CONTROL DE ILUMINACIÓN VIA INTERNET

M en C Juan Carlos Herrera Lozada
Profesor - Investigador del CIDETEC IPN
jlozada@ipn.mx
Jesús Alberto Ocotitla Hernández, Tania Alejandra Jaramillo Torres
Alumnos de Ingeniería en Informática de la UPIICSA IPN
alberto_ocotitla@hotmail.com tania_250985@hotmail.com

RESUMEN

En el presente artículo se plantea un sistema de control de iluminación vía Internet, en base a una interfaz hardware de puerto paralelo, que permite activar una carga alterna mediante una etapa de potencia con opto electrónica y un software desarrollado en Visual Basic 6, se utiliza una librería de enlace dinámico desarrollada en lenguaje C que permite enviar y recibir datos del puerto paralelo de la PC. El software de comunicación fue diseñado bajo el paradigma de las aplicaciones Cliente-Servidor, utilizando el protocolo de red TCP/IP, el cliente envía comandos en formato de texto al servidor, este último los interpreta y genera la señal correspondiente que se entrega a la interfaz hardware, logrando así controlar una carga alterna desde una máquina remota, siendo la aplicación en este caso, el control de iluminación. El sistema fue probado satisfactoriamente desde dos clientes remotos y un servidor utilizando la plataforma Windows.

1. INTRODUCCIÓN

Hoy en día la domótica es un tema que está acaparando cada día más atención entre las empresas y universidades del mundo, es uno de los campos que más avances presenta en cortos periodos de tiempo.

Podríamos definir a la domótica como la integración de la tecnología en el diseño inteligente de un recinto. En la domótica convergen varias ramas de la ingeniería, la base de todo el conjunto es la arquitectura, pues es importante crear espacios cómodos y distribuidos de manera inteligente, de igual manera las comunicaciones, los automatismos y el software juegan un papel relevante en los sistemas domóticos.

Actualmente en la mayoría de los recintos, el control de iluminación es automático, las luces se encienden gradualmente en proporción directa a la intensidad luminosa del ambiente. El presente trabajo propone una alternativa al uso de sensores en los sistemas de control de iluminación, y en su lugar ubica a una computadora de propósito general, que le otorga total control al usuario de las luces de un recinto, incluso de manera remota, vía Internet.

2. DESARROLLO

Para llevar a cabo la construcción de este sistema de tomaron en cuenta 4 aspectos relevantes, que delinearon el diseño final.

- El protocolo de comunicación
- La interfaz hardware
- La etapa de potencia
- La interfaz software local y remota

En la figura 1 se muestra el esquema general del proyecto y a continuación se detallará la implementación de cada una de las etapas derivadas de los aspectos relevantes del diseño.

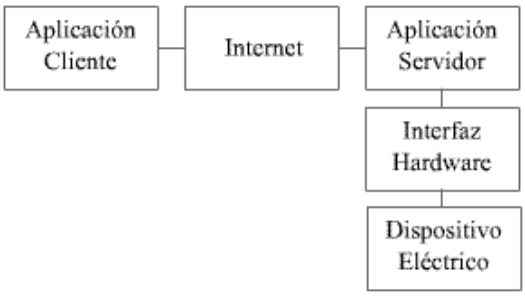

Figura 1. Esquema general del proyecto

2.1. El protocolo de comunicación

Todas las máquinas que están conectadas a Internet tienen asignadas una dirección IP, la dirección IP es un número de 32 bits, que se forma con 4 cifras de 8 bits separadas por un punto, por lo tanto el numero IP máximo que se puede encontrar es: 255.255.255.255, sin embargo, una máquina sin conexión a Internet también tiene una dirección IP, que es

generalmente: 127.0.0.1 a menos que la máquina forme parte de una red de área local.

Al igual que un usuario se comunica con el programa por medio del teclado o algún otro periférico, dos programas se pueden comunicar entre sí por medio de un protocolo de comunicación. Este tipo de programas reciben el nombre de aplicaciones *Cliente/Servidor* y permiten comunicar en consecuencia dos o más computadoras.

