lo normal. La tecnología de las estaciones modernas permite adaptar la velocidad automáticamente en función de la localización del dispositivo móvil, es decir, la distancia entre la estación y el dispositivo en cuestión.

CDMA

Con CDMA, para diferenciar a los distintos usuarios, en lugar de frecuencias separadas se usan códigos digitales únicos. Los códigos son conocidos tanto por la estación móvil (teléfono celular) como por la estación base, y se llaman "Secuencias de Código Pseudo-Aleatorio". Por lo tanto todos los usuarios comparten el mismo rango del espectro radioeléctrico.

Una llamada CDMA empieza con una transmisión a 9600 bits por segundo, la señal se amplifica para ser transmitida a aproximadamente 1.23 megabits por segundo. La amplificación implica que se aplica un código digital concreto a la señal generada por un usuario en una célula. Posteriormente la señal ensanchada se transmite, junto con el resto de señales que se generaron por otros usuarios por el mismo ancho de banda. Cuando las señales se reciben de los distintos usuarios se separan al hacer uso de los códigos distintivos y se regresan las distintas llamadas a una velocidad de 9600 bps.

El uso tradicional del espectro ensanchado o amplificado es el militar, debido a que una señal ensanchada es muy difícil de bloquear, de interferir y de identificar. Esto es así porque la potencia de estas señales está distribuida en un gran ancho de banda, de modo que las señales sólo aparecen como un ruido ligero. Todo lo contrario sucede con el

resto de tecnologías, que concentran la potencia de la señal en un ancho de banda estrecho, que se detecta fácilmente.

Sincronización

En la fase final del radio enlace, en el sentido de estación base a dispositivo móvil, una llamada no se transmite de forma continua. Cada cierto tiempo se conmuta entre los distintos usuarios y se transmite parte de su llamada con el pseudo-código correspondiente.

Este proceso se debe repetir continuamente para que un usuario no pierda la llamada al no reconocer su código concreto. Por ello las estaciones bas e deben estar sincronizadas con una referencia de tiempo común. Por ejemplo el sistema del posicionamiento global (GPS) usa esta técnica de sincronización, con base a satélites con un sistema de radio-navegación capaz de proporcionar, mediante medios prácticos y económicos, la posición, velocidad, y tiempo a un número ilimitado de usuarios, de forma continua.

Servicio de Mensajes Cortos (SMS)

Es un sistema para enviar y recibir mensajes de texto a teléfonos móviles, se creó como una parte del estándar de telefonía móvil GSM fase 1 en 1992. Dentro del SMS hay varias características que adhieren al estándar referido: a) un mensaje corto puede tener una longitud de hasta 160 caracteres, que consisten de palabras, números o una combinación alfanumérica, aunque también se pueden utilizar mensajes cortos con base en No-texto (por ejemplo, en formato binario); b) los mensajes cortos no se envían directamente del

remitente al receptor, sino que se transmiten a través de un centro de SMS; c) los mensajes cortos se pueden enviar y recibir simultáneamente con voz, datos y llamadas del fax. Esto se debe a que se usa un canal de radio dedicado durante la llamada: los mensajes cortos viajan sobre un canal dedicado a señalización independiente de los de tráfico.

Tecnología GSM

El Sistema Global para las Comunicaciones Móviles (GSM, proviene de "Groupe Special Mobile") es un sistema estándar, completamente definido, para la comunicación mediante teléfonos móviles que incorporan tecnología digital (Halonen, 2003). Por ser digital, cualquier cliente de GSM puede conectarse a través de su teléfono con su computador y puede hacer, enviar y recibir mensajes por e-mail, faxes, navegar por Internet, tener acceso seguro a la red informática de una compañía (LAN/Intranet), así como utilizar otras funciones digitales de transmisión de datos, incluyendo el Servicio de Mensajes Cortos (SMS) o mensajes de texto. Por su velocidad de transmisión y otras características, GSM constituye un estándar de segunda generación (2G). Su extensión a 3G se denomina UMTS y difiere en su mayor velocidad de transmisión, el uso de una arquitectura de red ligeramente distinta y sobre todo, en el empleo de diferentes protocolos de radio (W-CDMA).

MATERIALES Y MÉTODOS

El desarrollo del prototipo de riego implicó el uso de software, dispositivos electrónicos (para el control de dispositivos de salida y de comunicación bidireccional), información meteorológica (de entrada), e información del suelo(s) y cultivo (s).

La herramienta de desarrollo de software fue "NetBeans IDE 6.7.1" (plataforma de desarrollo de aplicaciones Java, Marca registrada de Sun Microsystems), que permitió la creación del software Riego Automático Inteligente (RAI) que integra la comunicación y el balance hídrico para el control del riego. Además se usó la suite Visual Studio 2008® (Microsoft Corporation) para desarrollar el software del dispositivo móvil (teléfono celular), además de una computadora portátil para el desarrollo e instalación del RAI. Para la componente de comunicación bidireccional de salida, se desarrolló un dispositivo electrónico que usa el puerto serial RS232 de una computadora y un modem-celular (de uso comercial) que se instaló vía USB.

En lo concerniente al control del riego, se realiza el balance hídrico a partir de la información meteorológica, suelo y del cultivo. El balance se realiza entre las salidas y entradas de agua al sistema, donde se compensan de una forma eficiente las pérdidas de agua en el sistema a partir de una función de abatimiento de agua en el suelo. El balance hídrico fue la herramienta para la toma de decisiones en la verificación de la disponibilidad de agua en el sistema cultivo-suelo. Es un método que se usa ampliamente en los estudios de zonificación agrícola, en el diseño e implementación y monitoreo de sistemas de riego y drenaje. Donde una deficiencia hídrica repercute en la productividad de los cultivos (Calvache *et al.,* 1997).

El envío de mensajes de texto a un celular y la recepción de comandos remotos se implementó con un modelo de comunicación PC-MÓDEM mediante el puerto serial de la PC. El módem funciona mediante un conjunto de instrucciones que se denominan

comandos AT (Tavernier, 1992), donde el software principal controla el envío y recepción de estos comandos y el posterior tratamiento de las respuestas.

COMANDOS AT

El control del módem GSM a través de la interfaz de usuario que se creo, con base en el uso de instrucciones o comandos Hayes o AT (Comandos AT, 2009). Todos los comandos de este protocolo comienzan por "AT" y terminan en "CR" (aunque este último carácter es configurable).

Tipos de Comandos AT

Existen dos tipos principales de comandos y que se utilizaron en el proyecto:

- Comandos que ejecutan acciones inmediatas: ATD marcación, ATA contestación o ATH desconexión
- Comandos que cambian algún parámetro del módem: ATS2=43 configuración del carácter de escape.

Modos De Funcionamiento Del Módem

El módem está en **modo de comandos** (command mode) si éste responde a los comandos que envía la computadora, que es posible configurar o realizar diversas operaciones: consulta, lectura o de marcado y conexión (A basic modem FAQ, 2009).

Cuando el módem se conecta con otra terminal pasa al **modo en línea** (online data mode), donde cualquier información que reciba del puerto serie del ordenador será enviada a la

terminal distante. El módem no procesa la información; solamente la trasmite a través de la línea de comunicación.

Para salir del modo en línea y regresar al modo comandos, se envía al módem +++ (secuencia de escape).

Listado de comandos utilizados en el proyecto

A continuación se presenta un listado de algunos comandos AT que se utilizaron en este proyecto, así como una breve explicación de las instrucciones que ejecuta el módem al recibir dichos comandos.

CONTROL DE ERRORES

AT+CMEE (Mobile Equipment Error). Este comando permite configurar el formato de la respuesta en caso de que se produzca algún error. Dependiendo del parámetro que se le provee, la respuesta sería ERROR, ERROR + un código numérico, o, ERROR + descripción del error. Se emplea este último modo para poder detallar el tipo de error que se produjo en el envío o recepción de cada uno de los comandos.

CÓDIGOS DE ACCESO

AT+CPIN (PIN Control). Se puede usar de dos modos distintos:

a) mediante la interrogación AT+CPIN? se puede determinar el estado del módem en cuanto a su control de acceso, es decir, si se debe ingresar códigos de acceso para poder utilizar el modem (Kinkoph, 1995).

b) Para introducir los códigos citados, se envía el comando AT+CPIN y el código requerido. En este caso se nos informará si el código es correcto (y la terminal se encuentra preparada), si es incorrecto.

MENSAJES SMS

Se puede dividir los comandos en dos tipos (Comandos AT Proyectos, 2009), según su funcionalidad:

a) Configuración del sistema de envío o recepción

- *AT+CMGF (Message Format)*. Configura el tipo de mensajes que se van a usar TEXTO (modo texto) o PDU (modo binario).
- *AT+CPMS (Preferred Message Storage).* Selecciona los valores de las 3 memorias existentes. La primera es la que permite la lectura, borrado y reenviado de mensajes, la segunda es la que contiene los mensajes escritos pero no enviados, mientras que la tercera es donde se almacenan los nuevos mensajes recibidos. Estas memorias pueden tomar los valores de módem o de tarjeta SIM.
- *AT+CSCA (Service Centre Address)*. Establece el número del centro de servicio, que corresponde al servidor, es específico para cada operadora e indica el destinatario intermedio entre el emisor y el receptor.
- *AT+CSMP (Set Text Mode Parameters).* Entre otros, configura el tiempo máximo de permanencia de un mensaje en el centro de servicio, desde su arribo hasta el envío al destinatario final.
- *+CNMI (New Message Indications to TE)*. Configura la metodología de notificación de un nuevo mensaje recibido.

b) Ejecución de acciones

- *AT+CMGR (Read Message).* Lee un mensaje almacenado en la posición que se pasa como parámetro.
- *AT+CMGD (Delete Message).* Borra un mensaje almacenado en la posición pasada como parámetro.
- *AT+CMGW (Write Message to Memory).* Memoriza un mensaje (el contenido del cual se pasa como parámetro) en la primera posición de memoria libre para su posterior envío, lectura, modificación, etc.
- *AT+CMGS (Send Message).* Envía un mensaje, cuyo contenido se debe escribir, a un destinatario especificado.
- *AT+CMSS (Send From Storage).* Envía un mensaje almacenado en una determinada posición de memoria.

IDENTIFICACIÓN

Muestra información referente al módem y a la tarjeta SIM.

- *AT+CGMI (Read MS Manufacturer Identification).* Fabricante.
- *AT+CGMM (Read MS Model Identification).* Modelo.
- *AT+CGMR (Read MS Revision Identification).*Revision.
- *AT+CGSN (Read MS Product Serial Number Identification).* Número de serie.
- *AT+CIMI (Subscriber Identification).* Número de serie de la tarjeta.

Como ejemplo de uso de los comando por medio del uso der la hyper-terminal de Windows u otro software que permita la comunicación a través del puerto serial de una computadora (Figura 1):

AT+CMGS="número de teléfono"

El modem deberá responder ">"

Teclear mensaje, p.e. “HOLA, Enviando SMS…”, no incluir comillas

Por último, se debe presionar las teclas CONTROL-Z

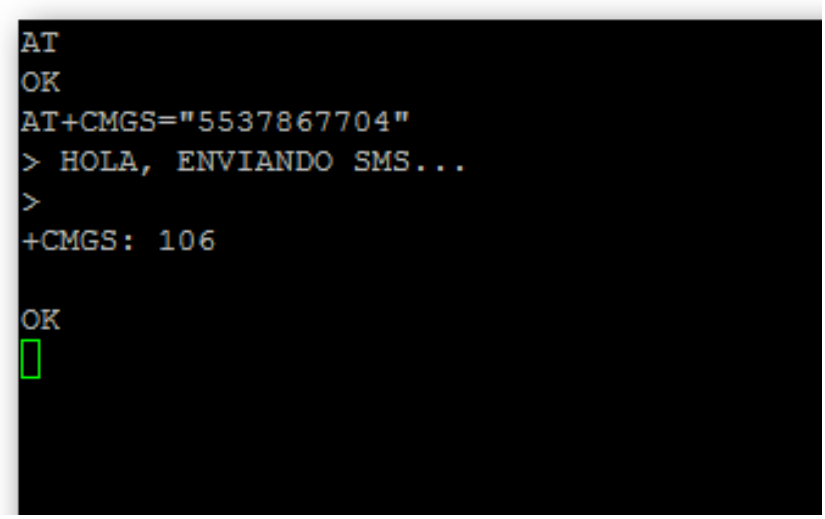


Figura 1. Envío de SMS desde la hyper-terminal de Windows

RESULTADOS Y DISCUSIÓN

El sistema usa la información meteorológica que se recupera de una estación meteorológica (tipo Campbell Scientific Inc. con los siguientes sensores: temperatura del ambiente, radiación global, precipitación, velocidad del viento y humedad relativa, en intervalos de una hora). A partir de esta información se determina la evapotranspiración de referencia. De los cultivos, se considera el tipo y tres funciones: el coeficiente del cultivo Kc, de profundidad de las raíces y la duración del cultivo en días. Del suelo, se necesita los porcentajes de arcilla, arena, materia orgánica y densidad aparente, para determinar la capacidad de campo, el punto de marchitez permanente, y la humedad aprovechable (dado

una profundidad como una función del crecimiento radicular del cultivo). Con la integración de estos elementos se hace el balance hídrico, donde el riego se realiza, sí el contenido de humedad resulta menor que un nivel de humedad definido previamente por el usuario (que dentro del software se define por la función de abatimiento permisible). Periódicamente, en intervalos que se le indican al sistema, se realiza la recuperación de datos, almacenamiento, cálculos necesarios para cada cultivo y la toma de decisiones para regar, con los criterios citados.

El envío de alarmas y mensajes al celular del usuario se hace por medio de mensajes en formato SMS, que es un sistema consolidado y en constante auge, dada su versatilidad y su bajo costo. De este modo, se debe controlar el módem enviándole una serie de instrucciones a través del puerto serial, para conseguir el propósito de establecer la comunicación deseada con otra terminal (el teléfono celular destino).

El teléfono celular cuanta con un sistema de administración de mensajes de texto, el cual tiene la función de informar al usuario que ha llegado un SMS y realizar las operaciones pertinentes como el alamcenamiento en memoria, para su uso posterior. Sin embargo, si el celular cuenta con un sistema operativo, es posible crear interfaces gráficas para interpretar los SMS entrantes debido a la capacidad de hardware y software del teléfono.

PUERTO SERIE RS-232

RS-232 (Recommended Standard 232, también conocido como Electronic Industries Alliance RS-232C) es una interfaz que designa una norma para el intercambio serie de datos binarios entre un DTE (Equipo terminal de datos) y un DCE (Data Communication

Equipment, Equipo de Comunicación de datos), aunque existen otras en las que también se utiliza la interfaz RS-232 (Alexon, 2007).

En particular, existen ocasiones en que interesa conectar otro tipo de equipamientos, como pueden ser computadoras. Evidentemente, en el caso de interconexión entre los mismos, se requerirá la conexión de un DTE (Data Terminal Equipment) con otro DTE. Para ello se utiliza una conexión entre los dos DTE sin usar modem, por ello se llama: null modem ó modem nulo (Hakala, 1996).

El RS-232 consiste en un conector tipo DB-25 (de 25 pines), aunque es normal encontrar la versión de 9 pines (DE-9), más barato e incluso más extendido para cierto tipo de periféricos (como el ratón serie del PC).

Un PC convencional suele disponer de 4 puertos COM, normalmente 2 internos integrados en la placa base y 2 más externos para el uso del usuario. El acceso a cada uno de ellos se realiza a través de sus direcciones BASE COM1=3F8, COM2=2F8, COM3=3E8, COM4=2E8.

Configuración del puerto serie

Cada uno de los puertos COM tiene 11 registros que son a los que debemos acceder para realizar las acciones requeridas.

BASE+0: tiene 3 registros

- *Reciver Buffer Register (RBR):* Registro buffer de recepción, con la función de recibir un dato del puerto.

- *Divisor Latch LSB (DLL):* Divisor de Velocidad, parte baja.
- *Transmitter Holding Register (THR):* Registro de retención de transmisión, con la función de transmitir un dato por el puerto.

BASE+1: tiene 2 registros

- *Divisor latch MSB (DLM):* Divisor de velocidad, parte alta.
- *Interrupt Enable Register (IER):* Registro para habilitar las interrupciones.

BASE +2:

- *Interrupt Identification Register (IIR):* Registro de identificación de interrupciones que controla la prioridad de las mismas.

BASE +3:

- *Line Control Register (LCR):* Registro de control de línea, que controla los parámetros de configuración del canal serie (bits de datos, bits de stop, tipo de paridad).

BASE +4:

- *Modem Control Register (MCR):* Registro de control del módem que activa las señales del mismo.

BASE +5:

- *Line Status Register (LSR):* Registro de estado de la línea, muestra errores, etc.

BASE +6:

- *Modem Status Register (MSR):* Registro de estado del módem.

BASE +7:

- *Scratch Register (SCR)*: Registro residual. Este registro de lectura/escritura no posee ningún cometido específico. Se puede utilizar por el programador para

almacenar valores temporales o para copiar el contenido de algún otro registro. Su razón de ser es que, de esta forma, el 16550A ocupa 8 direcciones de E/S, lo cual representa todas las expresables con los 3 bits menos significativos del bus de direcciones, con lo que no queda ninguna dirección sin asignar.

Parámetros del puerto Serial (Alexon, 1998)

CommPort

Indica el número del puerto serie usado, al modificar esta propiedad se puede cambiar el puerto de comunicación a usar.

Settings

Indica la velocidad, paridad, número de bits y bits de stop (parada) que se puede usar en la comunicación.

Los valores posibles para *velocidad* son (en bits por segundo):50, 100, 110, 300, 600, 1200, 2400, 4800, 9600, 14400, 19200 y 28800

*Los valores posibles para *paridad* son:

N - No envía bit de paridad ni hace comprobación de paridad en la recepción.

O - Envía y comprueba paridad, con el criterio de paridad IMPAR

E - Envía y comprueba paridad, con criterio de paridad PAR

Tanto el puerto serie al cual conectemos el módem, como la velocidad, paridad, número de bits de información y número de bits de stop (parada) se pueden configuran por el usuario mediante listas desplegables que ofrece el sistema al inicio de la interfaz de usuario.

Handshaking

Especifica el método de control sobre el flujo de información. En una comunicación serie se necesita conocer si el puerto puede enviar información y se debe indicar al módem que el canal de comunicación se encuentra preparado para transmitir información.

El Control de Flujo se puede hacer de dos formas:

- mediante las señales auxiliares del puerto (RTS, CTS, DSR, DTR), que son cables adicionales que tendrán una tensión positiva respecto a los 0 volts del equipo si esa señal está activada, o una tensión negativa si no lo está
- mediante señales especiales que se envían por los dos cables que transportan la información.

InBufferSize

Mediante esta propiedad establecemos el tamaño del Buffer (almacén de datos) de entrada. Este Buffer permite recibir datos sin que tenga que intervenir la aplicación continuamente para controlar el puerto de entrada.

OutBufferSize

Mediante esta propiedad se controla el tamaño del buffer de salida. El tamaño de los buffers depende de la aplicación y de la velocidad de comunicación de cada dispositivo.

Comandos remotos desde el celular

La aplicación que proporciona esta funcionalidad se desarrolló para Windows Móvil®. En la figura 2a se indica el cuadro de diálogo principal, con el que se puede hacer una consulta, enviar comandos para activar los dispositivos, definir alarmas y avisos vía correo electrónico, mensajes SMS y llevar a cabo la configuración del software.

Para configurar la aplicación, se especifica el número telefónico del módem celular, y se indica si se desea que se confirme – o no - el mensaje SMS (ver Figura 2b). La opción de consulta ofrece un campo en el que se pueden pedir las condiciones actuales para un cultivo establecido o bien de un puerto, es decir el estado (encendido o apagado) de algún dispositivo de riego de los que controlan el sistema (Figura 2c). La interfaz que permite el envío de comandos para encender/apagar un dispositivo asociado a un puerto se ilustra en la Figura 2d. Con las alarmas (Figura 2e) se visualizan los datos enviados por la computadora a través del módem GSM, dando los informes de la situación actual del cultivo en particular, y con los correos electrónicos e-mail y mensajes SMS, se puede, a partir del celular, enviar un mensaje a alguna cuenta de correo electrónico (Figura 2f).

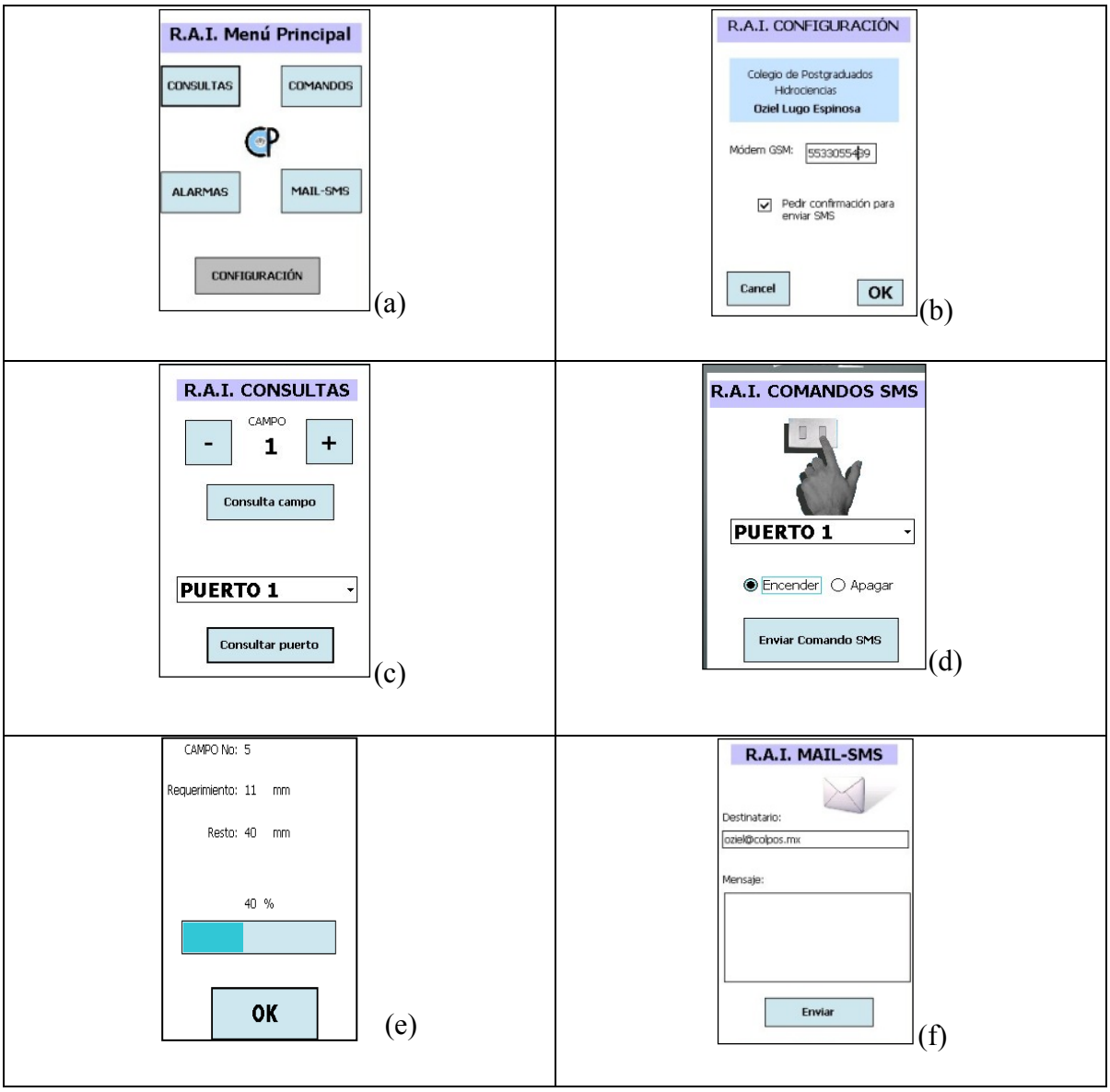


Figura 2. Pantallas de usuario en el celular.

Diálogo principal de RAI en el celular bajo Windows Móvil (a), configuración de RAI (b), consultas de campos o de puertos (c), envío de comandos (d), Alarmas (e), Avisos del celular a una cuenta de correo electrónico (f).

Mensajes de texto interceptados

Todos los lenguajes para programar en plataformas móviles celulares cuentan con una clase denominada "MessageInterceptor" o similar que, básicamente establece lo siguiente:

"siempre que el teléfono reciba un SMS que cumpla ciertas condiciones, envíalo a mi aplicación".

