



INSTITUTO POLITÉCNICO NACIONAL

**ESCUELA SUPERIOR DE INGENIERÍA
MECÁNICA Y ELÉCTRICA**

**SISTEMA DE MONITOREO DE
FALLAS EN CAMARAS IP**

TESIS

**QUE PARA OBTENER EL TITULO DE:
INGENIERO EN COMUNICACIONES Y ELECTRÓNICA**

PRESENTAN:

**RAFAEL GUZMÁN GARCÍA
GABRIEL HERNÁNDEZ HERRERA**

ASESORES:

**ING. JUAN CARLOS TORRES VILLASÁNCHEZ
M. en C. DAVID VÁZQUEZ ÁLVAREZ**



MÉXICO, D.F

2013

INSTITUTO POLITÉCNICO NACIONAL
ESCUELA SUPERIOR DE INGENIERÍA MECÁNICA Y ELECTRICA
UNIDAD PROFESIONAL "ADOLFO LÓPEZ MATEOS"

TEMA DE TESIS

**QUE PARA OBTENER EL TÍTULO DE
POR LA OPCIÓN DE TITULACIÓN
DEBERA(N) DESARROLLAR**

INGENIERO EN COMUNICACIONES Y ELECTRÓNICA
TESIS COLECTIVA Y EXAMEN ORAL INDIVIDUAL
C. RAFAEL GUZMAN GARCIA
C. GABRIEL HERNANDEZ HERRERA


"SISTEMA DE MONITOREO DE FALLAS EN CÁMARAS IP"

ELABORAR UN SISTEMA DE MONITOREO PARA CÁMARAS IP, QUE PERMITAN DETECTAR FALLAS DE IMAGEN Y DE COMUNICACIÓN EN ÉSTAS; REDUCIENDO COSTOS DE OPERACIÓN Y MANTENIMIENTO, MAXIMIZANDO TANTO LA EFICIENCIA COMO LA RENTABILIDAD DE ESTAS CÁMARAS CONECTADAS EN UN SISTEMA DE RED.

- VIDEO VIGILANCIA EN LA ACTUALIDAD
- SISTEMA DE MONITOREO
- PLANTEAMIENTO DE LA APLICACIÓN
- DESARROLLO DE LA APLICACIÓN.

MÉXICO D.F. A 29 DE ENERO DE 2013.

ASESORES


ING. JUAN CARLOS TORRES VILLASANCHEZ


M. ENC. DAVID VAZQUEZ ÁLVAREZ



M. EN C. DAVID VÁZQUEZ ÁLVAREZ
JEFE DEL DEPARTAMENTO ACADÉMICO DE
INGENIERÍA EN COMUNICACIONES Y ELECTRÓNICA

Agradecimiento

A:

A Dios por dejarme vivir este momento, por haberme dado la fuerza y la inteligencia de poder alcanzado esta meta.

A mis Padres: Rafael Guzmán Fuentes y María Matilde García González por todo el apoyo y cariño que me brindaron en este largo camino.

A mis Hermanos: Arturo Guzmán García, Eduardo Guzmán García y Jorge Guzmán García, por apoyar en todos aquellos momentos difíciles por los que pasado para poder conseguir esta meta.

A mi Tío: Pedro Guzmán Fuentes por todo ese gran apoyo en los malos y buenos momentos que me impulsaron para poder conseguir este gran sueño.

A mi hijo: Axel André Guzmán Soto le agradezco la paciencia el amor las sonrisas que fueron el motor que me impulsaron todos los días para poder seguir siempre adelante.

A mis amigos: Alejandro, Adonis, Gabriel, Michel, Bernardo, Luis, Manuel, les doy gracias a todos por compartir los buenos y malos momentos, y no solo les dedico esta tesis sino todos mis logros, ya que sin su apoyo no hubiese podido terminar mis estudios y así mismo terminar una carrera profesional.

Agradecimiento

A:

Mi madre Silvia Herrera, por darme la vida, quererme mucho, creer en mí y porque siempre me apoyaste. Mamá gracias por darme una carrera para mi futuro.

Mi padre Norberto Hernández, por los ejemplos de perseverancia y constancia que lo caracterizan y que me ha infundado siempre, por el valor mostrado para salir adelante y por tu amor.

Mis abuelos Estela López (QEPD), Ignacia Oledo, Magdaleno Herrera y Samuel Hernández (QEPD), por quererme y apoyarme siempre, esto también se lo debo a ustedes.

Mis hermanos, Carolina Hernández y Norberto Hernández, por estar conmigo y apoyarme siempre, los quiero mucho.

Mi sobrino, Rafael Hernández, para que veas en mí un ejemplo a seguir.

Todos mis amigos, Rafael, Daniel, Abraham, Rosario, Luis Antonio, Adrián, Ricardo, Armando, Julio por compartir los buenos y malos momentos.

Todos aquellos familiares y amigos que no recordé al momento de escribir esto.

CONTENIDO

OBJETIVO	3
JUSTIFICACIÓN	4
CAPITULO I Video Vigilancia en la Actualidad	5
1.1. ¿Qué es una Cámara IP?	6
1.1.2. Los Sistemas de Cámaras IP	6
1.2. Seguridad en la Sociedad	6
1.2.1. Video Vigilancia en el Metro del D.F.	7
1.3. Sistemas de Seguridad dentro de Instituciones Educativas	8
1.4. Video Digital sobre IP	9
1.5. Modelo de Referencia OSI	10
1.5.1. Antecedentes del Modelo OSI	10
1.5.2. Capas del Modelo OSI	10
1.6. Capa de Red	12
1.6.1. Direcciones IP	13
CAPITULO II Sistema de Monitoreo	15
2.1. Monitoreo de Redes de computadoras	16
2.2. Protocolo SNMP (Simple Network Management Protocol)	17
2.2.1. Necesidad de Administrar una Red	18
2.2.2. Definición de las Variables	19
2.2.3. Componentes Básicos	20
2.2.4. Comandos Básicos	20
2.2.5. Base de Información de Administración (MIB)	21
2.2.6. Mensajes SNMP	21
2.2.7. Trap	24
2.2.8. Arquitectura del Servicio	25

CAPITULO III	Planteamiento de la Aplicación	27
3.1.	Software utilizado para el desarrollo de la aplicación	28
3.1.1.	Procesamiento de imagen	28
3.1.2.	Generación de la aplicación	30
3.1.3.	Base de datos	31
3.1.4.	Gestión de red	33
3.2.	Plan de desarrollo de la aplicación	34
CAPITULO IV	Desarrollo de la aplicación	36
4.1.	Pantalla de bienvenida	37
4.2.	Acceso al usuario	38
4.3.	Menú Principal	41
4.4.	Reproductor de video	42
4.5.	Escaneo de imagen	45
4.6.	Gestión de red	48
4.7.	Reportes	49
4.8.	Navegador web	53
CONCLUSIONES		56
GLOSARIO TECNICO		57
ANEXO		58
BIBLIOGRAFIA		80

Objetivo:

Elaborar un sistema de monitoreo para cámaras IP, que permitan detectar fallas de imagen y de comunicación en éstas; reduciendo costos de operación y mantenimiento, maximizando tanto la eficiencia como la rentabilidad de estas cámaras conectadas en un sistema de red

Objetivos particulares:

Implementar una aplicación que permita comprobar errores de imagen en cámaras IP.

Implementar una aplicación que compruebe la comunicación de las cámaras que se encuentren dentro del sistema de red.

Generar una aplicación para la captura de reportes de fallas de las cámaras IP.

Justificación:

Actualmente las cámaras IP son una gran herramienta para sistemas de seguridad e implementaciones en la industria con sistemas inteligentes aplicando visión por computadora para la revisión de piezas.

Este proyecto se propuso debido a que al momento de estar realizando el servicio social en el Sistema de Transporte Colectivo “Metro” se observó que en las cámaras IP que utiliza esta institución, se presentaban diversas fallas. Para poder realizar una corrección a estas fallas, se supervisa cada cámara individualmente y poder observar si estaban presentes estas fallas; posteriormente se genera un reporte de error para cada cámara, y al ser un proceso administrativo, se tienen que realizar otros reportes en distintas áreas; esto para el seguimiento de la reparación de las mismas.

Al implementar este proyecto se pretende evitar el proceso de monitoreo de cada cámara, ya que es un proceso bastante tardado y que puede generar algunos problemas a futuro, las cámaras con que cuenta el STC deben presentar un buen funcionamiento en caso que se llegue a presentar algún incidente, se pueda responder de manera inmediata a éste, realizar una buena vigilancia y hacer sentir seguro al usuario.

CAPÍTULO

1

VÍDEO VIGILANCIA EN LA ACTUALIDAD

1.1. ¿Qué es una cámara IP?

Una Cámara IP es una cámara diseñada especialmente para enviar las señales de video (en algunos casos audio) a través de Internet o una red de área local (LAN) desde un explorador de internet.

En las cámaras IP se pueden integrar aplicaciones tales como la detección de presencia y grabación de imágenes, de manera que al momento de realizar una detección de presencia, se realice una grabación de imágenes de lo sucedido.

Debido a un importante avance en diversas ramas como la informática, las telecomunicaciones, las redes electrónicas y las tecnologías de multimedia estas han tenido un gran impacto en la actividad humana y la necesidad de automatizar procesos que actualmente se desarrollan manualmente, también buscar nuevas y diversas alternativas que puedan ser aplicables al medio, debido a tales necesidades se plantea el presente tema de investigación.

1.1.2. Los sistemas de cámaras IP

Funcionan parecido a los sistemas convencionales de CCTV (circuito cerrado de tv) con la diferencia que las imágenes pueden ser grabadas en la memoria de un procesador (hasta con 90 días de duración) y ser vistas por la central de monitoreo en tiempo real (generalmente el operador habré el sitio cuando el sistema de alarmas genera un evento fuera de lo común (desactivación de un comercio a horas de la madrugada), o en caso que el sistema genere eventos de robo o asalto como así también de incendio .

1.2. Seguridad en la sociedad

Actualmente existe en la sociedad un gran problema llamado *inseguridad*, es un problema sistémico e integral más que un problema de falta de vigilancia, o causado debido a la falta de seguridad misma.

Actualmente este problema recae en gran medida en la vigilancia pública y privada que se realiza tanto en diversos lugares ya sea en forma externa como interna. En el caso de la vídeo vigilancia esta puede ser llevada por diversos métodos como un circuito cerrado de televisión, programas de reconocimiento facial, sensores de proximidad, cámaras infrarrojas, cámaras robots, secuenciadores de vídeo, cámaras de intemperie con radiofrecuencia, cámaras de baja iluminación, en total oscuridad, de interiores visibles u ocultas, cámaras acuáticas.

Estos sistemas de seguridad se han implementado en diversos lugares tales como cajeros automáticos, tiendas departamentales, centros de entretenimiento, bancos,

escuelas, cárceles, calles, plazas, tráfico vehicular, seguridad infantil, medio ambiente, hospitales, empresas y casas, estos pueden ser implementados en “cualquier lugar que lo requiera”.

Debido a que actualmente ha habido un gran aumento en la inseguridad, la sociedad ha tenido que recurrir a diversos servicios que les brinden mayor protección, uno de los más requeridos es el sistema a través de cámaras de vídeo que se ha mantenido desarrollando en gran medida, comenzando con los circuitos cerrados de televisión hasta terminar actualmente con las cámaras IP.

Estos sistemas de vigilancia se están volviendo muy comunes en edificios de oficinas, escuelas e incluso en las calles. La vigilancia se ha convertido en un componente integral de los medios de control de acceso, enriquecidos con sistemas biométricos y sistemas de rastreo.

1.2.1. Vídeo vigilancia en el metro del D.F.

Hoy en día el Metro es uno de los medios de transporte con mayor demanda entre de parte de la población capitalina, esto conlleva que también sea un lugar con alto índice de delincuencia en sus diversas estaciones a lo largo de su red. Esto llevo al gobierno capitalino a implementar un sistema que prevenga y pueda controlar dicho problema y con esto lograr proteger a los usuarios.

El Sistema de Transporte Colectivo Metro cuenta actualmente con una infraestructura de aproximadamente 2500 cámaras que integran el sistema de vídeo vigilancia en las estaciones de mayor con mayor demanda. Además se tiene el proyecto de instalación de fibra óptica para la colocación de las cámaras en el interior de las instalaciones del transporte subterráneo. Para ello se requiere que en varias de las estaciones del Metro se realice la instalación de esta fibra óptica para colocar el equipo y transportar la imagen.

El sistema de vídeo vigilancia del Metro está conectado al centro de mando “C4” de la Secretaría de Seguridad Pública para atender diversos tipos de situaciones de emergencia.

A futuro se planea que las cámaras cuenten con un micrófono para mantener comunicación con los usuarios esto para hacer señalamientos cuando se presentarse alguna situación de peligro o cuando realicen actividades de vandalismo tales como grafiti, desórdenes y otros.

1.3. Sistemas de seguridad dentro de instituciones educativas

Con el fin de que la comunidad de la UNAM (Universidad Nacional Autónoma de México) pueda observar el avance de las obras de la construcción del edificio 18, se puso en operación una página Web comunicada con una cámara de vídeo IP.

Esta página Web puede ser consultada desde cualquier computadora conectada a la red del Instituto ya sea por interfaz alámbrica o inalámbrica. También se puede consultar desde cualquier lugar con servicio de Internet con una cuenta de usuario válida.

La cámara que se utiliza en la UNAM esta conecta directamente a la red de datos y utiliza los protocolos de comunicación de Internet (TCP/IP: Protocolo de Control de Transmisión/Protocolo de Internet) para transmitir el vídeo a un servidor, que a su vez es consultado a través de una página Web. La cámara también ofrece rutas programadas, paradas predefinidas para realizar acercamiento (*zoom*) hacia algún detalle en particular que se requiera revisar, fotografiar o grabar en vídeo; ofrece visión nocturna y generación de alarmas con la ayuda de sensores, puede girar en su eje vertical y también en el horizontal y maneja diferentes algoritmos de compresión para usar un menor ancho de banda de la red de datos. Esta solución es muy flexible y económica, a diferencia de los sistemas con cámaras analógicas y cable coaxial, ya que este tipo de cámaras IP pueden ser conectadas en cualquier punto de una red de datos tanto alámbrica como inalámbrica y pueden ser controladas y administradas desde cualquier punto donde exista servicio de Internet.

Al igual que la UNAM, el Instituto Politécnico Nacional y diferentes empresas invirtieron recursos para comprar cámaras de vídeo vigilancia que serán instaladas en escuelas públicas del DF, de las cuales se incluyen desde el nivel básico hasta nivel medio superior. Dicho proyecto consiste en implementar una red de vigilancia con el objetivo de cubrir la vigilancia en las inmediaciones de planteles educativos, verificar los problemas de narcomenudeo, maltrato y abuso contra los estudiantes.

En la mayoría de las escuelas, ya sea del nivel medio superior o del nivel superior y áreas administrativas del Instituto Politécnico Nacional actualmente no se cuenta con este tipo de sistemas de vídeo vigilancia que ofrezcan seguridad a la población estudiantil, docente, personal administrativo y de los bienes materiales.

Existen otras áreas del Instituto, tal como el edificio inteligente donde se cuenta con equipo de seguridad conformado por cámaras de vídeo. Sin embargo debido al problema de inseguridad que se tiene, tanto interna como externamente se ha solicitado apoyo a diferentes instituciones gubernamentales para instalar este tipo de equipos en todas sus unidades.

1.4. Vídeo digital sobre IP

La característica Plug and Play permite a las cámaras IP direccionables ser colocadas en cualquier lugar dentro de la infraestructura. Los equipos electrónicos que manejan actualmente tráfico IP se han vuelto parte integral de los sistemas de vigilancia. Ya que los vídeos se almacenan en formato digital, pueden ser vistos en cualquier lugar de la red con nuevas capacidades de seguridad para los archivos administrados como parte de las políticas de seguridad de la red. Además, éstos pueden ser vistos simultáneamente desde varios puntos de la red. No sólo es fácil de implementar, sino también es extremadamente versátil y las redes no son sobrecargadas con otro protocolo.

Las transmisiones son "nativas" esto elimina la necesidad de sistemas de cableado separado. TCP/IP (*Protocolo de Control de Transmisión/Protocolo de Internet*) se ha convertido en el estándar para las redes. Su arquitectura abierta permite que varios sistemas puedan compartir el espacio de red y aprovechar estas nuevas tecnologías para aumentar su capacidad, confiabilidad, escalabilidad y accesibilidad de los recursos de red. Con la habilidad de utilizar la infraestructura existente, un edificio puede volverse totalmente automatizado utilizando un sistema de cableado. Esta automatización puede incluir además del CCTV, control de accesos, sistemas de fuego, sistemas de seguridad, voz, y tráfico de red. Los administradores y los usuarios de la red no estarán en un puesto de control y/o administración de estos sistemas ya que puede realizarse desde cualquier estación de trabajo con acceso a la red y esto también aplica para el personal de seguridad. La cámara digital se vuelve ahora el punto de falla, no el centro de control, ya que es fácil hacer redundancia en los servidores.

Un sistema de CCTV basado en el protocolo IP es completamente diferente de las otras soluciones. Las cámaras, servidores de vídeo y teclados IP pueden colocarse en cualquier punto. Los teclados IP pueden controlar actualmente las funciones PTZ (*Pan, Tilt and Zoom; Vista Panorámica, Inclinación y Ampliación*) de cualquier cámara con base en su dirección IP. Como cualquier protocolo IP, las funciones de administración son incorporadas en la transmisión. Esto incluye DSP (*Procesamiento de Señales Digitales*), manejo de alarmas, grabación, capacidades de búsqueda y/o archivo, calendarización y automatización. Estas funciones de administración y control utilizan SNMP (*Protocolo de Manejo de Red Simple*) y otros cuadros de control, todas ellas parte del estándar IP.

Estas cámaras pueden equiparse con características avanzadas tales como sensores de movimiento, PTZ automatizado y salidas análogas de vídeo.

1.5. Modelo de referencia OSI (interconexión de sistema abierto)

1.5.1. Antecedentes del modelo OSI

En 1979, ISO (*Organización Internacional para la Estandarización*) definió su modelo de arquitectura de red OSI (Interconexión de sistemas abiertos). Este modelo fue adoptado en 1980 por el CCITT (*Comité Consultivo Internacional de Telefonía y Telegrafía*) en su recomendación X.200.

La comunicación de datos comprende 2 aspectos principales:

- *Transporte*: involucra todas las funciones relacionadas con la transferencia de datos entre dos usuarios.
- *Manipulación*: los datos deben ser liberados en una forma inteligible. En algunos casos los datos deben ser convertidos.

1.5.2. Capas del modelo OSI

Las redes de computadoras, proveen al usuario de una serie de servicios, e internamente poseen funciones. Las cuales son realizadas por las capas o niveles de la arquitectura que posee el tipo de red. Las arquitecturas de las redes tienen una serie de capas superpuestas (una encima de otra), en la que cada una desempeña su función.

Las funciones y características de las capas son las siguientes:

- Permiten fraccionar el desarrollo del protocolo que usa.
- Las capas facilitan el entendimiento del funcionamiento global de un protocolo.
- Facilitan las compatibilidades, tanto de software como de hardware de los distintos sistemas conectados.
- La arquitectura o estructuras de capas son flexibles a la hora de modificarlas.

a) CAPA FÍSICA

Es responsable del transporte de bits. Dependiendo del tipo de enlace físico los bits se representan de una manera en la que puedan ser transportados a través del medio.