Existiendo un programa *cliente* en una computadora en México y otro programa *servidor* en una computadora en Japón, ambos programas se pueden comunicar a través de Internet o de una LAN (Local Area Network). y compartir ó adquirir información, incluso sin que los dueños de las computadoras lo sepan.

Internet usa el protocolo TCP/IP (Transmision Control Protocol / Internet Protocol), es el que se encarga de recibir paquetes de información y redirigirlos al usuario que los solicitó. Este protocolo es el preferido por la mayoría de los desarrolladores de aplicaciones *Cliente-Servidor* ya que posee una característica que UDP (User Datagram Protocol) no tiene, TCP/IP puede verificar que el paquete de información haya llegado con éxito al destinatario, concretando así la transacción de manera fiable.

Por el contrario UDP solo manda el paquete con la información y no verifica que haya llegado satisfactoriamente, poniendo de esta manera en peligro al paquete, ya que puede no llegar entero al destinatario y por lo tanto no sirve si el paquete no llega en su totalidad.

Sin embargo hay aplicaciones en las que el protocolo UDP es más adecuado, por ejemplo: en el monitoreo de la temperatura de una red de boyas a lo largo de una porción de mar. En este caso, los clientes reportan periódicamente la temperatura obtenida de la lectura de los sensores, y la envían al servidor, el cual los muestra en un panel y los almacena en una base de datos, dado que se reporta miles de veces por día la temperatura de una boya, no afecta al desempeño general del sistema que se pierdan algunas lecturas, pues sabemos de antemano que no existiran grandes variaciones en los datos, más aun cuando las lecturas se realizan en lapsos cortos de tiempo. En este ejemplo no se requiere establecer una conexión y esto hace que la transmisión de los datos se haga de manera más rápida aunque implica un riesgo de perdida de datos. TCP/IP requiere de el establecimiento de una conexión previa a la transmisión de datos, lo cual provoca una ligera perdida de velocidad pero garantiza la correcta entrega de los datos.

El sistema propuesto en este artículo, requiere establecer una conexión y tener la certeza de que los datos llegan completos al destinatario, pues como veremos más adelante, los datos que envía el cliente, definirán el comportamiento y los comandos que ha de ejecutar la aplicación servidor, por lo tanto optamos por implementar el sistema utilizando el protocolo de red TCP/IP.

2.2. La interfaz hardware

Como veremos más adelante, la etapa de potencia requiere de una señal digital que controle la carga alterna, es por eso que requerimos de una interfaz hardware que permita de manera sencilla entregar una señal digital a la etapa de potencia.

En el sistema propuesto elegimos el puerto paralelo para construir la interfaz hardware, debido a 2 razones principales:

- Amplia documentación y ejemplos
- Fácil implementación

La tabla 1 muestra la distribución de los registros del puerto paralelo en un conector DB25.

Tabla 1. Correspondencia registros-pines

PINES	REGISTRO
1,14,16,17	Control
2-9	Datos
10-13, 15	Estado
18-25	Masa

Existen otros puertos que podríamos utilizar, tales como el puerto serie o el USB (Universal Serial Bus), pero el puerto paralelo tiene la ventaja de que podemos controlar directamente los 8 bits de su registro de datos, y estos bits no cambian su estado hasta que se modifiquen por software, esto nos permite controlar la iluminación de hasta 8 focos al mismo tiempo. No utilizamos los 5 bits del registro de estado en esta aplicación, pero se podría implementar una etapa adicional de adquisición de datos, para asi tener un sistema de

control de lazo cerrado, ya que en el presente trabajo, la aplicación cliente no puede saber si en realidad se encendió el foco.

Las figuras 2 y 3 muestran la distribución de los pines en los conectores correspondientes.