A continuación se muestra el código principal que se utilizó en el proyecto:

```
using Microsoft.WindowsMobile.PocketOutlook;
using Microsoft.WindowsMobile.PocketOutlook.MessageInterception;
//librerias a utilizar
const string smsCommandId = "Mi_aplicacion";
const string smsComparisonValue = "cadena_a_comparar";
MessageInterceptor intercept =   new MessageInterceptor(InterceptionAction.NotifyAndDelete);
intercept.MessageCondition = new MessageCondition();
intercept.MessageCondition.CaseSensitive = true;
intercept.MessageCondition.ComparisonType =   MessagePropertyComparisonType.StartsWith;
// en este caso, el caso de comparacion es para determiner si el mensaje comienza con una palabra clave
intercept.MessageCondition.ComparisonValue = smsComparisonValue;
intercept.EnableApplicationLauncher(smsCommandId);
intercept.MessageReceived +=  new MessageInterceptorEventHandler(intercept_MessageReceived);
void intercept_MessageReceived(object sender,
  MessageInterceptorEventArgs e) {
  if (e.Message is SmsMessage) {
    SmsMessage sms = (SmsMessage)e.Message;
    // el cuerpo del SMS contiene el texto que se recibió
  }
}
```

CONCLUSIONES

El uso de las tecnologías móviles en este caso el celular, son una herramienta que es posible usar para el envío de alarmas y avisos en tiempo real a un teléfono celular vía mensajes de texto. Una opción que permite tomar accionas oportunas que pueden apoyar a mejorar la producción y calidad de los sistemas productivos. En el caso del riego, el agricultor puede hacer uso de estas tecnologías en la toma de decisiones con el fin de alcanzar una mayor eficiencia en la aplicación del limitado recurso agua.

El desarrollo de comandos SMS a través de un celular permite el control remoto a distancia, donde la única limitante es la cobertura GSM de la compañía celular con la que se tiene contratado el servicio.

La disminución de costos resultante del constante desarrollo de la tecnología de comunicación, especialmente en relación con la telefonía celular, es quizá la causa principal del constante aumento del uso de dichos dispositivos. Esto a su vez permite el monitoreo en tiempo real de los cultivos, que se traduce en beneficios inmediatos en cuanto a la aplicación del riego.

REFERENCIAS BIBLIOGRÁFICAS

A Basic Modem FAQ. [en línea]. [Citado el 21 de diciembre de 2009]. Disponible para World Wide Web: http://www.modemshop.com/a_simple_modem_faq.html

Águila, M. F. Entwicklung eines vollautomatischen Bewässerungsregelungssystems für den Freilandgemüsebau. Editorial Verlag Grauer, Beuren – Stuttgart, Alemania. ISBN 3 – 86186–434–7. 2003.

Alexon, Jan. Serial Port Complete: COM Ports, USB Virtual COM Ports, and Ports for Embedded Systems Free, Lakeview Research, 2007, 380 p.

Alexon, Jan. Serial Port Complete: Programing and Circuits for RS-232 and RS-485 Links and Networks, Lakeview Research, 1998, 326 p.

Comandos AT Proyectos. [en línea]. [Citado el 15 de septiembre de 2009]. Disponible para World Wide Web: http://bluehack.elhacker.net/proyectos/comandosat/comandosat.html

Comandos AT. [en línea]. [Citado el 17 de septiembre de 2009]. Disponible para World Wide Web: http://www.eveliux.com/mx/comandos-at.php

Baryy, John; Lee, Eduard; Messerschmitt, David. "Digital Communication". 2004

Castro P., M,. Sistema de riego automatizado en tiempo real con balance hídrico, medición de humedad del suelo y lisímetro, 2008, Agricultura técnica en México. México. ISBN 0568-2517.

Doorenbos J. y Pruitt, W. Las necesidades de agua de los cultivos. Estudio. FAO: Riego y drenaje N° 24. Roma, FAO, 1984,181p.

Calvache, M.; Reichard, K.; Bacchi, O. Efecto de épocas de deficiencia hídrica en evapotranspiración actual de cultivo. Congreso Brasileiro de agrometeorología, 1997, 670 p.

Ganssle, J. y Barr, M., Embedded Systems Dictionary. CMP Books. San Francisco, USA, 2003, 291 p.

Hakala, David. Módems, a su alcance. Osborne-Mc Graw Hill. 1996. 343 p.

Halonen, Timo; Romero, Javier, Melero, Juan, "GSM/GPRS/EDGE PERFORMANCE", Editorial Wiley John and Sons, Segunda Edición, Inglaterra, 2003.

IMTA, Instituto Mexicano de Tecnología del Agua. [en línea].Morelos, México. [Citado el 29 de agosto de 2009]. Disponible para World Wide Web:

http://www.imta.gob.mx/

Kinkoph, Sherry, Módems y servicios en línea fácil. Prentice Hall. 1995. 376 p

Monteith, J. L. & M.H. Unsworth. Principles of environmental physics, 2nd Ed., Edward Arnold, London, UK, 1990,291 p.

Noergaard, T. Embedded Systems Architecture. A Comprehensive Guide For Engineers And Programmers. Elsevier. USA. 2005. 642 p.

Silva V.M.A. Meteorología e Climatología. Brasilia: INMET, Gráfica y Editora Pax, 2001. 532 p.

Sony Ericsson Mobile Communications International. GT47/GT48. Technical Description. 2003

Tavernier, C. Módems, Técnica y Realización. Prentice Hall. 1992. 157 p.

CONTROL ELECTRÓNICO DE DISPOSITIVOS DE RIEGO POR MEDIO DE UNA COMPUTADORA PERSONAL

IRRIGATION DEVICES ELECTRONIC CONTROL WITH A PERSONAL COMPUTER

Oziel Lugo Espinosa[1], Abel Quevedo Nolasco[2], Juan R. Bauer Mengelberg[3], David Hebert del Valle Paniagua[4], Enrique Palacios Vélez[5] y Miguel Águila Marín[6]

Colegio de Postgraduados, México

Postgrado de Hidrociencias, Km. 36.5 México-Texcoco, Texcoco, Edo. de México. CP. 56230.

1. oziel@colpos.mx, (01595) 9523488; 2. anolasco@colpos.mx, 58045900 Ext. 1072 y 1383; 3. jbauer@colpos.mx, 58045900 Ext. 1462; 4.dhvallep@colpos.mx, 58045900 Ext. 1465; 5. epalacios@colpos.mx, 58045900 Ext. 1174;
6. fmaguila@yahoo.com.

RESUMEN

Los modelos matemáticos tienen dificultades para reproducir el fenómeno complejo de la evapotranspiración, lo cual hace necesario disponer de instrumentos de alto costo para alimentar dichas ecuaciones, como es el caso del modelo de Penman-Monteith. Si se tienen los datos necesarios para estimar la evapotranspiracion, se puede aplicar el método del balance hidrico climatológico para determinar la cantidad de agua y el momento adecuado del riego. Es posible automatizar mediante el uso de una computadora y un programa de software el cálculo de requerimientos hídricos de uno o varios cultivos y en consecuencia la acción de encendido y apagado de distintos dispositivos físicos de riego. Cuando el programa de software determina que se debe regar, se envían señales digitales de salida por el puerto serial RS232 de la computadora hacia un dispositivo electrónico para accionar los dispositivos físicos de riego y complementarios, v.gr. electroválvulas, ventiladores o lámparas. Como las computadoras funcionan internamente con bajos niveles energía eléctrica, no son adecuados para tomar directamente acciones de control sobre los actuadores del sistema. La función de las interfaces de potencia es proporcionar la energía eléctrica necesaria para que las señales lógicas generadas por el controlador puedan actuar sobre los elementos físicos. Mientras que las señales eléctricas en electrónica general se

usan para transportar *información*, en electrónica de potencia su función es la de transportar *energía*. En el presente trabajo se describe el proceso de construcción de una interfaz electrónica (control) y de potencia (acción) para el encendido y apagado de dispositivos físicos con el uso de una computadora personal.
Palabras clave: automatización, riego, interface electrónica, multicultivo.

ABSTRACT

The mathematical models have difficulties to reproduce the complex phenomenon of the evapotranspiration, which does necessary to have high-priced instruments to feed the aforementioned equations, as the case is of Penman Monteith's model. If they have the necessary data to estimate the evapotranspiracion, the method of the hydric climatological balance to determine the quantity of water and the moment made suitable of irrigation can be applicable. It is possible to automatize by means of the use of a computer and a program of software the calculation of hydric requests of one or several cultivations and in consequence the action of ignition and extinguished of several physical devices of irrigation. As the computers run internally on low levels electric power, they are not fit to take actions of control on the actuators of the system directly. The show of the interfaces of potency is to provide the necessary electric power in order that they may perform on the logic signs generated by the control on the physical elements. While the electric signs in general electronics are used to transport *information themselves*, in electronics of potency his show is the transporting of energy. The process of construction of an electronic (control) and potency interface (action) for the ignition and extinguished of physical devices with the use of a personal computer is showed in the present work
Keywords: automatization, irrigation, hydric balance, multi-crop.

INTRODUCCIÓN

La creciente preocupación por la disponibilidad de agua, aunada a la consideración de los costos asociados a este vital insume de los cultivos, hace que los sistemas modernos de riego tratan de optimizar la cantidad de agua suministrada para satisfacer los requerimientos hídricos ideales de las plantas. Para estimar los requerimientos de agua, se emplean tanto modelos matemáticos como estimaciones indirectas de la evapotranspiración de la planta. Los modelos matemáticos tienen dificultades para reproducir el fenómeno de la evapotranspiración, dada la complejidad del mismo. Esto a su vez hace necesario disponer de instrumentos de alto costo como estaciones meteorologicas equipadas con diferentes sensores para medir diferentes variables del clima necesarias para alimentar dichas ecuaciones: este es el caso en el modelo de Penman-Monteith (Allen et al.,1998). Si se dispone de los datos necesarios para estimar la evapotranspiracion, se puede aplicar el método del balance hidrico climatologico para determinar la cantidad de agua y el momento adecuado del riego necesario para satisfacer las necesidades de la planta. El balance hídrico se calcula a partir de las variables meteorológicas, además de la información de los cultivos y suelos.

Con base en lo anterior, el objetivo del presente artículo es describir el desarrollo de la etapa de salida de señales tanto de control como de acción sobre distintos dispositivos para el riego, que incluyen las electroválvulas y bombas de agua, pero no excluyen el uso de otros equipos.

MATERIALES Y MÉTODOS

El desarrollo del prototipo de riego implicó el uso de software, desarrollo de una interface electrónica (para el control de los dispositivos de salida y de comunicación bidireccional), información meteorológica (de entrada), e información del suelo(s) y cultivo (s). La herramienta de desarrollo de software fue el entorno “NetBeans IDE 6.7.1” (plataforma de aplicaciones Java®), que permitió la creación del software Riego Inteligente Automático (RAI) para la comunicación y el control de dispositivos de riego. Además se usó la suite Visual Studio 2008® (Microsoft Corporation) para desarrollar el software del dispositivo móvil (teléfono celular), además de una computadora portátil para el desarrollo e

instalación del RAI. Para la componente de comunicación bidireccional de salida, se desarrolló un dispositivo electrónico que usa el puerto serial RS232 de una computadora) y un módem-celular (de uso comercial) que se instaló usando un puerto USB.

Para la componente de control se pueden utilizar distintas estrategias de riego (Castro, 2008). En este trabajo, se tiene como algoritmo de control al balance hídrico, que se calcula a partir de la información meteorológica, del suelo y del cultivo. El balance se realiza entre las salidas y entradas de agua al sistema, donde se compensan de una forma eficiente las pérdidas de agua en el sistema a partir de una función de abatimiento de agua en el suelo. El balance hídrico fue la herramienta para la toma de decisiones en la verificación de la disponibilidad de agua en el sistema cultivo-suelo. A continuación se detalla la componente de control electrónico y de potencia que en conjunto son los encargados del encendido/apagado de los dispositivos fisicos del riego.

SISTEMAS DE CONTROL

Un sistema de control es aquél que se encarga de obtener el resultado deseado de un sistema o proceso, manipulando las variables que afectan dicho resultado. Ganssle y Barr (2003) definen un sistema de control como un sistema integrado cuya función es manipular un dispositivo físico. Algunos ejemplos de sistemas de control comunes son los termostatos, elevadores y sistemas de dirección de vehículos.

Los componentes básicos de un sistema de control se pueden describir como *objetivos, controlador* y *resultados* o *salidas*. Los objetivos son llamados *señal de referencia* y definen el estado que se desea de la variable física de interés. El controlador se integra por elementos que permiten convertir la señal de referencia en una *señal de control*. Estos elementos pueden ser mecánicos, eléctricos, hidráulicos, neumáticos o electrónicos. La señal de control que genera el sistema se debe aplicar entonces al proceso que se desea controlar, una *Planta*. La señal de control modifica la salida entregada por la planta, la cual es la variable final cuyo valor se desea controlar. Las figuras 1 y 2 muestran los diagramas de un controlador y una planta. Aunque ambos sistemas tienen la misma representación en sus respectivos diagramas de bloques, el controlador se diseña para añadirse a la planta y controlar un proceso físico en ella.

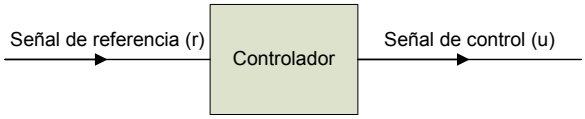


Figura 1. Diagrama de bloques de un sistema de control.

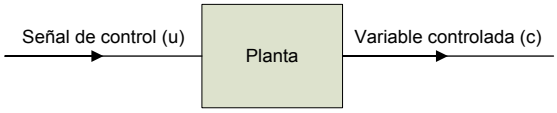


Figura 2. Diagrama de bloques de una planta.

La figura 3 muestra los dos conceptos anteriores que se integran en un solo sistema. En la misma figura se incluye el efecto de las perturbaciones externas. Estas señales entran al sistema después de aplicada la acción de control. El sistema mostrado en la figura 3a es llamado *sistema de control en lazo abierto*, mientras que el ilustrado con la figura 3b es un *sistema de control en lazo cerrado*. En ambos sistemas la variable de interés se ha marcado con la letra *c* y puede representar una temperatura, una posición o cualquier otra magnitud física que se desee controlar.

En los sistemas de control en lazo abierto no se tiene información del efecto de las perturbaciones externas (*n*), por lo que el controlador no puede corregir la señal de control (*u*) para minimizar su efecto negativo. La señal de referencia (*r*) que se alimenta al controlador permanece constante a lo largo del proceso. Los sistemas de control en lazo cerrado incluyen, como elemento adicional, un sensor encargado de monitorear la salida de la planta y compararla con la referencia original para generar la señal que se alimenta al controlador (*b*). Esta señal incluye información del efecto de las perturbaciones externas y así permite corregir la respuesta global del sistema (Kuo, 1996; Ogata, 1998).

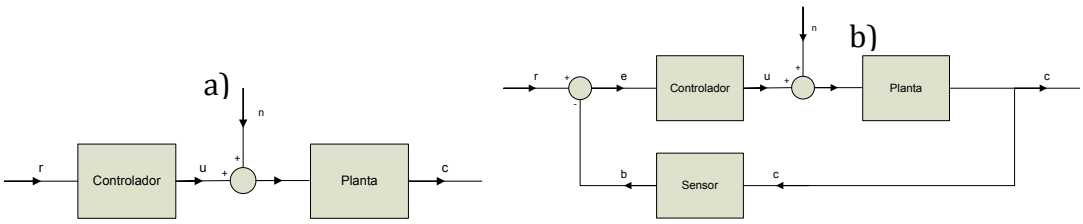


Figura 3. a) Sistema de control en lazo abierto y b)Sistema de control en lazo cerrado.

Donde:

r= señal de referencia

u=señal de control

n=perturbaciones externas

c= variable controlada

La figura 4 ilustra un sistema de control de temperatura en un invernadero, como un ejemplo de un sistema en lazo cerrado. En este caso, hay un solo actuador, un calefactor de gas, por lo que sólo puede elevar la temperatura del invernadero (planta).

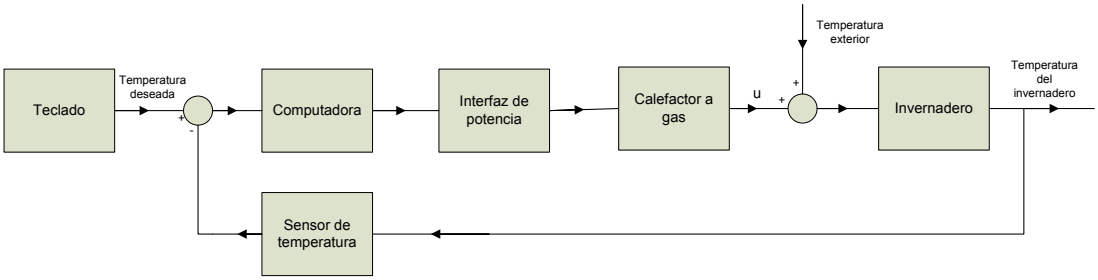


Figura 4. Sistema de control automático de temperatura.

CONTROLADORES

Morais y Boaventura (2000) diseñaron un sistema de control con módulos con base en microcontroladores para adquisición de datos y manejo de actuadores. Estos módulos están conectados a una estación central con una computadora personal a través de un bus de comunicaciones flexible que incluye radiofrecuencia, RS-485, CAN y GSM. Serôdio *et al.* (2001) reportan un sistema de módulos con base en microcontroladores 80C592, con una configuración muy semejante a la utilizada en este trabajo, ya que los módulos incluyen comunicación RS-232, memoria externa y reloj-calendario. Estos microcontroladores incorporan una interfaz CAN, que los autores utilizaron para comunicar el controlador con los actuadores. Los autores recomiendan este bus por presentar buen desempeño en ambientes ruidosos. Todos los trabajos que se mencionan con anterioridad, se enfocan a la automatización de invernaderos.

Como su nombre indica, el elemento más importante de los sistemas de control automático es el controlador. Si bien los controladores pueden ser dispositivos mecánicos, hidráulicos, eléctricos o incluso químicos, los controladores electrónicos digitales están cada vez más difundidos, debido a la flexibilidad que ofrecen para cambiar las estrategias de control.

Los controladores electrónicos digitales más sencillos tienen su base en la lógica combinatoria y secuencial: se los denomina *circuitos digitales*. Estos circuitos constituyen la base para las computadoras electrónicas más complejas, que constituyen la principal opción para la implementación física de los sistemas de control automático.

Los sistemas integrados (*embedded systems*) son computadoras diseñadas para realizar una función específica y carecen de flexibilidad, por lo que se les considera la contraparte de las *computadoras de propósito general*. Los sistemas integrados forman parte de teléfonos celulares, máquinas registradoras, electrodomésticos, equipos médicos, semáforos y muchas otras aplicaciones (Ganssle y Barr, 2003). Sus características centrales son su diseño cerrado o con muy poca flexibilidad, que permite la optimización del tamaño físico y el costo de los componentes. Muchos sistemas electrónicos se consideran sistemas integrados y existe cierta controversia acerca de la terminología, ya que algunos sistemas integrados actuales tienen cierto grado de flexibilidad en la programación, como en el caso de los teléfonos celulares (Noergaard, 2005)

Los controladores lógicos programables (*programmable logic controller, PLC*) son computadoras de nivel medio, dotadas de entradas y salidas adecuadas a las necesidades de la automatización industrial, ámbito en que son ampliamente usados (Moriyón, 2007). Entre sus características se encuentra el modo de programación, que utiliza la llamada *programación escalera* (*ladder logic*). La programación escalera resulta muy intuitiva desde el punto de vista de un ingeniero eléctrico industrial y permite cambios rápidos en los procesos automatizados. Los PLC trabajan de manera secuencial y los cambios se limitan generalmente a redefinir el orden en que se realiza una serie de acciones predefinida, o bien eliminar o añadir nuevas secciones. Los puertos de entrada y salida de los PLC están diseñados de acuerdo a los estándares industriales más difundidos, de manera que los sensores y actuadores se puedan conectar directamente, con un mínimo de modificaciones al programa.

En sentido estricto, los sistemas de registro de datos (*data logger systems*) permiten obtener y registrar información de sensores para su posterior análisis. Se caracterizan por estar equipados con memoria para almacenar grandes volúmenes de datos y pueden tener interfaces muy sencillas (teclados y pantallas de cristal líquido) o bien conectarse a una computadora personal para acceder a los datos desde ésta. Los registradores de datos pueden estar diseñados para monitorear ciertas variables físicas predefinidas (algunos pueden leer una sola variable), pero existen modelos más avanzados que se pueden programar para definir los sensores que se utilizarán. En el siguiente paso, los fabricantes han añadido a estos dispositivos cierta capacidad de procesamiento y control. Actualmente existen en el mercado modelos programables cada vez más flexibles, capaces de realizar cálculos con los datos que obtienen de los sensores y con puertos de control de propósito general.

La capacidad de procesamiento de las computadoras personales, junto con la disminución en su costo final, han permitido la diversificación de las funciones que realizan. Los sistemas de control automático que se apoyan en las computadoras personales tienden a elevar la complejidad del software de control y simplificar el diseño de hardware. Dado que las computadoras se fabrican en serie, con elementos estandarizados, es necesario diseñar periféricos de entrada y salida que permitan al software de control actuar sobre los elementos físicos del sistema (sensores y actuadores) y utilicen los canales estándar de comunicación. Estos canales son básicamente tres: puertos seriales, también llamados puertos COM (utilizan el estándar EIA/TIA 232, antes RS232); puertos seriales universales, USB; y puertos paralelos LPT, que utilizan algunos modelos de impresoras y escáneres, que cada vez son menos. Entre las ventajas de las computadoras personales está el alto grado de complejidad que se puede alcanzar en los cálculos, la gran capacidad de almacenamiento de datos, la posibilidad de diseñar interfaces de usuario muy intuititvas y el fácil acceso a la Internet. Aunque todas estas posibilidades pueden alcanzarse con los sistemas descritos antes, en todos los casos representa etapas adicionales de diseño de hardware y software, mientras que en las computadoras personales alcanza el diseño del software necesario. Por otro lado, los controladores con base en computadoras personales, precisamente por ser sistemas de propósito general, no están optimizados y generalmente incluyen muchos elementos innecesarios en el control.

INTERFACES DE POTENCIA

Las computadoras electrónicas funcionan internamente con bajos niveles de potencia eléctrica. Estos niveles, si bien permiten realizar cálculos matemáticos complejos, no son suficientes para soportar acciones de control sobre los actuadores del sistema de control. Por eso, se emplean interfaces de potencia, cuya función es la de proporcionar la energía necesaria para que las señales lógicas que se generan por el controlador puedan actuar sobre los elementos físicos. Mientras que en electrónica general las señales eléctricas se usan para transportar *información*, en electrónica de potencia sirven para transportar *energía*.

Los interruptores que se usan en las etapas de salida de los sistemas de control automático caen en dos grandes categorías: los electromecánicos y los semiconductores.

Los interruptores electromecánicos incluyen los relés y los contactores. Ambos están compuestos por una bobina que, al energizarse, atrae placas metálicas, provocando que se cierre un circuito eléctrico. Cuanto se retirar la corriente de control, las placas vuelven a su posición original por efecto de un muelle o resorte. De este modo, estos dispositivos están compuestos de dos secciones acopladas magnéticamente: la bobina de control y los contactos y muelles de salida. La primera sección maneja una corriente eléctrica mucho menor a la segunda. La principal diferencia entre relés y contactores está en la intensidad de corriente que pueden manejar: generalmente se llama contactor a un relé que maneja cargas superiores a los 15 amperios, terminología que no es rígida. Una característica de los relés es que con frecuencia presentan tres terminales de salida, dos de las cuales se encuentran en cortocircuito cuando el relé está inactivo (*contacto normalmente cerrado*). Al activar la bobina de control, la tercera terminal entra en contacto con una de las anteriores, que se desconectan entre sí (*contacto normalmente abierto*).

Los dispositivos semiconductores que se utilizan como interruptores son los transistores y los tiristores (Mandado, 1991), ambos están construidos en capas alternadas de semiconductores con carga positiva y negativa (*tipo p* y *tipo n)* que permiten o impiden la circulación de una corriente eléctrica grande entre dos terminales, en función de la presencia de una corriente o un voltaje relativamente pequeños en una terminal de control. De acuerdo a sus características específicas, algunos transistores y tiristores se usan para controlar cargas de corriente alterna o corriente directa, de bajo o alto voltaje, y de diferente

capacidad de corriente y velocidad de conmutación. A diferencia de lo que sucede en los relés, no existe movimiento y contacto mecánico, esto hace que los dispositivos semiconductores tengan mayor tiempo de vida, generen menos ruido electromagnético, y sean más rápidos y silenciosos. Sin embargo, tienden a ser más caros y requieren un diseño electrónico más complejo. Además, permiten la circulación de pequeñas cantidades de corriente eléctrica cuando están desactivados, lo que puede energizar cargas sensibles. Finalmente, presentan mayor impedancia cuando están activados, lo que resulta en una mayor disipación de calor.

Un caso especial de los tiristores son los *relés de estado sólido*, que son dispositivos encapsulados, constituidos por uno o más tiristores y terminales de control (Couëdic, 1999), listos para uso de manera similar a los relés. Ofrecen las ventajas y desventajas de los tiristores y la simplicidad de uso de los relés, aunque a un costo mucho mayor.