Define voltajes, tiempo de duración de los pulsos, el número de patas que tiene el conector de la interface y sus funciones, la forma de establecer la conexión inicial y de interrumpirla, etc.

Generalmente será un cable aunque no se descarta cualquier otro medio de transmisión como ondas o enlaces vía satélite.

b) CAPA DE ENLACE DE DATOS

- Asegura que la información sea transmitida sin errores entre nodos adyacentes de la red sin importar el medio de transmisión utilizado.
- Maneja tramas de datos como unidad de transmisión de datos.
- Crea los límites de la trama.
- Resuelve problemas de daño, pérdida o duplicidad de tramas.
- Participa en la regulación de flujo de tramas entre los nodos.

c) CAPA DE RED

Define la forma en que un mensaje se transmite a través de distintos tipos de redes hasta llegar a su destino. El protocolo principal de esta capa es el Protocolo de Internet (IP) aunque también se encuentran a este nivel los protocolos ARP, ICMP e IGMP.

- Se encarga de que los datos sean enviados a su correcto destino, determinando la ruta de transmisión.
- La unidad de transmisión de datos en esta capa es el paquete de datos.
- Participa en el control de congestión de la red.
- Puede llevar la contabilidad del número de paquetes o bits que se enviaron a cada cliente para cuestiones de facturación.
- Puede resolver problemas de interconexión de redes heterogéneas.

d) CAPA DE TRANSPORTE

La capa de transporte (protocolos TCP y UDP) ya no se preocupa de la ruta que siguen los mensajes hasta llegar a su destino. Sencillamente, considera que la comunicación extremo a extremo está establecida y la utiliza. Además añade la noción de puertos, como se tratará más adelante.

- Acepta los datos de la capa de sesión, los divide, siempre que sea necesario, en unidades más pequeñas (la capa de red generalmente pone un límite en el tamaño de los mensajes que acepta), los pasa a la capa de red y asegura que todos ellos lleguen correctamente a su destino.
- A partir de la capa de red, las 4 capas superiores restantes manejan mensajes como unidad de transmisión de datos.
- Detecta fallas en la red y realiza las acciones correspondientes.
- Solicita el establecimiento de un nuevo enlace, en el caso de que falle un enlace de la red.

e) CAPA DE SESIÓN

- Es un tipo de sistema operativo para la comunicación de datos.

- Permite que los usuarios de diferentes computadoras puedan establecer sesiones entre ellos.
- Realiza el control del diálogo.
- Lleva a cabo la función de sincronización, es decir, inserta puntos de verificación en el flujo de datos, con objeto de que solamente tengan que retransmitirse los datos que se encuentren en seguida del último punto de verificación cuando se reanuda el servicio después de una caída de la red.

f) CAPA DE PRESENTACIÓN

- Permite a dispositivos que intercambian información, entenderse o interpretarse entre ellos independientemente de la codificación que utilicen para los caracteres, por ejemplo, código ASCII (*American Standard Code for Information Interchange; Código Estadounidense Estándar para el Intercambio de Información*) y EBCDIC (*Extended Binary Coded Decimal Interchange Code; Código Extendido de Binario Codificado Decimal*).
- Convierte los datos transmitidos a una forma inteligible para el dispositivo terminal.
- Maneja aspectos de representación de la información, por ejemplo: la compresión de datos y la criptografía.

g) CAPA DE APLICACIÓN

Proporciona los distintos servicios de Internet: correo electrónico, páginas Web, FTP, TELNET.

Contiene una variedad de protocolos que hacen posible ofrecer una serie de aplicaciones al usuario final, por ejemplo:

- Correo electrónico.
- Transferencia de archivos.
- Terminal virtual (telnet).
- Directorio electrónico.

1.6. Capa de red

Se explicará detalladamente la capa de red, porque dentro de ésta se tienen las direcciones IP, el protocolo IP y la máscara de subred. Los cuales son importantes para establecer la comunicación entre el servidor y los usuarios.

Por medio de las direcciones IP se hace referencia al punto de publicación que se transmitirá por internet, ésta conexión se basa en el protocolo IP.

El cometido de la capa de red es hacer que los datos lleguen desde el origen al destino, aun cuando ambos no estén conectados directamente. Es decir que se

encarga de encontrar un camino manteniendo una tabla de enrutamiento y atravesando los equipos que sean necesarios, para hacer llevar los datos al destino.

Los equipos encargados de realizar este encaminamiento se denominan ruteadores. Adicionalmente la capa de red debe gestionar la congestión en la red, que es el fenómeno que se produce cuando una saturación de un nodo tira la red, es decir, provoca una falla en toda la red.

La capa de red se encarga de fragmentar cada mensaje en paquetes de datos llamados datagramas IP y de enviarlos de forma independiente a través de la red de redes. Cada datagrama IP incluye un campo con la dirección IP de destino. Esta información se utiliza para encaminar los datagramas a través de las redes necesarias que los hagan llegar hasta su destino.

1.6.1. Direcciones IP

La dirección IP es el identificador de cada host dentro de su red de redes. Cada anfitrión conectado a una red tiene una dirección IP asignada, la cual debe ser distinta a todas las demás direcciones que estén vigentes en ese momento en el conjunto de redes visibles por el anfitrión. En el caso de Internet, no puede haber dos computadoras con 2 direcciones IP públicas iguales.

Es posible tener dos computadoras con la misma dirección IP siempre y cuando pertenezcan a redes independientes entre sí (sin ningún camino posible que las comuniquen).

Las direcciones IP se clasifican en:

Direcciones IP públicas. Son visibles en todo Internet. Una computadora con una IP pública es accesible (visible) desde cualquier otra computadora conectada a Internet. Para conectarse a Internet es necesario tener una dirección IP pública.

Direcciones IP privadas (reservadas). Son visibles únicamente por otros anfitriones de su propia red o de otras redes privadas interconectadas por ruteadores. Se utilizan en las empresas para los puestos de trabajo. Las computadoras con direcciones IP privadas pueden salir a Internet por medio de un ruteador (o *proxy*) que tenga una IP pública. Sin embargo, desde Internet no se puede acceder a computadoras con direcciones IP privadas. A su vez, las direcciones IP pueden ser:

Direcciones IP estáticas (fijas). Un host que se conecte a la red con dirección IP estática siempre lo hará con una misma IP. Las direcciones IP públicas estáticas son las que utilizan los servidores de Internet con objeto de que estén siempre localizables por los usuarios de Internet. Estas direcciones hay que contratarlas.

Direcciones IP dinámicas. Un anfitrión que se conecte a la red mediante dirección IP dinámica, cada vez lo hará con una dirección IP diferente. Las direcciones IP públicas dinámicas son las que se utilizan en las conexiones a Internet mediante módem. Los proveedores de Internet utilizan direcciones IP dinámicas debido a que tienen más clientes que direcciones IP. Las direcciones IP están formadas por 4 bytes (32 bits). Se suelen representar de la forma a.b.c.d donde cada una de estas letras es un número comprendido entre el 0 y el 255, esto es conocido como octeto. Por ejemplo la dirección IP del servidor de IBM es 129.42.18.99

CAPÍTULO

2

SISTEMA DE MONITOREO

2.1. Monitoreo de redes de computadoras

La palabra monitoreo no tiene una definición exacta, pero en el contexto computacional sería “una función que busca conocer cómo se están realizando ciertas tareas que han sido diseñadas para un plan operativo y de presupuesto”.

Para el área tecnológica el concepto de monitoreo tiene un enfoque práctico, en materia de redes, es supervisar todas las funciones que implica el trabajo de una red de computadoras. El monitoreo está ligado con el concepto de inteligencia competitiva la cual se define como: “conocimiento generado a partir del análisis resultante de la información sobre el entorno de la organización, que está disponible lícitamente”.

En base a este conocimiento generado, se pueden realizar acciones específicas en el caso de que una infraestructura de redes pueda tener algún tipo de problema, esta sería la diferencia entre el simple análisis de datos y monitorear una red. En el monitoreo además de realizar un análisis detallado acerca de las acciones que suceden en la red, también se realizan acciones de supervisar y reaccionar ante algún imprevisto.

Estos imprevistos se pueden traducir en eventos como es el caso de “problemas de ruido en la transmisión que crean situaciones que no existen como tales como direcciones de computadoras que no pertenecen a ninguno de los nodos y errores en la información, por mencionar algunos”.

A grandes rasgos lo que se pretende en el proyecto de tesis es “una verificación sobre la información contenida en un paquete que viaja a lo largo de toda la red, además de los protocolos manejados en la red, si esta información no es válida por alguna razón, se declara inválido el paquete escribiendo una bandera de error” .

Si esto sucede con cada una de las computadoras conectadas en red que se encuentran bajo análisis “se llevará cuenta de los errores que ocurren en la red, si una computadora se da cuenta de que el número de errores excedió a la cuenta permitida, le informa a la computadora que está "monitoreando" a la red, para que esta pueda declarar una condición de error y mostrarla en el servidor de toda la red”.

En la estructura de un sistema de monitoreo se tienen limitantes y funciones específicas importantes, las cuales se muestran más adelante. Otro aspecto importante para resaltar en la definición de monitoreo es que al momento de mostrar el error en el servidor encargado de monitorear la red, el administrador deberá tener una reacción a este acontecimiento, el cual es dar seguimiento al error y restaurar la infraestructura de la red para su correcto funcionamiento, es decir tener una respuesta a los eventos inesperados del sistema.

Vistos todos los aspectos anteriormente mencionados se puede dar una definición de monitoreo en el área de redes y es: “Análisis detallado que surge a partir del estudio sobre la red supervisada y que nos da un conocimiento de su funcionamiento y en caso de tener algún error dar acción inmediata a su restablecimiento”.

La situación actual en el tema de monitoreo de redes no se encuentra establecida del todo, ya que existen pocas herramientas que le dan seguimiento al error a la red analizada, existen algunos productos como son: Solarwinds, ActiveXperts, AdvenNet y eEye todas de GIA Software las cuales en conjunto son herramientas de alto nivel que sirven para la administración, monitoreo y corrección de errores en la red. La desventaja es que son herramientas de alto nivel, la cual tienen un costo bastante elevado y que sólo ofrecen soluciones de manera local, es decir, estando en un servidor en el mismo lugar físico de la red supervisada.

La aportación en este proyecto de tesis será una herramienta de menor costo, fácil de usar y que el administrador lo utilice de manera remota, todos estos detalles del proyecto se mencionarán más adelante en el apartado de objetivo general y objetivos específicos.

En esta parte se puede concluir que el tema de monitoreo de redes es apenas un desarrollo que no se ha explotado del todo y que puede tener muchas variantes, para cuestiones de nuestro proyecto de monitoreo de redes se realizará mediante una aplicación Web de manera remota.

2.2. Protocolo SNMP (Simple Network Management Protocol)

En una internet existen varias redes conectadas entre sí con el uso de routers y un protocolo de interconexión de redes, de modo que los routers usan el protocolo para manejar las características de las redes y proporcionar un servicio uniforme entre ellas, aunque cada red use una tecnología distinta y unas reglas específicas de transmisión, los hosts de cada red la ven de igual manera.

La principal tecnología de interconexión de redes es el conjunto de protocolos de Internet TCP/IP, que fue creado por la Agencia de Proyectos de Investigación Avanzada de Defensa (DARPA) y que son los que se usan en el Internet, pero también en la interconexión de redes menores (redes locales).

SNMP se sitúa en el tope de la capa de transporte de la pila OSI, o por encima de la capa UDP de la pila de protocolos TCP/IP. Siempre en la capa de transporte. Gráficamente se podría ver como se muestra a continuación:

Aplicación SNMP
Transporte (TCP,UDP,...)
Interred(IP)
Interface de red
Física

SNMP es un Protocolo Simple de Administración de Red (SNMP), es un protocolo de la capa de aplicación que facilita el intercambio de información de administración entre dispositivos de redes.

2.2.1. Necesidad de Administrar una Red

Los problemas que se presentan en la interconexión de redes son principalmente dos:

Dispositivos diferentes: La interconexión de redes permite diferentes tipos de dispositivos y estos son de diversos vendedores, todos ellos soportando el protocolo TCP/IP. Usar una tecnología de interconexión abierta permitió que existieran las redes formadas por dispositivos de distintos fabricantes, por lo que para administrar estas redes, habrá que usar una tecnología de administración de redes abierta.

Administraciones diferentes: Como se permite la interconexión entre redes de distinto propósito y distinto tamaño, hay que tener en cuenta que también están administradas, gestionadas y financiadas de distinta forma, tales como HP OpenView, Novell, NMS, IBM Net View o Sun Net Manager y entidades administrativas que pueden incluir Hosts, Routers. Hubs o en nuestro caso ATM's.

El envío de la información queda a cargo del agente de SNMP situado en el cajero, es decir, el Cajero desde el punto de vista del servicio SNMP, opera como **SERVIDOR**. En tanto que el equipo receptor de la tramas enviadas por el agente, opera como **CLIENTE**.

Evolución Histórica del Protocolo Simple de Gestión de Red (SNMP)

El protocolo Snmpv1 fue diseñado a mediados de los 80 por Case, McCloghrie, Rose, and Waldbusser, como una solución a los problemas de comunicación entre diferentes tipos de redes.

En un principio, su principal meta era el lograr una solución temporal hasta la llegada de protocolos de gestión de red con mejores diseños y más completos. Pero esos administradores de red no llegaron y SNMPv1 se convirtió en la única opción para la gestión de red.

El manejo de este protocolo era simple, se basaba en el intercambio de información de red a través de mensajes (PDU's). Además de ser un protocolo fácilmente extensible a toda la red, debido a esto su uso se estandarizo entre usuarios y empresas que no querían demasiadas complicaciones en la gestión de sus sistemas informáticos dentro de una red.

No obstante este protocolo no era perfecto, además no estaba pensado para poder gestionar la inmensa cantidad de redes que cada día iban apareciendo. Para subsanar sus carencias surgió la versión 2 (SNMP v2). Las mayores innovaciones respecto a la primera versión son:

Introducción de mecanismos de seguridad, totalmente ausentes en la versión 1.

Estos mecanismos protegen la privacidad de los datos, confieren autenticación a los usuarios y controlan el acceso.

2.2.2. Definición de las variables.

Se añaden estructuras de la tabla de datos para facilitar el manejo de los datos. El hecho de poder usar tablas hace aumentar el número de objetos capaces de gestionar, con lo que el aumento de redes dejo de ser un problema.

Realmente esta versión 2 no supuso más que un parche, es más hubo innovaciones como los mecanismos de seguridad que se quedaron en pura teoría, no se llegaron a implementar.

SNMP en su última versión (SNMPv3) posee cambios significativos con relación a sus predecesores, sobre todo en aspectos de seguridad, sin embargo no ha sido mayoritariamente aceptado en la industria.

2.2.3. Componentes Básicos

Una red administrada a través de SNMP consiste de tres componentes claves:

- Dispositivos administrados;
- Agentes;
- Sistemas administradores de red (NMS's).

Un dispositivo administrado es un nodo de red que contiene un agente SNMP y reside en una red administrada. Estos recogen y almacenan información de administración, la cual es puesta a disposición de los NMS's usando SNMP. Los dispositivos administrados (llamados elementos de red), pueden ser routers, servidores de acceso, switches, bridges, hubs, computadores o impresoras.

Un agente es un módulo de software de administración de red que reside en un dispositivo administrado. Un agente posee un conocimiento local de información de administración (memoria libre, número de paquetes IP recibidos, rutas, etcétera), la cual es traducida a un formato compatible con SNMP y organizada en jerarquías.

Un NMS ejecuta aplicaciones que supervisan y controlan a los dispositivos administrados. Los NMS's proporcionan el volumen de recursos de procesamiento y memoria requeridos para la administración de la red. Uno o más NMS's deben existir en cualquier red administrada.

2.2.4. Comandos Básicos

Los dispositivos administrados son supervisados y controlados usando cuatro comandos SNMP básicos: lectura, escritura, notificación y operaciones transversales.

El comando de lectura es usado por un NMS para supervisar elementos de red. El NMS examina diferentes variables que son mantenidas por los dispositivos administrados.

El comando de escritura es usado por un NMS para controlar elementos de red. El NMS cambia los valores de las variables almacenadas dentro de los dispositivos administrados.

El comando de notificación es usado por los dispositivos administrados para reportar eventos en forma asíncrona a un NMS. Cuando cierto tipo de evento ocurre, un dispositivo administrado envía una notificación al NMS.

Las operaciones transversales son usadas por el NMS para determinar qué variables soporta un dispositivo administrado y para recoger secuencialmente información en tablas de variables, como por ejemplo, una tabla de rutas.

2.2.5. Base de información de Administración SNMP (MIB)

Una Base de Información de Administración (MIB) es una colección de información que está organizada jerárquicamente. Las MIB's son accedidas usando un protocolo de administración de red, como SNMP.

Un objeto administrado (algunas veces llamado objeto MIB, objeto, o MIB) es uno de cualquier número de características específicas de un dispositivo administrado. Los objetos administrados están compuestos de una o más instancias de objeto, que son esencialmente variables.

Existen dos tipos de objetos administrados: Escalares y tabulares. Los objetos escalares definen una simple instancia de objeto. Los objetos tabulares definen múltiples instancias de objeto relacionadas que están agrupadas conjuntamente en tablas MIB.

Un ejemplo de un objeto administrado es atInput, que es un objeto escalar que contiene una simple instancia de objeto, el valor entero que indica el número total de paquetes AppleTalk de entrada sobre una interfaz de un enrutadores.

Un identificador de objeto (object ID) únicamente identifica un objeto administrado en la jerarquía MIB. La jerarquía MIB puede ser representada como un árbol con una raíz anónima y los niveles, que son asignados por diferentes organizaciones.

2.2.6. Mensajes SNMP

Para realizar las operaciones básicas de administración nombradas anteriormente, el protocolo SNMP utiliza un servicio no orientado a la conexión (UDP) para enviar un pequeño grupo de mensajes (PDUs) entre los administradores y agentes. La utilización de un mecanismo de este tipo asegura que las tareas de administración de red no afecten el rendimiento global, ya que se evita la utilización de mecanismos de control y recuperación como los de un servicio orientado a la conexión.

Los puertos comúnmente utilizados para SNMP son los siguientes:

Número	Descripción
161	SNMP
162	SNMP-trap

Los paquetes utilizados para enviar consultas y respuestas SNMP poseen el siguiente formato:

Versión	Comunidad	SNMP PDU
---------	-----------	----------

Versión: Número de versión de protocolo que se está utilizando (por ejemplo 1 para SNMPv1).

Comunidad: Nombre o palabra clave que se usa para la autenticación. Generalmente existe una comunidad de lectura llamada "public" y una comunidad de escritura llamada "private".

SNMP PDU: Contenido de la unidad de datos del protocolo, el que depende de la operación que se ejecute.

Los mensajes GetRequest, GetNextRequest, SetRequest y GetResponse utilizan la siguiente estructura en el campo SNMP PDU:

Tipo	Identificador	Estado de error	Índice de error	Enlazado de variables
------	---------------	-----------------	-----------------	-----------------------

Identificador: Es un número utilizado por el NMS y el agente para enviar solicitudes y respuesta diferentes en forma simultánea;

Estado e índice de error: Sólo se usan en los mensajes GetResponse´ (en las consultas siempre se utiliza cero). El campo "índice de error" sólo se usa cuando "estado de error" es distinto de 0 y posee el objetivo de proporcionar información adicional sobre la causa del problema. El campo "estado de error" puede tener los siguientes valores:

- 0: No hay error.**
- 1: Demasiado grande.**
- 2: No existe esa variable.**
- 3: Valor incorrecto.**
- 4: El valor es de solo lectura.**
- 5: Error genérico.**

Enlazado de variables: Es una serie de nombres de variables con sus valores correspondientes (codificados en ASN.1).