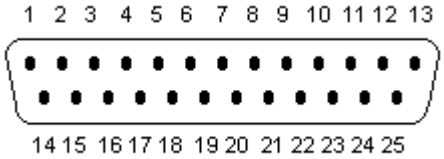

Figura 2. Distribución de los pines en un conector DB25 macho

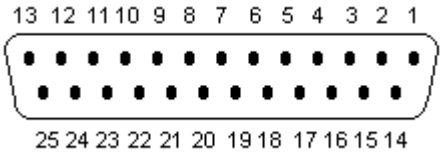

Figura 3. Distribución de los pines en un conector DB25 hembra

Para la construcción de la interfaz utilizamos un cable de puerto paralelo macho a hembra, ya que tenemos un conector DB25 hembra en la placa base de la PC y un conector DB25 macho en la tarjeta que construimos. Utilizamos 8 LEDs verdes, para poder apreciar el estado del registro de datos, de los 8 bits del registro de datos, solo ocupamos uno para entregar a la etapa de potencia.

La figura 4 muestra el aspecto final de la interfaz hardware de puerto paralelo que construimos.

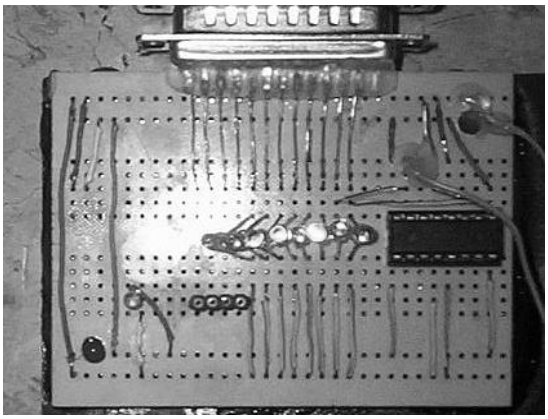

Figura 4. Interfaz hardware de puerto paralelo

2.3. La etapa de potencia

Las instalaciones eléctricas en las casas utilizan corriente alterna a 127 v, por lo tanto en el diseño se incluyó una etapa de potencia a base de opto electrónica que nos permite controlar una carga alterna, y así poder encender o apagar un foco. La razón fundamental para llevar a cabo acoplamiento óptico y aislamiento eléctrico es por protección de la etapa digital ya que si ocurre un corto en la etapa de potencia el optoacoplador protege toda la circuitería digital, de lo contrario, podríamos causar daños en la placa madre de la computadora.

Para controlar la etapa de potencia utilizamos el bit 1 del registro de datos, correspondiente al pin 2 del conector.

La figura 5 muestra el circuito que utilizamos como base para el control de la carga alterna, se requiere un montaje por cada foco que se desee controlar.

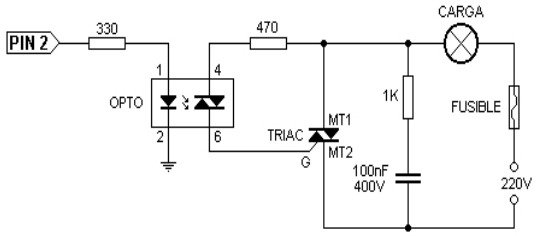

Figura 5. Circuito para controlar una carga alterna

Para la construcción de nuestra etapa de potencia cambiamos algunos valores de los componentes, a continuación se mencionan los componentes utilizados.

- Optoacoplador MOC3011
- Triac MAC228
- Resistencia 4.7 K a 5 W
- Resistencia 330 Ohms a ½ W
- Foco 100 W
- Tablilla fenólica

La serie de MOC30X son dispositivos de TRIAC aislados de forma óptica. Estos dispositivos contienen un diodo infrarrojo y una luz activada de silicón con un switch bilateral, que funciona como un TRIAC. Son interfaces entre controles digitales y TRIAC de potencia para controlar cargas alternas.

El Optoacoplador MOC3011 viene en un encapsulado DIP de 6 pines, de los cuales solo se conectan 4 tal como lo muestra la figura 6.