COMUNICACIONES ELECTRÓNICAS

Los sistemas de control deben transmitir información entre sí, que incluye mediciones de sensores, valores de referencia, señales de activación de salidas y comunicación entre los elementos internos del sistema. La comunicación de datos implica que la información es digital tanto en la fuente como en el destino, aunque durante la transmisión se puede efectuar tanto en forma digital como analógica. Esto quiere decir que se pueden utilizar canales analógicos (como la modulación en amplitud o frecuencia) para transmitir información digital (Tomasi, 2003).

Algunas de las interfaces de comunicaciones seriales que más se utilizan en los sistemas de control automático son la EIA/TIA-232 (antes RS-232), EIA-485, I^2C, USB y Ethernet.

La interfaz EIA/TIA-232 se estableció en 1962 por la Asociación de Industriales Electrónicos (AIE, 2009) y sufrió su cuarta modificación en 1987. Por las especificaciones técnicas del cable y usando una velocidad máxima de bits de 20kbps, la longitud máxima de la interfaz RS-232 es de aproximadamente 15m (Tomasi, 2003). Esta interfaz era estándar en las computadoras personales, aunque en los últimos modelos se desplazó por el

bus USB. Es importante el hecho que únicamente se contempla comunicación punto a punto, es decir, con este sistema se pueden conectar entre sí sólo dos elementos.

La interfaz USB (*universal serial bus*) se introdujo en 1995, estandarizada por un comité en que participan las principales compañías productoras de equipos de cómputo y periféricos. La intención era desplazar los puertos paralelo y serial por un bus más rápido, con un conector estándar y con mayor facilidad de interconexión, que permite conexiones en estrella de hasta 127 dispositivos en un máximo de 5 niveles de profundidad. Utiliza una línea balanceada para combatir el ruido y permite distancias de hasta 5m por cable; mediante interconectados mediante el uso de repetidores, proporcionan distancias de hasta 25 m. Aunque hay una amplia variedad de periféricos USB para computadoras personales, no es posible controlarlos de manera directa, puesto que usan programas cerrados.

Todos los sistemas de comunicaciones electrónicas tienen que lidiar con problemas de ruido eléctrico (cualquier energía eléctrica indeseable que cae dentro de la banda útil de la señal de interés) (Tomasi, 2003), existen muchas fuentes de ruido eléctrico, entre las que cabe destacar las siguientes:

- Ruido causado por el hombre (producido por mecanismos que producen chispas como conmutadores de motores, sistemas de encendido automotriz, lámparas fluorescentes).
- Ruido interno (producido por la circulación de corriente eléctrica en el interior de los circuitos y por efectos térmicos de esta circulación).
- Distorsión armónica (generada cuando se producen ondas derivadas de la señal principal que después interfieren con ésta) y
- Distorsión por intermodulación (se genera cuando se mezclan las frecuencias inadecuadas en un circuito que suma o resta señales).

La *interferencia* es una consecuencia de la distorsión (Barcells, 1992), pues se genera cuando una señal produce ondas en una frecuencia que no debería estar presente y que afecta a otra señal independiente. Un ejemplo de esto se produce al usar un teléfono y notar fallas en la comunicación cuando se enciende un motor eléctrico o se acerca a líneas de alto voltaje.

Si bien los circuitos digitales tienen mayor inmunidad al ruido que los analógicos, también se ven afectados por aquél. Un problema de la comunicación de datos es que la corrupción de tan solo un bit puede generar datos muy diferentes y un comportamiento errático. Precisamente esta circunstancia hace necesarios los procedimientos de control de errores en las comunicaciones digitales. Estos procedimientos incluyen la *paridad* (Sklar, 2004) y la *comprobación de redundancia cíclica (*Stremler, 1993). La paridad es un método muy sencillo que permite reconocer muchos errores de cambio de bits en un byte particular, mientras que la comprobación de redundancia cíclica (CRC) permite identificar errores en una trama compuesta de varios bytes que se envían como un paquete.

RESULTADOS Y DISCUSIÓN

Para el desarrollo de este trabajo, se construyó una interfaz de control (figura 5) con base en un pic para interpretar las señales emitidas por la computadora, y una etapa de potencia compuesta por optoacopladores (figura 6) y relés para accionar los periféricos del sistema (electroválvulas).

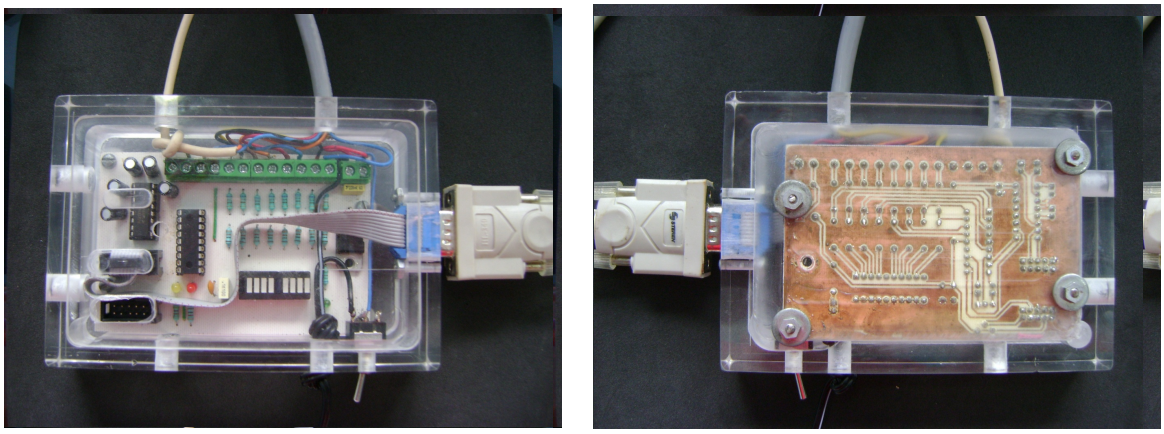


Figura 5. Circuito electrónico

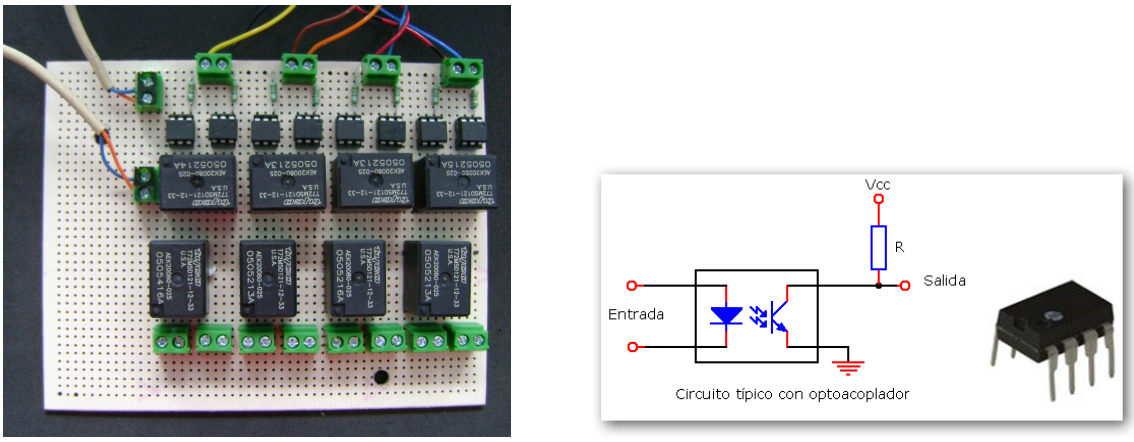


Figura 6. Interface de potencia y optoacoplador

SOFTWARE DE CONTROL

La programación de microcontroladores, que anteriormente se hacía en el lenguaje ensamblador específico para cada dispositivo (Pallas, 2007), ahora se efectúa usando compiladores de nivel intermedio que facilitan la interpretación y la portabilidad del código entre aplicaciones. En este trabajo se programaron los microcontroladores con el compilador CCS-C (Ccs Inc, 2009) que permite codificar en lenguaje C y genera posteriormente los archivos en código máquina que son grabados físicamente en el dispositivo. Se puede ver la ventana de trabajo del compilador CCS-C en la figura 7. Para grabar los programas se utilizó el software PIC600 (Steren, 2009) y el hardware USBPIC600 (figura 8).

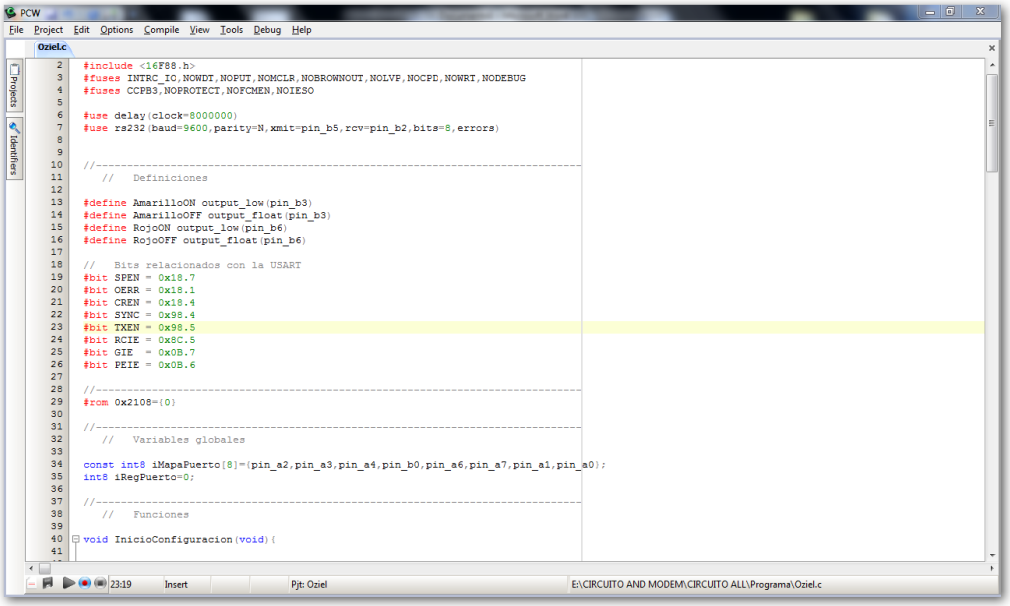


Figura 7. Ventana ejemplo del software de grabación PIC 600

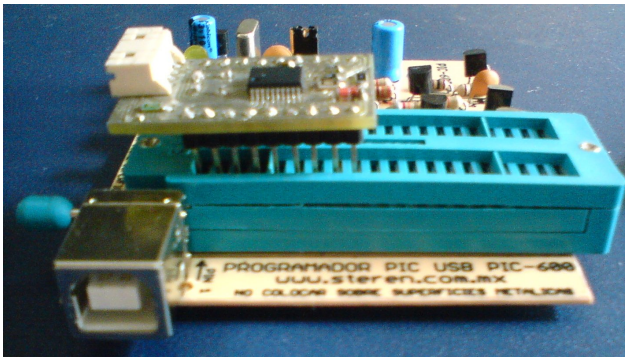


Figura 8. Grabador de microcontroladores Pic600

Los programas en los microcontroladores difieren en su forma de ejecución de los programas tradicionales que corren en computadoras personales. Los microcontroladores ejecutan su programa en ciclo de manera infinita, ya que se espera que sean parte de sistemas integrados, aunque también rutinas de interrupción que permiten ejecutar segmentos de código adicionales al programa principal. Las interrupciones se generan con determinados eventos, por ejemplo un cambio de voltaje en ciertas entradas, el paso de un intervalo pequeño de tiempo específico o una entrada de comunicaciones seriales. Mientras estos eventos no ocurren, el programa principal se mantiene en ejecución. Para la

comunicación de la computadora con el circuito electrónico, se utilizó el PIC18F88 (figura 9) de Microchip (Michochip, 2009).

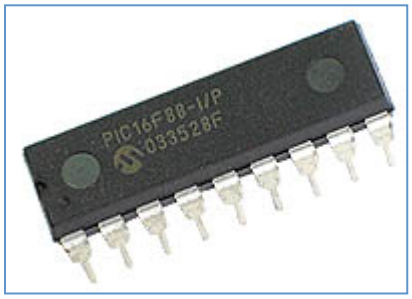


Figura 9. PIC 16F88

DIAGRAMAS ELECTRÓNICOS

A continuación se presentan los diagramas electrónicos que se utilizan para la realización de la interfaz de control, y el código insertado en el pic para la interpretación de las señales de la computadora. La figura 10 muestra el regulador de corriente cuya función es proteger al circuito electrónico de descargas no deseadas.

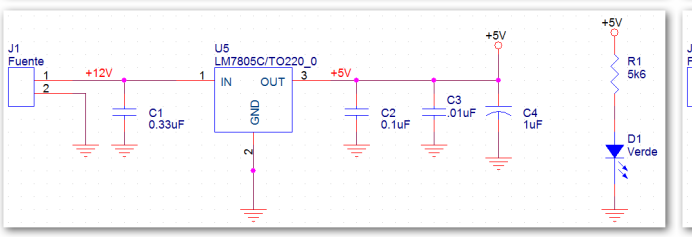


Figura 10. Regulador de corriente.

La figura 11 muestra el convertidor de voltaje para el puerto serial RS232 debido a que los voltajes utilizados por el pic y el puerto serial de la computadora son distintos. Para que exista una comunicación entre ambos dispositivos, el circuito electrónico max232 funciona como un “traductor” de voltajes entre ambos componentes.

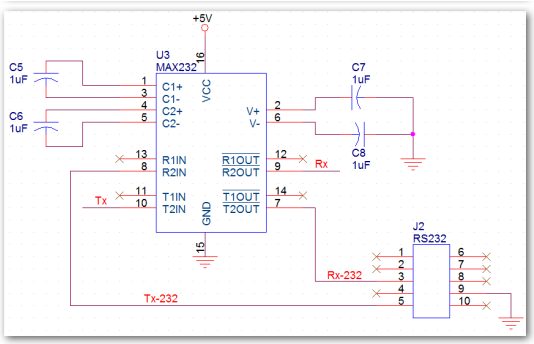


Figura 11. Max232. Convertidor de voltaje para el puerto serial RS232

En la figura 12 se puede observar la configuración de fábrica del Pic 16f88, asi también como las terminales usadas en el proyecto.

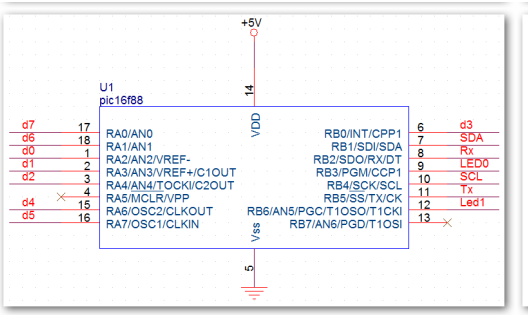


Figura 12. PIC 16F88

A continuación, se muestra el cógido fuente escrito en lenguaje C que se compiló y cargó al PIC 16f88 para el control de dispositivos desde una computadora.

CÓDIGO FUENTE

```
#include <16F88.h>
#fuses INTRC_IO, NOWDT, NOPUT, NOMCLR, NOBROWNOUT, NOLVP, NOCPD,
NOWRT, NODEBUG
#fuses CCPB3, NOPROTECT, NOFCMEN, NOIESO
#use delay(clock=8000000)
#use rs232(baud=9600,parity=N,xmit=pin_b5,rcv=pin_b2,bits=8,errors)

 //   Definiciones
#define AmarilloON output_low(pin_b3)
#define AmarilloOFF output_float(pin_b3)
#define RojoON output_low(pin_b6)
```

```
#define RojoOFF output_float(pin_b6)

//   Bits relacionados con la USART
#bit SPEN = 0x18.7
#bit OERR = 0x18.1
#bit CREN = 0x18.4
#bit SYNC = 0x98.4
#bit TXEN = 0x98.5
#bit RCIE = 0x8C.5
#bit GIE  = 0x0B.7
#bit PEIE = 0x0B.6
//-------------------------------------------------------------------------------
#rom 0x2108={0}

 //   Variables globales
const int8 iMapaPuerto[8]={pin_a2,pin_a3,pin_a4,pin_b0,pin_a6,pin_a7,pin_a1,pin_a0};
int8 iRegPuerto=0;
//-------------------------------------------------------------------------------

//   Funciones
void InicioConfiguracion(void){
   disable_interrupts(global);
   setup_timer_0(RTCC_INTERNAL);
   setup_timer_1(T1_DISABLED);
   setup_timer_2(T2_DISABLED,0,1);
   setup_ccp1(CCP_OFF);
   setup_uart(TRUE);                  // Requires: #use rs232
   setup_comparator(NC_NC_NC_NC);
   setup_oscillator(OSC_8MHZ|OSC_INTRC);
   setup_adc(ADC_OFF);
   setup_adc_ports(NO_ANALOGS);
   disable_interrupts(int_rtcc);
   disable_interrupts(int_rb);
   disable_interrupts(int_ext);
   disable_interrupts(int_ad);
   disable_interrupts(int_tbe);
   enable_interrupts(int_rda);
   disable_interrupts(int_timer1);
   disable_interrupts(int_timer2);
   disable_interrupts(int_ccp1);
   disable_interrupts(int_ssp);
   disable_interrupts(int_eeprom);
   disable_interrupts(int_timer0);
   disable_interrupts(int_comp);
   disable_interrupts(int_osc_fail);
   enable_interrupts(global);
}
```

```
void SacaPuertoMapa(int8 iDato){
int8 i;
  for(i=0;i<8;i++){
    if(bit_test(iDato,i))   output_high(iMapaPuerto[i]);
    else                    output_low(iMapaPuerto[i]);
  }
}

//-----------------------------------------------------------------------
//      Interrupciones
/*
  Estructura del dato que entra:
    ---------------------------------------------------------------------------------------------------
  |   b7   |    b6  |    b5  |    b4  |    b3  |    b2  |   b1   |   b0   |
  ---------------------------------------------------------------------------------------------------
  |  Preg  |  E/-A  |  Bit5  |  Bit4  |  Bit3  |  Bit2  |  Bit1  |  Bit0  |
  ---------------------------------------------------------------------------------------------------

  Preg: 1=PC pide el estado del puerto ; 0= PC ordena el estado del puerto
  E/-A:   1=Pin de salida =5V            ;   0=Pin de salida =0V
  Bit5->Bit0:   Número del pin de salida a que hace referencia el comando
*/

#int_rda
void rda_isr(void){
int8 iDatoEntra;
int8 iNumSalida;

  RojoON;
  if(OERR){
    CREN=0;
    delay_ms(10);
    CREN=1;                 // Manejo de errores de recepción
  }

  disable_interrupts(int_rda);
  if(kbhit()){
    iDatoEntra=getc();
    iNumSalida = iDatoEntra&0x3F;              // Máscara con el número de salida
    if( bit_test(iDatoEntra,7) ){              // Piden consultar estado
      if( iNumSalida<8 )
        putc( (int8)bit_test(iRegPuerto,iNumSalida) );
      else
        putc(0);                               // Aquí la revisión de pines extras
    }
    else{
```

```
        if( bit_test(iDatoEntra,6) ){              // Nos piden encender salida
          if( iNumSalida<8 )
            output_high(iMapaPuerto[iNumSalida]);
            bit_set(iRegPuerto,iNumSalida);
        }
        else{                                      // Nos piden apagar salida
          if( iNumSalida<8 )
            output_low(iMapaPuerto[iNumSalida]);
            bit_clear(iRegPuerto,iNumSalida);
        }
        write_eeprom(0x08,iRegPuerto);
      }
    }
    enable_interrupts(int_rda);
    RojoOFF;
}
//-------------------------------------------------------------------------------
void main(){
    InicioConfiguracion();  //configura el PIC
    iRegPuerto=read_eeprom(0x08);
    SacaPuertoMapa(iRegPuerto);
    AmarilloOFF;
    RojoOFF;
    putc(0x55);
//-------------------------------------------------------------------------------
    while(1){
      AmarilloON;
      delay_ms(1000); //retraso de 1000 milisegundos
      AmarilloOFF;
      delay_ms(1000); //retraso de 1000 milisegundos
    }
//-------------------------------------------------------------------------------
}
```

La figura 13 muestra el diagrama de flujo del código fuente anterior.

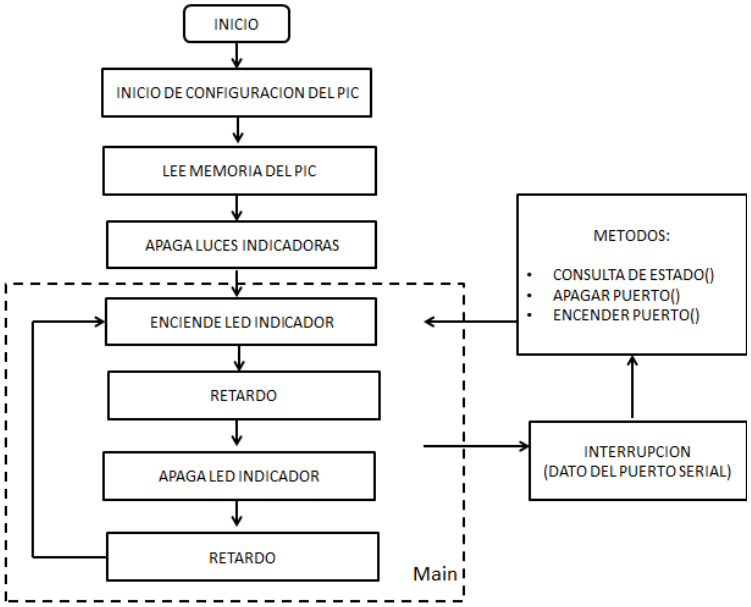


Figura 13. Diagrama de flujo del programa en el PIC

CONCLUSIONES

El diseño del circuito electrónico cumplió satisfactoriamente las tareas de procesamiento de las señales enviadas desde la computadora hacia los distintos dispositivos físicos. Se logró una transferencia confiable de los datos digitales de la computadora hacia señales eléctricas de control.

Como parte complementaria se logró encender/apagar algún dispositivo por medio de teléfonos celulares, lo que a su vez posibilita la automatización del riego, en tiempo real, a partir de variables de suelo, clima y cultivo, con el fin de de alcanzar una mayor eficiencia en la aplicación del limitado recurso agua.

Es posible aplicar el resultado obtenido en cualquier rama derivada de la automatización ya que una vez que se tiene el control sobre algún dispositivo electrico desde una compuardora, se puede desarrollar el software necesario para su control y automatización.

REFERENCIAS BIBLIOGRÁFICAS

AIE. Asociación de la industria eléctrica. [en línea]. [Citado el 3 de diciembre de 2009]. Disponible para World Wide Web: http://www.aie.cl/

Balcells Josep, Francesc Daura, Rafael Esparza, Ramón Pallás, Interferencias Electromagnéticas en sistemas electrónicos. Serie Mundo Electrónico, Ed. Marcombo 1992.

Castro P., M,. Sistema de riego automatizado en tiempo real con balance hídrico, medición de humedad del suelo y lisímetro, 2008, Agricultura técnica en México. México. ISBN 0568-2517.

Ccs Inc. [en línea]. [Citado el 16 de diciembre de 2009]. Disponible para World Wide Web: http://www.ccsinfo.com/

Couëdic, Marc. Circuitos integrados para tiristores y triacs. Alfaomega Grupo Editor. Boixareu Editores, Marcombo, México, 1999, 191 p.

Ganssle, J. y Barr, M., Embedded Systems Dictionary. CMP Books. San Francisco, USA, 2003, 291 p.

Kuo, B. 1996. Sistemas de control automático. Prentice Hall. México. 905 p.

Mandado, Enrique. Sistemas Electrónicos digitales 7ºedición, Marcombo boixareu editors Barcelona-México, 1991, 884 p.f

Microchip. [en línea]. [Citado el 20 de diciembre de 2009]. Disponible para World Wide Web: http://www.microchip.com

Monteith, J. L. & M.H. Unsworth. Principles of environmental physics, 2nd Ed., Edward Arnold, London UK, 1990,291 p.

Morais, R. y Boaventura, J. 2000. *Agrotronics: A Distributed Data Aquisition And Control Network For Agriculture Environments*. Acta Horticulturae 534: 319-325.

Moriyón Roberto; Alfonseca Manuel; Alfonseca Enrique . Teoría de autómatas y lenguajes formales. McGRAW-HILL/INTERAMERICANA DE ESPAÑA, S.A.U. 2007, 464 páginas

Noergaard, T. Embedded Systems Architecture. A Comprehensive Guide For Engineers And Programmers. Elsevier. USA. 2005. 642 p.

Ogata, K. 1998. *Ingeniería de control moderna.* Pearson Educación. México. 997 p.

Pallás, Ramón, Valdés, Fernando, Microcontroladores: fundamentos y aplicacones con pic. Marcombo, ediciones técnicas, 2007, 344 páginas

Serôdio, C., Boaventura Cunha, J., Morais, R., Couto, C. y Monteiro J. 2001. A Networked Platform For Agrcultural Management Systems. Computers and electronics in agriculture 31: 75-90.