GetRequest

A través de este mensaje el NMS solicita al agente retornar el valor de un objeto de interés mediante su nombre. En respuesta el agente envía una respuesta indicando el éxito o fracaso de la petición. Si la petición fue correcta, el mensaje resultante también contendrá el valor del objeto solicitado. Este mensaje puede ser usado para recoger un valor de un objeto, o varios valores de varios objetos, a través del uso de listas.

GetNextRequest

Este mensaje es usado para recorrer una tabla de objetos. Una vez que se ha usado un mensaje GetRequest para recoger el valor de un objeto, puede ser utilizado el mensaje GetNextRequest para repetir la operación con el siguiente objeto de la tabla. Siempre el resultado de la operación anterior será utilizado para la nueva consulta. De esta forma un NMS puede recorrer una tabla de longitud variable hasta que haya extraído toda la información para cada fila existente.

SetRequest

Este tipo de mensaje es utilizado por el NMS para solicitar a un agente modificar valores de objetos. Para realizar esta operación el NMS envía al agente una lista de nombres de objetos con sus correspondientes valores.

GetResponse

Este mensaje es usado por el agente para responder un mensaje GetRequest, GetNextRequest, o SetRequest. En el campo "Identificador de Request" lleva el mismo identificador que el "request" al que está respondiendo.

2.2.7. Trap

Una trap es generada por el agente para reportar ciertas condiciones y cambios de estado a un proceso de administración. El formato de la PDU es diferente:

Tipo	Enterprise	Dirección del agente	Tipo genérico de trap	Tipo específico de trap	Timestamp	Enlazado de variables
------	------------	----------------------	-----------------------	-------------------------	-----------	-----------------------

Enterprise: Identificación del subsistema de gestión que ha emitido el trap.

Dirección del agente: Dirección IP del agente que ha emitido el trap.

Tipo genérico de trap.

Cold start (0): Indica que el agente ha sido inicializado o reinicializado.

Warm start (1): Indica que la configuración del agente ha cambiado.

Link down (2): Indica que una interfaz de comunicación se encuentra fuera de servicio (inactiva).

Link up (3): Indica que una interfaz de comunicación se encuentra en servicio (activa).

Authentication failure (4): Indica que el agente ha recibido un requerimiento de un NMS no autorizado (normalmente controlado por una comunidad).

EGP neighbor loss (5): Indica que en sistemas en que los routers están utilizando el protocolo EGP, un equipo colindante se encuentra fuera de servicio.

Enterprise (6): En esta categoría se encuentran todos los nuevos traps incluidos por los vendedores.

Tipo específico de trap: Es usado para traps privados (de fabricantes), así como para precisar la información de un determinado trap genérico.

Timestamp: Indica el tiempo que ha transcurrido entre la reinicialización del agente y la generación del trap.

Enlazado de variables: Se utiliza para proporcionar información adicional sobre la causa del mensaje.

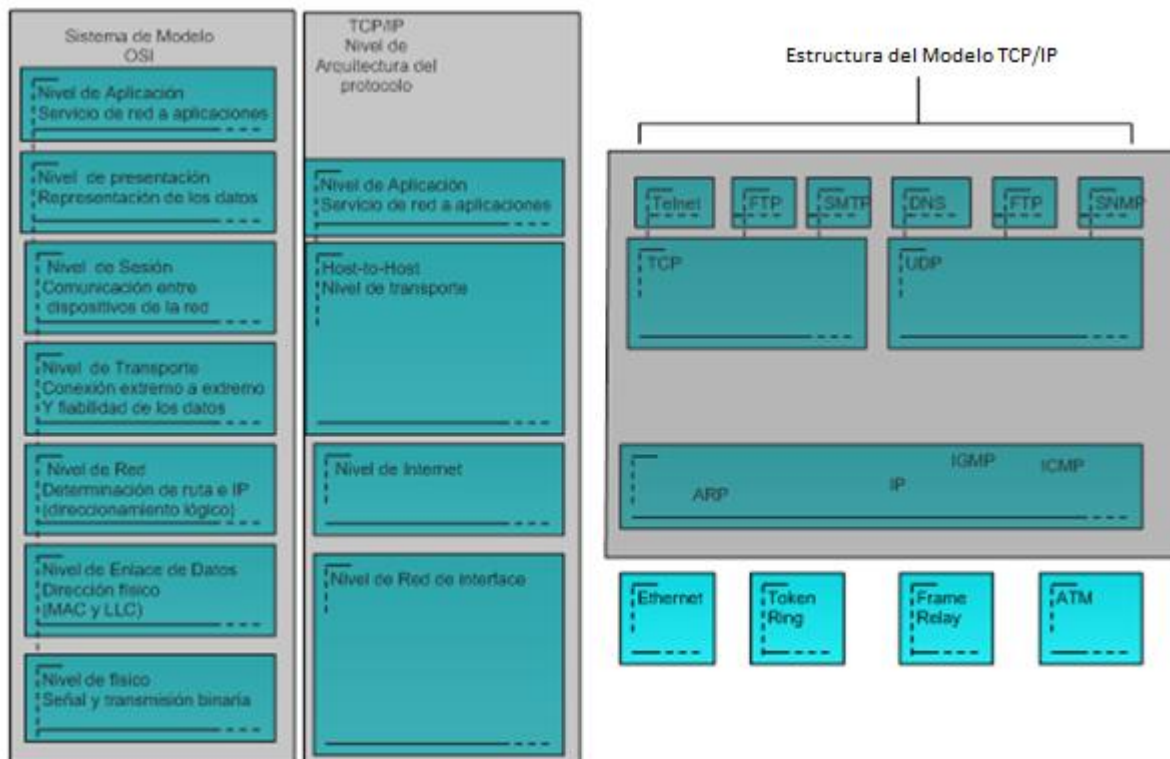
GetBulkRequest

Este mensaje es usado por un NMS que utiliza la versión 2 del protocolo SNMP típicamente cuando es requerida una larga transmisión de datos, tal como la recuperación de largas tablas. En este sentido es similar al mensaje GetNextRequest usado en la versión 1 del protocolo, sin embargo, GetBulkRequest es un mensaje que implica un método mucho más rápido y eficiente, ya que a través de un solo mensaje es posible solicitar la totalidad de la tabla.

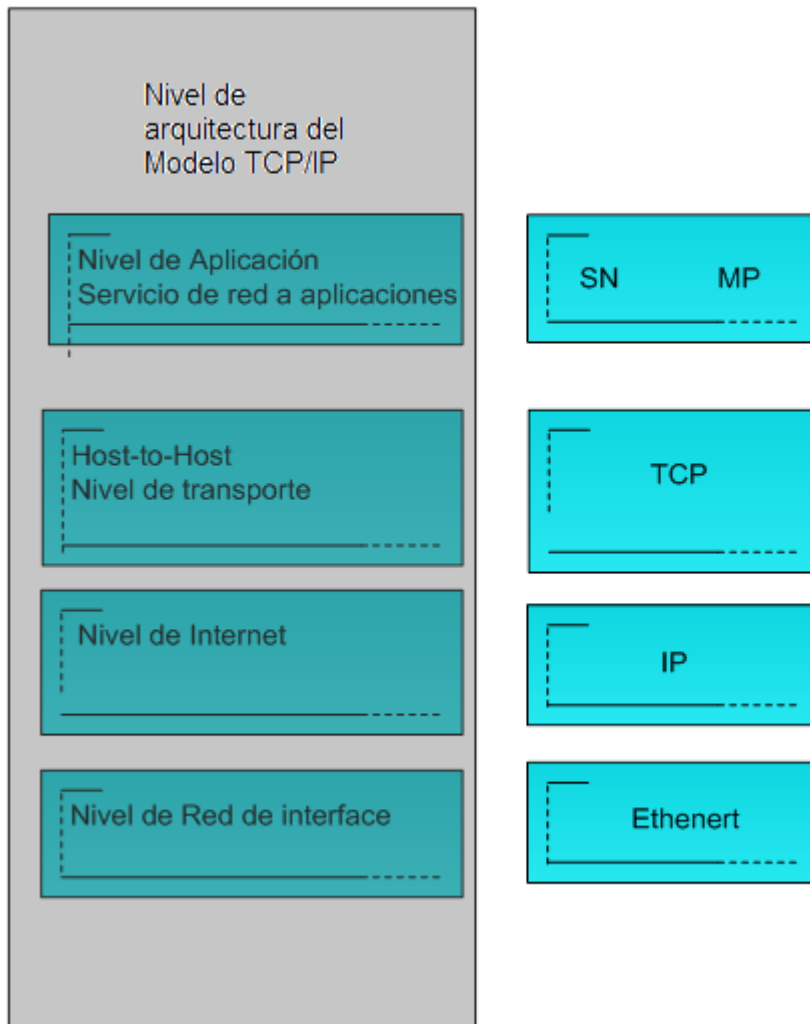
Solicitud de Información (InformRequest)

Un NMS que utiliza la versión 2 del protocolo SNMP transmite un mensaje de este tipo a otro NMS con las mismas características, para notificar información sobre objetos administrados.

2.2.8. Arquitectura del Servicio



Como puede observarse en la figura, el servicio SNMP forma parte de la Suite del modelo TCP/IP. En el caso que nos atañe, en la capa de aplicación (Application Layer) se sitúa el servicio SNMP; en la capa de transporte (Transport Layer) se tiene la base del TCP; en la capa de Internet (Internet Layer), es por medio de la IP; y finalmente en la capa de interface de red (Network Interface Layer). Aislando la arquitectura tipo utilizada, tenemos la siguiente figura :



CAPÍTULO

3

PLANTEAMIENTO DE LA APLICACIÓN

Debido a que esta aplicación se implementara en un entorno grafico que sea amigable con el usuario, se ha pensado generar una aplicación que se pueda instalar en el sistema más utilizado actualmente el cual es Windows. Para lo cual se requiere conocer los programas que se implementaran para generar la aplicación que sea sencilla de utilizar, estos usuarios solo necesitan saber manejar el sistema operativo antes mencionado y de esta manera las empresas que utilicen este software no tengan que dar una capacitación o que solo una persona sea la encargada de utilizar dicha aplicación.

La aplicación debe de ser implementada en algún lenguaje que nosotros podamos programar, esto para poder hacer más sencilla la resolución de problemas que se puedan llegar a presentar al estarla programando, sin embargo también podemos ver las alternativas que se ofrecen actualmente y de esta manera hacer un análisis sobre qué ventajas y desventajas ofrecen los programas que nosotros escogimos para programar.

3.1. Software utilizado para el desarrollo de la aplicación

3.1.1. Procesamiento de imagen

MATLAB

Es un software de desarrollo en el cual se pueden realizar diferentes proyectos, algunos ejemplos de estos proyectos pueden ser la manipulación de matrices, resolver ecuaciones de segundo grado, el procesamiento digital de imágenes, la implementación de algoritmos, entre otros.

Este software es muy usado en diversas universidades para efectos educativos; en este caso se hará uso del software para poder hacer la manipulación de una imagen tomada con la cámara IP para hacer la comparación de esta con otra que tomaremos como una imagen base.

Las dos imágenes a utilizar primero tienen que ser convertidas a matrices para hacer una comparación, este método es conocido como "Correlación" y dependiendo del resultado de esta comparación, podremos determinar si en la imagen que está captando la cámara IP existe algún problema en la imagen que está captando. Se escoge este método para hacer la comparación ya que es de los más simples y que arrojan mejores resultados en la comparación de imágenes.

Métodos de comparación de imágenes

Dentro de los métodos de comparación de imágenes se encuentran la *correlación* y la *convolución*.

1. Correlación

La correlación de imágenes tiene como objeto la localización de un área en una imagen de manera automática. Para poder hacer la correlación en una imagen, se tiene que conocer un punto de la misma, esto puede ser un objeto que se presente y de esta manera poder seleccionarla para realizar una búsqueda. Esta área u objeto seleccionado será conocido como matriz de búsqueda o área de búsqueda y dentro de ella localizaremos determinadas formas que definan un objeto específico, las cuales llamaremos matriz patrón; en otros casos suele llamarse como ventana de búsqueda.

Al momento de tener la matriz patrón esta será la plantilla a utilizar en la comparación de imágenes, esto al tomar una nueva imagen que será conocida como matriz muestra, y por medio de una diferencia o resta de matrices nos resultara una matriz la cual es el resultado de la anterior operación matemática. Al momento de comprobar esta matriz podremos observar si hay diferencia entre la matriz patrón y la matriz muestra.

2. Convolución

La convolución de imágenes tiene como objeto la fusión dos imágenes f y g en una tercera imagen que representará la magnitud en la que se superpone la imagen f y una versión trasladada e invertida de la imagen g .

Correlación vs Convolución.

En procesamiento de imágenes, no hay un orden temporal y causal, por lo tanto la convolución y la correlación funcionan casi de la misma manera. Pero la convolución se utiliza para describir transformaciones de sistemas lineales, como

suaviza miento o filtrado, mientras que la correlación es utilizada como medida de similitud en el contexto de casa o correspondencia de patrones o de planillas.

El programa Matlab nos parece la mejor opción ya que es un software fácil de programar y que tiene un buen desempeño al momento de hacer el análisis de imágenes, esto nos permite poder hacer la implementación de código matemático de una manera sencilla, sin tener que estar desarrollando una gran cantidad de código, con esto se optimiza el tiempo de programación, ya que dé al no utilizar este programa tardaríamos bastante tiempo en generar un código similar en algún otro compilador.

3.1.2. Generación de la aplicación

Microsoft Visual Studio 2010

Microsoft Visual Studio 2010 es un entorno de desarrollo integrado para sistemas operativos Windows. Soporta varios lenguajes de programación tales como Visual C++, Visual C#, Visual J#, ASP.NET y Visual Basic .NET, aunque actualmente se han desarrollado las extensiones necesarias para muchos otros.

La versión más reciente es la .NET que se incluye en el paquete Visual Studio .NET de Microsoft. Esta versión combina la sencillez del BASIC con un poderoso lenguaje de programación Visual que juntos permiten desarrollar robustos programas de 32 bits para plataformas Windows. Esta fusión de sencillez y la estética permitió ampliar mucho más el monopolio de Microsoft, ya que el lenguaje sólo es compatible con plataformas Windows, un sistema operativo de la misma empresa.

Visual Basic ya no es más "un lenguaje para principiantes" sino que es una perfecta alternativa para los programadores de cualquier nivel que deseen desarrollar aplicaciones compatibles con Windows. Para aquellos programadores avanzados o para quienes deseen incorporar funciones más complejas una alternativa mejor al Visual Basic es el Visual C++.

Java

Otro lenguaje de programación orientado a objetos, fue desarrollado por Sun Microsystems. El lenguaje tiene mucha relación con el lenguaje C y C++, pero contiene un modelo de objetos más simple y sin herramientas de bajo nivel que llegue a producir errores. La memoria es gestionada mediante un recolector de basura.

En mayo de 2007, Sun Microsystems liberó gran parte de las tecnologías Java bajo la licencia GNU GPL, de tal forma que prácticamente todo el Java de Sun es ahora software libre (aunque la biblioteca de clases de Sun que se requiere para ejecutar los programas Java aún no lo es).

A pesar de que se tiene a dos grandes desarrolladores de aplicaciones, se optó por elegir el desarrollador de Microsoft Visual Studio, ya que es un sistema en el cual ya se tiene práctica, además esto ayudaría a que no se tenga la necesidad de aprender un lenguaje de programación nuevo y que tal vez si se presenta algún problema se pueda solucionar de una manera más rápida.

3.1.3. Base de Datos

SQL Server

SQL Server es un sistema de gestión de base de datos desarrollado por la empresa Microsoft. Microsoft SQL Server es la alternativa de Microsoft a otros gestores de bases de datos como es *MySQL*.

Características:

- Estabilidad y seguridad.
- Soporta procedimientos almacenados.
- Permite trabajar en modo cliente-servidor, donde la información y datos se alojan en el servidor y los terminales o clientes de la red sólo acceden a la información.
- Además permite administrar información de otros servidores de datos.

Este sistema incluye una versión reducida, llamada SQL Express Edition, se distribuye en forma *gratuita* con el mismo motor de base de datos pero orientado a proyectos más pequeños.

MySQL

MySQL es un sistema de gestión de bases de datos multiusuario, multiplataforma y de código abierto.

MySQL pertenece a la compañía sueca MySQL AB, a la que le pertenece casi todos los derechos del código fuente.

MySQL es muy popular en aplicaciones web y suele combinarse con el popular lenguaje PHP.

Características

- MySQL está escrito en C y C++
- Emplea el lenguaje SQL para consultas a la base de datos.
- MySQL Server está disponible de manera gratuita bajo licencia GPL.
- Trabaja en diversas plataformas: GNU/Linux, Mac OS X, NetBSD, Novell NetWare, OpenBSD, OS/2 Warp, QNX, SGI IRIX, Solaris, SunOS, Microsoft Windows (95, 98, ME, NT, 2000, XP y Vista).

Tabla de comparación entre MySQL y SQL Server

Característica	MySQL	SQL Server Express	SQL Server
Costo	Libre y de pago	Libre	De pago
Open Source	Si	No	No
Plataformas	Linux, Windows y muchas otras	Sólo Windows	Sólo Windows
Limite de tamaño de la base de datos	Limitado por el sistema operativo	10Gb	Limitado por el sistema operativo
Compatibilidad ACID	Depende del motor de almacenamiento	Si	Si
Transacciones	Si	Si	Si
Servicio de reportes	No	Si	Si
Posibilidad de elegir diferentes formas de almacenamiento	Si	No	No
Claves Foráneas	Depende del motor	Si	Si
Vistas	Si	Si	Si
Procedimientos almacenados	Si	Si	Si
Triggers	Si	Si	Si
Cursores	Si	Si	Si
Subconsultas	Si	Si	Si
Replicación	Si	Limitado	Si
Funciones definidas por el usuario (UDF)	Si	Si	Si

<http://www.latindevelopers.com/articulos/sql-server/diferencias-entre-mysql-y-sql-server.php>

A pesar de que el programa MySQL nos brinda muy buenas opciones para el manejo de los reportes de error, consideramos una mejor opción usar Microsoft SQL Server ya que el desarrollador creo la implementación en los 2 programas (Microsoft Visual Studio 2010 y Microsoft SQL Server 2008) de una manera sencilla y sin problemas.

Agregado a esto como se observa en las características de los programas, en el desarrollo de base de datos Microsoft SQL Server 2008 brinda mayor seguridad y estabilidad que MySQL, y también agrega algunas características adicionales de gestión de base de datos.

3.1.4. Gestión de Red

SNMPc Network Manager

Castle Rock Computing fue la primera compañía en ofrecer un Windows basado en sistema de gestión SNMP. SNMPc es un sistema seguro de gestión de red que ofrece monitoreo en tiempo real para su infraestructura de red.

SNMPc utiliza una arquitectura de agente de votación distribuidos para proporcionar una solución de alto rendimiento capaz de vigilar redes de varios cientos de dispositivos a decenas de miles de personas.