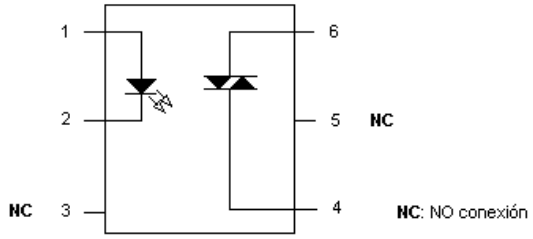

Figura 6. Configuración de pines del MOC3011

El MAC228, es un componente que realiza una función análoga al tiristor, pero para corriente alterna. Al igual que el tiristor tiene dos estados de funcionamiento: bloqueo y conducción. El paso de bloqueo al de conducción se realiza por aplicación de un impulso de corriente en la puerta, y el paso del estado de conducción al de bloqueo por la disminución de la corriente por debajo de la corriente de mantenimiento.

El TRIAC conduce la corriente entre los dos ánodos (A1 y A2) cuando se aplica una señal a la puerta (G) en un sentido o en el inverso, por ello, es un dispositivo bidireccional, el encapsulado del triac es idéntico al de los tiristores, el MAC228 tolera 8 Amperes a 800 V.

La figura 7 muestra el símbolo y la configuración de los pines del MAC228.

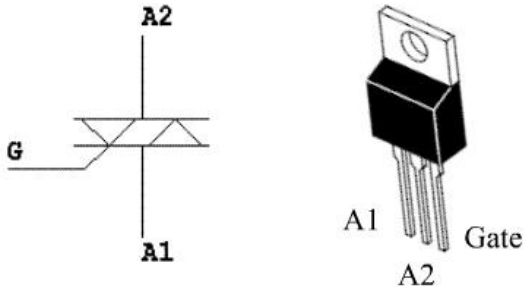

Figura 7. Símbolo y configuración del MAC228

En el diseño original, el TRIAC que elegimos fue el MAC223, pues el MAC228 queda sobrado para los 127 V de una instalación eléctrica común, pero no encontramos uno de menor capacidad. La figura 8 muestra la tarjeta que diseñamos y que nos permite controlar un foco utilizando la señal proveniente del pin 2 del puerto paralelo.

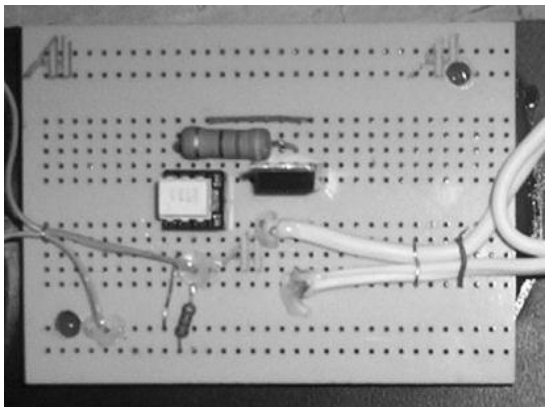

Figura 8. Etapa de potencia

Para que fuese fácil de transportar, optamos por ubicar todo el conjunto sobre una base de plástico, tal como se muestra en la figura 9, que nos muestra la interfaz de puerto paralelo y la etapa de potencia.

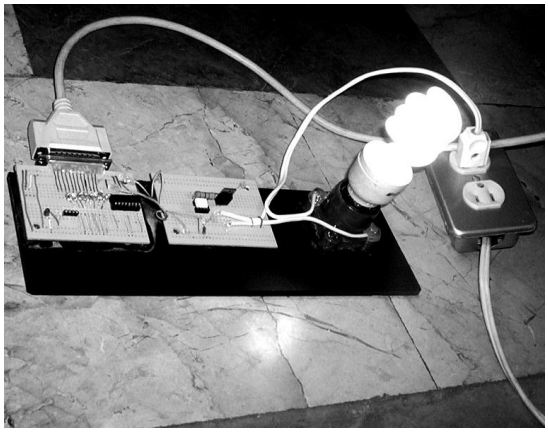

Figura 9. Interfaz hardware y etapa de potencia

2.4. La interfaz software local y remota

En el aspecto del software necesitamos poder enviar datos a través del puerto paralelo, Visual Basic no tiene rutinas propias para hacer esto, por lo tanto al igual que sucede con otros lenguajes interpretados como JAVA, se tiene que utilizar una librería de enlace dinámico (DLL) desarrollada en un lenguaje compilado como C, para el caso particular se utilizó la IO.DLL, que es gratuita, y se puede bajar de la siguiente dirección:

http://www.geekhideout.com

Para utilizar las funciones de la librería, se agrega un modulo al proyecto de VB, en donde se declaran las funciones, además se debe ubicar el archivo IO.DLL en el directorio de sistema de la máquina.