Sklar, Bernard. Digital Communications, Fundamentals and Applications. Prentice-Hall. Englewood Cliffs, USA. 2004, 1079 p.

Steren. [en línea]. [Citado el 13 de diciembre de 2009]. Disponible para World Wide Web: http://www.steren.com.mx/catalogo/interior3.asp?pdto=PIC-600

Stremler, F.G. Introducción a los Sistemas de Comunicación. Addison-Wesley Iberoamericana, Estados Unidos, 1993. 742 p.

Tomasi, W. Sistemas de comunicaciones electrónicas. Pearson Educación. México, 2003, 976 p.

CONCLUSIONES

Se logró el objetivo planteado: el desarrollo del prototipo que permite accionar los dispositivos de salida a partir de un sistema de control - donde se toma la decisión a partir de un balance hídrico - de manera autorregulada. Como parte complementaria se logró encender/apagar algún dispositivo por medio de teléfonos celulares, lo que a su vez posibilita la automatización del riego, en tiempo real, a partir de variables de suelo, clima y cultivo, con el fin de alcanzar una mayor eficiencia en la aplicación del limitado recurso agua.

Se utilizó los valores del cultivo de calabaza zucchini grey (*cucúrbita spp*.) para probar el sistema y comparar resultados, obteniendo los mismos datos que Castro (2008), con su proyecto de automatización de riego con base en tres estrategias distintas de riego.

La utilización de tecnologías de comunicación como teléfonos celulares, facilita un seguimiento e incluso el control del sistema en tiempo real, al obtener información del estado del sistema y enviar comandos u órdenes de ejecución, lo que se traduce en beneficios inmediatos en cuanto a la aplicación del riego.

Para que el prototipo realice los cálculos adecuados (cantidad de agua y momento adecuado) para cada cultivo, es necesario introducir en el sistema los parámetros del suelo donde se establece cada cultivo y los datos característicos de cada cultivo (Kc).

La programación orientada a objetos, junto con las diferentes tecnologías utilizadas, facilitó la inserción necesarios en el programa principal de diversos algoritmos y la alimentación de información (meteorológicos, cultivo, suelo, geográficos), indispensables para la estimación objetiva del riego.

RECOMENDACIONES

Se debe tener en cuenta el ruido eléctrico, es decir, la energía eléctrica indeseable generan por la computadora y los dispositivos físicos que se desean controlar, por ello es importante tener un acoplamiento óptico entre los dispositivos de salida y la computadora, lo que evita el ruido y resulta en un funcionamiento adecuado del sistema. Para contar con el desarrollo completo del control de riego en un sistema abierto, falta el desarrollo de un sistema de adquisición automatizada de datos meteorológicos.

LITERATURA CITADA

ARTÍCULO 1

Águila, M. F. Entwicklung eines vollautomatischen Bewässerungsregelungssystems für den Freilandgemüsebau. Editorial Verlag Grauer, Beuren – Stuttgart, Alemania. ISBN 3 – 86186–434–7. 2003.

Bralts, V. F., M. A. Driscoll and F. Kelly S. Microcomputer based irrigation management and control system. ASAE, Paper No. 86-1223. St Joseph, MI.USA,1986.

Castro P., M,. Sistema de riego automatizado en tiempo real con balance hídrico, medición de humedad del suelo y lisímetro, 2008, Agricultura técnica en México. México. ISBN 0568-2517.

CIMIS. CIMIS Agricultural Resources Book. California Irrigation Management information system CIMIS, California Department of Water Resources. 2005.

Doorenbos J. y Pruitt, W. Las necesidades de agua de los cultivos. Estudio. FAO: Riego y drenaje N° 24. Roma, FAO, 1984,181p.

Calvache, M.; Reichard, K.; Bacchi, O. Efecto de épocas de deficiencia hídrica en evapotranspiración actual de cultivo. Congreso Brasileiro de agrometeorología, 1997, p.668-670.

IMTA, Instituto Mexicano de Tecnología del Agua. [en línea].Morelos, México. [Citado el 29 de agosto de 2009]. Disponible para World Wide Web: http://www.imta.gob.mx/

INIFAP, Instituto Nacional de Investigaciones Forestales, Agrícolas y Pecuarias, Red Nacional De Estaciones Estatales Agroclimatológicas, [en línea], México. [Citado el 27 de agosto de 2009]. Disponible para World Wide Web: http://clima.inifap.gob.mx/redclima/

Mesonet [en línea]. Oklahoma, EU. [Citado el 10 de septiembre de 2009]. Disponible para World Wide Web: http://www.mesonet.org/

Morais, R. y Boaventura, J. 2000. Agritronics: A Distributed Data Aquisition and Control Network For Agriculture Environments. Acta Horticulturae 534: 319-325.

Moreno A., S., L. Tijerina Ch., R. Acosta H., V.M. Ruiz C., F.S. Zazueta R., y G. Crespo P.

Automatización de un sistema de riego localizado aplicado a una plantación de durazno. Agrociencia, Vol. 33, 1996,2: 191-197

Noergaard, T. Embedded Systems Architecture. A Comprehensive Guide For Engineers And Programmers. Elsevier. USA. 2005. 642 p.

Palacios V. E., & A. Excebio. Introducción a la teoría de la operación de distritos de riego. Segunda reimpresión corregida. Centro de Hidrociencias. Colegio de Postgraduados, México. 1989.

Silva V.M.A. Meteorología e Climatología. Brasilia: INMET, Gráfica y Editora Pax, 2001. 532 p.

Tomasi, W., Sistemas de comunicaciones electrónicas. Pearson Educación. México, 2003, 976 p.

United Nations. Water for people water for life. Executive Summary of the UN Worl Water Development Report. Educational Scientific and Cultural Organuzation (UNESCO), Paris, France, 2003, 604 p.

Wessels, W.P.J., W.H. Steyn and J.H. Moolman. Automatic microirrigation and salt injection system for research and commercial applications. Proceeding of the fifth international microirrigation Congress. Orlando Fl., USA. ASAE,1995, p. 116-122.

ARTÍCULO 2

A Basic Modem FAQ. [en línea]. [Citado el 21 de diciembre de 2009]. Disponible para World Wide Web: http://www.modemshop.com/a_simple_modem_faq.html

Águila, M. F. Entwicklung eines vollautomatischen Bewässerungsregelungssystems für den Freilandgemüsebau. Editorial Verlag Grauer, Beuren – Stuttgart, Alemania. ISBN 3 – 86186–434–7. 2003.

Alexon, Jan. Serial Port Complete: COM Ports, USB Virtual COM Ports, and Ports for Embedded Systems Free, Lakeview Research, 2007, 380 p.

Alexon, Jan. Serial Port Complete: Programing and Circuits for RS-232 and RS-485 Links and Networks, Lakeview Research, 1998, 326 p.

Comandos AT Proyectos. [en línea]. [Citado el 15 de septiembre de 2009].

Disponible para World Wide Web:
http://bluehack.elhacker.net/proyectos/comandosat/comandosat.html

Comandos AT. [en línea]. [Citado el 17 de septiembre de 2009].
Disponible para World Wide Web:
http://www.eveliux.com/mx/comandos-at.php

Baryy, John; Lee, Eduard; Messerschmitt, David. "Digital Communication". 2004

Castro P., M,. Sistema de riego automatizado en tiempo real con balance hídrico, medición de humedad del suelo y lisímetro, 2008, Agricultura técnica en México. México. ISBN 0568-2517.

Doorenbos J. y Pruitt, W. Las necesidades de agua de los cultivos. Estudio. FAO: Riego y drenaje N° 24. Roma, FAO, 1984,181p.

Calvache, M.; Reichard, K.; Bacchi, O. Efecto de épocas de deficiencia hídrica en evapotranspiración actual de cultivo. Congreso Brasileiro de agrometeorología, 1997, 670 p.

Ganssle, J. y Barr, M., Embedded Systems Dictionary. CMP Books. San Francisco, USA, 2003, 291 p.

Hakala, David. Módems, a su alcance. Osborne-Mc Graw Hill. 1996. 343 p.

Halonen, Timo; Romero, Javier, Melero, Juan, "GSM/GPRS/EDGE PERFORMANCE", Editorial Wiley John and Sons, Segunda Edición, Inglaterra, 2003.

IMTA, Instituto Mexicano de Tecnología del Agua. [en línea].Morelos, México. [Citado el 29 de agosto de 2009]. Disponible para World Wide Web: http://www.imta.gob.mx/

Kinkoph, Sherry, Módems y servicios en línea fácil. Prentice Hall. 1995. 376 p

Monteith, J. L. & M.H. Unsworth. Principles of environmental physics, 2nd Ed., Edward Arnold, London, UK, 1990,291 p.

Noergaard, T. Embedded Systems Architecture. A Comprehensive Guide For Engineers And Programmers. Elsevier. USA. 2005. 642 p.

Silva V.M.A. Meteorología e Climatología. Brasilia: INMET, Gráfica y Editora Pax, 2001. 532 p.

Sony Ericsson Mobile Communications International. GT47/GT48. Technical Description. 2003

Tavernier, C. Módems, Técnica y Realización. Prentice Hall. 1992. 157 p

ARTÍCULO 3

AIE. Asociación de la industria eléctrica. [en línea]. [Citado el 3 de diciembre de 2009]. Disponible para World Wide Web: http://www.aie.cl/

Balcells Josep, Francesc Daura, Rafael Esparza, Ramón Pallás, Interferencias Electromagnéticas en sistemas electrónicos. Serie Mundo Electrónico, Ed. Marcombo 1992.

Castro P., M,. Sistema de riego automatizado en tiempo real con balance hídrico, medición de humedad del suelo y lisímetro, 2008, Agricultura técnica en México. México. ISBN 0568-2517.

Ccs Inc. [en línea]. [Citado el 16 de diciembre de 2009]. Disponible para World Wide Web: http://www.ccsinfo.com/

Couëdic, Marc. Circuitos integrados para tiristores y triacs. Alfaomega Grupo Editor. Boixareu Editores, Marcombo, México, 1999, 191 p.

Ganssle, J. y Barr, M., Embedded Systems Dictionary. CMP Books. San Francisco, USA, 2003, 291 p.

Kuo, B. 1996. Sistemas de control automático. Prentice Hall. México. 905 p.

Mandado, Enrique. Sistemas Electrónicos digitales 7ºedición, Marcombo boixareu editors Barcelona-México, 1991, 884 p.f

Microchip. [en línea]. [Citado el 20 de diciembre de 2009]. Disponible para World Wide Web: http://www.microchip.com

Monteith, J. L. & M.H. Unsworth. Principles of environmental physics, 2nd Ed., Edward Arnold, London UK, 1990,291 p.

Morais, R. y Boaventura, J. 2000. *Agrotronics: A Distributed Data Aquisition And Control Network For Agriculture Environments*. Acta Horticulturae 534: 319-325.

Moriyón Roberto; Alfonseca Manuel; Alfonseca Enrique . Teoría de autómatas y lenguajes formales. McGRAW-HILL/INTERAMERICANA DE ESPAÑA, S.A.U. 2007, 464 páginas

Noergaard, T. Embedded Systems Architecture. A Comprehensive Guide For Engineers And Programmers. Elsevier. USA. 2005. 642 p.

Ogata, K. 1998. *Ingeniería de control moderna.* Pearson Educación. México. 997 p.

Pallás, Ramón, Valdés, Fernando, Microcontroladores: fundamentos y aplicacones con pic. Marcombo, ediciones técnicas, 2007, 344 páginas

Serôdio, C., Boaventura Cunha, J., Morais, R., Couto, C. y Monteiro J. 2001. A Networked Platform For Agrcultural Management Systems. Computers and electronics in agriculture 31: 75-90.

Sklar, Bernard. Digital Communications, Fundamentals and Applications. Prentice-Hall. Englewood Cliffs, USA. 2004, 1079 p.

Steren. [en línea]. [Citado el 13 de diciembre de 2009]. Disponible para World Wide Web: http://www.steren.com.mx/catalogo/interior3.asp?pdto=PIC-600

Stremler, F.G. Introducción a los Sistemas de Comunicación. Addison-Wesley Iberoamericana, Estados Unidos, 1993. 742 p.

Tomasi, W. Sistemas de comunicaciones electrónicas. Pearson Educación. México, 2003, 976 p.