Características principales:

- Monitor de dispositivos SNMP, enlaces WAN, servidores y aplicaciones
- Protocolo de Internet versión 6 (IPv6)
- Compatible con SNMP v1, v2c y SNMP v3
- Distribuidor Independiente - Gestiona cualquier dispositivo SNMP de cualquier fabricante
- Las alarmas automáticas
- Se ejecuta en servicios de Windows
- De consola remota y el acceso de Java
- Muestra en tiempo real el MIB
- Detección de redes automatizado

Net-SNMP

Net-SNMP es un software utilizado para la gestión del protocolo SNMP (v1, v2c y v3). Soporta diferentes tipos de protocolo tales como IPv4, IPv6 IPX AAL5 y otros más. Contiene una librería genérica, un conjunto de aplicaciones de líneas de comando.

Net-SNMP esta hospedado en el servidor de SourceForge y es uno de los programas más utilizados ya que se encuentra en el rango de los 100 más descargados. Puede ser instalado en la mayoría de los sistemas operativos que se encuentran actualmente tales como Linux, OpenBSD, Solaris y Mac OS X.

Debido a que para hacer la gestión de una red con diversos dispositivos, necesitamos un programa que nos proporcione las mejores prestaciones y el menor número de fallas al estar gestionando las cámaras conectadas a la aplicación, por lo cual se seleccionara el programa SNMPc ya que el programa que nos ofrece mejores prestaciones que el programa Net-SNMP.

3.2. Plan de desarrollo de la aplicación

Después de establecer cuales con los programas que se van a utilizar, procederemos a hacer un plan de trabajo a seguir para el desarrollo de la aplicación.

Para poder realizar este diseño la plataforma principal de desarrollo será Microsoft Visual Studio 2010, en el cual se programará la interfaz gráfica de la aplicación y en donde se implementaran diferentes tipos de funciones en forma de botones, que nos proporcionaran las herramientas para poder hacer el análisis de la imagen y de la red.

Al ser una aplicación de seguridad se implementara un acceso controlado, esto se realizara por medio de un usuario con contraseña valida, de tal manera que personas ajenas al área donde se utilice el software no puedan hacer ningún uso de este.

Al ser un sistema de gestión de fallas se requiere tener un control de los incidentes que se lleguen a presentar, y posteriormente darles una solución, para esto implementaremos una base de datos que será implementada a través del programa Microsoft SQL Server 2008, esta base de datos contendrá la descripción de la cámara que llegase a presentar alguna falla, con esto se pretende identificar rápidamente la cámara y dar solución a los problemas. Los datos que se contendrán en el reporte son los siguientes:

- Número de reporte
- Dirección IP
- Tipo de cámara
- Fabricante de la cámara
- Modelo de la cámara
- Falla de cámara
- Falla de red
- Ubicación de la cámara

Al momento de obtener los datos anteriores, el sistema procederá a generar un reporte global (que puede ser impreso), éste proporcionará la información que el personal de mantenimiento requiere para reparar la cámara. Estos reportes podrían ayudar a resolver problemas en el futuro, por ejemplo, cuando una cámara presente fallas iguales o similares, comprobaremos en el registro si se presentó alguna cámara con los mismos problemas y solucionarlo de una manera más rápida.

Incluirá la función de detectar fallas que se generen en la red, con esto se obtendrá un sistema de video sólido, ya que si llegase a fallar alguna cámara podría presentarse una falta de vigilancia que tal vez repercutiría en algún problema de mayor índole, para esto utilizaremos el programa llamado SNMPc, que es de los mejores programas para gestión de red que existen actualmente en el mercado.

Incluirá la comparación de imagen para poder hacer la detección de fallas en la cámara IP, esto se lograra al implementar un código que nosotros generaremos en el programa MATLAB, para poder escribir este código de comparación de imagen se utilizara el método de correlación ya explicado anteriormente, ya que es un método bastante sencillo para el procedimiento que se requiere.

Se integrará una interfaz de control de la cámara, donde se podrá observar la imagen que se está capturando en ese momento, y además en caso que a cámara lo permita se podrá tener control de la misma al poderla mover hacía varias direcciones, con esto la podremos observar en un ángulo de visión más extenso.

Contará con un explorador con salida a internet, esto para poder conectarse con las cámaras que actualmente se encuentran de manera gratuita en la red, con esto podemos comprobar que el programa este ejecutándose correctamente o descartar que se presente alguna falla con la conexión a la cámara que no se encuentre reportada con el sistema de red.

Dado que algunas cámaras tienen la función de realizar grabaciones de video, la aplicación contará con conexión hacia el servidor donde se encontrarán los videos que se graben, y en caso que se requiera observar la grabación de una cámara en específico pueda hacerse tan solo con seleccionar esta cámara.

CAPÍTULO

A

DESARROLLO DE LA APLICACIÓN

En este capítulo se explicará cómo fue desarrollada la aplicación en cuestión, también se describirán los problemas que se generaron mientras se desarrollaba esta aplicación, y de esta manera alguien que llegase a tomar esta aplicación como referencia y se encuentre con estos problemas, pueda solucionarlos.

4.1. Pantalla de Bienvenida



Fig. 4.1. Pantalla de Bienvenida

Para poder hacer la pantalla de entrada (Fig. 4.1) de esta aplicación se utilizó un SplashScreen o plantilla de presentación que se encuentra en Visual Studio 2010 para presentar la aplicación, esta herramienta puede ser programada para que cada vez que la aplicación se muestre mientras otros componentes de la aplicación son ejecutados. Con esta herramienta podemos observar el nombre de la aplicación, datos específicos como creadores, nombre de la compañía, también nos muestra la versión de la aplicación en la que se trabaja y así conocer si la versión está actualizada o estamos utilizando con alguna versión anterior.

Esta herramienta es muy útil para poder darle una buena imagen a nuestra aplicación ya que hoy en día la mayoría de las aplicaciones gráficas debe contener una imagen agradable, amigable para el usuario. Cuando termina de ejecutar los componentes necesarios para la aplicación el SplashScreen se cerrará y dará paso al siguiente proceso, esto será de manera automática sin que el usuario tenga que cerrar manualmente la presentación o seleccionar algún botón.

4.2. Acceso al Usuario



Fig. 4.2. Pantalla de Acceso al Usuario

Cuando la pantalla de bienvenida se cierre, la aplicación nos mostrará una pantalla de acceso (Fig. 4.2). La pantalla de acceso de usuario trabaja se creó al establecer una conexión Visual Studio 2010 con una base de datos desarrollada previamente en SQL Server 2008, esta base fu nombrada database, donde fueron creadas las tablas Rol y Usuario.

La tabla Rol contiene los siguientes campos:

CodRol. Este campo contiene el nombre del rol,

Usuario. Este campo contiene los siguientes campos:

CodUsuario: este campo almacenará el código usuario o número del usuario.

CodRol: este campo es la relación con la tabla Rol, en donde, ya se indicó previamente el valor que almacena, en siguientes campos identificamos y tenemos el control de los usuarios estos son los vales nombre, apellidos, dirección, usuario y contraseña. Esta parte solo mencionamos que tipos de valores utilizaremos para poder tener la autenticación de los usuarios.

Para poder realizar el acceso al sistema seguiremos utilizando Visual Studio en conjunto con otras herramientas. La primera de estas herramientas utilizadas fue Forms o ventana de Windows, en un lenguaje de visual Basic en blanco, posteriormente comenzamos a dar formato a Forms colocándole un nombre específico, para este caso se utilizó el nombre de “Acceso”, posteriormente se realizó el cambio del icono que lleva por defecto a otro que identificara Forms, también se colocó un fondo para dar una mejor presentación a nuestro trabajo.

Después procedimos a colocar un cuadro de texto o label en la parte superior de la ventana, en este indicaremos el texto que llevara el encabezado; enseguida colocaremos un PictureBox, donde colocaremos una imagen en la posición que deseemos para poder dar una buena presentación a la ventana, continuamos con la colocación de otro label en donde estableceremos que es campo de Rol, enseguida de este label colocaremos un ComboBox, en donde se podrá mostrar los tipos de roles almacenados en la base de datos, después de colocar nuestro ComboBox colocaremos un label adicional con el texto Usuario y colocaremos un TextBox en donde se escribirá el nombre del usuario que será evaluado.

Colocamos otro label con el texto de contraseña y nuevamente colocaremos un TextBox pero a diferencia del anterior lo configuraremos para que no se muestre lo que estemos escribiendo, así se quedará establecido el perfil que se maneja en las contraseñas, que es ocultar los caracteres para tener una mayor confidencialidad, este campo también es evaluado.

Procederemos a insertar un botón el cual llamaremos Entrar, este se programó con un método en donde analizará los campos Rol, Usuario y Contraseña ya antes mencionados, para poder realizar esta acción se programó primero una clase que se nombró “conexión”, en ésta generamos el código para poder hacer la conexión a la base de datos, esto se realizó al importar valores de SQL Server que previamente habían sido generados, después importar los datos de SQL Server realizamos una conexión hacia la base de datos donde pasaremos los siguientes valores: el nombre de nuestro localhost, el nombre de la base de datos a la que se requiere tener acceso, en esta se coloca la palabra “true” para indicar que es una autenticación Windows, también se generó el código para abrir y cerrar la conexión con SQL Server 2008.

Después de ser generado la clase conexión, se tuvo que generar una nueva clase que la nombramos “datos”, ésta la utilizamos declarando los parámetros que capturaremos en este caso serán usuario, contraseña, rol; para poder capturar los datos necesarios se tuvieron que generar métodos para poder realizar esta captura, en donde nos tendrá que regresar los valores capturados en el TextBox y en el ComboBox previamente insertados; tendremos que generar otro miembro de nuestra clase en donde asignaremos nuestros valores previamente capturados.

Para poder guardar nuestros valores capturados generamos una nueva clase que nombramos “funciones”, en esta clase importamos una de las características de SQL Server llamada “Sqlclient”, también se crea una función con el nombre de “validar” la cual va capturar los datos de nuestra clase datos.

Con esta nueva función se crea una excepción en donde conectamos la base por medio de la función “conectado” que hemos heredado previamente. Con nuestra variable declarada utilizaremos un comando programado en SQL Server que nombraremos “validar” comprobaremos estos valores capturados Con nuestros campos TextBox y ComboBox y una vez almacenados en nuestra clase “datos” podremos validarlos con los valores que tenemos almacenados en nuestra base de datos, si estos llegan a ser iguales se mandará un mensaje de error mediante un MsgBox, y para terminar nuestra función desconectamos.

Después de haber comprobado las clases, se procede a programar nuestro “Button”, importando nuestros valores de SQL, continuaremos con la declaración de variables públicas, y posteriormente continuamos con la creación de una función en donde se realizó un método para poder cargar nuestra variable “ComboBox” con los valores almacenados en la base de datos, todo esto se cargará en el cuerpo de Windows Forms, esto fue útil para continuar con la programación de nuestro “Button”, ya que dentro de la estructura de éste pasamos los valores capturados anteriormente en nuestra clase “datos” y de esta manera validados a través de las clases de funciones, si todos los valores son correctos nos dará acceso a nuestra aplicación, pero si los datos son incorrectos nos mandará un mensaje de error mediante un MsgBox con la leyenda “Error en el Usuario, Contraseña, o Rol”.

Con ésto se concluye la creación del acceso para los usuarios, como se puede observar es un método bastante seguro ya que las cuentas de usuario no se encuentra alojadas dentro nuestra aplicación si no en una base de datos y para que un usuario pueda identificarse tiene que pasar por varios procesos dentro de nuestra aplicación como por SQL Server, esto nos proporciona seguridad en la aplicación.

4.3. Menú Principal



Fig. 4.3. Menú Principal

Después de hacer una identificación correcta y pasar el filtro de seguridad, se desplegará el menú principal (Fig. 4.3) en donde se muestran las funciones principales de la aplicación para poder desarrollar este el menú, volvemos a crear un nuevo archivo con Windows Forms, al cual nombramos “menu” y le dimos formato cambiando el icono, también se colocó un fondo, posteriormente pasamos a colocar un “PictureBox” en donde colocamos una imagen de una cámara IP, se colocan seis botones los cuales se nombraron con los siguientes nombres: Video, Scanner de Imagen, Scanner de red, Reporte de fallas, Navegador y Salir. Los primeros cinco abren los procesos principales de la aplicación y el último nos permite abandonar la aplicación en su totalidad.

Nuestro menú realizará el llamado de las aplicaciones cuando éstas sean requeridas para su utilización por medio de un botón. La forma en la cual se programó cada uno de sus botones para que realicen el llamado de los archivos con extensión “.exe” de cada una de las aplicaciones anteriormente programadas y ejecutadas.

Cada Botón abre un proceso distinto ya asignado. Estos procesos serán explicados más adelante, la ventaja de utilizar nuestro menú de esta manera es que podemos abrir los distintos procesos programados en diversos lenguajes, esto no permite tener un manejo versátil de nuestras aplicaciones lo único en común que tienen cada una de las aplicaciones es que son procesos ejecutables.

4.4. Reproductor de Video

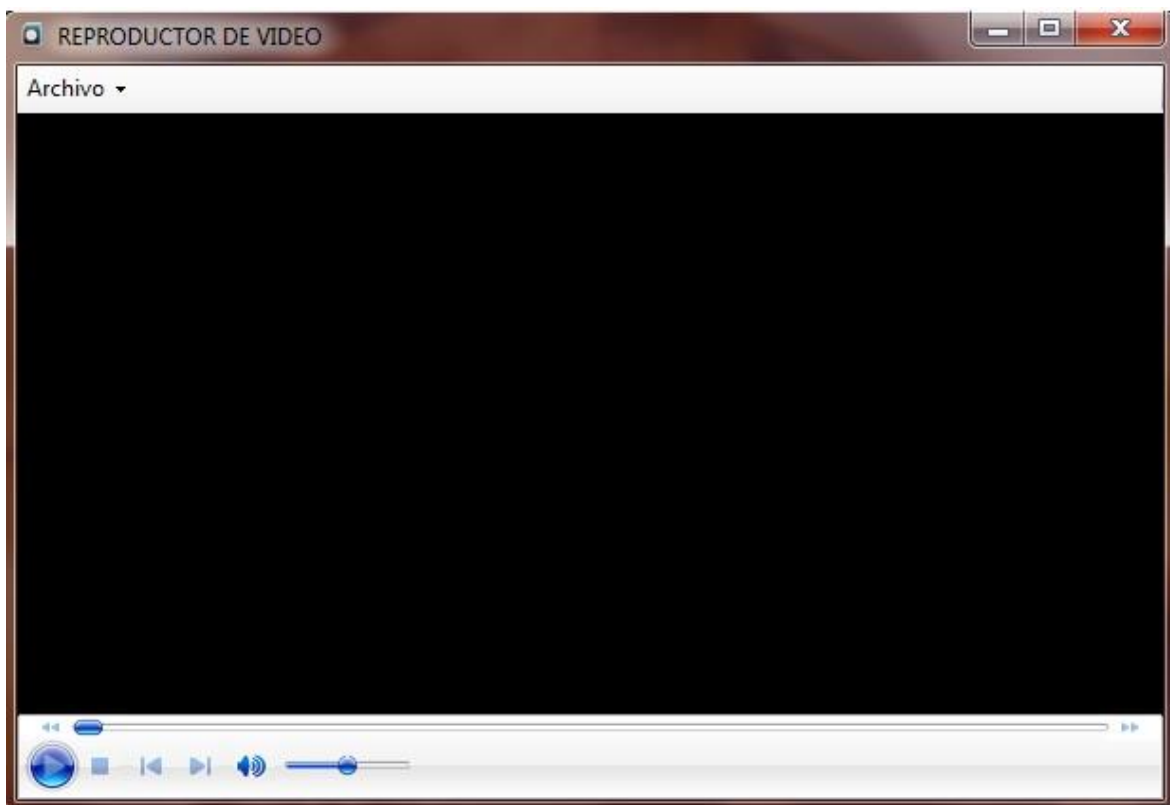


Fig. 4.4. Reproductor de Video

Ya que la mayoría de las cámaras IP cuentan con una opción de grabar video, se pensó que era buena idea hacer la implementación de un reproductor para todas las grabaciones que se llegasen a hacer.

El primer paso para crear el menú es un reproductor de video (Fig. 4.4) creado en Visual Studio 2010, para poder realizarlo se crea un nuevo archivo de Windows Forms y lo llamaremos "Reproductor de video", mediante las herramientas y componentes de Visual Studio aplicaremos el componente de Windows Media Player, también utilizaremos un OpenFileDialog y un ToolStrip.

Agregaremos primero al Forms llamado Reproductor de video, la versión completa del reproductor Windows Media Player nos permitió observar los videos en formatos como wmv, mp3, mpg y avi, que son los formatos más utilizados por las cámaras IP al hacer grabaciones, este complemento es una herramienta de Visual Studio 2010 muy útil para nuestra aplicación.

Teniendo el complemento dentro de Windows Forms colocaremos un ToolStrip que es la barra que utiliza Windows para desplegar listas de opciones en las aplicaciones, además agregaremos el complemento OpenFileDialog lo dentro de ToolStrip para poder abrir una ventana de dialogo, donde podremos tener acceso a los videos guardados con esto tenemos una manera sencilla de navegar dentro de Windows.

Agregado el complemento "OpenFileDialog" se programó el "ToolStrip" de la siguiente manera, se creó una lista con dos opciones: la primera con el texto de abrir y la segunda con el texto de salir.

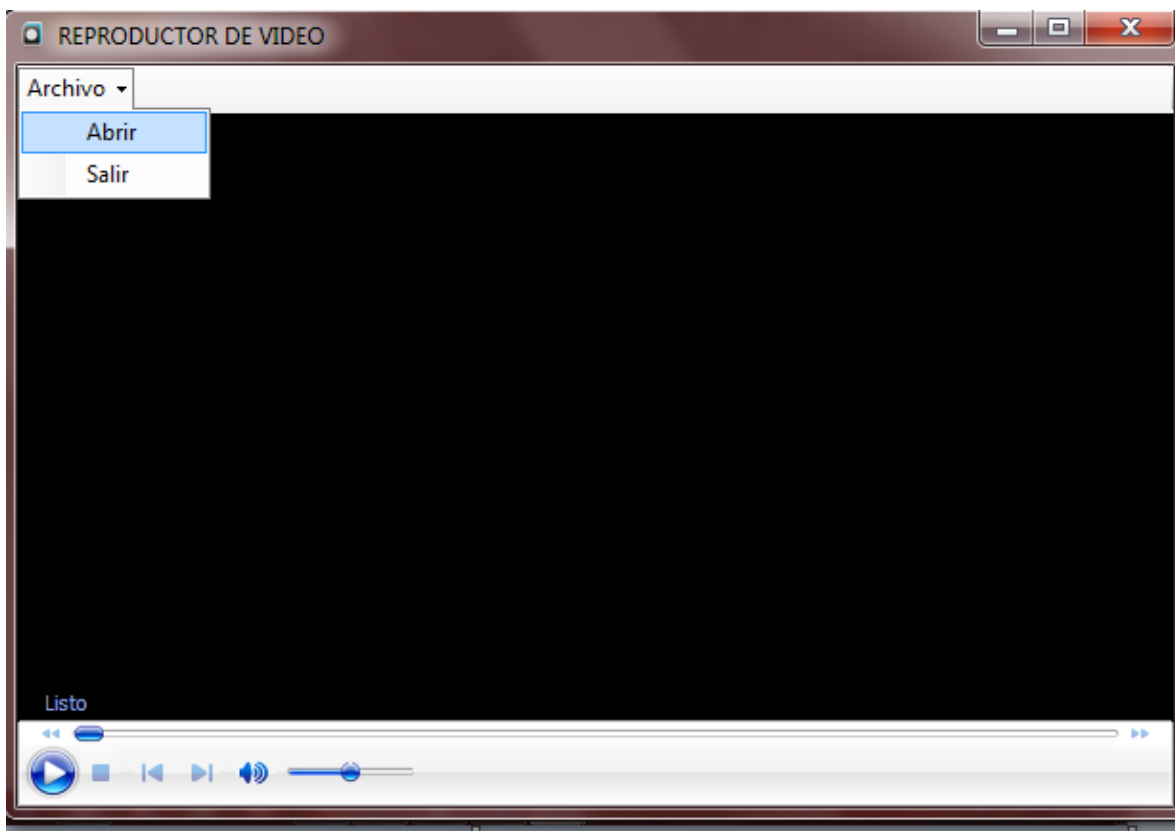


Fig. 4.5. Menú "Archivo" con opciones "Abrir" y "Salir"

Habiendo implementado las opciones de abrir y salir (Fig. 4.5), procedimos a implementar las funciones que van a desarrollar; empezamos con el proceso de abrir en donde usaremos la función "OpenFileDialog", ésta nos proporciona el acceso a las carpetas y a los archivos, con esto podemos reproducir los videos debido al complemento de Windows Media Player introducido anteriormente.