A continuación se muestra la declaración de las funciones que utilizamos de la IO.DLL.

```
‘ Función que permite poner a 1 el bit que queramos
‘ del puerto especificado
Public Declare Sub SetPortBit Lib "IO.DLL" (ByVal Port As Integer, ByVal Bit As Byte)

‘ Función que permite poner a 0 el bit que queramos
‘ del puerto especificado
Public Declare Sub ClrPortBit Lib "IO.DLL" (ByVal Port As Integer, ByVal Bit As Byte)

‘ El puerto al que vamos a leer o escribir
Public Out_Port As Integer
```

Y ahora la utilización de las funciones desde un formulario, en donde establecemos el puerto a leer o escribir, y dos rutinas, una para encender el foco y otra para apagarlo.

```
‘ Establece la dirección del registro de datos del puerto
‘ paralelo
Const Out_Port = &H378

‘ Pone a 0 el bit 2 del registro de datos
Sub Apagar_Foco()
ClrPortBit Out_Port, 2
End Sub

‘ Pone a 1 el bit 2 del registro de datos
Sub Encender_Foco()
SetPortBit Out_Port, 2
End Sub
```

El lenguaje que elegimos para el desarrollo del software de comunicación, también fue Visual Basic 6, debido a que existen muchos ejemplos en Internet y se proporciona el control WINSOCK con la instalación del Visual Studio, que nos permite fácilmente implementar comunicaciones utilizando el protocolo de red TCP/IP.

El control Winsock no está en la barra de controles del IDE de Visual Basic, para agregarlo manualmente debemos ir al menú “Proyecto” y dar clic sobre la opción “Componentes”, así se nos muestra una lista de todos los componentes registrados, escogemos “Microsoft WinSock Control” y Aceptamos.

Antes de profundizar en los aspectos de la programación del control Winsock, vamos a mencionar algunas características de las aplicaciones Cliente-Servidor:

- El cliente obtiene servicios del servidor
- El servidor proporciona un puerto al que debe conectarse el cliente
- Un SOCKET es el extremo de un enlace bidireccional
- Se utiliza un SOCKET en el cliente y otro en el servidor
- El cliente es quién inicia la comunicación entre las dos máquinas.
- El servidor es quien acepta las peticiones de conexión
- Puede haber varios clientes para un único servidor

Ahora que ya tenemos una noción de la forma en que se comunican este tipo de aplicaciones vamos a examinar las diferentes propiedades, métodos y eventos que nos proporciona el control Winsock.

Principales propiedades:

- **LocalIP**: Propiedad que devuelve el número IP de la máquina local.
- **LocalPort**: Propiedad que establece el puerto que se deja a la escucha cuando se esta en modo servidor.
- **RemotePort**: Propiedad que establecer el puerto al cual vamos a conectarnos estando en modo cliente

.

Principales métodos:

- **Listen:** Método que pone a la escucha de una petición de conexión al puerto local.
- **Accept**: Método que acepta una petición de conexión.
- **GetData**: Método por el cual obtenemos datos a través de la conexión.
- **SendData**: Método por el cual enviamos datos a través de la conexión.

Principales eventos:

- **ConnectionRequest:** Evento que se produce cuando un cliente solicita una conexión al servidor.
- **Connect:** Evento que se produce cuando el equipo local se conecta al equipo remoto.

- **Close:** Evento que se produce cuando el equipo remoto cierra la conexión.
- **DataArrival:** Se produce este evento cuando llegan los datos que se enviaron a través de la conexión.

La figura 10 muestra la relación existente entre las propiedades, métodos y eventos del control Winsock, al utilizarlo en modo cliente y en modo servidor.