ANEXOS

A: CÓDIGO FUENTE SISTEMA RAI

```
package raicelular;
import java.util.GregorianCalendar;
import javax.swing.JOptionPane;

public class Main {
public static boolean circuito_activado=false;
public static boolean modem_activado=false;
public static String centro_mensajes = "5512235110";
//private static String centro_mensajes_telcel="5512235110";
public static String destinatarioSMS = "5537867704";
//public static String emisorSMS="5537867704";
public static JFMenu miventana=new JFMenu();
public static Reloj miReloj=new Reloj();
public static int resultadoConsulta=0;
public static boolean quiereMails=false;
public static boolean quiereSMS=false;
public static boolean quiereControlSMS=false;
public static boolean estaConfiguradoCircuito=false;
public static boolean estaConfiguradoModem=false;

    public static void main(String[] args) {
        miventana.carga_puertos();
        miventana.setLocationRelativeTo(null);
        miventana.setVisible(true);
    }

    public static boolean isCircuito_activado() {
        return circuito_activado;
    }

    public static void setCircuito_activado(boolean circuito_activado) {
        Main.circuito_activado = circuito_activado;
    }

    public static boolean isModem_activado() {
        return modem_activado;
    }

    public static void setModem_activado(boolean modem_activado) {
        Main.modem_activado = modem_activado;
    }


package raicelular;
import java.util.Observable;
import java.util.Observer;
import javax.swing.Timer;
import java.awt.event.*;
import java.util.Calendar;
import java.util.GregorianCalendar;
public class Reloj extends Observable
 {
```

```
        GregorianCalendar calendario=new GregorianCalendar();


    public Reloj()
     {
      //el tiempo esta en milisegundos

      Timer timer = new Timer (10000, new ActionListener ()
                  {
                   public void actionPerformed(ActionEvent e)
             {
                 setChanged();
                 notifyObservers (calendario.get(Calendar.MINUTE));
             }
         });

         timer.start();
    }

        public void inicia()
     {
         Reloj modelo = new Reloj();
         modelo.addObserver(new Observer()
             {

                          public void update (Observable unObservable,
Object dato)
                          {
                          int intensidad=
Main.miventana.smsclass.verifica_senial();
Main.miventana.jProgressBarSeÃ±al.setValue(intensidad);

                          }

                }

         );

    }

}
package raicelular;
import javax.mail.*;
import javax.mail.internet.*;
import java.util.Properties;
import javax.swing.JOptionPane;
public class SendAuthentication {
private String mensajeSMS="";
  public void Send(String destinatario, String mensaje)
{
    this.mensajeSMS=mensaje;
    String host ="smtp.gmail.com";
    String from ="mail@gmail.com";
    String to =destinatario;
   Properties prop = new Properties();
    prop.setProperty("mail.smtp.host", host);
    prop.setProperty("mail.smtp.auth", "true");
```

```
    prop.setProperty("mail.smtp.starttls.enable", "true");
    prop.setProperty("mail.smtp.port","587");

    try{
        SMTPAuthentication auth = new
SMTPAuthentication("mail","password");
        auth.getPasswordAuthentication();
        Session session = Session.getDefaultInstance(prop, auth);
        Message msg = getMessage(session, from, to);
        System.out.println("Enviando ...");
        Transport.send(msg);
        System.out.println("Mail enviado!");
        JOptionPane.showMessageDialog(null, "Mail enviado", "R.A.I.",
JOptionPane.PLAIN_MESSAGE);
    }
    catch (Exception e)
    {
        System.out.println("ERROR "+e);
        ExceptionManager.ManageException(e);
        JOptionPane.showMessageDialog(null, "ERROR al enviar el mensaje",
"R.A.I.", JOptionPane.PLAIN_MESSAGE);
    }

}
  private MimeMessage getMessage(Session session, String from, String to)
{
    try{
        MimeMessage msg = new MimeMessage(session);
        msg.setText(this.mensajeSMS);
        msg.addRecipient(Message.RecipientType.TO, new
InternetAddress(to));
        msg.setFrom(new InternetAddress(from,"R.A.I."));
        msg.setSubject("NotificaciÃ³n de Alarma");
        return msg;
    }
    catch (java.io.UnsupportedEncodingException ex)
    {
        ExceptionManager.ManageException(ex);
        return null;
    }

    catch (MessagingException ex)
    {
        ExceptionManager.ManageException(ex);
        return null;
    }
}

}



package raicelular;
import javax.mail.*;
public class SMTPAuthentication extends javax.mail.Authenticator{
    private String username;
      private String password;
```

```
    public SMTPAuthentication(String username, String password) {
            super();
            this.username = username;
            this.password = password;     }
        @Override
    public PasswordAuthentication getPasswordAuthentication() {
   return new PasswordAuthentication(username, password);
     }

}



package raicelular;
import gnu.io.CommPortIdentifier;
import gnu.io.SerialPort;
import gnu.io.SerialPortEvent;
import gnu.io.SerialPortEventListener;
import java.io.IOException;
import java.io.InputStream;
import java.io.OutputStream;
import java.util.ArrayList;
import java.util.Collections;
import java.util.Enumeration;
import java.util.List;
import javax.swing.DefaultListModel;
import javax.swing.JOptionPane;

public class JFMenu extends javax.swing.JFrame implements Runnable,
SerialPortEventListener {
   public ClaseSMS smsclass =new ClaseSMS();
   public SendAuthentication mailClass = new SendAuthentication();
    JDconfiguracionCircuito form_config_circuito=new
    JDconfiguracionCircuito( this, true);
    JDconfiguracionModem form_config_modem=new
    JDconfiguracionModem(this,true);
    public static boolean estaEnSincroniacion=false;
    public int[] puertos=new int[9];
    public int contador=0;
    DefaultListModel listmodel =new DefaultListModel();
    private Enumeration listaDePuertos;
    public static int consulta_bits=0;  //0=apagado     1=prendido
    consulta el puerto n para saber si esta prendido o apagado
    public Thread threadDeLectura;
    private List list;

    static SerialPort puertoSerialCircuito;
    static SerialPort puertoSerialModem;
    static OutputStream flujoSalidaCircuito;
    static OutputStream flujoSalidaModem;
    static CommPortIdentifier portIdCircuito;
    static CommPortIdentifier portIdModem;
    static InputStream flujoEntradaCircuito;
    static InputStream flujoEntradaModem;

    public int numPuertoCircuito=0;
    public int numPuertoModem=0;
```

```
    public int databitsCircuito=3;
    public int databitsModem=3;
    public int paridadCircuito=2;
    public int paridadModem=2;
    public int velocidadCircuito=11;
    public int velocidadModem=11;
    public int stopbitsCircuito=0;
    public int stopbitsModem=0;
    public int controlFlujoCircuito=0;
    public int controlFlujoModem=0;
    public String puertoCircuito="";
    public String puertoModem="";

public JFMenu() {   //constructor
initComponents();
this.carga_puertos();
deshabilita_checkbox_Puertos();

}
    @SuppressWarnings("unchecked")
    private void initComponents() {
        buttonGroupCompanias = new javax.swing.ButtonGroup();
        jPanelPuertos = new javax.swing.JPanel();
        jPanel1 = new javax.swing.JPanel();
        jButtonConfigPuertos = new javax.swing.JButton();
        jCheckBoxControlSMS = new javax.swing.JCheckBox();
        jCheckBoxConAlarmasSMS = new javax.swing.JCheckBox();
        jLabel1 = new javax.swing.JLabel();
        jPanel3 = new javax.swing.JPanel();
        jCheckBoxPort1 = new javax.swing.JCheckBox();
        jCheckBoxPort2 = new javax.swing.JCheckBox();
        jCheckBoxPort3 = new javax.swing.JCheckBox();
        jCheckBoxPort4 = new javax.swing.JCheckBox();
        jCheckBoxPort5 = new javax.swing.JCheckBox();
        jCheckBoxPort6 = new javax.swing.JCheckBox();
        jCheckBoxPort7 = new javax.swing.JCheckBox();
        jCheckBoxPort8 = new javax.swing.JCheckBox();
        jButtonConsulta = new javax.swing.JButton();
        jPanelPuertos1 = new javax.swing.JPanel();
        jPanel2 = new javax.swing.JPanel();
        jLabel2 = new javax.swing.JLabel();
        jProgressBarSeÃ±al = new javax.swing.JProgressBar();
        jButtonConfigModem = new javax.swing.JButton();
        jLabel4 = new javax.swing.JLabel();
        jPanel4 = new javax.swing.JPanel();
        jLabel3 = new javax.swing.JLabel();
        jButton1 = new javax.swing.JButton();
        jCheckBoxConMails = new javax.swing.JCheckBox();
        jTextFieldDestinatarioMail = new javax.swing.JTextField();
        jLabel5 = new javax.swing.JLabel();

setDefaultCloseOperation(javax.swing.WindowConstants.EXIT_ON_CLOSE);
setTitle("RAI    -OZIEL-");
        setBackground(new java.awt.Color(51, 204, 0));
        setResizable(false);
jPanelPuertos.setBorder(javax.swing.BorderFactory.createTitledBorder(java
x.swing.BorderFactory.createEtchedBorder(null, java.awt.Color.blue),
```

```
"CIRCUITO ELECTRÃ‘NICO",
javax.swing.border.TitledBorder.DEFAULT_JUSTIFICATION,
javax.swing.border.TitledBorder.DEFAULT_POSITION, new
java.awt.Font("Tahoma", 1, 14), new java.awt.Color(0, 0, 153))); //
NOI18N
jPanel1.setBorder(javax.swing.BorderFactory.createEtchedBorder());

        jButtonConfigPuertos.setFont(new java.awt.Font("Tahoma", 1, 12));
        jButtonConfigPuertos.setText("CONFIGURAR");
        jButtonConfigPuertos.addActionListener(new
java.awt.event.ActionListener() {
            public void actionPerformed(java.awt.event.ActionEvent evt) {
                jButtonConfigPuertosActionPerformed(evt);
            }
        });

        jCheckBoxControlSMS.setText("Control SMS");
        jCheckBoxControlSMS.addActionListener(new
java.awt.event.ActionListener() {
            public void actionPerformed(java.awt.event.ActionEvent evt) {
                jCheckBoxControlSMSActionPerformed(evt);
            }
        });

        jCheckBoxConAlarmasSMS.setText("Enviar alarmas SMS");
        jCheckBoxConAlarmasSMS.addActionListener(new
java.awt.event.ActionListener() {
            public void actionPerformed(java.awt.event.ActionEvent evt) {
                jCheckBoxConAlarmasSMSActionPerformed(evt);
            }
        });

        jLabel1.setIcon(new
javax.swing.ImageIcon(getClass().getResource("/images/circuito1.gif")));
// NOI18N

        javax.swing.GroupLayout jPanel1Layout = new
javax.swing.GroupLayout(jPanel1);
        jPanel1.setLayout(jPanel1Layout);
        jPanel1Layout.setHorizontalGroup(

jPanel1Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADI
NG)
            .addGroup(javax.swing.GroupLayout.Alignment.TRAILING,
jPanel1Layout.createSequentialGroup()
                .addContainerGap()

.addGroup(jPanel1Layout.createParallelGroup(javax.swing.GroupLayout.Align
ment.TRAILING)
                    .addGroup(jPanel1Layout.createSequentialGroup()
                        .addComponent(jButtonConfigPuertos,
javax.swing.GroupLayout.PREFERRED_SIZE, 120,
javax.swing.GroupLayout.PREFERRED_SIZE)
                        .addGap(8, 8, 8))

.addGroup(jPanel1Layout.createParallelGroup(javax.swing.GroupLayout.Align
ment.LEADING)
```

```
                        .addComponent(jCheckBoxControlSMS)
                        .addComponent(jCheckBoxConAlarmasSMS)))

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)
                .addComponent(jLabel1,
javax.swing.GroupLayout.PREFERRED_SIZE, 133,
javax.swing.GroupLayout.PREFERRED_SIZE)
                .addContainerGap())
        );
        jPanel1Layout.setVerticalGroup(

jPanel1Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADI
NG)
            .addGroup(javax.swing.GroupLayout.Alignment.TRAILING,
jPanel1Layout.createSequentialGroup()
                .addContainerGap()

.addGroup(jPanel1Layout.createParallelGroup(javax.swing.GroupLayout.Align
ment.TRAILING)
                    .addComponent(jLabel1,
javax.swing.GroupLayout.PREFERRED_SIZE, 119, Short.MAX_VALUE)
                    .addGroup(jPanel1Layout.createSequentialGroup()
                        .addComponent(jButtonConfigPuertos,
javax.swing.GroupLayout.PREFERRED_SIZE, 38,
javax.swing.GroupLayout.PREFERRED_SIZE)
                        .addGap(18, 18, 18)
                        .addComponent(jCheckBoxControlSMS)

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED, 6,
Short.MAX_VALUE)
                        .addComponent(jCheckBoxConAlarmasSMS)
                        .addGap(11, 11, 11)))
                .addContainerGap())
        );


jPanel3.setBorder(javax.swing.BorderFactory.createTitledBorder(javax.swin
g.BorderFactory.createEtchedBorder(), "Estado"));

        jCheckBoxPort1.setText("Puerto 1");
        jCheckBoxPort1.addActionListener(new
java.awt.event.ActionListener() {
            public void actionPerformed(java.awt.event.ActionEvent evt) {
                jCheckBoxPort1ActionPerformed(evt);
            }
        });

        jCheckBoxPort2.setText("Puerto 2");
        jCheckBoxPort2.addActionListener(new
java.awt.event.ActionListener() {
            public void actionPerformed(java.awt.event.ActionEvent evt) {
                jCheckBoxPort2ActionPerformed(evt);
            }
        });

        jCheckBoxPort3.setText("Puerto 3");
```

```
        jCheckBoxPort3.addActionListener(new
java.awt.event.ActionListener() {
            public void actionPerformed(java.awt.event.ActionEvent evt) {
                jCheckBoxPort3ActionPerformed(evt);
            }
        });

        jCheckBoxPort4.setText("Puerto 4");
        jCheckBoxPort4.addActionListener(new
java.awt.event.ActionListener() {
            public void actionPerformed(java.awt.event.ActionEvent evt) {
                jCheckBoxPort4ActionPerformed(evt);
            }
        });

        jCheckBoxPort5.setText("Puerto 5");
        jCheckBoxPort5.addActionListener(new
java.awt.event.ActionListener() {
            public void actionPerformed(java.awt.event.ActionEvent evt) {
                jCheckBoxPort5ActionPerformed(evt);
            }
        });

        jCheckBoxPort6.setText("Puerto 6");
        jCheckBoxPort6.addActionListener(new
java.awt.event.ActionListener() {
            public void actionPerformed(java.awt.event.ActionEvent evt) {
                jCheckBoxPort6ActionPerformed(evt);
            }
        });

        jCheckBoxPort7.setText("Puerto 7");
        jCheckBoxPort7.addActionListener(new
java.awt.event.ActionListener() {
            public void actionPerformed(java.awt.event.ActionEvent evt) {
                jCheckBoxPort7ActionPerformed(evt);
            }
        });

        jCheckBoxPort8.setText("Puerto 8");
        jCheckBoxPort8.addActionListener(new
java.awt.event.ActionListener() {
            public void actionPerformed(java.awt.event.ActionEvent evt) {
                jCheckBoxPort8ActionPerformed(evt);
            }
        });

        jButtonConsulta.setFont(new java.awt.Font("Tahoma", 1, 12));
        jButtonConsulta.setText("CONSULTA PUERTOS");
        jButtonConsulta.addActionListener(new
java.awt.event.ActionListener() {
            public void actionPerformed(java.awt.event.ActionEvent evt) {
                jButtonConsultaActionPerformed(evt);
            }
        });
```

```
        javax.swing.GroupLayout jPanel3Layout = new
javax.swing.GroupLayout(jPanel3);
        jPanel3.setLayout(jPanel3Layout);
        jPanel3Layout.setHorizontalGroup(

jPanel3Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADI
NG)
            .addGroup(jPanel3Layout.createSequentialGroup()
                .addComponent(jCheckBoxPort1)

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)
                .addComponent(jCheckBoxPort2)

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)
                .addComponent(jCheckBoxPort3)

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)
                .addComponent(jCheckBoxPort4))
            .addGroup(jPanel3Layout.createSequentialGroup()
                .addComponent(jCheckBoxPort5)

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)
                .addComponent(jCheckBoxPort6)

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)
                .addComponent(jCheckBoxPort7)

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.UNRELATED)
                .addComponent(jCheckBoxPort8))
            .addGroup(jPanel3Layout.createSequentialGroup()
                .addGap(53, 53, 53)
                .addComponent(jButtonConsulta))
        );

jPanel3Layout.linkSize(javax.swing.SwingConstants.HORIZONTAL, new
java.awt.Component[] {jCheckBoxPort1, jCheckBoxPort2, jCheckBoxPort3,
jCheckBoxPort4, jCheckBoxPort5, jCheckBoxPort6, jCheckBoxPort7,
jCheckBoxPort8});

        jPanel3Layout.setVerticalGroup(

jPanel3Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADI
NG)
            .addGroup(jPanel3Layout.createSequentialGroup()
                .addContainerGap(javax.swing.GroupLayout.DEFAULT_SIZE,
Short.MAX_VALUE)

.addGroup(jPanel3Layout.createParallelGroup(javax.swing.GroupLayout.Align
ment.BASELINE)
                    .addComponent(jCheckBoxPort1)
                    .addComponent(jCheckBoxPort2)
                    .addComponent(jCheckBoxPort3)
                    .addComponent(jCheckBoxPort4))

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.UNRELATED)
```

```
.addGroup(jPanel3Layout.createParallelGroup(javax.swing.GroupLayout.Align
ment.BASELINE)
                    .addComponent(jCheckBoxPort5)
                    .addComponent(jCheckBoxPort6)
                    .addComponent(jCheckBoxPort7)
                    .addComponent(jCheckBoxPort8))
addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.UNRELATED)
                .addComponent(jButtonConsulta,
javax.swing.GroupLayout.PREFERRED_SIZE, 33,
javax.swing.GroupLayout.PREFERRED_SIZE)
                .addContainerGap())       );
        jPanel3Layout.linkSize(javax.swing.SwingConstants.VERTICAL, new
java.awt.Component[] {jCheckBoxPort1, jCheckBoxPort2, jCheckBoxPort3,
jCheckBoxPort4, jCheckBoxPort5, jCheckBoxPort6, jCheckBoxPort7,
jCheckBoxPort8});
        javax.swing.GroupLayout jPanelPuertosLayout = new
javax.swing.GroupLayout(jPanelPuertos);
        jPanelPuertos.setLayout(jPanelPuertosLayout);
        jPanelPuertosLayout.setHorizontalGroup(

jPanelPuertosLayout.createParallelGroup(javax.swing.GroupLayout.Alignment
.LEADING)
            .addGroup(jPanelPuertosLayout.createSequentialGroup()
                .addContainerGap()
.addGroup(jPanelPuertosLayout.createParallelGroup(javax.swing.GroupLayout
.Alignment.LEADING)
.addComponent(jPanel3, javax.swing.GroupLayout.PREFERRED_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE)
                    .addComponent(jPanel1,
javax.swing.GroupLayout.PREFERRED_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE))
                .addContainerGap(javax.swing.GroupLayout.DEFAULT_SIZE,
Short.MAX_VALUE))
        );
        jPanelPuertosLayout.setVerticalGroup(

jPanelPuertosLayout.createParallelGroup(javax.swing.GroupLayout.Alignment
.LEADING)
            .addGroup(jPanelPuertosLayout.createSequentialGroup()
                .addComponent(jPanel1,
javax.swing.GroupLayout.PREFERRED_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE)

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)
                .addComponent(jPanel3,
javax.swing.GroupLayout.DEFAULT_SIZE, 132, Short.MAX_VALUE)
                .addContainerGap())
        );


jPanelPuertos1.setBorder(javax.swing.BorderFactory.createTitledBorder(jav
ax.swing.BorderFactory.createEtchedBorder(null, java.awt.Color.blue),
"MODEM GSM", javax.swing.border.TitledBorder.DEFAULT_JUSTIFICATION,
```

```
javax.swing.border.TitledBorder.DEFAULT_POSITION, new
java.awt.Font("Tahoma", 1, 14), new java.awt.Color(0, 0, 153))); //
NOI18N


jPanel2.setBorder(javax.swing.BorderFactory.createEtchedBorder());

        jLabel2.setIcon(new
javax.swing.ImageIcon(getClass().getResource("/images/ModemGSM2.jpg")));
// NOI18N

        jProgressBarSeÃ±al.setBackground(new java.awt.Color(204, 204,
204));
        jProgressBarSeÃ±al.setFont(new java.awt.Font("Tahoma", 1, 12));
        jProgressBarSeÃ±al.setForeground(new java.awt.Color(0, 153, 0));
        jProgressBarSeÃ±al.setToolTipText("Intensidad de seÃ±al GSM");
        jProgressBarSeÃ±al.setStringPainted(true);

        jButtonConfigModem.setFont(new java.awt.Font("Tahoma", 1, 12));
        jButtonConfigModem.setText("CONFIGURAR");
        jButtonConfigModem.addActionListener(new
java.awt.event.ActionListener() {
            public void actionPerformed(java.awt.event.ActionEvent evt) {
                jButtonConfigModemActionPerformed(evt);
            }
        });

        jLabel4.setText("SeÃ±al GSM");

        javax.swing.GroupLayout jPanel2Layout = new
javax.swing.GroupLayout(jPanel2);
        jPanel2.setLayout(jPanel2Layout);
        jPanel2Layout.setHorizontalGroup(

jPanel2Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADI
NG)
            .addGroup(jPanel2Layout.createSequentialGroup()
                .addContainerGap()

.addGroup(jPanel2Layout.createParallelGroup(javax.swing.GroupLayout.Align
ment.LEADING)
                    .addGroup(jPanel2Layout.createSequentialGroup()

.addGroup(jPanel2Layout.createParallelGroup(javax.swing.GroupLayout.Align
ment.LEADING)
                            .addComponent(jButtonConfigModem,
javax.swing.GroupLayout.DEFAULT_SIZE, 120, Short.MAX_VALUE)
                            .addComponent(jProgressBarSeÃ±al, 0, 0,
Short.MAX_VALUE))

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED))
                    .addGroup(javax.swing.GroupLayout.Alignment.TRAILING,
jPanel2Layout.createSequentialGroup()
                        .addComponent(jLabel4)
                        .addGap(37, 37, 37)))
```

```
                .addComponent(jLabel2,
javax.swing.GroupLayout.PREFERRED_SIZE, 100,
javax.swing.GroupLayout.PREFERRED_SIZE)
                .addContainerGap())
        );
        jPanel2Layout.setVerticalGroup(

jPanel2Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADI
NG)
            .addGroup(jPanel2Layout.createSequentialGroup()
                .addContainerGap()

.addGroup(jPanel2Layout.createParallelGroup(javax.swing.GroupLayout.Align
ment.LEADING)
                    .addComponent(jLabel2)
                    .addGroup(javax.swing.GroupLayout.Alignment.TRAILING,
jPanel2Layout.createSequentialGroup()
                        .addComponent(jButtonConfigModem,
javax.swing.GroupLayout.DEFAULT_SIZE, 42, Short.MAX_VALUE)

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.UNRELATED)
                        .addComponent(jLabel4)
                        .addGap(1, 1, 1)
                        .addComponent(jProgressBarSeÃ±al,
javax.swing.GroupLayout.PREFERRED_SIZE, 30,
javax.swing.GroupLayout.PREFERRED_SIZE)))
                .addContainerGap())
        );

        javax.swing.GroupLayout jPanelPuertos1Layout = new
javax.swing.GroupLayout(jPanelPuertos1);
        jPanelPuertos1.setLayout(jPanelPuertos1Layout);
        jPanelPuertos1Layout.setHorizontalGroup(

jPanelPuertos1Layout.createParallelGroup(javax.swing.GroupLayout.Alignmen
t.LEADING)
            .addGroup(jPanelPuertos1Layout.createSequentialGroup()
                .addContainerGap()
                .addComponent(jPanel2,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE)
                .addContainerGap())
        );
        jPanelPuertos1Layout.setVerticalGroup(

jPanelPuertos1Layout.createParallelGroup(javax.swing.GroupLayout.Alignmen
t.LEADING)
            .addGroup(jPanelPuertos1Layout.createSequentialGroup()
                .addComponent(jPanel2,
javax.swing.GroupLayout.PREFERRED_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE)
                .addContainerGap())
        );


jPanel4.setBorder(javax.swing.BorderFactory.createTitledBorder(javax.swin
```

```
g.BorderFactory.createEtchedBorder(null, java.awt.Color.blue), "",
javax.swing.border.TitledBorder.DEFAULT_JUSTIFICATION,
javax.swing.border.TitledBorder.DEFAULT_POSITION, new
java.awt.Font("Tahoma", 0, 11), new java.awt.Color(0, 0, 153))); //
NOI18N

        jLabel3.setIcon(new
javax.swing.ImageIcon(getClass().getResource("/images/logo_colpos.gif")))
; // NOI18N

        jButton1.setFont(new java.awt.Font("Tahoma", 1, 18));
        jButton1.setIcon(new
javax.swing.ImageIcon(getClass().getResource("/images/Salir.png"))); //
NOI18N
        jButton1.addActionListener(new java.awt.event.ActionListener() {
            public void actionPerformed(java.awt.event.ActionEvent evt) {
                jButton1ActionPerformed(evt);
            }
        });

        jCheckBoxConMails.setText("Enviar alarmas MAIL");
        jCheckBoxConMails.addActionListener(new
java.awt.event.ActionListener() {
            public void actionPerformed(java.awt.event.ActionEvent evt) {
                jCheckBoxConMailsActionPerformed(evt);
            }
        });

        jTextFieldDestinatarioMail.setText("oziel@colpos.mx");
        jTextFieldDestinatarioMail.setEnabled(false);

        jLabel5.setText("Destinatario:");

        javax.swing.GroupLayout jPanel4Layout = new
javax.swing.GroupLayout(jPanel4);
        jPanel4.setLayout(jPanel4Layout);
        jPanel4Layout.setHorizontalGroup(

jPanel4Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADI
NG)
            .addGroup(jPanel4Layout.createSequentialGroup()

.addGroup(jPanel4Layout.createParallelGroup(javax.swing.GroupLayout.Align
ment.LEADING)
                    .addGroup(jPanel4Layout.createSequentialGroup()
                        .addContainerGap(135, Short.MAX_VALUE)
                        .addComponent(jCheckBoxConMails))
                    .addGroup(jPanel4Layout.createSequentialGroup()
                        .addComponent(jLabel3)

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.UNRELATED)

.addGroup(jPanel4Layout.createParallelGroup(javax.swing.GroupLayout.Align
ment.LEADING)
                            .addComponent(jLabel5)
                            .addComponent(jTextFieldDestinatarioMail,
javax.swing.GroupLayout.DEFAULT_SIZE, 173, Short.MAX_VALUE)))
```

```
                    .addGroup(javax.swing.GroupLayout.Alignment.TRAILING,
jPanel4Layout.createSequentialGroup()
                        .addContainerGap(195, Short.MAX_VALUE)
                        .addComponent(jButton1,
javax.swing.GroupLayout.PREFERRED_SIZE, 63,
javax.swing.GroupLayout.PREFERRED_SIZE)))
                .addContainerGap())
        );
        jPanel4Layout.setVerticalGroup(

jPanel4Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADI
NG)
            .addGroup(jPanel4Layout.createSequentialGroup()

.addGroup(jPanel4Layout.createParallelGroup(javax.swing.GroupLayout.Align
ment.LEADING, false)
                    .addGroup(jPanel4Layout.createSequentialGroup()
                        .addComponent(jCheckBoxConMails)
                        .addGap(4, 4, 4)
                        .addComponent(jLabel5)

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED,