Continuamos con la programación ahora de la opción salir en este método solo tendremos que aplicar la sentencia de cerrar utilizada por Visual Basic 2010 para cerrar las ventanas.

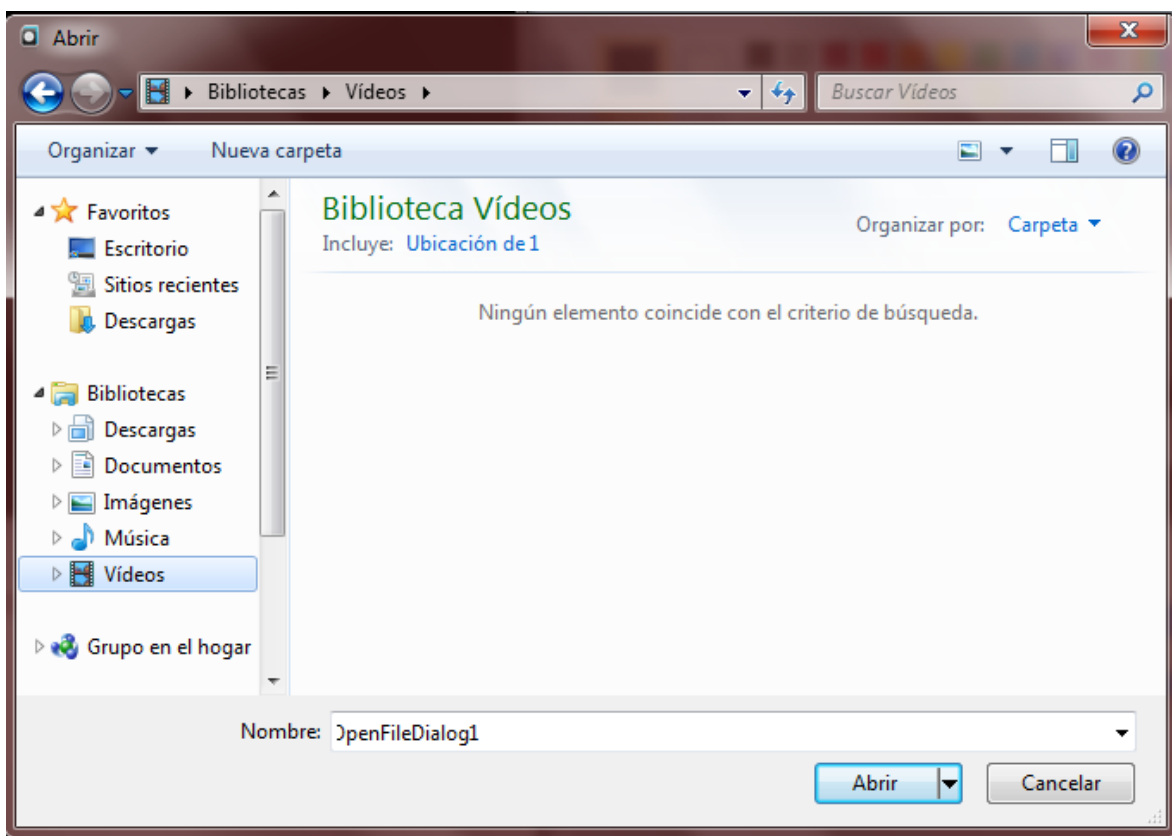


Fig. 4.6. Menú "Abrir"

En la Fig. 4.6 se muestra el resultado de la programación, esta es una ventana personalizada para poder navegar entre las carpetas como las que utiliza Windows.

Esta parte es de gran utilidad ya que evita la necesidad de abrir el reproductor que viene por defecto en Windows que requiere más recursos de la máquina, ya que ejecuta más procesos y el que se programó no, así mismo no perdemos la calidad y efectividad para poder reproducir los videos de las cámaras IP.

4.5. Escaneo de imagen



Fig. 4.7. Pantalla “Analizar Falla de Imagen”

Para poder implementar el análisis de imagen (Fig. 4.7) que se requiere para la aplicación, se consideró que la mejor opción era utilizar el programa MATLAB, ya que la interfaz que maneja es muy similar a la utilizada por Microsoft Visual Studio, también se consideró que debido a que se requiere el uso de funciones matemáticas, MATLAB es mejor que Microsoft Visual Studio y nos evitaría el escribir demasiado código para llegar al mismo fin. El método de correlación es el ideal para poder hacer la comparación de imágenes en que han sido convertidas en matrices.

La única desventaja con la que se encontró entre MATLAB y Visual Studio es en cuestión del desarrollo de su aplicaciones gráficas, en MATLAB el entorno grafico no es tan amigable como el de Visual Studio, tiene un ambiente grafico simple con pocas funciones, pero es compensado con la facilidad de poder realizar análisis matemáticos.

Para realizar nuestra aplicación de análisis de imagen se programó con el ambiente gráfico de MATLAB, para una mayor comodidad del usuario, ya que al implementarse en los botones de la aplicación se podrá analizar la imagen.

La programación de los botones fue la siguiente: en este ambiente grafico cada botón es una función, para empezar programaremos los primeros dos botones utilizando la función Callback, donde implementaremos el código para poder abrir una venta, en la cual podemos navegar dentro de las carpetas de Windows para poder obtener imágenes a ser analizarlas. Después obtener las imágenes muestra y la imagen a analizar, se tuvieron que guardar los valores a través de handles (función de almacenamiento), incluyendo la función guidata (hObject, handles) nos permitió pasar los valores de las imágenes convertidas en matrices.

Con los siguientes dos botones también se generaron dos nuevas funciones Callback, donde se procedió a guardar los valores de la imagen muestra y la imagen a analizar, pero convertidas en escala de grises (es necesario para que podamos utilizar nuestro método de correlación). Después de realizar la conversión de las dos imágenes a escala de grises utilizaremos nuevamente handles para poder almacenar nuestros valores, utilizaremos también guidata (hObject, handles) que nos permitirá utilizar los valores en la siguiente función.

En el último botón también se generará otra función Callback, donde implementaremos el método de correlación de imágenes ya mencionado en capítulo tres. Para implementar este método necesitamos obtener nuestros valores de las imágenes convertidas a escalas de grises, esto lo podremos realizar declarado dos nuevas variables, donde pasaremos los valores almacenados en nuestros handles, posteriormente declaramos dos nuevas variables de esta manera `a=double(A)`; esto nos permitirá poder trabajar con los valores necesarios para nuestro método.

Continuaremos con la lectura de nuestros valores antes de pasarlos al método de correlación, al haber realizado esto, procederemos a aplicar el método de correlación para poder comparar las imágenes convertidas en matrices. La única restricción para la utilización de este método es que las dos matrices deben ser del mismo tamaño para poder ser comparadas entre si.

Si nuestras imágenes después de ser evaluadas por el método de correlación muestran un resultado de la comparación menor a .80 tendremos un error en la imagen a analizar, esto se reflejara mediante un cuadro de aviso en donde nos indicara que la imagen tiene error. En caso contrario se desplegará un cuadro de aviso indicando que la imagen no tiene ningún error (Fig. 4.8).

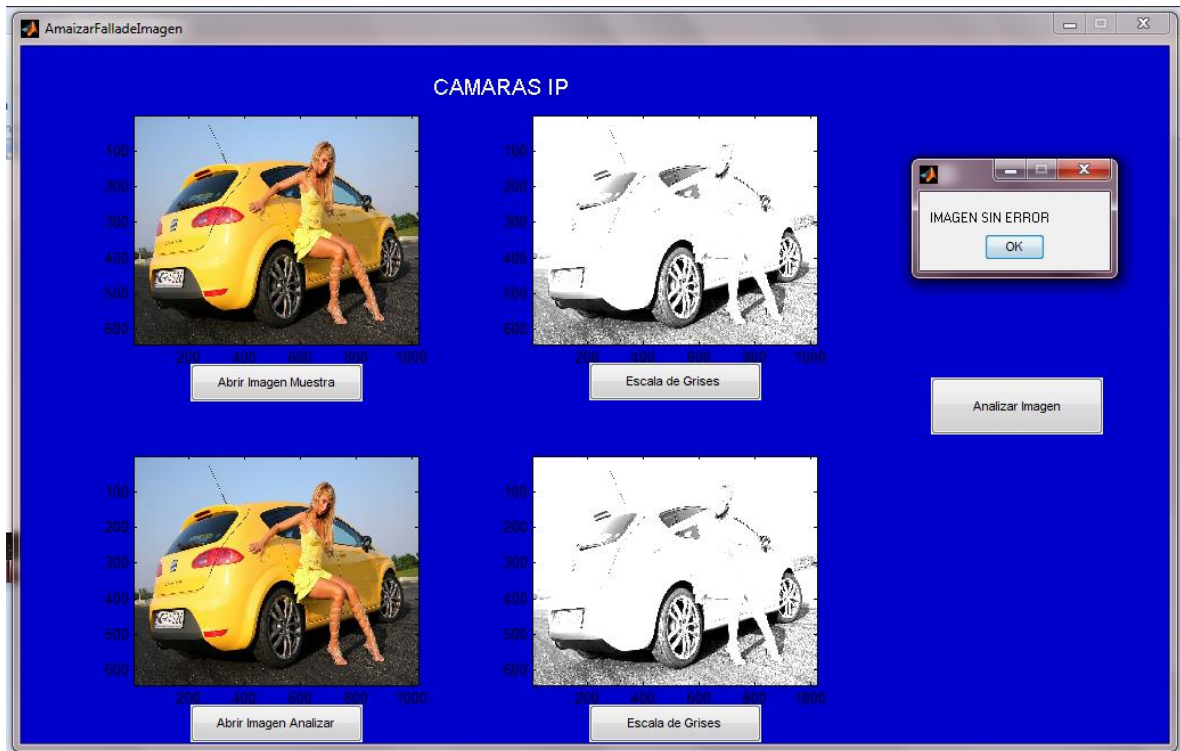


Fig. 4.8. Pantalla de demostración de análisis de Imagen

4.6. Gestion de Red

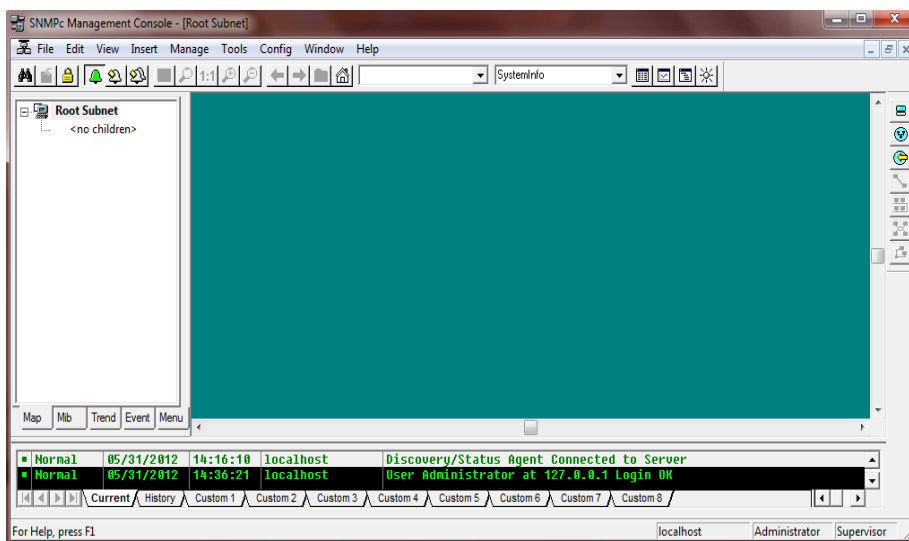


Fig. 4.9. Pantalla de inicio del programa SNMPc

Para poder hacer la gestión de la red se utilizará el programa SNMPc (Fig. 4.9) como complemento de la aplicación, esto debido a que es de las mejores aplicaciones para la gestión del protocolo SNMP que es uno de los protocolos con los cuales trabajan estas cámaras y de los más seguros.

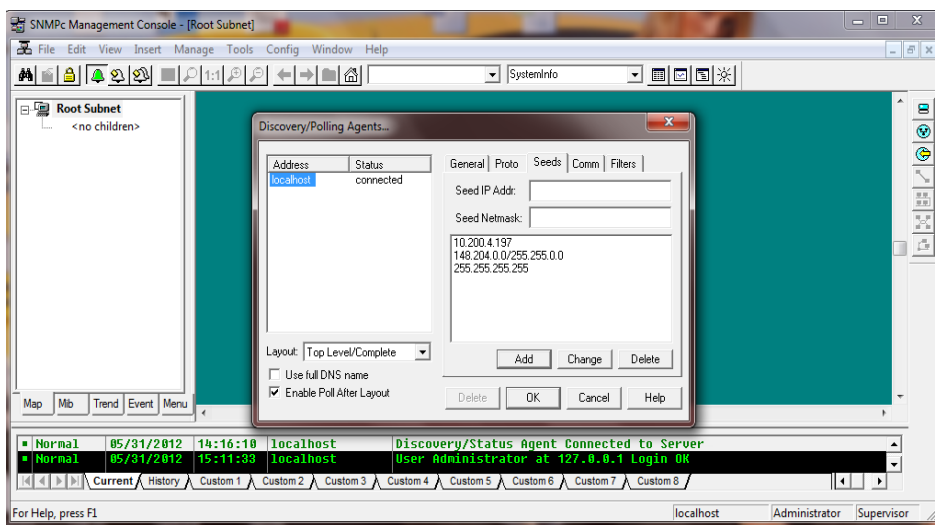


Fig. 4.10. Pantalla para ingreso de dirección IP en el programa SNMPc

En la pantalla principal colocamos la dirección IP local (Fig. 4.10) y snmpc realizó un mapa de la red con topologías y todo el contenido, esto es muy beneficioso para la aplicación, ya que así se pueden gestionar varias cámaras por medio de líneas de comando, por que la mayoría de los sistemas de vigilancia genera redes independientes de cada una de las cámaras y de los servidores.

4.7. Reportes

Direccion Ip	Marca de Camara	Modelo de la Camara	Tipo de Camara	Ubicacion de la Camara
192.168.0.18	Trendnet	TV-lp 110	fija exterior	Direccion general
192.168.0.19	Foscam	SKU-26358	fija interior	Jefatura de Ige
192.168.0.20	panasonic	BB-HCM521	fija exterior	Jefatura de Ige
192.168.0.21	sony	spn-cn23	fija	Biblioteca Nacion

Direccion Ip	Falla de Imagen	Falla de Red	Numero de Reincidencias	Numero de Folio	Fecha de Falla
192.168.0.18	Imagen borrosa ...	ninguna	2	123	15/05/12

Fig. 4.11. Pantalla de Reportes

Para poder tener un control de las fallas que presenten, se generó un sistema de reporte de fallas (Fig. 4.11), donde se maneja gran parte de la información sobre las cámaras IP, y de esta manera analizar si el sistema de seguridad tiene algún punto vulnerable, este sistema de reportes fue generado en el programa Visual Studio. Para poder realizar este sistema de reportes fue necesario crear una nueva aplicación, ya que contiene una conexión a una base de datos llamada Reporte, esta se encuentra alojada en SQL Server, para evitar algún conflicto de conexión se creó un nuevo proyecto el cual se llamara reporte de fallas de cámaras IP.

Una vez creado este nuevo proyecto, comenzaremos por dar el mismo formato que las aplicaciones anteriores, continuamos con la colocación de dos DataGridView que nos sirvieron para poder tener conexión con la base de datos y poder visualizar la información contenida en ella, para lograr esto se tuvo que configurar DataGridView mediante la opción "nueva conexión" que pertenece a este, después de seleccionar esta opción, se nos desplego una venta donde elegiremos el localhost, cuando la aplicación se conecta con SQL Server, otra ventana nos permitió observar todos los elementos que contenían las bases de datos, de esta manera seleccionamos la base de datos "Reporte" y sus elementos para así concluir la conexión con SQL Server, cuando se terminó este proceso permitió visualizar los campos y la información almacenada.

Para poder modificar esta información se tuvo que colocar en el nuevo proyecto los campos a manejar a través de varios “label”, donde se colocaron los nombres de los campos de nuestra base de datos y así identificar los datos que se manejan; en cada “label” colocamos “TextBox” para capturar los valores que deseamos ingresar, gestionar o modificar en nuestra base de datos, dimos formato a cada uno de los campos, después continuamos insertando seis botones que desempeñaran distintas funciones; sus nombres serán los siguientes: ingresar, buscar, actualizar, borrar, reporte, salir. Para que estos botones puedan ejecutar tareas, tendemos que configurar el “DataSet” a través de procesos.

El primero proceso que generamos fue el proceso de Insertar, para poder hacer esto se necesitó seleccionar una tabla de nuestra base de datos, y con la opción de crear una nueva consulta, se abrió una ventana con diversas opciones, en esta usamos la opción de “crear un nuevo proceso de almacenado”, una vez creado el proceso de almacenar pasaremos los primeros cinco valores de nuestra tabla “Descripción”, se repetirá el proceso para nuestra tabla “Fallas”.

El segundo proceso que se realizó fue el de Actualizar, para que pudiese ser llevado a cabo esta acción tendemos que seleccionar una de nuestras tablas y repetir los pasos antes mencionados para el proceso de “Insertar”, pero seleccionando “Update”. Con esta instrucción se actualizaron los cambios en nuestra tabla “Descripción”.

En el tercer proceso se realizó la función de buscar, es el mismo procedimiento que los dos procesos anteriores con la diferencia de haber usado la opción “Select”, donde sólo pasamos el valor de la llave primaria de nuestra base de datos que la denotaremos como Param1.

Para poder realizar la función de eliminar registros se tuvo que seleccionar la opción “Delete”, donde pasamos el valor de nuestra llave principal y así poder eliminar el mismo valor y todos los valores relacionados con ella, de igual manera utilizamos el valor obtenido de Param1 para poder efectuar esta función.