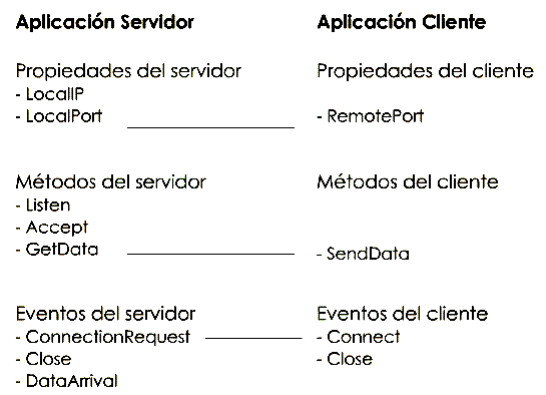

Figura 10. Relación entre propiedades, métodos y eventos del control Winsock

A continuación se muestra el código necesario para establecer una comunicación:

Del lado del servidor:

```
‘ Asigna un puerto y lo deja a la escucha
Sub Aceptar_Control_Remoto()
Winsock1.LocalPort = 888
Winsock1.Close
Winsock1.LocalPort = 888
Winsock1.Listen
End Sub

‘ Acepta una petición de conexión cuando se produzca
Private Sub Winsock1_ConnectionRequest(ByVal requestID
As Long)
Winsock1.Close
Winsock1.Accept requestID
End Sub

‘ Al llegar datos nuevos los envía a una variable tipo cadena
Private Sub Winsock1_DataArrival(ByVal bytesTotal As
Long)
Dim Mensaje As String
Winsock1.GetData Mensaje
Ens Sub

‘ Termina una conexión
Sub Cancelar_Control_Remoto()
Winsock1.Close
End Sub
```

Del lado del cliente:

```
‘ Establece un puerto remoto y una IP para intentar conexión
Private Sub Iniciar_Conexión()
On Error GoTo Error_Al_Conectar
Winsock1.RemotePort = 888
Winsock1.Close
Winsock1.RemotePort = 888
Winsock1.Connect 192.168.10.102, 888
Exit Sub
Error_Al_Conectar:
MsgBox "Imposible Conectar", vbExclamation, "Error"
Winsock1.Close
End Sub


‘ Envía la cadena “Hola Mundo” al equipo remoto
Private Sub Enviar_Datos()
On Error GoTo fallo
Winsock1.SendData "Hola Mundo"
Exit Sub
fallo:
MsgBox "No hay conexión", vbExclamation, "Error"
Winsock1.Close
End Sub

‘Finaliza la conexión
Private Sub Terminar_Conexión()
Winsock1.Close
End Sub
```

Como hemos visto, la comunicación entre dos máquinas remotas es fácil de implementar utilizando el control Winsock y gracias a que este control soporta el protocolo de red TCP/IP podemos tener la certeza de que los datos llegarán con éxito al destinatario.

Para los fines de este sistema, se agregó una etapa condicional en el evento *DataArrival* del servidor, así al reconocer una cadena el programa ejecuta la rutina correspondiente. Para agregar seguridad, se incluyó una etapa de autenticación en el cliente, lo cual nos brinda la certeza de que la aplicación servidor solamente tomará en cuenta aquellas ordenes de un cliente conocido.

```
‘ Al llegar datos busca que rutina ejecutar y verifica
‘ que la aplicación cliente se haya identificado
Private Sub Winsock1_DataArrival(ByVal bytesTotal As
Long)
Dim Mensaje As String
Winsock1.GetData Mensaje
If Contraseña _ valida = True then
        If Mensaje = “OFF” then Apaga_Foco
        If Mensaje = “ON” then Enciende_Foco
Else
        If Mensaje = “Contraseña” then
                Contraseña _ valida = True
        End if

End if
End Sub
```

3. RESULTADOS

Se diseñó y construyó un circuito que permite controlar una carga alterna mediante una señal digital, también se desarrolló un software que permite activar esta carga alterna de manera programada según los horarios que el usuario elija y guardar estos horarios, de tal manera que el software recuerda los horarios cuando se corta el suministro de energía eléctrica. Se garantiza que en todo momento este activo el sistema, pues la aplicación servidor se inicia junto con el sistema operativo.