javax.swing.GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE)
                        .addComponent(jTextFieldDestinatarioMail,
javax.swing.GroupLayout.PREFERRED_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE))
                    .addComponent(jLabel3))

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED,
javax.swing.GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE)
                .addComponent(jButton1)
                .addContainerGap())
        );

        javax.swing.GroupLayout layout = new
javax.swing.GroupLayout(getContentPane());
        getContentPane().setLayout(layout);
        layout.setHorizontalGroup(

layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
            .addGroup(layout.createSequentialGroup()
                .addContainerGap()
                .addComponent(jPanelPuertos,
javax.swing.GroupLayout.PREFERRED_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE)

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LE
ADING)
                    .addComponent(jPanelPuertos1,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE)
```

```
                    .addComponent(jPanel4,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE))
                .addContainerGap())
        );
        layout.setVerticalGroup(

layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
            .addGroup(layout.createSequentialGroup()
                .addContainerGap()

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LE
ADING)
                    .addComponent(jPanelPuertos,
javax.swing.GroupLayout.PREFERRED_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE)
                    .addGroup(layout.createSequentialGroup()
                        .addComponent(jPanelPuertos1,
javax.swing.GroupLayout.PREFERRED_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE)

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)
                        .addComponent(jPanel4, 0,
javax.swing.GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE)))
                .addContainerGap(javax.swing.GroupLayout.DEFAULT_SIZE,
Short.MAX_VALUE))
        );
    }


public void deshabilita_checkbox_Puertos(){
    this.jCheckBoxConAlarmasSMS.setEnabled(false);
    this.jCheckBoxControlSMS.setEnabled(false);
    this.jCheckBoxConMails.setEnabled(false);
    this.jCheckBoxPort1.setEnabled(false);
    this.jCheckBoxPort2.setEnabled(false);
    this.jCheckBoxPort3.setEnabled(false);
    this.jCheckBoxPort4.setEnabled(false);
    this.jCheckBoxPort5.setEnabled(false);
    this.jCheckBoxPort6.setEnabled(false);
    this.jCheckBoxPort7.setEnabled(false);
    this.jCheckBoxPort8.setEnabled(false);
    this.jButtonConsulta.setEnabled(false);
    this.jProgressBarSeÃ±al.setEnabled(false);
}

public void habilita_checkbox_Puertos(){
    this.jCheckBoxPort1.setEnabled(true);
    this.jCheckBoxPort2.setEnabled(true);
    this.jCheckBoxPort3.setEnabled(true);
    this.jCheckBoxPort4.setEnabled(true);
    this.jCheckBoxPort5.setEnabled(true);
    this.jCheckBoxPort6.setEnabled(true);
    this.jCheckBoxPort7.setEnabled(true);
    this.jCheckBoxPort8.setEnabled(true);
```

```
    jButtonConsulta.setEnabled(true);
    this.jCheckBoxConMails.setEnabled(true);
}


    private void jButton1ActionPerformed(java.awt.event.ActionEvent evt)
{
    int opcion=JOptionPane.showConfirmDialog(this,"Â¿Estas seguro de
querer salir del programa?", "R.A.I. System" ,
JOptionPane.YES_NO_OPTION);
       if (opcion==0) //si la respuesta es YES
       {
          System.exit(0);
       }
    }


    private void
jButtonConfigPuertosActionPerformed(java.awt.event.ActionEvent evt) {
       this.form_config_circuito.setLocationRelativeTo(null);
       this.form_config_circuito.setVisible(true);

    }//GEN-LAST:event_jButtonConfigPuertosActionPerformed

    private void
jButtonConfigModemActionPerformed(java.awt.event.ActionEvent evt) {//GEN-
FIRST:event_jButtonConfigModemActionPerformed
      this.form_config_modem.setLocationRelativeTo(null);
      this.form_config_modem.setVisible(true);
    }


private static void pausa(){
try{
    Thread.sleep(250);
   }
catch (InterruptedException e)
{
    System.out.println("ERROR en pausa(): "+e);
}
}
private void revisa_acciones(int codigo){
String estado="";
int puerto=0;

    switch (codigo){
            case 0: estado="APAGADO"; puerto=1; break;
            case 64: estado="ENCENDIDO"; puerto=1; break;
            case 1: estado="APAGADO"; puerto=2; break;
            case 65: estado="ENCENDIDO"; puerto=2; break;
            case 2: estado="APAGADO"; puerto=3; break;
            case 66: estado="ENCENDIDO"; puerto=4; break;
            case 3: estado="APAGADO"; puerto=4; break;
            case 67: estado="ENCENDIDO"; puerto=4; break;
            case 4: estado="APAGADO"; puerto=5; break;
            case 68: estado="ENCENDIDO"; puerto=5; break;
            case 5: estado="APAGADO"; puerto=6; break;
```

```
            case 69: estado="ENCENDIDO"; puerto=6; break;
            case 6: estado="APAGADO"; puerto=7; break;
            case 70: estado="ENCENDIDO"; puerto=7; break;
            case 7: estado="APAGADO"; puerto=8; break;
            case 71: estado="ENCENDIDO"; puerto=8; break;
            case 128: estado="APAGADO"; puerto=1; break;
            case 129: estado="ENCENDIDO"; puerto=2; break;
            case 130: estado="APAGADO"; puerto=3; break;
            case 131: estado="ENCENDIDO"; puerto=4; break;
            case 132: estado="APAGADO"; puerto=5; break;
            case 133: estado="ENCENDIDO"; puerto=6; break;
            case 134: estado="APAGADO"; puerto=7; break;
            case 135: estado="ENCENDIDO"; puerto=8; break;
    }
if(codigo<100){   //SI ES ACCION-PUERTOS
        if(Main.quiereSMS) smsclass.EnviaSMS(Main.destinatarioSMS, "OZIEL
VENTANA PUERTO "+puerto+" " +estado);
        if (Main.quiereMails)
mailClass.Send(this.jTextFieldDestinatarioMail.getText(), "PUERTO
"+puerto+" " +estado);
    }//fin del if principal

else{  //SI ES CONSULTA
        if(Main.quiereSMS){
               if (Main.resultadoConsulta==0){
                     smsclass.EnviaSMS(Main.destinatarioSMS, "OZIEL
VENTANA PUERTO "+puerto+" APAGADO");
                   }else{
                    smsclass.EnviaSMS(Main.destinatarioSMS, "OZIEL
VENTANA PUERTO "+puerto+" ENCENDIDO");
                   }
           }//fin del if quiere SMS
        if (Main.quiereMails){
                   if (Main.resultadoConsulta==0){

mailClass.Send(this.jTextFieldDestinatarioMail.getText(), "PUERTO
"+puerto+" APAGADO");
                   }else{

mailClass.Send(this.jTextFieldDestinatarioMail.getText(), "PUERTO
"+puerto+" ENCENDIDO");
                   }
        }//fin del quiereMails
 }//fin  del else principal


}
    private void jCheckBoxPort1ActionPerformed(java.awt.event.ActionEvent
evt) {
        if (this.jCheckBoxPort1.isSelected())
          enciende(64);
       else
            enciende(0);
    }

    private void jCheckBoxPort2ActionPerformed(java.awt.event.ActionEvent
evt) {
```

```
       if (this.jCheckBoxPort2.isSelected())
            enciende(65);
       else
         enciende(1);
    }

    private void jCheckBoxPort3ActionPerformed(java.awt.event.ActionEvent
evt) {
        if (this.jCheckBoxPort3.isSelected())
            enciende(66);
       else
         enciende(2);
    }

    private void jCheckBoxPort4ActionPerformed(java.awt.event.ActionEvent
evt) {
     if (this.jCheckBoxPort4.isSelected())
            enciende(67);
       else
         enciende(3);
    }

    private void jCheckBoxPort5ActionPerformed(java.awt.event.ActionEvent
evt) {
      if (this.jCheckBoxPort5.isSelected())
            enciende(68);
       else
         enciende(4);
    }

    private void jCheckBoxPort6ActionPerformed(java.awt.event.ActionEvent
evt) {
      if (this.jCheckBoxPort6.isSelected())
            enciende(69);
       else
         enciende(5);
}

    private void jCheckBoxPort7ActionPerformed(java.awt.event.ActionEvent
evt) {
          if (this.jCheckBoxPort7.isSelected())
            enciende(70);
       else
         enciende(6);
}

    private void jCheckBoxPort8ActionPerformed(java.awt.event.ActionEvent
evt) {
             if (this.jCheckBoxPort8.isSelected())
            enciende(71);
       else
         enciende(7);
}

    private void
jButtonConsultaActionPerformed(java.awt.event.ActionEvent evt) {
   try{
```

```
int el_bit=Integer.parseInt(JOptionPane.showInputDialog(this, "Num de
Puerto a consultar", "CONSULTA PUERTOS", JOptionPane.PLAIN_MESSAGE));

if(el_bit>0 & el_bit<9){
int bits_consulta=el_bit+127;
enciende(bits_consulta);
pausa();
}
else{
JOptionPane.showMessageDialog(this, "Debes ingresar un numero de puerto
vÃ¡lido (1-8)", "¿Sabes lo que estas haciendo?",
JOptionPane.INFORMATION_MESSAGE);
}
}
catch(NumberFormatException ex){
    System.out.println("Error de formato de numero");
}

if (Main.resultadoConsulta==0) {
   JOptionPane.showMessageDialog(this, "PUERTO APAGADO", "CONSULTA
PUERTOS", JOptionPane.INFORMATION_MESSAGE);

}
    else
    {
       JOptionPane.showMessageDialog(this, "PUERTO ENCENDIDO", "CONSULTA
PUERTOS", JOptionPane.INFORMATION_MESSAGE);
    }
    }//GEN-LAST:event_jButtonConsultaActionPerformed

    private void
jCheckBoxConMailsActionPerformed(java.awt.event.ActionEvent evt) {

        if (this.jCheckBoxConMails.isSelected())
       {
          Main.quiereMails=true;
          this.jTextFieldDestinatarioMail.setEnabled(true);
       }
       else
       {
           Main.quiereMails=false;
            this.jTextFieldDestinatarioMail.setEnabled(false);
       }
}

    private void
jCheckBoxControlSMSActionPerformed(java.awt.event.ActionEvent evt) {
          if (this.jCheckBoxControlSMS.isSelected())
       {
          Main.quiereControlSMS=true;
       }
       else
       {
           Main.quiereControlSMS=false;
       }
}
```

```
    private void
jCheckBoxConAlarmasSMSActionPerformed(java.awt.event.ActionEvent evt) {
          if (this.jCheckBoxConAlarmasSMS.isSelected())
       {
          Main.quiereSMS=true;
                }
       else
       {
           Main.quiereSMS=false;
       }
}



    public void enciende(int bites){

      try{
          JFMenu.flujoSalidaCircuito.write(bites);
          pausa();
          JFMenu.flujoSalidaCircuito.flush();
          revisa_acciones(bites);
      }
      catch (IOException e)
                      {
                        System.out.println("Error al encender los
puertos" +e);
                        e.printStackTrace();
                      }
  }


public void sincroniza_Puertos(){
    this.contador=0;
    enciende(128); // <---------------------------------PUERTO 1
    pausa();
    enciende(129); // <---------------------------------PUERTO 2
    pausa();
    enciende(130); // <---------------------------------PUERTO 3
    pausa();
    enciende(131); // <---------------------------------PUERTO 4
    pausa();
    enciende(132); // <---------------------------------PUERTO 5
    pausa();
    enciende(133); // <---------------------------------PUERTO 6
    pausa();
    enciende(134); // <---------------------------------PUERTO 7
    pausa();
    enciende(135); // <---------------------------------PUERTO 8
    pausa();
   }


public void carga_puertos() {
list = new ArrayList();
listmodel.removeAllElements();
form_config_circuito.jList_puertos.removeAll();
form_config_modem.jList_puertos.removeAll();
```

```
listaDePuertos = CommPortIdentifier.getPortIdentifiers();

        while (listaDePuertos.hasMoreElements()) {

            portIdCircuito = (CommPortIdentifier)
listaDePuertos.nextElement();
            if (portIdCircuito.getPortType() ==
CommPortIdentifier.PORT_SERIAL) {
                  if(no_se_encuentra_en_lista(portIdCircuito.getName()))
                    list.add(portIdCircuito.getName());


                }
            }

Collections.sort(list);
for (int i=0; i<list.size();i++){
    listmodel.add(i,list.get(i));
}
form_config_circuito.jList_puertos.setModel(listmodel);
form_config_modem.jList_puertos.setModel(listmodel);
}

private boolean no_se_encuentra_en_lista(String puerto){
int tam=list.size();
boolean resultado=true;

for (int i = 0; i < tam; i++) {
if(puerto.equals(list.get(i)) )
resultado=false;
    }
return resultado;
}

private void verifia_checkboxs(){
  if(this.puertos[0]==0) this.jCheckBoxPort1.setSelected(false);
  else this.jCheckBoxPort1.setSelected(true);
    if(this.puertos[1]==0) this.jCheckBoxPort2.setSelected(false);
  else this.jCheckBoxPort2.setSelected(true);
   if(this.puertos[2]==0) this.jCheckBoxPort3.setSelected(false);
  else this.jCheckBoxPort3.setSelected(true);
  if(this.puertos[3]==0) this.jCheckBoxPort4.setSelected(false);
  else this.jCheckBoxPort4.setSelected(true);
   if(this.puertos[4]==0) this.jCheckBoxPort5.setSelected(false);
  else this.jCheckBoxPort5.setSelected(true);
   if(this.puertos[5]==0) this.jCheckBoxPort6.setSelected(false);
  else this.jCheckBoxPort6.setSelected(true);
   if(this.puertos[6]==0) this.jCheckBoxPort7.setSelected(false);
  else this.jCheckBoxPort7.setSelected(true);
   if(this.puertos[7]==0) this.jCheckBoxPort8.setSelected(false);
  else this.jCheckBoxPort8.setSelected(true);
}
    public synchronized void run() {
          try {
            Thread.sleep(175);
        } catch (InterruptedException e)
```

```
        {
            System.out.println("ERROR: "+e);
        }
 }

public synchronized void serialEvent(SerialPortEvent event) {

    switch (event.getEventType()) {
     case SerialPortEvent.BI: break;
     case SerialPortEvent.OE: break;
     case SerialPortEvent.FE: break;
     case SerialPortEvent.PE: break;
     case SerialPortEvent.CD: break;
     case SerialPortEvent.CTS: break;
     case SerialPortEvent.DSR: break;
     case SerialPortEvent.RI: break;
     case SerialPortEvent.OUTPUT_BUFFER_EMPTY: break;
     case SerialPortEvent.DATA_AVAILABLE:
       try {


         while ( JFMenu.flujoEntradaCircuito.available() >0  ) {
           int resultado= flujoEntradaCircuito.read();
            // System.out.println("RESULTADO: "+resultado);
                    if (resultado==85){
                        for (int i = 0; i < 8; i++) {
                            puertos[i]=0;
                        }
                        this.setEnabled(false);
                        this.sincroniza_Puertos();
                    }
                    else{
                            if(this.contador<8){

                      this.puertos[contador]=resultado;
                       this.contador++;
                            }
                            else{
                       Main.resultadoConsulta=resultado;
                           }
                    }

          }
           this.verifia_checkboxs();
           this.setEnabled(true);

       } catch (IOException e) {
                  System.out.println("ERROR EVENTO PUERTO SERIAL: "+e);
                  JFMenu.puertoSerialCircuito.close();

       }
        break;

         default:{
                  System.out.println("ERROR SERIAL: ");
                  JFMenu.puertoSerialCircuito.close();
         }
```

```
    }

    }

    private javax.swing.ButtonGroup buttonGroupCompanias;
    private javax.swing.JButton jButton1;
    private javax.swing.JButton jButtonConfigModem;
    private javax.swing.JButton jButtonConfigPuertos;
    private javax.swing.JButton jButtonConsulta;
    public javax.swing.JCheckBox jCheckBoxConAlarmasSMS;
    public javax.swing.JCheckBox jCheckBoxConMails;
    public javax.swing.JCheckBox jCheckBoxControlSMS;
    public javax.swing.JCheckBox jCheckBoxPort1;
    public javax.swing.JCheckBox jCheckBoxPort2;
    public javax.swing.JCheckBox jCheckBoxPort3;
    public javax.swing.JCheckBox jCheckBoxPort4;
    public javax.swing.JCheckBox jCheckBoxPort5;
    public javax.swing.JCheckBox jCheckBoxPort6;
    public javax.swing.JCheckBox jCheckBoxPort7;
    public javax.swing.JCheckBox jCheckBoxPort8;
    private javax.swing.JLabel jLabel1;
    private javax.swing.JLabel jLabel2;
    private javax.swing.JLabel jLabel3;
    private javax.swing.JLabel jLabel4;
    private javax.swing.JLabel jLabel5;
    private javax.swing.JPanel jPanel1;
    private javax.swing.JPanel jPanel2;
    private javax.swing.JPanel jPanel3;
    private javax.swing.JPanel jPanel4;
    private javax.swing.JPanel jPanelPuertos;
    private javax.swing.JPanel jPanelPuertos1;
    public javax.swing.JProgressBar jProgressBarSeÃ±al;
    private javax.swing.JTextField jTextFieldDestinatarioMail;
}



package raicelular;
public class JDespera extends javax.swing.JDialog {
    public JDespera(java.awt.Frame parent, boolean modal) {
        super(parent, modal);
        initComponents();
        this.setResizable(false);
        this.setLocationRelativeTo(parent);
    }

    private void initComponents() {
        jPanel1 = new javax.swing.JPanel();
        jLabelTexto = new javax.swing.JLabel();
        jLabel2 = new javax.swing.JLabel();

setDefaultCloseOperation(javax.swing.WindowConstants.DO_NOTHING_ON_CLOSE)
;
        setTitle("ESPERE...");
        setBackground(new java.awt.Color(153, 153, 255));
        setCursor(new java.awt.Cursor(java.awt.Cursor.WAIT_CURSOR));
```

```
        setModal(true);
        setResizable(false);
        setUndecorated(true);

        jPanel1.setBackground(new java.awt.Color(153, 153, 255));

        jLabelTexto.setFont(new java.awt.Font("Tahoma", 1, 18)); //
NOI18N

jLabelTexto.setHorizontalAlignment(javax.swing.SwingConstants.CENTER);
        jLabelTexto.setText("CONFIGURANDO PUERTO...");

        jLabel2.setFont(new java.awt.Font("Tahoma", 1, 36)); // NOI18N
        jLabel2.setText("ESPERE");

        javax.swing.GroupLayout jPanel1Layout = new
javax.swing.GroupLayout(jPanel1);
        jPanel1.setLayout(jPanel1Layout);
        jPanel1Layout.setHorizontalGroup(

jPanel1Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADI
NG)
            .addGroup(jPanel1Layout.createSequentialGroup()

.addGroup(jPanel1Layout.createParallelGroup(javax.swing.GroupLayout.Align
ment.LEADING)
                    .addGroup(jPanel1Layout.createSequentialGroup()
                        .addContainerGap()
                        .addComponent(jLabelTexto,
javax.swing.GroupLayout.DEFAULT_SIZE, 487, Short.MAX_VALUE))
                    .addGroup(jPanel1Layout.createSequentialGroup()
                        .addGap(42, 42, 42)
                        .addComponent(jLabel2,
javax.swing.GroupLayout.PREFERRED_SIZE, 248,
javax.swing.GroupLayout.PREFERRED_SIZE)))
                .addContainerGap())
        );
        jPanel1Layout.setVerticalGroup(

jPanel1Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADI
NG)
            .addGroup(javax.swing.GroupLayout.Alignment.TRAILING,
jPanel1Layout.createSequentialGroup()
                .addContainerGap()
                .addComponent(jLabel2,
javax.swing.GroupLayout.PREFERRED_SIZE, 65,
javax.swing.GroupLayout.PREFERRED_SIZE)

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED,
javax.swing.GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE)
                .addComponent(jLabelTexto,
javax.swing.GroupLayout.PREFERRED_SIZE, 65,
javax.swing.GroupLayout.PREFERRED_SIZE)
                .addContainerGap())
        );
```

```
        javax.swing.GroupLayout layout = new
javax.swing.GroupLayout(getContentPane());
        getContentPane().setLayout(layout);
        layout.setHorizontalGroup(

layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
            .addGroup(layout.createSequentialGroup()
                .addContainerGap()
                .addComponent(jPanel1,
javax.swing.GroupLayout.PREFERRED_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE)
                .addContainerGap(javax.swing.GroupLayout.DEFAULT_SIZE,
Short.MAX_VALUE))
        );
        layout.setVerticalGroup(

layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
            .addGroup(layout.createSequentialGroup()
                .addContainerGap()
                .addComponent(jPanel1,
javax.swing.GroupLayout.PREFERRED_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE)
                .addContainerGap(javax.swing.GroupLayout.DEFAULT_SIZE,
Short.MAX_VALUE))
        );

}
    private javax.swing.JLabel jLabel2;
    public javax.swing.JLabel jLabelTexto;
    private javax.swing.JPanel jPanel1;
}
```

```
package raicelular;
import gnu.io.CommPortIdentifier;
import gnu.io.NoSuchPortException;
import gnu.io.PortInUseException;
import gnu.io.SerialPort;
import gnu.io.SerialPortEventListener;
import gnu.io.UnsupportedCommOperationException;
import java.io.IOException;
import java.util.TooManyListenersException;
import javax.swing.JOptionPane;

public class JDconfiguracionModem extends javax.swing.JDialog {

        public JDconfiguracionModem(java.awt.Frame parent, boolean modal)
{
        super(parent, modal);
        initComponents();
        this.jTextFieldDestAlarma.setColumns(10);
    }
```

```
    private void initComponents() {
        buttonGroupTelefonicas = new javax.swing.ButtonGroup();
        jPanel10 = new javax.swing.JPanel();
        jRadioButtonMovistar = new javax.swing.JRadioButton();
        jRadioButtonTelcel = new javax.swing.JRadioButton();
        jPanel11 = new javax.swing.JPanel();
        jTextFieldDestAlarma = new javax.swing.JTextField();
        jPanel7 = new javax.swing.JPanel();
        jScrollPane1 = new javax.swing.JScrollPane();
        jList_puertos = new javax.swing.JList();
        jLabel1 = new javax.swing.JLabel();
        jButton_ok = new javax.swing.JButton();
        jButton_cancelar = new javax.swing.JButton();
setDefaultCloseOperation(javax.swing.WindowConstants.DISPOSE_ON_CLOSE);
setTitle("CONFIGURACIÃ“N MODEM CELULAR");


jPanel10.setBorder(javax.swing.BorderFactory.createTitledBorder("Empresa
TelefÃ³nica Celular"));

        buttonGroupTelefonicas.add(jRadioButtonMovistar);
        jRadioButtonMovistar.setSelected(true);
        jRadioButtonMovistar.setText("Movistar");

        buttonGroupTelefonicas.add(jRadioButtonTelcel);
        jRadioButtonTelcel.setText("Telcel");

        javax.swing.GroupLayout jPanel10Layout = new
javax.swing.GroupLayout(jPanel10);
        jPanel10.setLayout(jPanel10Layout);
        jPanel10Layout.setHorizontalGroup(

jPanel10Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEAD
ING)
            .addGroup(jPanel10Layout.createSequentialGroup()
                .addGap(27, 27, 27)

.addGroup(jPanel10Layout.createParallelGroup(javax.swing.GroupLayout.Alig
nment.LEADING)
                    .addComponent(jRadioButtonTelcel)
                    .addComponent(jRadioButtonMovistar))
                .addContainerGap(41, Short.MAX_VALUE))
        );
        jPanel10Layout.setVerticalGroup(

jPanel10Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEAD
ING)
            .addGroup(jPanel10Layout.createSequentialGroup()
                .addComponent(jRadioButtonMovistar)

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED, 3,
Short.MAX_VALUE)
                .addComponent(jRadioButtonTelcel))
        );
```

```
jPanel11.setBorder(javax.swing.BorderFactory.createTitledBorder("Destinat
ario Alarmas"));

        jTextFieldDestAlarma.setFont(new java.awt.Font("Tahoma", 0, 14));
// NOI18N

jTextFieldDestAlarma.setHorizontalAlignment(javax.swing.JTextField.CENTER
);
        jTextFieldDestAlarma.setText("5537867704");

        javax.swing.GroupLayout jPanel11Layout = new
javax.swing.GroupLayout(jPanel11);
        jPanel11.setLayout(jPanel11Layout);
        jPanel11Layout.setHorizontalGroup(

jPanel11Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEAD
ING)
            .addGroup(jPanel11Layout.createSequentialGroup()
                .addContainerGap()
                .addComponent(jTextFieldDestAlarma,
javax.swing.GroupLayout.DEFAULT_SIZE, 115, Short.MAX_VALUE)
                .addContainerGap())
        );
        jPanel11Layout.setVerticalGroup(

jPanel11Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEAD
ING)
            .addComponent(jTextFieldDestAlarma,
javax.swing.GroupLayout.PREFERRED_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE)
        );


jPanel7.setBorder(javax.swing.BorderFactory.createTitledBorder("Seleccion
a el puerto"));

        jList_puertos.setModel(new javax.swing.AbstractListModel() {
            String[] strings = { "Item 1", "Item 2", "Item 3", "Item 4",
"Item 5" };
            public int getSize() { return strings.length; }
            public Object getElementAt(int i) { return strings[i]; }
        });
        jScrollPane1.setViewportView(jList_puertos);

        javax.swing.GroupLayout jPanel7Layout = new
javax.swing.GroupLayout(jPanel7);
        jPanel7.setLayout(jPanel7Layout);
        jPanel7Layout.setHorizontalGroup(

jPanel7Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADI
NG)
            .addGroup(jPanel7Layout.createSequentialGroup()
                .addContainerGap()
```

```
                .addComponent(jScrollPane1,
javax.swing.GroupLayout.PREFERRED_SIZE, 118,
javax.swing.GroupLayout.PREFERRED_SIZE)
                .addContainerGap(javax.swing.GroupLayout.DEFAULT_SIZE,
Short.MAX_VALUE))
        );
        jPanel7Layout.setVerticalGroup(

jPanel7Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADI
NG)
            .addGroup(jPanel7Layout.createSequentialGroup()
                .addComponent(jScrollPane1,
javax.swing.GroupLayout.DEFAULT_SIZE, 187, Short.MAX_VALUE)
                .addContainerGap())
        );

        jLabel1.setBackground(new java.awt.Color(0, 0, 0));

jLabel1.setHorizontalAlignment(javax.swing.SwingConstants.CENTER);
        jLabel1.setIcon(new
javax.swing.ImageIcon(getClass().getResource("/images/ModemGSM2.jpg")));
// NOI18N
        jLabel1.setOpaque(true);

        jButton_ok.setFont(new java.awt.Font("Tahoma", 1, 24)); // NOI18N
        jButton_ok.setIcon(new
javax.swing.ImageIcon(getClass().getResource("/images/Undo.png"))); //
NOI18N
        jButton_ok.setText("OK");
        jButton_ok.addActionListener(new java.awt.event.ActionListener()
{
            public void actionPerformed(java.awt.event.ActionEvent evt) {
                jButton_okActionPerformed(evt);
            }
        });

        jButton_cancelar.setFont(new java.awt.Font("Tahoma", 1, 12)); //
NOI18N
        jButton_cancelar.setText("CANCELAR");
        jButton_cancelar.addActionListener(new
java.awt.event.ActionListener() {
            public void actionPerformed(java.awt.event.ActionEvent evt) {
                jButton_cancelarActionPerformed(evt);
            }
        });

        javax.swing.GroupLayout layout = new
javax.swing.GroupLayout(getContentPane());
        getContentPane().setLayout(layout);
        layout.setHorizontalGroup(

layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
            .addGroup(layout.createSequentialGroup()
                .addContainerGap()
                .addComponent(jPanel7,
javax.swing.GroupLayout.PREFERRED_SIZE,
```

```
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE)

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.TR
AILING)
                    .addComponent(jButton_cancelar)