Teniendo todos los procesos, los utilizamos para poder programar nuestros primeros cuatro botones, los dos botones restantes los programamos más adelante; comenzamos por generar dos nuevos puntos de intuición que solamente usamos para los procesos de eliminar y de búsqueda, continuaremos primero con la programación de los botones de insertar, actualizar, ya que en estos dos botones la programación fue la misma: comenzamos por utilizar los valores de la tabla mediante la función “DescripcionTableAdapter”, enseguida colocamos el nombre del proceso que deseamos implementar, en este caso será la acción insertar datos; para que esto se lleve a cabo pasamos los valores capturados en nuestros “TextBox” insertados anteriormente los pasamos en forma de Param1, Param2... y para ser leídos usamos la misma función de la tabla pero esta vez usamos “fill”(Me.dataset. el nombre de nuestra tabla), al haber cumplido con esto, repetimos

el procedimiento para la tabla “Fallas”, aquí ingresaremos los valores restantes, empezando por el Param1,Param6...hasta el Param10, el motivo por el que pasamos nuevamente los valores de Param1 es debido a que la llave principal es la relación entre ambas tablas, al concluir limpiamos los campos con la propiedad .clear() de cada uno de los TextBox. Esto se repitió para el proceso de Actualizar, se cambió el acceso al proceso “Actualizar” y pasamos todos los valores de las tablas como el proceso anterior.

Continuamos la programación del botón “búsqueda”, a diferencia de los procesos anteriores solo introduciremos el valor de “Param1”, con este valor podremos visualizar todos los vales en nuestro “DataGridView”, que es la ventana directa hacia la base de datos, en la programación de esta usamos las mismas funciones, solo que llamamos al proceso de búsqueda que tendrá el valor de realizar la búsqueda que insertemos en TextBox1.

Para poder eliminar un registro usamos la misma programación de los botones anteriores, al borrar el valor de “Param1” en la tabla Fallas se eliminaron casi todos los valores a excepción del valor de la llave principal. Para que podamos eliminar este registro se tuvo que generar una nueva consulta, donde usaremos el comando de texto para eliminar sólo ese valor y lo llamamos de la misma manera que nuestra demás funciones.

Para poder programar nuestro botón de Reporte, se tuvo que generar el reporte mediante un complemento de Visual Studio llamado “CrystalReport”, este complemento se configuró dando una nueva conexión a nuestra base de datos, para hacer ésto, se tuvieron que desplegar las opciones secundarias al dar clic derecho en el campo de la base de datos, donde se nos desplego una ventana en la que se genera la nueva conexión hacia el localhost de SQL Server. En esta conexión encontraremos la base de datos y se obtuvieron las tablas que requerimos. Una vez obtenidos estos valores pasamos a dar formato al reporte, donde introduciremos los valores que requerimos que se presenten en el reporte; también se pueden agregar valores como fecha, hora, número de página, número de registro y colocar varias cosas para poder dar un mejor formato a nuestro reporte si así se desea.

Para poder visualizar nuestro reporte tendernos que crear un nuevo archivo de Forms, donde agregamos un CrystalReportViewer, que tuvimos que acoplar al tamaño del nuevo archivo de Forms, este se llamó “Imprimir Reporte”, con el ajuste terminado pasamos a utilizar a nuestro CrystalReport ya previamente configurado y con el formato deseado.

De esta manera observamos los valores de la base de datos que podrán ser impresos para ser analizados con más detalle. En la progresión solo tuvimos que llamar a “Imprime Reporte” y aparecerá la ventana con el reporte ya configurado (Fig. 4.12) y con todos los valores de nuestra base de datos.



Fig. 4.12. Reporte de Fallas para Imprimir

Por último programamos el botón de salir, donde usamos el comando `ME.close()`. Este comando nos permite cerrar por completo nuestra aplicación

La ventaja de incorporar un sistema de captura de reportes a la aplicación, es que permite tener un mejor control de las cámaras de vigilancia y las fallas que se presenten, con esta información podemos dictaminar si encontramos algún punto vulnerable en el sistema de vigilancia, también si alguna de las marcas de cámaras no es apta para ser implementada, ya que tal vez se llegasen a presentar demasiadas falla en un tiempo corto y se tendría que retirar.

4.8. Navegador web

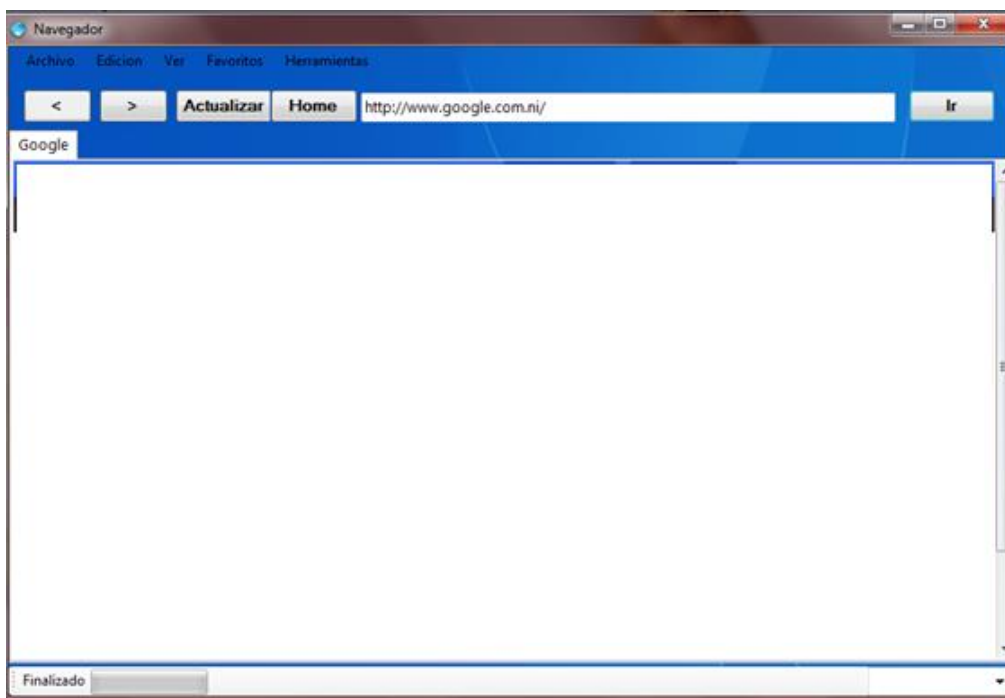


Fig. 4.13. Pantalla de Navegador Web

El navegador (Fig. 4.13) que se pensó implementar no es muy diferente a otros navegadores de uso común como Mozilla FireFox, Chrome, opera, etc. Este navegador lo usaremos para poder tener acceso a las cámaras dentro de nuestra red o fuera de ella. El navegador también se desarrollado utilizando Visual Studio, para ello utilizamos varias herramienta, sentencias ya mencionadas, estas sentencias solo serán mencionadas y se explicaran la nuevas herramientas, debido a que el funcionamiento y la programación son muy similares.

Comenzamos por generar un nuevo proyecto y lo nombramos “navegador web”, después colocaremos un “TabControl” (esta herramienta nos servirá para poder visualizar la páginas de internet, las cámaras IP), un “ProgressBar” (utilizado para poder ver los procesos que se van ejecutando o mientras carga la pagina), un “OpenFileDialog” y un “ToolStrip” para poder crear las opciones de nuestro navegador de la misma forma que creamos las opciones del reproductor de video, como se muestra en la imagen siguiente (Fig. 4.14):

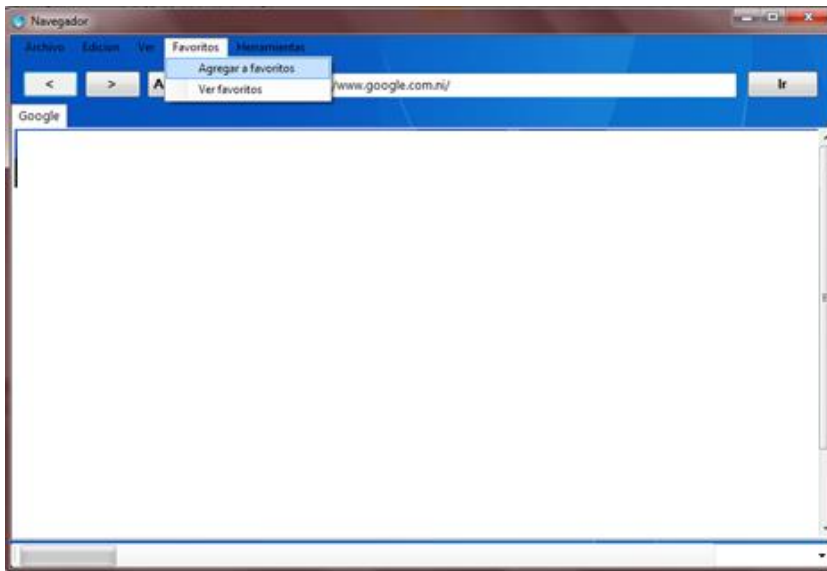


Fig. 4.14. Pestaña "Favoritos" del navegador

Después de programar cada opción al igual que nuestro reproductor, procedemos a crear nuevos Forms para poder generar las opciones y de esta manera puedan ser visibles en las ventanas independientes.

Estos nuevos Forms también pasaron por el proceso de formato, donde colocamos el fondo de nuestra aplicación y el icono correspondiente.

Una opción que se incluye en el navegador es la creación de favoritos (Fig. 4.15), donde podremos almacenar direcciones web de páginas que contienen cámaras IP públicas.

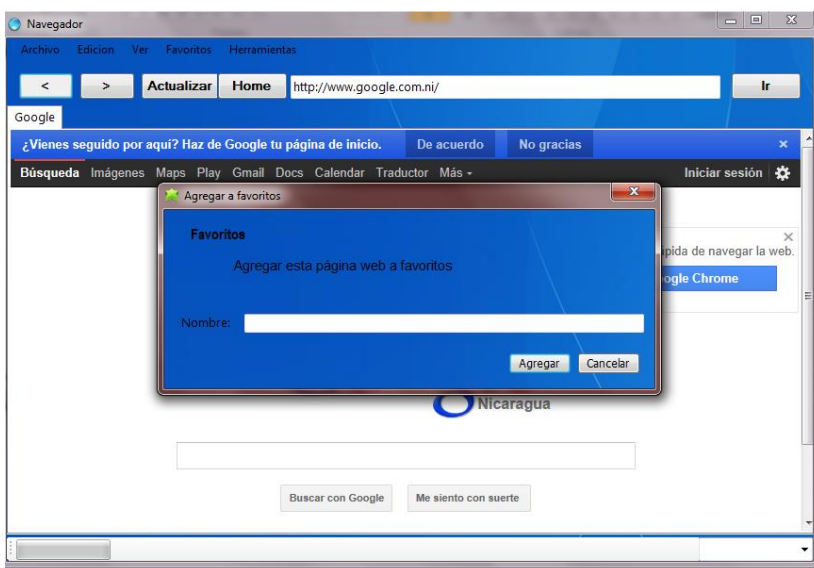


Fig. 4.15. Opción "Agregar Favoritos"

En la siguiente imagen (Fig. 4.16) se muestra como se puede registrar las direcciones IP de las cámaras, esta opción es un “Forms” nuevo, con sus botones, sus encabezados creados con un “label” y un “TextBox”, para capturar las direcciones o paginas que se deseen almacenar .

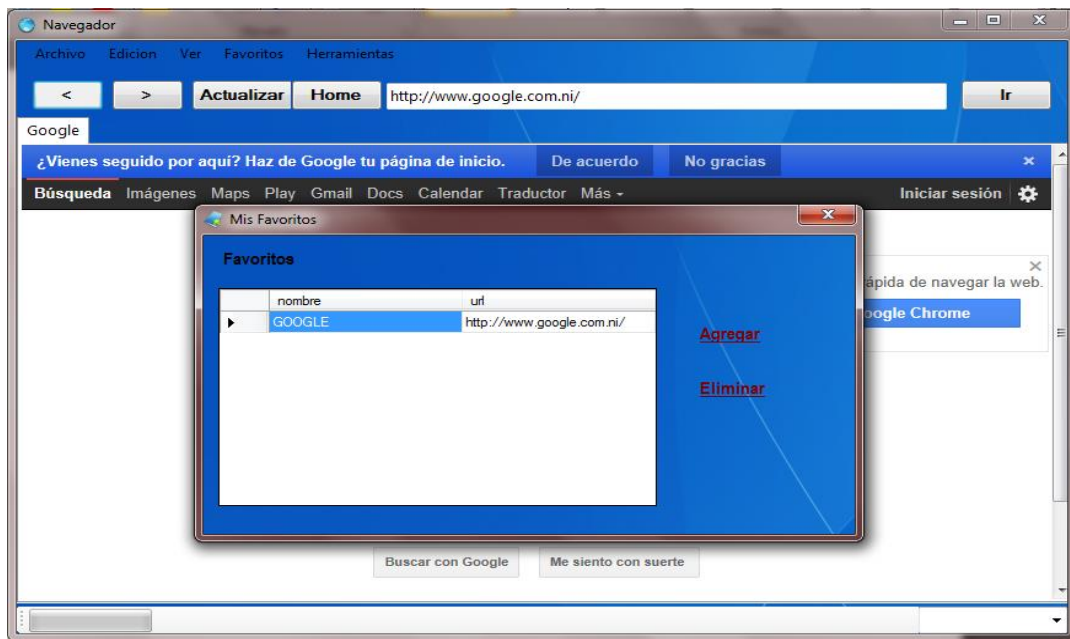


Fig. 4.16. Administración de “Favoritos”

Con este “Forms” desplegamos las direcciones almacenadas de las paginas que se deseen guardar y/o sean de interes para el usuario (como por ejemplo el manual de alguna de las camaras que se esta implemetando en el sistema de vigilancia).

Al hacer la implementación de los elementos mencionados anteriormente, se puede genera la aplicación para detectar fallas de caramas IP en un sistema de vigilancia ya establecido, esto es beneficioso para los usuarios para poder detectar las fallas que se presenten en dicho sistema, esto evitara un respuesta lenta al momento de realizar algún mantenimiento a las cámaras que integran este sistema de vigilancia.

CONCLUSIONES

La implementación de este sistema resulta muy útil para las empresas que consideren la seguridad dentro de sus instalaciones, la aplicación permite a estas empresas el poder tener un sistema con un mínimo de fallas en las cámaras IP que manejen. En caso que se llegase a presentar alguna falla en estas cámaras, el sistema podría determinar si esta falla que se presente es por parte de la conexión a la red o se debe a alguna falla de la imagen en esta. Con esto se puede optimizar el tiempo de mantenimiento que reciben estas cámaras, y poder solucionar esta situación de una manera más rápida.

Otro factor positivo es que cualquier persona con un mínimo conocimiento en computadoras puede hacer uso de esta aplicación, con esto si la persona encargada de manejar el sistema no llegase a asistir, cualquiera de sus compañeros podría hacer uso de la aplicación, esto teniendo un acceso valido.

Por lo tanto de nuestra parte se concluye que es un sistema muy completo, que debería ser tomado en cuenta para su implementación en todas las empresas que manejen seguridad a través de cámaras IP.

GLOSARIO TECNICO

Button: Herramienta de visual Studio que genera un botón interactivo

Callback: Nombre que asigna el ambiente grafico a las funciones ejecutables

CodRol: Nombre otorgado a un campo de la base de datos indicando en código del rol

CodUsuario: Nombre otorgado a un campo de la base de datos indicando en código de Usuario

ComboBox: Representa un control de cuadro combinado de Windows

CrystalReport: complemento de Visual Studio para poder generar reportes

CrystalReportViewer: Herramienta de Visual Studio que nos permite visualizar los reportes

DataGridView: Muestra los datos en una cuadrícula personalizable.

DataSet: un grupo de clases que describen una simple base de datos relacional en memoria

Delete: Eliminar datos

DescripcionTableAdapter: comunican la aplicación con una base de datos.

Fill: no se establecen varias propiedades de la DataTable y objetos de DataColumn (tales como claves principales, campos de incremento)

Forms : Crear aplicaciones basadas en Windows

guidata (hObject, handles): función para poder almacenar valores y heredarlos a otra función

handles : función de MATLAB para almacenar

label : proporciona una forma de mostrar texto debajo de los controles de programación

localhost : nombre reservado para el servidor que tienen todas las computadoras

ME.close() : función de Visual Studio para poder cerrar una aplicación

MsgBox: caja de mensajes

OpenFileDialog : de los formularios Windows Forms es un cuadro de diálogo preconfigurado

PictureBox : control para visualizar las imágenes

ProgressBar : herramienta utilizada para medir procesos

Select : seleccionar

SplashScreen : forma de la plantilla que se puede agregar a un proyecto de aplicación de Windows

Sqlclient : Representa un conjunto de métodos para crear instancias de la implementación de las clases de origen de datos

TabControl : expone los siguientes miembros. ... Apariencia, Obtiene o establece la apariencia visual de las fichas del control.

TextBox : caja de textos utilizada para obtener valores

ToolStrip : Proporciona un contenedor para los objetos de barras

Update: Actualizar datos

ANEXO

CÓDIGO DE PROGRAMACIÓN PARA MICROSOFT VISUAL STUDIO

PLANTILLA DE PRESENTACION

```
Public NotInheritable Class SplashScreen1
```

```
    'TODO: Este formulario se puede establecer fácilmente como pantalla de
    presentación para la aplicación desde la ficha "Aplicación"
    ' del Diseñador de proyectos ("Propiedades" bajo el menú "Proyecto").
```

```
    Private Sub SplashScreen1_Load(ByVal sender As Object, ByVal e As
    System.EventArgs) Handles Me.Load
```

```
        'Configure el texto del cuadro de diálogo en tiempo de ejecución según la
    información del ensamblado de la aplicación.
```

```
        'TODO: Personalice la información del ensamblado de la aplicación en el panel
    "Aplicación" del cuadro de diálogo
```

```
        ' propiedades del proyecto (bajo el menú "Proyecto").
```

```
        'Título de la aplicación
```

```
        If My.Application.Info.Title <> "" Then
```

```
            ApplicationTitle.Text = My.Application.Info.Title
```

```
        Else
```

```
            'Si falta el título de la aplicación, utilice el nombre de la aplicación
    sin la extensión
```

```
            ApplicationTitle.Text
```

```
            System.IO.Path.GetFileNameWithoutExtension(My.Application.Info.AssemblyName)
```

```
        End If
```

```
        'Dé formato a la información de versión usando el texto establecido en el
    control de versiones en tiempo de diseño como
```

```
        ' cadena de formato. Esto le permite una localización efectiva si lo desea.
```

```
        ' Se pudo incluir la información de compilación y revisión usando el
    siguiente código y cambiando el
```

```
        ' texto en tiempo de diseño del control de versiones a "Versión
    {0}.{1:00}.{2}.{3}" o algo parecido. Consulte
```

```
        ' String.Format() en la Ayuda para obtener más información.
```

```
        ,
```

```
        ,
```

```
            Version.Text = System.String.Format(Version.Text,
```

```
            My.Application.Info.Version.Major, My.Application.Info.Version.Minor,
```

```
            My.Application.Info.Version.Build, My.Application.Info.Version.Revision)
```

```

        Version.Text = System.String.Format(Version.Text,
My.Application.Info.Version.Major, My.Application.Info.Version.Minor)

        'Información de Copyright
        Copyright.Text = My.Application.Info.Copyright
    End Sub

    Private Sub ApplicationTitle_Click(sender As System.Object, e As System.EventArgs)
Handles ApplicationTitle.Click

    End Sub

    Private Sub MainLayoutPanel_Paint(sender As System.Object, e As
System.Windows.Forms.PaintEventArgs) Handles MainLayoutPanel.Paint

    End Sub
    Private Sub Copyright_Click(sender As System.Object, e As System.EventArgs)
Handles Copyright.Click

    End Sub
End Class