El software emplea un protocolo de red seguro, además de incluir una etapa de autenticación en la aplicación cliente, de esta forma tenemos una conexión fiable que permite controlar la iluminación de un recinto vía Internet.

Para lograr que el sistema active la carga alterna a cierta hora del día se requiere utilizar el control TIMER, este control se encuentra en la barra de controles del IDE de Visual Basic, y nos permite ejecutar rutinas cada determinado tiempo, por ejemplo si quisiéramos que el sistema nos muestre un mensaje a las 2 de la tarde el código seria el siguiente:

```
' Cuando se carga el formulario establece el intervalo del timer
' a medio segundo
Private Sub Form_Load()
Timer1.Interval = 500
End Sub

' Cada medio segundo el control
' verifica la hora, si son las 2
' de la tarde, entonces muestra el mensaje
Private Sub Timer1_Timer()
If Time = "02:00:00 p.m." Then MsgBox "Ya son las 2"
End Sub
```

Para guardar los horarios y recuperarlos cuando se inicie la aplicación, se utilizó el control RICHTEXTBOX, que nos permite guardar cadenas de texto de manera sencilla, a continuación un ejemplo que guarda una cadena y luego la recupera.

```
' Al dar clic en el boton Guardar
' se asigna una cadena al RichTextBox
' y se guarda la cadena en la ruta especificada
Private Sub Guardar_Click()
RichTextBox1.Text = "Esto es una prueba"
RichTextBox1.SaveFile "C:/prueba.dato"
End Sub

' Al dar clic en el boton Recuperar
' se carga en el RichTextBox el archivo
' especificado en la ruta y se muestra en un mensaje
Private Sub Recuperar_Click()
RichTextBox1.LoadFile "C:/prueba.dato"
MsgBox RichTextBox1.Text
End Sub
```

Este control no se encuentra en la barra de controles del IDE de Visual Basic, por lo tanto hay que cargarlo como se hizo con el control WINSOCK.

A continuación se describe el software, la figura 11 muestra la pantalla principal de la aplicación *Servidor*.

Figura 11. Pantalla principal del *Servidor*

El control manual nos permite encender o apagar un foco al presionar un botón, así como activar o desactivar el control programable.

El control programable nos permite establecer hasta 5 horarios en los cuales el foco debe apagarse o encenderse.

Esta pantalla también permite acceder a la configuración del control vía Internet. El control remoto es opcional, pues al poder establecer

horarios la utilización del control vía Internet se reserva para cuando sea realmente necesario.

La figura 12 muestra la pantalla de configuración del control vía Internet en la aplicación *Servidor*.

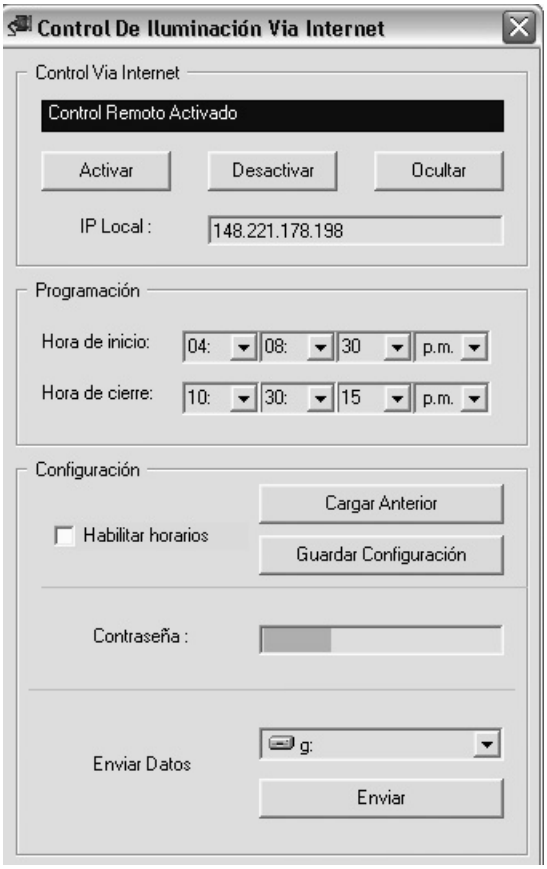

Figura 12. Pantalla de configuración del control vía Internet en la aplicación *Servidor*

El primer panel nos permite activar o desactivar el control vía Internet al presionar un botón, de igual manera nos muestra la IP local.