                    .addComponent(jLabel1,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE))

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.UNRELATED)

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LE
ADING, false)
                    .addComponent(jPanel10,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE)
                    .addComponent(jPanel11,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE)
                    .addComponent(jButton_ok,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE))
                .addContainerGap())
        );
        layout.setVerticalGroup(

layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
            .addGroup(layout.createSequentialGroup()

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.TR
AILING, false)
                    .addComponent(jPanel7,
javax.swing.GroupLayout.Alignment.LEADING,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE)
                    .addGroup(javax.swing.GroupLayout.Alignment.LEADING,
layout.createSequentialGroup()
                        .addContainerGap()

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LE
ADING)

.addGroup(javax.swing.GroupLayout.Alignment.TRAILING,
layout.createSequentialGroup()
                                .addComponent(jLabel1)

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED,
javax.swing.GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE)
                                .addComponent(jButton_cancelar,
javax.swing.GroupLayout.PREFERRED_SIZE, 38,
javax.swing.GroupLayout.PREFERRED_SIZE))
                            .addGroup(layout.createSequentialGroup()
                                .addComponent(jPanel11,
javax.swing.GroupLayout.PREFERRED_SIZE,
```

```
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE)

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)
                                .addComponent(jPanel10,
javax.swing.GroupLayout.PREFERRED_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE)

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED,
javax.swing.GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE)
                                .addComponent(jButton_ok)))))
                .addContainerGap())
        );
}



    private void jButton_okActionPerformed(java.awt.event.ActionEvent
evt) {//GEN-FIRST:event_jButton_okActionPerformed
if(this.jRadioButtonMovistar.isSelected())
    Main.centro_mensajes="+525512235110";   //<---------------- MOVISTAR
else
    Main.centro_mensajes="+5294100001410";    //<---------------- TELCEL

Main.destinatarioSMS=this.jTextFieldDestAlarma.getText();
Main.miventana.numPuertoModem=this.jList_puertos.getSelectedIndex();
Main.miventana.puertoModem
=this.jList_puertos.getSelectedValue().toString();

this.setVisible(false);

if (Main.miventana.smsclass.iniciaServicio(Main.miventana.puertoModem)){
    Main.miReloj.inicia();
    Main.miventana.jCheckBoxConAlarmasSMS.setEnabled(true);
    Main.miventana.jCheckBoxControlSMS.setEnabled(true);
      }

}//GEN-LAST:event_jButton_okActionPerformed

    private void
jButton_cancelarActionPerformed(java.awt.event.ActionEvent evt) {//GEN-
FIRST:event_jButton_cancelarActionPerformed
this.jList_puertos.setSelectedIndex(Main.miventana.numPuertoModem);
this.setVisible(false);
}//GEN-LAST:event_jButton_cancelarActionPerformed


private void configura_puerto(){
    try {
       JFMenu.portIdModem =
CommPortIdentifier.getPortIdentifier(Main.miventana.puertoModem);
        } catch (NoSuchPortException ex) {
          System.out.println("Puerto no encontrado " +ex);
        }
```

```
    System.out.println("Puerto a configurar: "
+JFMenu.portIdModem.getName());
    if( JFMenu.portIdModem.isCurrentlyOwned()){
    return;
}
    abre_puerto();
    int veloc=9600;
 try {
JFMenu.puertoSerialModem.setFlowControlMode(SerialPort.FLOWCONTROL_NONE);
JFMenu.puertoSerialModem.setSerialPortParams(veloc,SerialPort.DATABITS_8,
SerialPort.STOPBITS_1, SerialPort.PARITY_NONE);
     }

 catch (UnsupportedCommOperationException e) {
            System.out.println("ERROR AL CONFIGURAR EL PUERTO " +e);
            cierra_puerto();
            e.printStackTrace();
     }
}

private void cierra_puerto(){
JFMenu.puertoSerialModem.close();
}

private void abre_puerto(){
 try {

     if (JFMenu.portIdModem.isCurrentlyOwned()){
         JFMenu.puertoSerialModem.close();
         System.out.println("PUERTO YA INICIADO");
     }

     JFMenu.puertoSerialModem = (SerialPort)
JFMenu.portIdModem.open("OzielModem", 2000);

 } catch (PortInUseException e)

                    {
                        JOptionPane.showMessageDialog(this, "El puerto
estÃ¡ en uso por otra aplicaciÃ³n", "!Advertencia!",
JOptionPane.INFORMATION_MESSAGE);
                        System.out.println("ERROR AL ABRIR EL PUERTO
"+e);
                        return;
                    }

     try {
            JFMenu.flujoEntradaModem =
JFMenu.puertoSerialModem.getInputStream();
    } catch (IOException e) {
        System.out.println("ERROR AL OBTENER FLUJO DE ENTRADA DEL PUERTO
SERIAL - Modem "+e );
    }

       try
                  {
```

```
                    JFMenu.flujoSalidaModem =
JFMenu.puertoSerialModem.getOutputStream();
                }
             catch (IOException e)
                {
                    System.out.println("Error writing to output stream "
+ e);
                }
 try {
     JFMenu.puertoSerialModem.addEventListener((SerialPortEventListener)
Main.miventana);
         }
        catch (TooManyListenersException e) {
         System.out.println("ERROR " +e);        }

    JFMenu.puertoSerialModem.notifyOnDataAvailable(true);
}

    // Variables declaration - do not modify//GEN-BEGIN:variables
    private javax.swing.ButtonGroup buttonGroupTelefonicas;
    private javax.swing.JButton jButton_cancelar;
    private javax.swing.JButton jButton_ok;
    private javax.swing.JLabel jLabel1;
    public javax.swing.JList jList_puertos;
    private javax.swing.JPanel jPanel10;
    private javax.swing.JPanel jPanel11;
    private javax.swing.JPanel jPanel7;
    private javax.swing.JRadioButton jRadioButtonMovistar;
    private javax.swing.JRadioButton jRadioButtonTelcel;
    private javax.swing.JScrollPane jScrollPane1;
    private javax.swing.JTextField jTextFieldDestAlarma;
}

package raicelular;
import gnu.io.CommPortIdentifier;
import gnu.io.NoSuchPortException;
import gnu.io.PortInUseException;
import gnu.io.SerialPort;
import gnu.io.SerialPortEventListener;
import gnu.io.UnsupportedCommOperationException;
import java.io.IOException;
import java.util.TooManyListenersException;
import javax.swing.JOptionPane;
public class JDconfiguracionCircuito extends javax.swing.JDialog{
    public JDconfiguracionCircuito(java.awt.Frame parent, boolean modal)
{
        super(parent, modal);
        initComponents();
    }

    private void initComponents() {
        jPanel7 = new javax.swing.JPanel();
        jScrollPane1 = new javax.swing.JScrollPane();
        jList_puertos = new javax.swing.JList();
        jPanel6 = new javax.swing.JPanel();
        jPanel1 = new javax.swing.JPanel();
        jComboBox_databits = new javax.swing.JComboBox();
```

```
        jPanel4 = new javax.swing.JPanel();
        jComboBox_paridad = new javax.swing.JComboBox();
        jPanel5 = new javax.swing.JPanel();
        jComboBox_stopbits = new javax.swing.JComboBox();
        jPanel3 = new javax.swing.JPanel();
        jComboBox_flowcontrol = new javax.swing.JComboBox();
        jPanel_velocidad = new javax.swing.JPanel();
        jComboBox_velocidad = new javax.swing.JComboBox();
        jPanel8 = new javax.swing.JPanel();
        jButton_ok = new javax.swing.JButton();
        jButton_cancelar = new javax.swing.JButton();
        jLabel1 = new javax.swing.JLabel();

setDefaultCloseOperation(javax.swing.WindowConstants.DISPOSE_ON_CLOSE);
        setTitle("ConfiguraciÃ³n Puerto Serial CIRCUITO ELECTRONICO");
jPanel7.setBorder(javax.swing.BorderFactory.createTitledBorder(null,
"Selecciona el puerto",
javax.swing.border.TitledBorder.DEFAULT_JUSTIFICATION,
javax.swing.border.TitledBorder.DEFAULT_POSITION, new
java.awt.Font("Tahoma", 0, 11), new java.awt.Color(0, 0, 153))); //
NOI18N

        jList_puertos.setModel(new javax.swing.AbstractListModel() {
            String[] strings = { "Item 1", "Item 2", "Item 3", "Item 4",
"Item 5" };
            public int getSize() { return strings.length; }
            public Object getElementAt(int i) { return strings[i]; }
        });
        jScrollPane1.setViewportView(jList_puertos);

        javax.swing.GroupLayout jPanel7Layout = new
javax.swing.GroupLayout(jPanel7);
        jPanel7.setLayout(jPanel7Layout);
        jPanel7Layout.setHorizontalGroup(

jPanel7Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADI
NG)
            .addGroup(jPanel7Layout.createSequentialGroup()
                .addContainerGap()
                .addComponent(jScrollPane1,
javax.swing.GroupLayout.PREFERRED_SIZE, 118,
javax.swing.GroupLayout.PREFERRED_SIZE)
                .addContainerGap(javax.swing.GroupLayout.DEFAULT_SIZE,
Short.MAX_VALUE))
        );
        jPanel7Layout.setVerticalGroup(

jPanel7Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADI
NG)
            .addGroup(jPanel7Layout.createSequentialGroup()
                .addComponent(jScrollPane1,
javax.swing.GroupLayout.DEFAULT_SIZE, 213, Short.MAX_VALUE)
                .addContainerGap())
        );


jPanel6.setBorder(javax.swing.BorderFactory.createTitledBorder(null,
```

```
"ParÃ¡metros", javax.swing.border.TitledBorder.DEFAULT_JUSTIFICATION,
javax.swing.border.TitledBorder.DEFAULT_POSITION, new
java.awt.Font("Tahoma", 0, 11), new java.awt.Color(0, 0, 153)));

jPanel1.setBorder(javax.swing.BorderFactory.createTitledBorder(javax.swin
g.BorderFactory.createEtchedBorder(), "DataBits",
javax.swing.border.TitledBorder.DEFAULT_JUSTIFICATION,
javax.swing.border.TitledBorder.DEFAULT_POSITION, new
java.awt.Font("Tahoma", 1, 11), new java.awt.Color(0, 0, 102)));
        jComboBox_databits.setFont(new java.awt.Font("Tahoma", 1, 14));
        jComboBox_databits.setModel(new
javax.swing.DefaultComboBoxModel(new String[] { "5", "6", "7", "8" }));
        jComboBox_databits.setSelectedIndex(3);
        jComboBox_databits.addActionListener(new
java.awt.event.ActionListener() {
            public void actionPerformed(java.awt.event.ActionEvent evt) {
                jComboBox_databitsActionPerformed(evt);
            }
        });

        javax.swing.GroupLayout jPanel1Layout = new
javax.swing.GroupLayout(jPanel1);
        jPanel1.setLayout(jPanel1Layout);
        jPanel1Layout.setHorizontalGroup(

jPanel1Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADI
NG)
            .addGroup(javax.swing.GroupLayout.Alignment.TRAILING,
jPanel1Layout.createSequentialGroup()
                .addContainerGap()
                .addComponent(jComboBox_databits, 0, 54, Short.MAX_VALUE)
                .addContainerGap())
        );
        jPanel1Layout.setVerticalGroup(

jPanel1Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADI
NG)
            .addGroup(jPanel1Layout.createSequentialGroup()
                .addComponent(jComboBox_databits,
javax.swing.GroupLayout.PREFERRED_SIZE, 30,
javax.swing.GroupLayout.PREFERRED_SIZE)
                .addContainerGap(javax.swing.GroupLayout.DEFAULT_SIZE,
Short.MAX_VALUE))
        );


jPanel4.setBorder(javax.swing.BorderFactory.createTitledBorder(javax.swin
g.BorderFactory.createEtchedBorder(), "Paridad",
javax.swing.border.TitledBorder.DEFAULT_JUSTIFICATION,
javax.swing.border.TitledBorder.DEFAULT_POSITION, new
java.awt.Font("Tahoma", 1, 11), new java.awt.Color(0, 0, 102)));

        jComboBox_paridad.setFont(new java.awt.Font("Tahoma", 1, 12));
        jComboBox_paridad.setModel(new
javax.swing.DefaultComboBoxModel(new String[] { "EVEN", "MARK", "NONE",
"ODD", "SPACE" }));
        jComboBox_paridad.setSelectedIndex(2);
```

```
        javax.swing.GroupLayout jPanel4Layout = new
javax.swing.GroupLayout(jPanel4);
        jPanel4.setLayout(jPanel4Layout);
        jPanel4Layout.setHorizontalGroup(

jPanel4Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADI
NG)
            .addGroup(jPanel4Layout.createSequentialGroup()
                .addContainerGap()
                .addComponent(jComboBox_paridad, 0, 112, Short.MAX_VALUE)
                .addContainerGap())
        );
        jPanel4Layout.setVerticalGroup(

jPanel4Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADI
NG)
            .addGroup(jPanel4Layout.createSequentialGroup()
                .addComponent(jComboBox_paridad,
javax.swing.GroupLayout.PREFERRED_SIZE, 30,
javax.swing.GroupLayout.PREFERRED_SIZE)
                .addContainerGap(javax.swing.GroupLayout.DEFAULT_SIZE,
Short.MAX_VALUE))
        );


jPanel5.setBorder(javax.swing.BorderFactory.createTitledBorder(javax.swin
g.BorderFactory.createEtchedBorder(), "StopBits",
javax.swing.border.TitledBorder.DEFAULT_JUSTIFICATION,
javax.swing.border.TitledBorder.DEFAULT_POSITION, new
java.awt.Font("Tahoma", 1, 11), new java.awt.Color(0, 0, 102))); //
NOI18N

        jComboBox_stopbits.setFont(new java.awt.Font("Tahoma", 1, 12));
        jComboBox_stopbits.setModel(new
javax.swing.DefaultComboBoxModel(new String[] { "1", "1_5", "2" }));

        javax.swing.GroupLayout jPanel5Layout = new
javax.swing.GroupLayout(jPanel5);
        jPanel5.setLayout(jPanel5Layout);
        jPanel5Layout.setHorizontalGroup(

jPanel5Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADI
NG)
            .addGroup(jPanel5Layout.createSequentialGroup()
                .addContainerGap()
                .addComponent(jComboBox_stopbits, 0, 54, Short.MAX_VALUE)
                .addContainerGap())
        );
        jPanel5Layout.setVerticalGroup(

jPanel5Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADI
NG)
            .addGroup(jPanel5Layout.createSequentialGroup()
                .addComponent(jComboBox_stopbits,
javax.swing.GroupLayout.PREFERRED_SIZE, 30,
javax.swing.GroupLayout.PREFERRED_SIZE)
```

```
                .addContainerGap(javax.swing.GroupLayout.DEFAULT_SIZE,
Short.MAX_VALUE))
        );


jPanel3.setBorder(javax.swing.BorderFactory.createTitledBorder(javax.swin
g.BorderFactory.createEtchedBorder(), "Control de flujo",
javax.swing.border.TitledBorder.DEFAULT_JUSTIFICATION,
javax.swing.border.TitledBorder.DEFAULT_POSITION, new
java.awt.Font("Tahoma", 1, 11), new java.awt.Color(0, 0, 102))); //
NOI18N

        jComboBox_flowcontrol.setFont(new java.awt.Font("Tahoma", 1,
12));
        jComboBox_flowcontrol.setModel(new
javax.swing.DefaultComboBoxModel(new String[] { "None", "RTSCTS IN",
"RTSCTS OUT", "XON XOFF IN", "XON XOFF OUT" }));

        javax.swing.GroupLayout jPanel3Layout = new
javax.swing.GroupLayout(jPanel3);
        jPanel3.setLayout(jPanel3Layout);
        jPanel3Layout.setHorizontalGroup(

jPanel3Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADI
NG)
            .addGroup(jPanel3Layout.createSequentialGroup()
                .addContainerGap()
                .addComponent(jComboBox_flowcontrol, 0, 204,
Short.MAX_VALUE)
                .addContainerGap())
        );
        jPanel3Layout.setVerticalGroup(

jPanel3Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADI
NG)
            .addGroup(jPanel3Layout.createSequentialGroup()
                .addComponent(jComboBox_flowcontrol,
javax.swing.GroupLayout.PREFERRED_SIZE, 30,
javax.swing.GroupLayout.PREFERRED_SIZE)
                .addContainerGap(javax.swing.GroupLayout.DEFAULT_SIZE,
Short.MAX_VALUE))
        );


jPanel_velocidad.setBorder(javax.swing.BorderFactory.createTitledBorder(j
avax.swing.BorderFactory.createEtchedBorder(), "Bits por segundo",
javax.swing.border.TitledBorder.DEFAULT_JUSTIFICATION,
javax.swing.border.TitledBorder.DEFAULT_POSITION, new
java.awt.Font("Tahoma", 1, 11), new java.awt.Color(0, 0, 102))); //
NOI18N

        jComboBox_velocidad.setFont(new java.awt.Font("Tahoma", 1, 12));
        jComboBox_velocidad.setModel(new
javax.swing.DefaultComboBoxModel(new String[] { "75", "110", "134",
"150", "300", "600", "1200", "1800", "2400", "4800", "7200", "9600",
"14400", "19200", "38400", "57600", "115200", "128000" }));
        jComboBox_velocidad.setSelectedIndex(11);
```

```
        javax.swing.GroupLayout jPanel_velocidadLayout = new
javax.swing.GroupLayout(jPanel_velocidad);
        jPanel_velocidad.setLayout(jPanel_velocidadLayout);
        jPanel_velocidadLayout.setHorizontalGroup(

jPanel_velocidadLayout.createParallelGroup(javax.swing.GroupLayout.Alignm
ent.LEADING)
            .addGroup(jPanel_velocidadLayout.createSequentialGroup()
                .addContainerGap()
                .addComponent(jComboBox_velocidad, 0, 112,
Short.MAX_VALUE)
                .addContainerGap())
        );
        jPanel_velocidadLayout.setVerticalGroup(

jPanel_velocidadLayout.createParallelGroup(javax.swing.GroupLayout.Alignm
ent.LEADING)
            .addGroup(jPanel_velocidadLayout.createSequentialGroup()
                .addComponent(jComboBox_velocidad,
javax.swing.GroupLayout.PREFERRED_SIZE, 30,
javax.swing.GroupLayout.PREFERRED_SIZE)
                .addContainerGap(javax.swing.GroupLayout.DEFAULT_SIZE,
Short.MAX_VALUE))
        );

        javax.swing.GroupLayout jPanel6Layout = new
javax.swing.GroupLayout(jPanel6);
        jPanel6.setLayout(jPanel6Layout);
        jPanel6Layout.setHorizontalGroup(

jPanel6Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADI
NG)
            .addGroup(javax.swing.GroupLayout.Alignment.TRAILING,
jPanel6Layout.createSequentialGroup()
                .addContainerGap()

.addGroup(jPanel6Layout.createParallelGroup(javax.swing.GroupLayout.Align
ment.TRAILING)
                    .addComponent(jPanel3,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE)
                    .addGroup(jPanel6Layout.createSequentialGroup()

.addGroup(jPanel6Layout.createParallelGroup(javax.swing.GroupLayout.Align
ment.TRAILING)
                            .addComponent(jPanel5,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE)
                            .addComponent(jPanel1,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE))

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)

.addGroup(jPanel6Layout.createParallelGroup(javax.swing.GroupLayout.Align
ment.LEADING)
```

```
                            .addComponent(jPanel4,
javax.swing.GroupLayout.Alignment.TRAILING,
javax.swing.GroupLayout.PREFERRED_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE)
                            .addComponent(jPanel_velocidad,
javax.swing.GroupLayout.Alignment.TRAILING,
javax.swing.GroupLayout.PREFERRED_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE))))
                .addContainerGap())
        );
        jPanel6Layout.setVerticalGroup(

jPanel6Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADI
NG)
            .addGroup(jPanel6Layout.createSequentialGroup()

.addGroup(jPanel6Layout.createParallelGroup(javax.swing.GroupLayout.Align
ment.TRAILING, false)
                    .addComponent(jPanel1,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE)
                    .addComponent(jPanel4,
javax.swing.GroupLayout.PREFERRED_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE))

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)

.addGroup(jPanel6Layout.createParallelGroup(javax.swing.GroupLayout.Align
ment.LEADING)
                    .addComponent(jPanel_velocidad,
javax.swing.GroupLayout.PREFERRED_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE)
                    .addComponent(jPanel5,
javax.swing.GroupLayout.PREFERRED_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE))

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)
                .addComponent(jPanel3,
javax.swing.GroupLayout.PREFERRED_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE)
                .addContainerGap())
        );

        jButton_ok.setFont(new java.awt.Font("Tahoma", 1, 24));
        jButton_ok.setIcon(new
javax.swing.ImageIcon(getClass().getResource("/images/Undo.png"))); //
NOI18N
        jButton_ok.setText("OK");
        jButton_ok.addActionListener(new java.awt.event.ActionListener()
{
            public void actionPerformed(java.awt.event.ActionEvent evt) {
```

```
                jButton_okActionPerformed(evt);
            }
        });

        jButton_cancelar.setFont(new java.awt.Font("Tahoma", 1, 12));
        jButton_cancelar.setText("CANCELAR");
        jButton_cancelar.addActionListener(new
java.awt.event.ActionListener() {
            public void actionPerformed(java.awt.event.ActionEvent evt) {
                jButton_cancelarActionPerformed(evt);
            }
        });

        jLabel1.setIcon(new
javax.swing.ImageIcon(getClass().getResource("/images/serial.jpg"))); //
NOI18N

        javax.swing.GroupLayout jPanel8Layout = new
javax.swing.GroupLayout(jPanel8);
        jPanel8.setLayout(jPanel8Layout);
        jPanel8Layout.setHorizontalGroup(

jPanel8Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADI
NG)
            .addGroup(jPanel8Layout.createSequentialGroup()
                .addContainerGap()

.addGroup(jPanel8Layout.createParallelGroup(javax.swing.GroupLayout.Align
ment.LEADING)
                    .addGroup(jPanel8Layout.createSequentialGroup()
                        .addComponent(jButton_ok,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE)
                        .addContainerGap())
                    .addGroup(javax.swing.GroupLayout.Alignment.TRAILING,
jPanel8Layout.createSequentialGroup()

.addGroup(jPanel8Layout.createParallelGroup(javax.swing.GroupLayout.Align
ment.TRAILING)
                            .addComponent(jButton_cancelar,
javax.swing.GroupLayout.Alignment.LEADING,
javax.swing.GroupLayout.DEFAULT_SIZE, 133, Short.MAX_VALUE)
                            .addComponent(jLabel1,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE))
                        .addGap(12, 12, 12))))
        );
        jPanel8Layout.setVerticalGroup(

jPanel8Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADI
NG)
            .addGroup(jPanel8Layout.createSequentialGroup()
                .addContainerGap()
                .addComponent(jButton_cancelar,
javax.swing.GroupLayout.PREFERRED_SIZE, 34,
javax.swing.GroupLayout.PREFERRED_SIZE)
                .addGap(34, 34, 34)
```

```
                .addComponent(jLabel1,
javax.swing.GroupLayout.PREFERRED_SIZE, 55,
javax.swing.GroupLayout.PREFERRED_SIZE)

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED, 36,
Short.MAX_VALUE)
                .addComponent(jButton_ok)
                .addContainerGap())
        );

        javax.swing.GroupLayout layout = new
javax.swing.GroupLayout(getContentPane());
        getContentPane().setLayout(layout);
        layout.setHorizontalGroup(

layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
            .addGroup(layout.createSequentialGroup()
                .addContainerGap()
                .addComponent(jPanel7,
javax.swing.GroupLayout.PREFERRED_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE)
                .addGap(18, 18, 18)
                .addComponent(jPanel6,
javax.swing.GroupLayout.PREFERRED_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE)
                .addGap(18, 18, 18)
                .addComponent(jPanel8,
javax.swing.GroupLayout.PREFERRED_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE)
                .addContainerGap(javax.swing.GroupLayout.DEFAULT_SIZE,
Short.MAX_VALUE))
        );
        layout.setVerticalGroup(

layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
            .addGroup(layout.createSequentialGroup()
                .addContainerGap()

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LE
ADING)
                    .addComponent(jPanel6,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE)
                    .addComponent(jPanel7,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE)
                    .addComponent(jPanel8,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE))
                .addContainerGap())
        );

        pack();
    }// </editor-fold>//GEN-END:initComponents
```

```
private void jButton_cancelarActionPerformed(java.awt.event.ActionEvent
evt) {//GEN-FIRST:event_jButton_cancelarActionPerformed
this.jList_puertos.setSelectedIndex(Main.miventana.numPuertoCircuito);
this.jComboBox_databits.setSelectedIndex(Main.miventana.databitsCircuito)
;
this.jComboBox_paridad.setSelectedIndex(Main.miventana.paridadCircuito);
this.jComboBox_velocidad.setSelectedIndex(Main.miventana.velocidadCircuit
o);
this.jComboBox_stopbits.setSelectedIndex(Main.miventana.stopbitsCircuito)
;
this.jComboBox_flowcontrol.setSelectedIndex(Main.miventana.controlFlujoCi
rcuito);
this.setVisible(false);
}//GEN-LAST:event_jButton_cancelarActionPerformed

private void jButton_okActionPerformed(java.awt.event.ActionEvent evt)
{okActionPerformed

 configura_y_sincroniza();
}

private void jComboBox_databitsActionPerformed(java.awt.event.ActionEvent
evt) {//
    // TODO add your handling code here:
}//



public void configura_y_sincroniza() {

Main.miventana.numPuertoCircuito=this.jList_puertos.getSelectedIndex();
Main.miventana.databitsCircuito=this.jComboBox_databits.getSelectedIndex(
);
Main.miventana.paridadCircuito=this.jComboBox_paridad.getSelectedIndex();
Main.miventana.velocidadCircuito=this.jComboBox_velocidad.getSelectedInde
x();
Main.miventana.stopbitsCircuito=this.jComboBox_stopbits.getSelectedIndex(
);
Main.miventana.controlFlujoCircuito=this.jComboBox_flowcontrol.getSelecte
dIndex();
Main.miventana.puertoCircuito=this.jList_puertos.getSelectedValue().toStr
ing();
configura_puerto();
Main.miventana.habilita_checkbox_Puertos();
Main.miventana.sincroniza_Puertos();
Main.estaConfiguradoCircuito=true;
this.setVisible(false);
JOptionPane.showMessageDialog(null, "CIRCUITO CONFIGURADO, OK", "AVISO",
JOptionPane.PLAIN_MESSAGE);

}

private void configura_puerto(){
    System.out.println("Puerto a configurar: "
+Main.miventana.puertoCircuito);
```

```
     try {
       JFMenu.portIdCircuito =
CommPortIdentifier.getPortIdentifier(Main.miventana.puertoCircuito);
        } catch (NoSuchPortException ex) {
          System.out.println("Puerto no encontrado " +ex);
        }

    if( JFMenu.portIdCircuito.isCurrentlyOwned()){
        String propietario= JFMenu.portIdCircuito.getCurrentOwner();
        System.out.println("PUERTO USADO POR: "+propietario);
    return;
}
    abre_puerto();
    int
veloc=Integer.parseInt(this.jComboBox_velocidad.getSelectedItem().toStrin
g());
 try {
     switch (Main.miventana.controlFlujoCircuito){
         case
0:JFMenu.puertoSerialCircuito.setFlowControlMode(SerialPort.FLOWCONTROL_N
ONE); break;
          case
1:JFMenu.puertoSerialCircuito.setFlowControlMode(SerialPort.FLOWCONTROL_R
TSCTS_IN); break;
           case
2:JFMenu.puertoSerialCircuito.setFlowControlMode(SerialPort.FLOWCONTROL_R
TSCTS_OUT); break;
            case
3:JFMenu.puertoSerialCircuito.setFlowControlMode(SerialPort.FLOWCONTROL_X