```

ACCESO

```

Imports System.Data.Sql
Imports System.Data.SqlClient

Public Class Acceso
    Public Nombre, Apellidos As String

    Private Sub btnAccesar_Click(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles btnAccesar.Click

        Try
            Dim dts As New Datos
            Dim func As New Funciones

            dts.nomusuario = txtUsuario.Text
            dts.passusuario = txtPassword.Text
            dts.rolusuario = cbRol.SelectedValue

            If func.validar(dts) = True Then

                Dim Principal As New Principal
                Principal.Show()

                Me.Hide()
            End If
        Catch ex As Exception
            MessageBox.Show(ex.Message)
        End Try
    End Sub
End Class

```

```

Else

    MsgBox("Error en el Usuario,Contraseña, o Rol",
MsgBoxStyle.Information)
    txtUsuario.Clear()
    txtPassword.Clear()

End If

Catch ex As Exception
    MsgBox(ex.Message)

End Try
End Sub

Private Sub cargar_combo(ByVal ComboBox As ComboBox, ByVal sql As String)
    Dim striconexion As String = " Data Source=ASPIREONE721\SQLEXPRESS; Initial
Catalog = DataBase; Integrated Security = True"
    Dim conexion As New SqlConnection(striconexion)

Try

    ' Abrimos la conexion a sql server

    conexion.Open()
    'Pasamos la consulta sql y la conexion al sql comand

    Dim cmd As New SqlCommand(sql, conexion)

    'Inicializar un nuevo sqlDataAdapter

    Dim da As New SqlDataAdapter(cmd)

    ' crear y Llamar el Dataset
    Dim ds As New DataSet
    da.Fill(ds)

    ComboBox.DataSource = ds.Tables(0)

    ' Asignar el campo ala propiedad DisplayMember Y Valumember del comboBox

    ComboBox.DisplayMember = ds.Tables(0).Columns(1).Caption.ToString
    ComboBox.ValueMember = ds.Tables(0).Columns(0).Caption

Catch ex As Exception

```



```

        MessageBox.Show(ex.Message.ToString, "Error", MessageBoxButtons.OK,
        MessageBoxIcon.Error)

    Finally
        If conexion.State = ConnectionState.Open Then
            conexion.Close()
        End If

    End Try

End Sub

Private Sub Acceso_Load(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles MyBase.Load

    cargar_combo(cbRol, "Select CodRol,TipoRol From Rol")
End Sub

Private Sub TextBox1_TextChanged(sender As System.Object, e As System.EventArgs)
Handles txtUsuario.TextChanged

End Sub
End Class

```

PAGINA PRINCIPAL

```

Imports System.Net
Imports System.Data.SQLite
Imports System.Xml

Public Class Principal

    #Region "enumerados"

    Private Enum Exec
        OLECMDID_OPTICAL_ZOOM = 63
        OLECMDID_CUT = 11
        OLECMDID_COPY = 12
        OLECMDID_PASTE = 13
        OLECMDID_DELETE = 33
        OLECMDID_SELECTALL = 17
        OLECMDID_FIND = 32
    End Enum

    Private Enum ExecOpt
        OLECMDEXEOPT_DODEFAULT = 0
        OLECMDEXEOPT_PROMPTUSER = 1
        OLECMDEXEOPT_DONPROMPTUSER = 2
    End Enum

```

```

OLECMDEXECOPT_SHOWHELP = 3
End Enum
#End Region

#Region "Variables"
Dim i As Integer = 0
Dim valorZoom As String = 100
#End Region

#Region "CadenaDeConexion"
Dim con As New SqlConnection("Data Source=|DataDirectory|\nav.db;User
instance=True")
#End Region

#Region "FuncionesAgregadas"
'Procedimiento NuevaPestaña
'modo = 0 ; entonces irá a la página de inicio
'modo = 1; entonces duplicara la pestaña
Public Sub NuevaPestaña(ByVal modo As Byte, url As String)
Dim browser As New WebBrowser
TabControl1.TabPages.Add("Nueva pestaña")
TabControl1.SelectTab(i)
browser.Name = "Navegador"
browser.Dock = DockStyle.Fill
TabControl1.SelectedTab.Controls.Add(browser)
AddHandler browser.ProgressChanged, AddressOf Loading
AddHandler browser.DocumentCompleted, AddressOf Done
i = i + 1
CType(TabControl1.SelectedTab.Controls.Item(0),
WebBrowser).ScriptErrorsSuppressed = True
If modo = 0 Then
CType(TabControl1.SelectedTab.Controls.Item(0),
WebBrowser).Navigate(LeerHome)
Else
CType(TabControl1.SelectedTab.Controls.Item(0), WebBrowser).Navigate(url)
End If

End Sub

Function LeerHome()
Dim myXmlDocument As New XmlDocument()
myXmlDocument.Load(My.Application.Info.DirectoryPath + "\home.xml")
Dim node As XmlNode
node = myXmlDocument.DocumentElement
Return node.ChildNodes.Item(0).InnerText
End Function

Private Sub Loading(ByVal sender As Object, ByVal e As
Windows.Forms.WebBrowserProgressChangedEventArgs)

```

```

        Try
            progreso_ProgressBar.Maximum = e.MaximumProgress
            progreso_ProgressBar.Value = e.CurrentProgress
        Catch ex As Exception

        End Try
    End Sub

    Private Sub Done(ByVal sender As Object, ByVal e As
Windows.Forms.WebBrowserDocumentCompletedEventArgs)
        TabControl1.SelectedTab.Text = CType(TabControl1.SelectedTab.Controls.Item(0),
WebBrowser).DocumentTitle
        urlTextBox.Text = CType(TabControl1.SelectedTab.Controls.Item(0),
WebBrowser).Url.ToString
    End Sub

    Private Sub HacerZoom(ByVal valor As String)
        Try
            Dim res As Object = Nothing
            Dim MyWeb As Object
            MyWeb = CType(TabControl1.SelectedTab.Controls.Item(0),
WebBrowser).ActiveXInstance
            MyWeb.ExecWB(Exec.OLECMDID_OPTICAL_ZOOM, ExecOpt.OLECMDEXECOPT_PROMPTUSER,
Integer.Parse(valor))
        Catch ex As Exception
            MessageBox.Show(ex.Message.ToString)
        End Try
    End Sub

    Public Sub AgregarAHistorial()
        Dim url As String
        url = urlTextBox.Text
        'Iniciar la conexion
        con.Open()
        'Insertamos un nuevo registro en la tabla historial
        Dim insertar As String = "insert into historial(url,fecha) values (' " + url +
" ', '" + Now.Date.ToString + "')";
        Dim comando1 As New SQLiteCommand(insertar, con)
        comando1.ExecuteNonQuery()
        ' Cerramos la Conexión
        con.Close()
    End Sub

#End Region

#Region "OtrosEventos"

    Private Sub TabControl1_Selected(ByVal sender As System.Object, ByVal e As
System.Windows.Forms.TabControlEventArgs) Handles TabControl1.Selected

```

```

        Try
            urlTextBox.Text = CType(TabControl1.SelectedTab.Controls.Item(0),
WebBrowser).Url.ToString
        Catch ex As Exception
        End Try
    End Sub

    Private Sub ComboBox1_SelectedIndexChanged(ByVal sender As System.Object, ByVal e
As System.EventArgs) Handles ComboBox1.SelectedIndexChanged
        valorZoom = Replace(ComboBox1.SelectedItem, Chr(37), "")
        HacerZoom(valorZoom)
    End Sub

    Private Sub Timer1_Tick(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles Timer1.Tick
        estadoLabel.Text = CType(TabControl1.SelectedTab.Controls.Item(0),
WebBrowser).StatusText
    End Sub

#End Region

#Region "EventoLoad"
    Private Sub Principal_Load(sender As System.Object, e As System.EventArgs) Handles
MyBase.Load
        TabControl1.TabPages.Add("Nueva pestaña")
        Dim browser As New WebBrowser
        browser.Name = "Navegador"
        browser.Dock = DockStyle.Fill
        TabControl1.SelectedTab.Controls.Add(browser)
        CType(TabControl1.SelectedTab.Controls.Item(0),
WebBrowser).ScriptErrorsSuppressed = True
        CType(TabControl1.SelectedTab.Controls.Item(0), WebBrowser).Navigate(leerHome)
        AddHandler browser.ProgressChanged, AddressOf Loading
        AddHandler browser.DocumentCompleted, AddressOf Done
        i = i + 1
    End Sub
#End Region

#Region "MenuArchivo"
    Private Sub AñadirPestañaMI_Click(sender As System.Object, e As System.EventArgs)
Handles AñadirPestañaMI.Click
        NuevaPestaña(0, Nothing)
    End Sub

    Private Sub DuplicarPestañaMI_Click(sender As System.Object, e As
System.EventArgs) Handles DuplicarPestañaMI.Click
        NuevaPestaña(1, urlTextBox.Text)
    End Sub

```

```

Private Sub QuitarPestañaMI_Click(sender As System.Object, e As System.EventArgs)
Handles QuitarPestañaMI.Click
    If Not TabControl1.TabPages.Count = 1 Then
        'Liberamos los recursos usados por el control
        CType(TabControl1.SelectedTab.Controls.Item(0), WebBrowser).Dispose()
        'Removemos el tabPage del tabcontrol
        TabControl1.TabPages.RemoveAt(TabControl1.SelectedIndex)
        'Actualizamos el tab seleccionado
        TabControl1.SelectTab(TabControl1.TabPages.Count - 1)
        i = i - 1
    End If
End Sub

Private Sub NuevaVentanaMI_Click(sender As System.Object, e As System.EventArgs)
Handles NuevaVentanaMI.Click
    Dim nueva As New Principal
    nueva.StartPosition = FormStartPosition.CenterParent
    nueva.Show()
End Sub

Private Sub AbrirMI_Click(sender As System.Object, e As System.EventArgs) Handles
AbrirMI.Click
    OpenFileDialog1.Title = "Abrir archivo"
    OpenFileDialog1.FileName = ""
    OpenFileDialog1.Filter = "Paginas web(*.html)|*.html|Paginas
web(*.mth)|*.mht|Paginas web(*.aspx)|*.aspx"
    OpenFileDialog1.ShowDialog()
    Dim url As String = OpenFileDialog1.FileName
    CType(TabControl1.SelectedTab.Controls.Item(0), WebBrowser).Navigate(url)
    urlTextBox.Text = url
    AgregarAHistorial()
End Sub

Private Sub GuardarComoMI_Click(sender As System.Object, e As System.EventArgs)
Handles GuardarComoMI.Click
    CType(TabControl1.SelectedTab.Controls.Item(0), WebBrowser).ShowSaveAsDialog()
End Sub

Private Sub ConfigurarPaginaMI_Click(sender As System.Object, e As
System.EventArgs) Handles ConfigurarPaginaMI.Click
    CType(TabControl1.SelectedTab.Controls.Item(0),
WebBrowser).ShowPageSetupDialog()
End Sub

Private Sub VistaPreviaMI_Click(sender As System.Object, e As System.EventArgs)
Handles VistaPreviaMI.Click
    CType(TabControl1.SelectedTab.Controls.Item(0),
WebBrowser).ShowPrintPreviewDialog()
End Sub

```

```

    Private Sub ImprimirMI_Click(sender As System.Object, e As System.EventArgs)
Handles ImprimirMI.Click
        CType(TabControl1.SelectedTab.Controls.Item(0), WebBrowser).ShowPrintDialog()
    End Sub

    Private Sub PropiedadesMI_Click(sender As System.Object, e As System.EventArgs)
Handles PropiedadesMI.Click
        CType(TabControl1.SelectedTab.Controls.Item(0),
WebBrowser).ShowPropertiesDialog()
    End Sub

    Private Sub SalirMI_Click(sender As System.Object, e As System.EventArgs) Handles
SalirMI.Click
        Me.Close()
    End Sub
#End Region

#Region "MenuEdicion"
    Private Sub CortarMI_Click(sender As System.Object, e As System.EventArgs) Handles
CortarMI.Click
        Dim MyWeb As Object
        MyWeb = CType(TabControl1.SelectedTab.Controls.Item(0),
WebBrowser).ActiveXInstance
        MyWeb.ExecWB(Exec.OLECMID_CUT, ExecOpt.OLECMDEXECOPT_DONPROMPTUSER)
    End Sub

    Private Sub CopiarMI_Click(sender As System.Object, e As System.EventArgs) Handles
CopiarMI.Click
        Dim MyWeb As Object
        MyWeb = CType(TabControl1.SelectedTab.Controls.Item(0),
WebBrowser).ActiveXInstance
        MyWeb.ExecWB(Exec.OLECMID_COPY, ExecOpt.OLECMDEXECOPT_DONPROMPTUSER)
    End Sub

    Private Sub PegarMI_Click(sender As System.Object, e As System.EventArgs) Handles
PegarMI.Click
        Dim MyWeb As Object
        MyWeb = CType(TabControl1.SelectedTab.Controls.Item(0),
WebBrowser).ActiveXInstance
        MyWeb.ExecWB(Exec.OLECMID_PASTE, ExecOpt.OLECMDEXECOPT_DONPROMPTUSER)
    End Sub

    Private Sub BorrarMI_Click(sender As System.Object, e As System.EventArgs) Handles
BorrarMI.Click
        Try
            Dim res As Object = Nothing
            Dim MyWeb As Object
            MyWeb = CType(TabControl1.SelectedTab.Controls.Item(0),
WebBrowser).ActiveXInstance

```

```

        MyWeb.ExecWB(Exec.OLECMDID_DELETE, ExecOpt.OLECMDEXECOPT_DONPROMPTUSER)
    Catch ex As Exception
        MessageBox.Show(ex.Message.ToString)
    End Try
End Sub

Private Sub BuscarMI_Click(sender As System.Object, e As System.EventArgs) Handles
BuscarMI.Click
    Dim MyWeb As Object
    MyWeb = CType(TabControl1.SelectedTab.Controls.Item(0),
WebBrowser).ActiveXInstance
    MyWeb.ExecWB(Exec.OLECMDID_FIND, ExecOpt.OLECMDEXECOPT_PROMPTUSER)
End Sub

Private Sub SeleccionarTodoMI_Click(sender As System.Object, e As
System.EventArgs) Handles SeleccionarTodoMI.Click
    Dim MyWeb As Object
    MyWeb = CType(TabControl1.SelectedTab.Controls.Item(0),
WebBrowser).ActiveXInstance
    MyWeb.ExecWB(Exec.OLECMDID_SELECTALL, ExecOpt.OLECMDEXECOPT_DONPROMPTUSER)
End Sub
#End Region

#Region "MenuVer"
Private Sub AumentarZoomMI_Click(sender As System.Object, e As System.EventArgs)
Handles AumentarZoomMI.Click
    If valorZoom = 1000 Then
        Return
    End If
    valorZoom += 25
    ComboBox1.Text = valorZoom.ToString + "%"
    HacerZoom(valorZoom)
End Sub

Private Sub DisminuirZoomMI_Click(sender As System.Object, e As System.EventArgs)
Handles DisminuirZoomMI.Click
    If valorZoom = 25 Then
        Return
    End If
    valorZoom -= 25
    ComboBox1.Text = valorZoom.ToString + "%"
    HacerZoom(valorZoom)
End Sub

Private Sub PaginaDeInicioMI_Click(sender As System.Object, e As System.EventArgs)
Handles PaginaDeInicioMI.Click
    CType(TabControl1.SelectedTab.Controls.Item(0), WebBrowser).GoHome()
End Sub

```

```

    Private Sub PaginaActualMI_Click(sender As System.Object, e As System.EventArgs)
Handles PaginaActualMI.Click
        CType(TabControl1.SelectedTab.Controls.Item(0),
WebBrowser).Navigate(urlTextBox.Text)
    End Sub
    Private Sub DetenerMI_Click(sender As System.Object, e As System.EventArgs)
Handles DetenerMI.Click
        CType(TabControl1.SelectedTab.Controls.Item(0), WebBrowser).Stop()
    End Sub

    Private Sub ActualizarMI_Click(sender As System.Object, e As System.EventArgs)
Handles ActualizarMI.Click
        CType(TabControl1.SelectedTab.Controls.Item(0), WebBrowser).Refresh()
    End Sub

#End Region

    Private Sub AtrasBtn_Click(sender As System.Object, e As System.EventArgs) Handles
AtrasBtn.Click
        CType(TabControl1.SelectedTab.Controls.Item(0), WebBrowser).GoBack()
    End Sub

    Private Sub AdelanteBtn_Click(sender As System.Object, e As System.EventArgs)
Handles AdelanteBtn.Click
        CType(TabControl1.SelectedTab.Controls.Item(0), WebBrowser).GoForward()
    End Sub

    Private Sub ActualizarBtn_Click(sender As System.Object, e As System.EventArgs)
Handles ActualizarBtn.Click
        CType(TabControl1.SelectedTab.Controls.Item(0), WebBrowser).Refresh()
    End Sub

    Private Sub HomeBtn_Click(sender As System.Object, e As System.EventArgs) Handles
HomeBtn.Click
        CType(TabControl1.SelectedTab.Controls.Item(0), WebBrowser).GoHome()
    End Sub

    Private Sub GoBtn_Click(sender As System.Object, e As System.EventArgs) Handles
GoBtn.Click
        'Si el campo URL esta vacio
        If urlTextBox.Text.Trim(" ") = "" Then
            Return
        End If
        'Navego a la pagina web
        CType(TabControl1.SelectedTab.Controls.Item(0),
WebBrowser).Navigate(urlTextBox.Text)
        AgregarAHistorial()
    End Sub

```



```

Private Sub BorraHistorialMI_Click(sender As System.Object, e As System.EventArgs)
Handles BorraHistorialMI.Click
    Dim resultado As DialogResult
    resultado = MessageBox.Show("Realmente desea borrar el historial?", "Navegador
web", MessageBoxButtons.YesNo, MessageBoxIcon.Question)
    If resultado = DialogResult.Yes Then
        'Iniciar la conexion
        con.Open()
        Dim consultaBorrado As String = "delete from historial"
        Dim comandoBorrar As New SQLiteCommand(consultaBorrado, con)
        comandoBorrar.ExecuteNonQuery()
        'Cerrar la conexion
        con.Close()
    End If
End Sub

Private Sub CodigoFuenteMI_Click(sender As System.Object, e As System.EventArgs)
Handles CodigoFuenteMI.Click
    Try
        Dim cliente As New WebClient

        Dim url As String = CType(TabControl1.SelectedTab.Controls.Item(0),
WebBrowser).Url.ToString
        Dim html As String = cliente.DownloadString(New Uri(url))
        Dim objVerSource As New VistaCodigo(html)
        objVerSource.Show()
    Catch ex As Exception
        MessageBox.Show("Sorry! " + ex.Message, "Navegador web",
MessageBoxButtons.OK, MessageBoxIcon.Error)
    End Try
End Sub

Private Sub PáginaDeInicioMI_Click(sender As System.Object, e As System.EventArgs)
Handles PáginaDeInicioMI.Click
    Dim paginaIni As New PaginaInicioD
    paginaIni.ShowDialog()
End Sub

Private Sub AgregarAFavoritosToolStripMenuItem_Click(sender As System.Object, e As
System.EventArgs) Handles AgregarAFavoritosToolStripMenuItem.Click
    Dim agregarFavD As New AgregarAFavoritosD
    agregarFavD.ShowDialog()
End Sub

Private Sub VerFavoritosToolStripMenuItem_Click(sender As System.Object, e As
System.EventArgs) Handles VerFavoritosToolStripMenuItem.Click
    Dim fav As New FavoritosD