El siguiente panel nos permite establecer un horario en el cual se inicia el control vía Internet y otro horario en el cual se termina.

El panel de configuración nos permite activar o desactivar los horarios, establecer la contraseña y guardar la configuración, así como enviar la contraseña y la IP a una memoria USB para poder autenticarnos y controlar la iluminación desde una máquina remota.

La figura 13 muestra la pantalla de la aplicación cliente tal como se ve al iniciar el programa.

Figura 13. Pantalla de la aplicación *cliente*

Los pasos para controlar el foco desde la aplicación cliente son:

- Elegir la unidad de la memoria USB en la cual tenemos los datos de conexión.
- Clic en el botón cargar, si en la unidad seleccionada no están los datos el sistema da un mensaje de error.
- Presionar el botón de Iniciar la conexión
- Presionar el botón de autenticar, si no hay conexión a Internet o la aplicación servidor no tiene activo el control vía Internet el sistema mostrara un error.
- Ahora podemos apagar o encender el foco.

La figura 14 muestra la ventana del cliente cuando se han cargado los datos de conexión correctamente.

Figura 14. Se han cargado los datos desde la memoria USB

La figura 15 muestra la ventana del cliente cuando se ha presionado el botón de autenticar y se ha realizado correctamente. La figura 16 muestra la ventana cuando se ha presionado el botón de encender el foco, si se presiona este botón antes de identificarse o antes de iniciar la conexión se muestra un mensaje de error.

Figura 15. Autenticación

Figura 16. Encendiendo el foco

En la versión final de la aplicación cliente, la autenticación se realiza cuando se inicia la conexión, de igual manera en la versión final los datos de conexión no son visibles al usuario, sin embargo se muestran en este trabajo con la finalidad de tener mayor claridad en el funcionamiento del software.

El funcionamiento del sistema se puede apreciar en un video que muestra el funcionamiento del software así como de la interfaz hardware y la etapa de potencia en la siguiente dirección de Internet:

http://www.youtube.com/watch?v=U-9SzV6Czy4

4. CONCLUSIONES

Utilizando como base este programa, podemos tener infinidad de rutinas que podemos llamar a ejecución desde una máquina remota. Una de las aplicaciones posibles sería el telecomando de robots vía Internet.

El presente trabajo demuestra la utilidad de una formación interdisciplinaria, al integrar conocimientos que tradicionalmente se considerarían aislados, como son: las redes, la electrónica y el desarrollo de software

El desarrollo de este sistema, es muy económico, y es un paso en dirección de la independencia tecnológica del país en cuanto a las aplicaciones existentes en la domótica.

El utilizar una computadora permite definir el comportamiento del sistema, pues se puede establecer que se comporte como un automatismo, o bien como un mando a distancia, dada la naturaleza computacional de la propuesta, el dispositivo remoto puede hacer uso de las nuevas tecnologías de cómputo móvil, y con esto, manipular la iluminación desde una PDA o un teléfono inteligente, esto sin importar la ubicación de la persona, siempre y cuando se encuentre dentro del radio de una red inalámbrica (WLAN).

El sistema desarrollado tiene buenas posibilidades de impactar positivamente en el nivel de calidad de vida de las personas con alguna discapacidad.

5.BIBLIOGRAFÍA

[1] Motorola, Data sheets MAC228
[2] Motorola, Data sheets MOC3011
[3] http://www.geekhideout.com/iodll.shtml
[4] http://www.elguruprogramador.com.ar