ONXOFF_IN); break;
             case
4:JFMenu.puertoSerialCircuito.setFlowControlMode(SerialPort.FLOWCONTROL_X
ONXOFF_OUT); break;
        }
     switch (Main.miventana.databitsCircuito){//0,1,2,3
         case 0:
             switch (Main.miventana.stopbitsCircuito){ //0,1,2
                 case 0:
                     switch (Main.miventana.paridadCircuito){ //0,1,2,3,4
                         case 0:
JFMenu.puertoSerialCircuito.setSerialPortParams(veloc,SerialPort.DATABITS
_5,SerialPort.STOPBITS_1, SerialPort.PARITY_EVEN);    break;
                         case 1:
JFMenu.puertoSerialCircuito.setSerialPortParams(veloc,SerialPort.DATABITS
_5,SerialPort.STOPBITS_1, SerialPort.PARITY_MARK);    break;
                         case 2:
JFMenu.puertoSerialCircuito.setSerialPortParams(veloc,SerialPort.DATABITS
_5,SerialPort.STOPBITS_1, SerialPort.PARITY_NONE);    break;
                         case 3:
JFMenu.puertoSerialCircuito.setSerialPortParams(veloc,SerialPort.DATABITS
_5,SerialPort.STOPBITS_1, SerialPort.PARITY_ODD);    break;
                         case 4:
JFMenu.puertoSerialCircuito.setSerialPortParams(veloc,SerialPort.DATABITS
_5,SerialPort.STOPBITS_1, SerialPort.PARITY_SPACE);    break;
                                     }  break;

                 case 1:
```

```
                    switch (Main.miventana.paridadCircuito){ //0,1,2,3,4
                        case 0:
JFMenu.puertoSerialCircuito.setSerialPortParams(veloc,SerialPort.DATABITS
_5,SerialPort.STOPBITS_1_5, SerialPort.PARITY_EVEN);    break;
                        case 1:
JFMenu.puertoSerialCircuito.setSerialPortParams(veloc,SerialPort.DATABITS
_5,SerialPort.STOPBITS_1_5, SerialPort.PARITY_MARK);    break;
                        case 2:
JFMenu.puertoSerialCircuito.setSerialPortParams(veloc,SerialPort.DATABITS
_5,SerialPort.STOPBITS_1_5, SerialPort.PARITY_NONE);    break;
                        case 3:
JFMenu.puertoSerialCircuito.setSerialPortParams(veloc,SerialPort.DATABITS
_5,SerialPort.STOPBITS_1_5, SerialPort.PARITY_ODD);    break;
                        case 4:
JFMenu.puertoSerialCircuito.setSerialPortParams(veloc,SerialPort.DATABITS
_5,SerialPort.STOPBITS_1_5, SerialPort.PARITY_SPACE);    break;
                                    }  break;

                case 2:
                    switch (Main.miventana.paridadCircuito){ //0,1,2,3,4
                        case 0:
JFMenu.puertoSerialCircuito.setSerialPortParams(veloc,SerialPort.DATABITS
_5,SerialPort.STOPBITS_2, SerialPort.PARITY_EVEN);    break;
                        case 1:
JFMenu.puertoSerialCircuito.setSerialPortParams(veloc,SerialPort.DATABITS
_5,SerialPort.STOPBITS_2, SerialPort.PARITY_MARK);    break;
                        case 2:
JFMenu.puertoSerialCircuito.setSerialPortParams(veloc,SerialPort.DATABITS
_5,SerialPort.STOPBITS_2, SerialPort.PARITY_NONE);    break;
                        case 3:
JFMenu.puertoSerialCircuito.setSerialPortParams(veloc,SerialPort.DATABITS
_5,SerialPort.STOPBITS_2, SerialPort.PARITY_ODD);    break;
                        case 4:
JFMenu.puertoSerialCircuito.setSerialPortParams(veloc,SerialPort.DATABITS
_5,SerialPort.STOPBITS_2, SerialPort.PARITY_SPACE);    break;
                                    }  break;
             }//fin del switch stopbits      break;

case 1:
            switch (Main.miventana.stopbitsCircuito){ //0,1,2
                case 0:
                    switch (Main.miventana.paridadCircuito){ //0,1,2,3,4
                        case 0:
JFMenu.puertoSerialCircuito.setSerialPortParams(veloc,SerialPort.DATABITS
_6,SerialPort.STOPBITS_1, SerialPort.PARITY_EVEN);    break;
                        case 1:
JFMenu.puertoSerialCircuito.setSerialPortParams(veloc,SerialPort.DATABITS
_6,SerialPort.STOPBITS_1, SerialPort.PARITY_MARK);    break;
                        case 2:
JFMenu.puertoSerialCircuito.setSerialPortParams(veloc,SerialPort.DATABITS
_6,SerialPort.STOPBITS_1, SerialPort.PARITY_NONE);    break;
                        case 3:
JFMenu.puertoSerialCircuito.setSerialPortParams(veloc,SerialPort.DATABITS
_6,SerialPort.STOPBITS_1, SerialPort.PARITY_ODD);    break;
                        case 4:
JFMenu.puertoSerialCircuito.setSerialPortParams(veloc,SerialPort.DATABITS
_6,SerialPort.STOPBITS_1, SerialPort.PARITY_SPACE);    break;
```

```
                                    }  break;

                 case 1:
                    switch (Main.miventana.paridadCircuito){ //0,1,2,3,4
                        case 0:
JFMenu.puertoSerialCircuito.setSerialPortParams(veloc,SerialPort.DATABITS
_6,SerialPort.STOPBITS_1_5, SerialPort.PARITY_EVEN);    break;
                        case 1:
JFMenu.puertoSerialCircuito.setSerialPortParams(veloc,SerialPort.DATABITS
_6,SerialPort.STOPBITS_1_5, SerialPort.PARITY_MARK);    break;
                        case 2:
JFMenu.puertoSerialCircuito.setSerialPortParams(veloc,SerialPort.DATABITS
_6,SerialPort.STOPBITS_1_5, SerialPort.PARITY_NONE);    break;
                        case 3:
JFMenu.puertoSerialCircuito.setSerialPortParams(veloc,SerialPort.DATABITS
_6,SerialPort.STOPBITS_1_5, SerialPort.PARITY_ODD);    break;
                        case 4:
JFMenu.puertoSerialCircuito.setSerialPortParams(veloc,SerialPort.DATABITS
_6,SerialPort.STOPBITS_1_5, SerialPort.PARITY_SPACE);    break;
                                    }  break;

                case 2:
                    switch (Main.miventana.paridadCircuito){ //0,1,2,3,4
                        case 0:
JFMenu.puertoSerialCircuito.setSerialPortParams(veloc,SerialPort.DATABITS
_6,SerialPort.STOPBITS_2, SerialPort.PARITY_EVEN);    break;
                        case 1:
JFMenu.puertoSerialCircuito.setSerialPortParams(veloc,SerialPort.DATABITS
_6,SerialPort.STOPBITS_2, SerialPort.PARITY_MARK);    break;
                        case 2:
JFMenu.puertoSerialCircuito.setSerialPortParams(veloc,SerialPort.DATABITS
_6,SerialPort.STOPBITS_2, SerialPort.PARITY_NONE);    break;
                        case 3:
JFMenu.puertoSerialCircuito.setSerialPortParams(veloc,SerialPort.DATABITS
_6,SerialPort.STOPBITS_2, SerialPort.PARITY_ODD);    break;
                        case 4:
JFMenu.puertoSerialCircuito.setSerialPortParams(veloc,SerialPort.DATABITS
_6,SerialPort.STOPBITS_2, SerialPort.PARITY_SPACE);    break;
                                    }  break;
             }//fin del switch stopbits      break;



case 2:
            switch (Main.miventana.stopbitsCircuito){ //0,1,2
                case 0:
                    switch (Main.miventana.paridadCircuito){ //0,1,2,3,4
                        case 0:
JFMenu.puertoSerialCircuito.setSerialPortParams(veloc,SerialPort.DATABITS
_7,SerialPort.STOPBITS_1, SerialPort.PARITY_EVEN);    break;
                        case 1:
JFMenu.puertoSerialCircuito.setSerialPortParams(veloc,SerialPort.DATABITS
_7,SerialPort.STOPBITS_1, SerialPort.PARITY_MARK);    break;
                        case 2:
JFMenu.puertoSerialCircuito.setSerialPortParams(veloc,SerialPort.DATABITS
_7,SerialPort.STOPBITS_1, SerialPort.PARITY_NONE);    break;
```

```
                        case 3:
JFMenu.puertoSerialCircuito.setSerialPortParams(veloc,SerialPort.DATABITS
_7,SerialPort.STOPBITS_1, SerialPort.PARITY_ODD);    break;
                        case 4:
JFMenu.puertoSerialCircuito.setSerialPortParams(veloc,SerialPort.DATABITS
_7,SerialPort.STOPBITS_1, SerialPort.PARITY_SPACE);    break;
                                    }  break;

                 case 1:
                    switch (Main.miventana.paridadCircuito){ //0,1,2,3,4
                        case 0:
JFMenu.puertoSerialCircuito.setSerialPortParams(veloc,SerialPort.DATABITS
_7,SerialPort.STOPBITS_1_5, SerialPort.PARITY_EVEN);    break;
                        case 1:
JFMenu.puertoSerialCircuito.setSerialPortParams(veloc,SerialPort.DATABITS
_7,SerialPort.STOPBITS_1_5, SerialPort.PARITY_MARK);    break;
                        case 2:
JFMenu.puertoSerialCircuito.setSerialPortParams(veloc,SerialPort.DATABITS
_7,SerialPort.STOPBITS_1_5, SerialPort.PARITY_NONE);    break;
                        case 3:
JFMenu.puertoSerialCircuito.setSerialPortParams(veloc,SerialPort.DATABITS
_7,SerialPort.STOPBITS_1_5, SerialPort.PARITY_ODD);    break;
                        case 4:
JFMenu.puertoSerialCircuito.setSerialPortParams(veloc,SerialPort.DATABITS
_7,SerialPort.STOPBITS_1_5, SerialPort.PARITY_SPACE);    break;
                                    }  break;

                case 2:
                    switch (Main.miventana.paridadCircuito){ //0,1,2,3,4
                        case 0:
JFMenu.puertoSerialCircuito.setSerialPortParams(veloc,SerialPort.DATABITS
_7,SerialPort.STOPBITS_2, SerialPort.PARITY_EVEN);    break;
                        case 1:
JFMenu.puertoSerialCircuito.setSerialPortParams(veloc,SerialPort.DATABITS
_7,SerialPort.STOPBITS_2, SerialPort.PARITY_MARK);    break;
                        case 2:
JFMenu.puertoSerialCircuito.setSerialPortParams(veloc,SerialPort.DATABITS
_7,SerialPort.STOPBITS_2, SerialPort.PARITY_NONE);    break;
                        case 3:
JFMenu.puertoSerialCircuito.setSerialPortParams(veloc,SerialPort.DATABITS
_7,SerialPort.STOPBITS_2, SerialPort.PARITY_ODD);    break;
                        case 4:
JFMenu.puertoSerialCircuito.setSerialPortParams(veloc,SerialPort.DATABITS
_7,SerialPort.STOPBITS_2, SerialPort.PARITY_SPACE);    break;
                                    }  break;
             }//fin del switch stopbits      break;

case 3:
             switch (Main.miventana.stopbitsCircuito){ //0,1,2
                case 0:
                    switch (Main.miventana.paridadCircuito){ //0,1,2,3,4
                        case 0:
JFMenu.puertoSerialCircuito.setSerialPortParams(veloc,SerialPort.DATABITS
_8,SerialPort.STOPBITS_1, SerialPort.PARITY_EVEN);    break;
                        case 1:
JFMenu.puertoSerialCircuito.setSerialPortParams(veloc,SerialPort.DATABITS
_8,SerialPort.STOPBITS_1, SerialPort.PARITY_MARK);    break;
```

```
                        case 2:
JFMenu.puertoSerialCircuito.setSerialPortParams(veloc,SerialPort.DATABITS
_8,SerialPort.STOPBITS_1, SerialPort.PARITY_NONE);    break;
                        case 3:
JFMenu.puertoSerialCircuito.setSerialPortParams(veloc,SerialPort.DATABITS
_8,SerialPort.STOPBITS_1, SerialPort.PARITY_ODD);    break;
                        case 4:
JFMenu.puertoSerialCircuito.setSerialPortParams(veloc,SerialPort.DATABITS
_8,SerialPort.STOPBITS_1, SerialPort.PARITY_SPACE);    break;
                                    }  break;

                 case 1:
                    switch (Main.miventana.paridadCircuito){ //0,1,2,3,4
                        case 0:
JFMenu.puertoSerialCircuito.setSerialPortParams(veloc,SerialPort.DATABITS
_8,SerialPort.STOPBITS_1_5, SerialPort.PARITY_EVEN);    break;
                        case 1:
JFMenu.puertoSerialCircuito.setSerialPortParams(veloc,SerialPort.DATABITS
_8,SerialPort.STOPBITS_1_5, SerialPort.PARITY_MARK);    break;
                        case 2:
JFMenu.puertoSerialCircuito.setSerialPortParams(veloc,SerialPort.DATABITS
_8,SerialPort.STOPBITS_1_5, SerialPort.PARITY_NONE);    break;
                        case 3:
JFMenu.puertoSerialCircuito.setSerialPortParams(veloc,SerialPort.DATABITS
_8,SerialPort.STOPBITS_1_5, SerialPort.PARITY_ODD);    break;
                        case 4:
JFMenu.puertoSerialCircuito.setSerialPortParams(veloc,SerialPort.DATABITS
_8,SerialPort.STOPBITS_1_5, SerialPort.PARITY_SPACE);    break;
                                    }  break;

                case 2:
                    switch (Main.miventana.paridadCircuito){ //0,1,2,3,4
                        case 0:
JFMenu.puertoSerialCircuito.setSerialPortParams(veloc,SerialPort.DATABITS
_8,SerialPort.STOPBITS_2, SerialPort.PARITY_EVEN);    break;
                        case 1:
JFMenu.puertoSerialCircuito.setSerialPortParams(veloc,SerialPort.DATABITS
_8,SerialPort.STOPBITS_2, SerialPort.PARITY_MARK);    break;
                        case 2:
JFMenu.puertoSerialCircuito.setSerialPortParams(veloc,SerialPort.DATABITS
_8,SerialPort.STOPBITS_2, SerialPort.PARITY_NONE);    break;
                        case 3:
JFMenu.puertoSerialCircuito.setSerialPortParams(veloc,SerialPort.DATABITS
_8,SerialPort.STOPBITS_2, SerialPort.PARITY_ODD);    break;
                        case 4:
JFMenu.puertoSerialCircuito.setSerialPortParams(veloc,SerialPort.DATABITS
_8,SerialPort.STOPBITS_2, SerialPort.PARITY_SPACE);    break;
                                    }  break;
             }//fin del switch stopbits      break;

            }

 }
        catch (UnsupportedCommOperationException e) {
            System.out.println("ERROR AL CONFIGURAR EL PUERTO " +e);
            cierra_puerto();
```

```
    }
}


private void cierra_puerto(){
JFMenu.puertoSerialCircuito.close();
}

private void abre_puerto(){

 try {

        JFMenu.puertoSerialCircuito = (SerialPort)
JFMenu.portIdCircuito.open("Oziel", 2000);
                }
   catch (PortInUseException e){

JOptionPane.showMessageDialog(this, "El puerto estÃ¡ en uso por otra
aplicaciÃ³n", "!Advertencia!", JOptionPane.INFORMATION_MESSAGE);
                                        System.out.println("ERROR AL
ABRIR EL PUERTO "+e);

                                    /*if
(JFMenu.portIdCircuito.isCurrentlyOwned()){
       JFMenu.puertoSerialCircuito.close();
       System.out.println("PUERTO YA INICIADO");
       }

       abre_puerto();*/      System.exit(0);
                              }//fin del catch

     try {
            JFMenu.flujoEntradaCircuito =
JFMenu.puertoSerialCircuito.getInputStream();
    } catch (IOException e) {
         System.out.println("ERROR AL OBTENER FLUJO DE ENTRADA DEL PUERTO
SERIAL - CIRCUITO "+e );
    }

       try
                  {
                    JFMenu.flujoSalidaCircuito =
JFMenu.puertoSerialCircuito.getOutputStream();
                  }
              catch (IOException e)
                  {
                    System.out.println("ERROR AL OBTENER FLUJO DE SALIDA
DEL PUERTO SERIAL - CIRCUITO "+e);
                  }

 try {

JFMenu.puertoSerialCircuito.addEventListener((SerialPortEventListener)
Main.miventana);
        }
       catch (TooManyListenersException e) {
        System.out.println("ERROR " +e);
```

```
        }
    JFMenu.puertoSerialCircuito.notifyOnDataAvailable(true);
}
    private javax.swing.JButton jButton_cancelar;
    private javax.swing.JButton jButton_ok;
    private javax.swing.JComboBox jComboBox_databits;
    private javax.swing.JComboBox jComboBox_flowcontrol;
    private javax.swing.JComboBox jComboBox_paridad;
    private javax.swing.JComboBox jComboBox_stopbits;
    private javax.swing.JComboBox jComboBox_velocidad;
    private javax.swing.JLabel jLabel1;
    public javax.swing.JList jList_puertos;
    private javax.swing.JPanel jPanel1;
    private javax.swing.JPanel jPanel3;
    private javax.swing.JPanel jPanel4;
    private javax.swing.JPanel jPanel5;
    private javax.swing.JPanel jPanel6;
    private javax.swing.JPanel jPanel7;
    private javax.swing.JPanel jPanel8;
    private javax.swing.JPanel jPanel_velocidad;
    private javax.swing.JScrollPane jScrollPane1;
    // End of variables declaration//GEN-END:variables


}

package raicelular;
public class ExceptionManager {
public static void ManageException (Exception e)
{
    System.out.println ("Se ha producido un error");
    System.out.println (e.getMessage());
    e.printStackTrace(System.out);
}
}




package raicelular;
public class ComandosClass {
  public void ejecutaComando(String sms){
    int indice= sms.indexOf(" ");
    String comando=sms.substring(0, indice);
    String parametro =sms.substring(indice).trim();
if(comando.equals("CONSULTA_CAMPO")){
 Main.miventana.smsclass.EnviaSMS(Main.destinatarioSMS, "OZIEL ALARMA 5
10 12 23 75");

 else if(comando.equals("CONSULTA_PUERTO")){
        int bits_consulta=Integer.parseInt(parametro)+127;
        Main.miventana.enciende(bits_consulta);
                if (Main.resultadoConsulta==0) {
Main.miventana.smsclass.EnviaSMS(Main.destinatarioSMS, "OZIEL VENTANA EL
PUERTO "+parametro+ " se encuentra ENCENDIDO");
                }
                else
```

```
                    {
                Main.miventana.smsclass.EnviaSMS(Main.destinatarioSMS,
"OZIEL VENTANA EL PUERTO "+parametro+ " se encuentra APAGADO");
                    }
}
else if (comando.equals("PRENDER_PUERTO")){
    switch (Integer.parseInt(parametro)){
            case 1:
                Main.miventana.enciende(64);
                Main.miventana.jCheckBoxPort1.setSelected(true);
                break;
            case 2:
                Main.miventana.enciende(65);
                Main.miventana.jCheckBoxPort2.setSelected(true);
                break;
             case 3:
                Main.miventana.enciende(66);
                Main.miventana.jCheckBoxPort3.setSelected(true);
                break;
             case 4:
                Main.miventana.enciende(67);
                Main.miventana.jCheckBoxPort4.setSelected(true);
                break;
            case 5:
                Main.miventana.enciende(68);
                Main.miventana.jCheckBoxPort5.setSelected(true);
                break;
            case 6:
                Main.miventana.enciende(69);
                Main.miventana.jCheckBoxPort6.setSelected(true);
                break;
            case 7:
                Main.miventana.enciende(70);
                Main.miventana.jCheckBoxPort7.setSelected(true);
                break;
            case 8:
                Main.miventana.enciende(71);
                Main.miventana.jCheckBoxPort8.setSelected(true);
                break;
        }

}else if(comando.equals("APAGAR_PUERTO")){
        switch (Integer.parseInt(parametro)){
            case 1:
                Main.miventana.enciende(0);
                Main.miventana.jCheckBoxPort1.setSelected(false);
                break;
            case 2:
                Main.miventana.enciende(1);
                Main.miventana.jCheckBoxPort2.setSelected(false);
                break;
             case 3:
                Main.miventana.enciende(2);
                Main.miventana.jCheckBoxPort3.setSelected(false);
                break;
             case 4:
                Main.miventana.enciende(3);
```

```
                Main.miventana.jCheckBoxPort4.setSelected(false);
                break;
            case 5:
                Main.miventana.enciende(4);
                Main.miventana.jCheckBoxPort5.setSelected(false);
                break;
            case 6:
                Main.miventana.enciende(5);
                Main.miventana.jCheckBoxPort6.setSelected(false);
                break;
            case 7:
                Main.miventana.enciende(6);
                Main.miventana.jCheckBoxPort7.setSelected(false);
                break;
            case 8:
                Main.miventana.enciende(7);
                Main.miventana.jCheckBoxPort8.setSelected(false);
                break;
        }

}else if(comando.equals("RM")){
int indice2=parametro.indexOf(" ");
String destinatario = parametro.substring(0, indice2);
String mensaje=parametro.substring(indice2).trim();
Main.miventana.mailClass.Send(destinatario, mensaje);
}
else        {
         Main.miventana.smsclass.EnviaSMS(Main.destinatarioSMS, "OZIEL
VENTANA COMANDO INVALIDO");

        }
  }
}


package raicelular;
import java.io.IOException;
import javax.swing.JOptionPane;
import org.smslib.AGateway.GatewayStatuses;
import org.smslib.GatewayException;
import org.smslib.ICallNotification;
import org.smslib.IGatewayStatusNotification;
import org.smslib.IInboundMessageNotification;
import org.smslib.IOutboundMessageNotification;
import org.smslib.InboundMessage;
import org.smslib.Message.MessageTypes;
import org.smslib.OutboundMessage;
import org.smslib.SMSLibException;
import org.smslib.Service;
import org.smslib.TimeoutException;
import org.smslib.modem.SerialModemGateway;
//import org.apache.log4j.BasicConfigurator;

public class ClaseSMS  {
    public ComandosClass comandos=new ComandosClass();
    SerialModemGateway miModem;
    Service srv=new Service();
```

```
    OutboundMessage msg;

  public boolean iniciaServicio(String puertoSerial){
      try {
    OutboundNotification salidaNotification = new OutboundNotification();
    InboundNotification inboundNotification = new InboundNotification();
    CallNotification callNotification = new CallNotification();
    GatewayStatusNotification statusNotification = new
GatewayStatusNotification();

            miModem = new SerialModemGateway("modemGSM.oziel",
puertoSerial, 9600, "Sony", "");
            miModem.setInbound(true); //definimos que el modem puede
recibir mensajes
            miModem.setOutbound(true); //definimos que el modem puede
enviar mensajes
            miModem.setSimPin("0000"); //se establece el PIN de seguridad
del chip
            //pasamos los parametros necesarios para establecer el
servicio de envio y recepcion
            //de mensajes
            srv.setOutboundNotification(salidaNotification);
            srv.setInboundNotification(inboundNotification);
            srv.setCallNotification(callNotification);
            srv.setGatewayStatusNotification(statusNotification);
            srv.addGateway(miModem);
            srv.startService(); //inicia el servicio
            System.out.println("INICIO DE MODEM GSM CORRECTO");
            JOptionPane.showMessageDialog(null, "INICO DE MODEM CORRECTO,
OK", "AVISO", JOptionPane.PLAIN_MESSAGE);
            Main.estaConfiguradoModem=true;
            return true;

        } catch (SMSLibException ex) {
            System.out.println("ERROR AL INICIAR SERVICIO MODEM
GSM,SMSLibException, "+ex);
            //JOptionPane.showMessageDialog(null, ex.getStackTrace(),
"AVISO", JOptionPane.PLAIN_MESSAGE);
            ex.printStackTrace();
            return false;
        } catch (IOException ex) {
            System.out.println("ERROR AL INICIAR SERVICIO MODEM
GSM,IOException, "+ex);
           // JOptionPane.showMessageDialog(null, ex, "AVISO",
JOptionPane.PLAIN_MESSAGE);
            return false;
        } catch (InterruptedException ex) {
            System.out.println("ERROR AL INICIAR SERVICIO MODEM
GSM,InterruptedException, "+ex);
            //JOptionPane.showMessageDialog(null, ex, "AVISO",
JOptionPane.PLAIN_MESSAGE);
            return false;     }

    }

public int verifica_senial(){
       int senial=0;
```

```
        try {
              senial = miModem.getSignalLevel();
                } catch (TimeoutException ex) {
                   System.out.println("ERROR AL INICIAR SERVICIO MODEM
GSM,TimeoutException, "+ex);
                } catch (GatewayException ex) {
                   System.out.println("ERROR AL INICIAR SERVICIO MODEM
GSM,GatewayExceptioon, "+ex);
                } catch (IOException ex) {
                   System.out.println("ERROR AL INICIAR SERVICIO MODEM
GSM,IOException, "+ex);
                } catch (InterruptedException ex) {
                    System.out.println("ERROR AL INICIAR SERVICIO MODEM
GSM,InterruptedException, "+ex);
            }

   return senial;
}


      public void EnviaSMS(String numCel, String mensaje) // Se envia el
mensaje sincronamente
      {
                try {
                    msg = new OutboundMessage(numCel, mensaje);
                    msg.setFrom("RAI OZIEL");
                    System.out.println("ENVIANDO...");
                    srv.sendMessage(msg);
                } catch (TimeoutException ex) {
                    System.out.println("ERROR AL ENVIAR MENSAJE, MODEM
SMS TimeoutException,  "+ex);
                } catch (GatewayException ex) {
                    System.out.println("ERROR AL ENVIAR MENSAJE, MODEM
SMS GatewayException,  "+ex);
                } catch (IOException ex) {
                    System.out.println("ERROR AL ENVIAR MENSAJE, MODEM
SMS IOException,  "+ex);
                } catch (InterruptedException ex) {
                    System.out.println("ERROR AL ENVIAR MENSAJE, MODEM
SMS InterruptedException,  "+ex);
                }
        System.out.println("MENSAJE ENVIADO CORRECTAMENTE");
        JOptionPane.showMessageDialog(null, "SMS ENVIADO CORRECTAMENTE",
"AVISO", JOptionPane.PLAIN_MESSAGE);
      }

    public void finaliza_Servicio(){ //Detiene el servicio que se
encuentra en ejecucion
        try {
            srv.stopService();
        } catch (TimeoutException ex) {
            System.out.println("Error al finalizar el servicio, TIEMPO DE
EXPIRACION "+ex);
        } catch (GatewayException ex) {
           System.out.println("Error al finalizar el servicio, ERROR DEL
MODEM "+ex);
        } catch (IOException ex) {
```

```
            System.out.println("Error al finalizar el servicio, ERROR IO
"+ex);
        } catch (InterruptedException ex) {
            System.out.println("Error al finalizar el servicio, ERROR
INTERRUPCION "+ex);
        }
    }

      public class OutboundNotification implements
IOutboundMessageNotification
      {
            public void process(String elModem, OutboundMessage msg)
            {
                  System.out.println("Notificacion de salida por parte
de: " + elModem);
                  System.out.println(msg);
            }
      }//fin de la clase OutboundNotification


    public   class InboundNotification implements
IInboundMessageNotification
      {
           public void process(String gatewayId, MessageTypes msgType,
InboundMessage msg)
            {
                  if (msgType == MessageTypes.INBOUND) {
                        System.out.println(">>> Nuevo mensaje SMS: " );}
                  else if (msgType == MessageTypes.STATUSREPORT){
                        System.out.println(">>>Nuevo mensaje de status:
");}

                        System.out.println("MENSAJE: "+msg.getText());
                        if(Main.quiereControlSMS)
                        comandos.ejecutaComando(msg.getText());


                        try   // Eliminamos el mensaje de memoria una vez
que llego
                        {
                            srv.deleteMessage(msg);
                        }
                        catch (Exception e)
                        {
                            System.out.println("Error al tratar de
eliminar el mensaje.");
                            e.printStackTrace();
                        }
                }
      }//fin  de la clase InboundNotification


    public class CallNotification implements ICallNotification
      {
            public void process(String gatewayId, String callerId)
            {
```

```
                  System.out.println(">>> LLamada entrante de: " +
callerId);
            }
      } //fin de la clase CallNotification



      public class GatewayStatusNotification implements
IGatewayStatusNotification
      {
            public void process(String gatewayId, GatewayStatuses
oldStatus, GatewayStatuses newStatus)
            {
                  System.out.println(">>> Cambio de STATUS " + ", OLD: "
+ oldStatus + " -> NEW: " + newStatus);
            }
      }

}
```