```

```

        fav.ShowDialog()
    End Sub

    Private Sub HistorialToolStripMenuItem_Click(sender As System.Object, e As
System.EventArgs) Handles HistorialToolStripMenuItem.Click
        Dim histo As New HistorialD
        histo.ShowDialog()
    End Sub

    Private Sub MenuStrip1_ItemClicked(sender As System.Object, e As
System.Windows.Forms.ToolStripItemClickedEventArgs) Handles MenuStrip1.ItemClicked

    End Sub
End Class

```

MENU

```

Public Class Principal

    Private Sub PictureBox1_Click(sender As System.Object, e As System.EventArgs)
Handles PictureBox1.Click

    End Sub

    Private Sub Button5_Click(sender As System.Object, e As System.EventArgs) Handles
Button5.Click
        GetAttr("C:\Users\Gabo\Documentos\Visual Studio 2010\Projects\NavegadorWeb\NavegadorWeb\bin\Debug\NavegadorWeb.exe")
        Shell("C:\Users\Gabo\Documentos\Visual Studio 2010\Projects\NavegadorWeb\NavegadorWeb\bin\Debug\NavegadorWeb.exe")
    End Sub

    Private Sub Button1_Click(sender As System.Object, e As System.EventArgs) Handles
Button1.Click
        Reproductor.Show()
    End Sub

    Private Sub Button4_Click(sender As System.Object, e As System.EventArgs) Handles
Button4.Click
        GetAttr("C:\Users\Gabo\Documentos\Visual Studio 2010\Projects\Reporte de
fallas Camaras Ip\Reporte de fallas Camaras Ip\bin\Debug\Reporte de fallas Camaras
Ip.exe")
        Shell("C:\Users\Gabo\Documentos\Visual Studio 2010\Projects\Reporte de fallas
Camaras Ip\Reporte de fallas Camaras Ip\bin\Debug\Reporte de fallas Camaras Ip.exe")
    End Sub

```

```

    Private Sub Button6_Click(sender As System.Object, e As System.EventArgs) Handles
Button6.Click
        End
    End Sub

    Private Sub Button3_Click(sender As System.Object, e As System.EventArgs) Handles
Button3.Click
        GetAttr("C:\Program Files\SNMPc Network Manager\crcstart.exe"
Shell("C:\Program Files\SNMPc Network Manager\crcstart.exe")

    End Sub

    Private Sub Button2_Click(sender As System.Object, e As System.EventArgs) Handles
Button2.Click
        GetAttr("C:\Users\Gabo\Documentos\Visual Studio
2010\Projects\Camara\Camara\bin\Debug\Camara.exe")
        Shell("C:\Users\Gabo\Documentos\Visual Studio
2010\Projects\Camara\Camara\bin\Debug\Camara.exe")

    End Sub
End Class

```

NAVEGADOR

```

Imports System.Windows.Forms
Imports System.Xml

Public Class PaginaInicioD

    Dim url As String

    Private Sub cambiarHomePage()
        Dim myXmlDocument As New XmlDocument
        myXmlDocument.Load(My.Application.Info.DirectoryPath + "\home.xml")
        Dim node As XmlNode
        node = myXmlDocument.DocumentElement
        If node.Name = "home" Then
            node.ChildNodes.Item(0).InnerText = urlLabel.Text
        End If
        myXmlDocument.Save(My.Application.Info.DirectoryPath + "\home.xml")
    End Sub

    Private Sub OK_Button_Click(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles OK_Button.Click
        cambiarHomePage()
    End Sub

```

```

        Me.DialogResult = System.Windows.Forms.DialogResult.OK
        Me.Close()
    End Sub

    Private Sub Cancel_Button_Click(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles Cancel_Button.Click
        Me.DialogResult = System.Windows.Forms.DialogResult.Cancel
        Me.Close()
    End Sub

    Private Sub PaginaInicioD_Load(sender As System.Object, e As System.EventArgs)
Handles MyBase.Load
        url = CType(Principal.TabControl1.SelectedTab.Controls.Item(0),
WebBrowser).Url.ToString
        urlLabel.Text = url
    End Sub
End Class

```

REPORTE DE FALLAS

```
Public Class Reporte
```

```

    Private Sub Reporte_Load(sender As System.Object, e As System.EventArgs) Handles
MyBase.Load
        'TODO: esta línea de código carga datos en la tabla 'ReporteDataSet.Fallas'
Puede moverla o quitarla según sea necesario.
        Me.FallasTableAdapter.Fill(Me.ReporteDataSet.Fallas)
        'TODO: esta línea de código carga datos en la tabla
'ReporteDataSet.Descripcion' Puede moverla o quitarla según sea necesario.
        Me.DescripcionTableAdapter.Fill(Me.ReporteDataSet.Descripcion)

    End Sub

    Private Sub Button1_Click(sender As System.Object, e As System.EventArgs) Handles
Button1.Click
        Me.DescripcionTableAdapter.Insertar(TextBox1.Text, TextBox2.Text,
TextBox3.Text, TextBox4.Text, TextBox5.Text)
        Me.DescripcionTableAdapter.Fill(Me.ReporteDataSet.Descripcion)
        Me.FallasTableAdapter.Insertar2(TextBox1.Text, TextBox6.Text, TextBox7.Text,
TextBox8.Text, TextBox9.Text, TextBox10.Text)
        Me.FallasTableAdapter.Fill(Me.ReporteDataSet.Fallas)
        TextBox1.Clear()
        TextBox2.Clear()
        TextBox3.Clear()
        TextBox4.Clear()
        TextBox5.Clear()
        TextBox6.Clear()

```

```

        TextBox7.Clear()
        TextBox8.Clear()
        TextBox9.Clear()
        TextBox10.Clear()
    End Sub

    Private Sub Button2_Click(sender As System.Object, e As System.EventArgs) Handles
Button2.Click
        Me.DescripcionTableAdapter.Actulizar(TextBox1.Text,           TextBox2.Text,
        TextBox3.Text, TextBox4.Text, TextBox5.Text)
        Me.DescripcionTableAdapter.Fill(Me.ReporteDataSet.Descripcion)
        Me.FallasTableAdapter.Actualizar2(TextBox1.Text, TextBox6.Text, TextBox7.Text,
        TextBox8.Text, TextBox9.Text, TextBox10.Text)
        Me.FallasTableAdapter.Fill(Me.ReporteDataSet.Fallas)
        TextBox1.Clear()
        TextBox2.Clear()
        TextBox3.Clear()
        TextBox4.Clear()
        TextBox5.Clear()
        TextBox6.Clear()
        TextBox7.Clear()
        TextBox8.Clear()
        TextBox9.Clear()
        TextBox10.Clear()

    End Sub

    Private Sub Button3_Click(sender As System.Object, e As System.EventArgs) Handles
Button3.Click
        Me.FallasTableAdapter.Borrar(TextBox1.Text)
        Me.FallasTableAdapter.Fill(Me.ReporteDataSet.Fallas)
        Me.DescripcionTableAdapter.Borrar2(TextBox1.Text)
        Me.DescripcionTableAdapter.Fill(Me.ReporteDataSet.Descripcion)
        TextBox1.Clear()
    End Sub

    Private Sub Button4_Click(sender As System.Object, e As System.EventArgs) Handles
Button4.Click
        Me.DescripcionTableAdapter.FillBy(Me.ReporteDataSet.Descripcion,
        TextBox1.Text)
        Me.FallasTableAdapter.FillBy2(Me.ReporteDataSet.Fallas, TextBox1.Text)
        TextBox1.Clear()

    End Sub

```

```
Private Sub Button7_Click(sender As System.Object, e As System.EventArgs) Handles  
Button7.Click
```

```
    ImprimirRepor.Show()
```

```
End Sub
```

```
Private Sub Button5_Click(sender As System.Object, e As System.EventArgs) Handles  
Button5.Click
```

```
    Me.Close()
```

```
End Sub
```

```
End Class
```

REPRODUTOR

```
Public Class Reprodutor
```

```
Private Sub AbrirToolStripMenuItem_Click(sender As System.Object, e As  
System.EventArgs) Handles AbrirToolStripMenuItem.Click
```

```
    OpenFileDialog1.ShowDialog()
```

```
    AxWindowsMediaPlayer1.URL = OpenFileDialog1.FileName
```

```
End Sub
```

```
Private Sub SalirToolStripMenuItem_Click(sender As System.Object, e As  
System.EventArgs) Handles SalirToolStripMenuItem.Click
```

```
    Me.Close()
```

```
End Sub
```

```
End Clas
```

CÓDIGO DE PROGRAMACIÓN PARA MATLAB

```
function varargout = AmaizarFalladeImagen(varargin)
% AMAIZARFALLADEIMAGEN M-file for AmaizarFalladeImagen.fig
%     AMAIZARFALLADEIMAGEN, by itself, creates a new AMAIZARFALLADEIMAGEN or raises
the existing
%     singleton*.
%
%     H = AMAIZARFALLADEIMAGEN returns the handle to a new AMAIZARFALLADEIMAGEN or
the handle to
%     the existing singleton*.
%
%     AMAIZARFALLADEIMAGEN('CALLBACK',hObject,eventData,handles,...) calls the local
function named CALLBACK in AMAIZARFALLADEIMAGEN.M with the given input
arguments.
%
%     AMAIZARFALLADEIMAGEN('Property','Value',...) creates a new AMAIZARFALLADEIMAGEN
or raises the
%     existing singleton*. Starting from the left, property value pairs are
%     applied to the GUI before AmaizarFalladeImagen_OpeningFcn gets called. An
%     unrecognized property name or invalid value makes property application
%     stop. All inputs are passed to AmaizarFalladeImagen_OpeningFcn via varargin.
%
%     *See GUI Options on GUIDE's Tools menu. Choose "GUI allows only one
instance to run (singleton)".
%
% See also: GUIDE, GUIDATA, GUIHANDLES

% Edit the above text to modify the response to help AmaizarFalladeImagen

% Last Modified by GUIDE v2.5 01-Jun-2012 14:06:28

% Begin initialization code - DO NOT EDIT
gui_Singleton = 1;
gui_State = struct('gui_Name',       mfilename, ...
                  'gui_Singleton',   gui_Singleton, ...
                  'gui_OpeningFcn', @AmaizarFalladeImagen_OpeningFcn, ...
                  'gui_OutputFcn',  @AmaizarFalladeImagen_OutputFcn, ...
                  'gui_LayoutFcn',  [ ], ...
                  'gui_Callback',    [ ]);

if nargin && ischar(varargin{1})
    gui_State.gui_Callback = str2func(varargin{1});
end

if nargout
```

```

        [varargout{1:nargout}] = gui_mainfcn(gui_State, varargin{:});
    else
        gui_mainfcn(gui_State, varargin{:});
    end
% End initialization code - DO NOT EDIT

% --- Executes just before AmaizarFalladeImagen is made visible.
function AmaizarFalladeImagen_OpeningFcn(hObject, eventdata, handles, varargin)
% This function has no output args, see OutputFcn.
% hObject    handle to figure
% eventdata  reserved - to be defined in a future version of MATLAB
% handles     structure with handles and user data (see GUIDATA)
% varargin    command line arguments to AmaizarFalladeImagen (see VARARGIN)

% Choose default command line output for AmaizarFalladeImagen
handles.output = hObject;

% Update handles structure
guidata(hObject, handles);

% UIWAIT makes AmaizarFalladeImagen wait for user response (see UIRESUME)
% uiwait(handles.figure1);

% --- Outputs from this function are returned to the command line.
function varargout = AmaizarFalladeImagen_OutputFcn(hObject, eventdata, handles)
% varargout  cell array for returning output args (see VARARGOUT);
% hObject    handle to figure
% eventdata  reserved - to be defined in a future version of MATLAB
% handles     structure with handles and user data (see GUIDATA)

% Get default command line output from handles structure
varargout{1} = handles.output;

% --- Executes on button press in Abrir1.
function Abrir1_Callback(hObject, eventdata, handles) %#ok<DEFNU>
[nombre dire ]=uigetfile('*.jpg','Abrir Imagen de Muestra');
if nombre == 0
    return
end
muestra=imread(fullfile(dire,nombre));
axes(handles.axes1)
image(muestra)
handles.mut=muestra;
guidata(hObject,handles)

```



```

% --- Executes on button press in Abrir2.
function Abrir2_Callback(hObject, eventdata, handles) %#ok<DEFNU>
[nombre dire ]=uigetfile('*.jpg','Abrir Imagen para Analizar');
if nombre == 0
    return
end
analizar=imread(fullfile(dire,nombre));
axes(handles.axes2)
image(analizar)
handles.analiz=analizar;
guidata(hObject,handles)

% --- Executes on button press in gris1.
function gris1_Callback (hObject, eventdata, handles) %#ok<DEFNU>
im=handles.mut;
gr=rgb2gray(im );
axes(handles.axes3)
image(gr)
colormap gray
handles.muestra=gr;
guidata(hObject,handles)

% --- Executes on button press in gris2.
function gris2_Callback(hObject, eventdata, handles) %#ok<DEFNU>
in=handles.analiz;
gri=rgb2gray(in );
axes(handles.axes4)
image(gri)
colormap gray
handles.analizar=gri;
guidata(hObject,handles)

% --- Executes on button press in Analizar.
function Analizar_Callback(hObject, eventdata, handles) %#ok<DEFNU>
A=handles.muestra;
B=handles.analizar;
a=double(A);
b=double(B);
a = a - mean2(a);
b = b - mean2(b);
r = sum(sum(a.*b))/sqrt(sum(sum(a.*a))*sum(sum(b.*b)));
if r > .80;
    msgbox('IMAGEN SIN ERROR ' )
    if r<.80;
        msgbox(' ERROR EN IMAGEN ' )
    end
end
end

```

Costos de la aplicación

El costo total de desarrollo de la aplicación se reporta en las siguientes tablas.

Costos de Recursos Humanos

Producto o servicio	Cantidad		Total
		+	
Salario promedio mensual de un programador	12 Meses	\$15,000.00	\$180,000.00
		Tota de costos	\$180,000.00

Recursos Humanos: Ciento Ochenta Mil 00/100 pesos

Costo de Recursos Materiales

Producto o servicio	Cantidad	Costo	Total
Costo de Visual Studio 2010 Profesional	1	\$15,867.00	\$15,867.00
Coste de SQL Server 2008	1	\$24,999.00	\$24,999.00
Coste de SNMPC	1	\$ 10,950	\$10,950.00
Costo de Matlab	1	\$1,500.00	\$1,500.00
		Tola de costos	\$53,316.00

Recursos Materiales: Cincuenta y Tres Mil Trecientos Dieciséis 00/100 pesos.

La adquisición de los costos de recursos materiales y recursos humanos dio un costo de \$233,316.00 (Doscientos treinta y tres mil trescientos *dieciséis* 00/100 pesos).

Este cálculo fue hecho en base a un salario mensual que se paga a un programador Junior.

BASE DE DATOS SQL SERVER

Tabla de Usuario

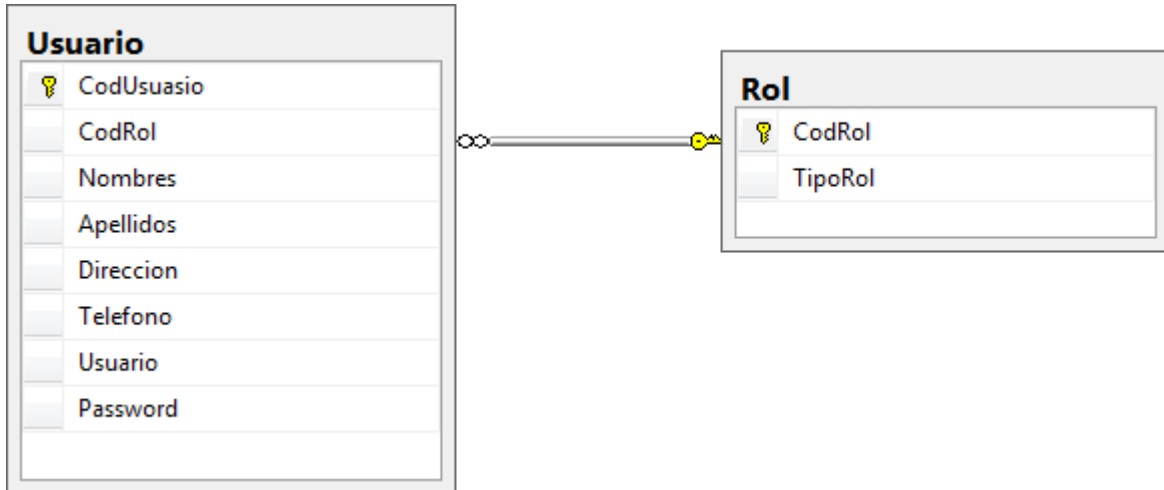
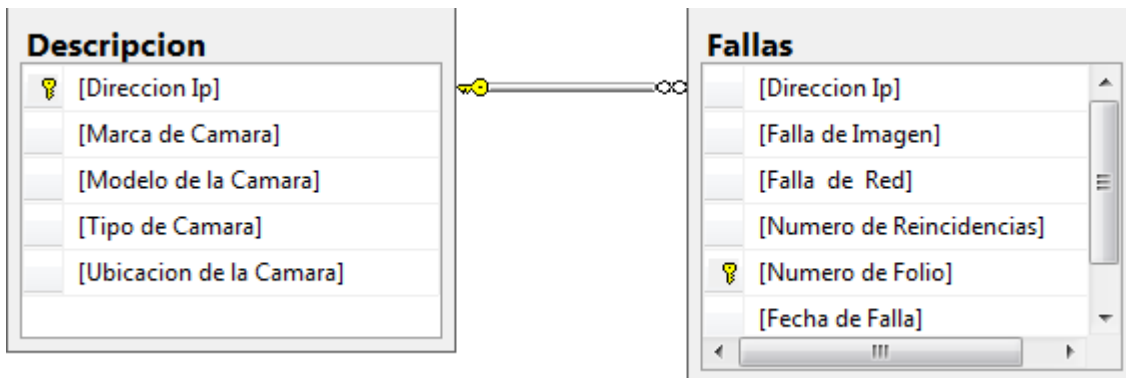


Tabla de Reporte de Fallas



BIBLIOGRAFIA

UNAM, Instituto de Ingeniería *Sistema de Video Vigilancia IP* México.1995 - 2007, <http://www.ii.unam.mx>.

BARAJA, Saulo *Curso de Protocolos de TCP/IP* <http://www.saulo.net/pub/tcpip>

BLACK, Uyless *Redes de Computadores Protocolos, normas e interfaces* Editorial Alfaomega.

EL ECONOMISTA S.A. de C.V. *Instalarán sistema de vídeo vigilancia en el*

Metro. Copyright © 1994-2000,
<http://www.economista.com.mx/sinprivilegios/articulos/2007-05-09-35931>.

STALLINGS, William *Comunicaciones y Redes de Computadores* Madrid, 2000. Editorial Prentice Hall.

TALENS, Oliag Sergio, **HERNÁNDEZ**, Orallo José *INTERNET Redes de computadores y sistemas de información* Segunda edición, 1998. Editorial Paraninfo.

Cisco Connection Online
<http://www.cisco.com>

SQL Server 2008
<http://www.microsoft.com/sqlserver/en/us/default.aspx>

Visual Studio
<http://www.microsoft.com/visualstudio/en-us>

Matlab
<http://www.mathworks.com/>

Java
<http://www.java.com/es/>

MySQL
<http://www.mysql.com/>

SNMPc
<http://www.castlerock.com/>