UNIVERSIDAD DE EL SALVADOR
FACULTAD MULTIDISCIPLINARIA DE OCCIDENTE
DEPARTAMENTO DE INGENIERIA Y ARQUITECTURA

TRABAJO DE GRADUACIÓN

TEMA:

"DISEÑO DE UN SISTEMA DE SEGURIDAD BASADO EN SENSORES CONTROLADOS MEDIANTE TECNOLOGÍA INFORMATICA, PARA LA FACULTAD MULTIDISCIPLINARIA DE OCCIDENTE DE LA UNIVERSIDAD DE EL SALVADOR"

PRESENTADO POR:

MANCIA MAGAÑA, TERESA ELIZABETH
QUINTANILLA MEDINA, KRISTEL MARIA

PARA OPTAR AL TÍTULO DE:

INGENIERO DE SISTEMAS INFORMÁTICOS

DOCENTE DIRECTOR:

ING. ERNESTO ALEXANDER CALDERON PERAZA

FEBRERO DE 2008
SANTA ANA, EL SALVADOR, CENTROAMÉRICA

UNIVERSIDAD DE EL SALVADOR

RECTOR:

MSC. RUFINO ANTONIO QUEZADA SÁNCHEZ

VICERRECTOR ACADÉMICO:

MSC. MIGUEL ÁNGEL PÉREZ RAMOS

VICERRECTOR ADMINISTRATIVO:

MSC. OSCAR NOÉ NAVARRETE

SECRETARIO GENERAL:

LIC. DOUGLAS VLADIMIR ALFARO CHÁVEZ

FACULTAD MULTIDISCIPLINARIA DE OCCIDENTE

DECANO:

LIC. JORGE MAURICIO RIVERA

VICEDECANO:

MSC. ELADIO EFRAÍN ZACARÍAS ORTEZ

SECRETARIO:

LIC. VÍCTOR HUGO MERINO QUEZADA

DEPARTAMENTO DE INGENIERIA

JEFE DEL DEPARTAMENTO:

ING. RAÚL ERNESTO MARTÍNEZ BERMUDEZ

DEDICATORIA

A mi Padre Celestial, que desde que me creo ha estado conmigo en cada paso y proyecto de mi vida. Infinitas gracias mi Dios por amarme como soy y con amor eterno.

A mi Madre Maria Santísima, porque me ha cuidado con esmero y cariño en todas las dificultades en mí caminar.

A mis Padres, Teresa Isabel de Mancia y Juan Adán Mancia, por su amor y apoyo incondicional, por su paciencia y el ejemplo que me dan cada día en luchar por la vida.

A mis hermanos, Luís Enrique y Juan José, por su apoyo y amor incondicional, y procurar ayudarme en alcanzar esta meta.

A mi compañera de Tesis, Kristel Quintanilla, por la amistad brindada y acompañarme en culminar esta meta.

A mis amigas y amigos, que me han acompañado y animado desde que comencé a emprender esta meta. Por que sin ellos no podríamos compartir las bondades que Dios nos regala en la vida.

A mi primo casi hermano Ricardo Polanco, porque a pesar de la distancia, me ha acompañado en este largo caminar.

Al Ing. Alexander Calderón, por brindarnos su amistad y apoyo, para la realización de nuestro trabajo de grado.

A mis compañeras(os) de estudio, porque hemos compartido este largo caminar y una buena amistad, en el transcurso de los años.

Al Ing. Francisco Segovia y el Téc. Juan José Aguilar, que son parte del equipo, de la empresa ALTESA, S.A. de C.V. Que nos compartieron sus conocimientos para el desarrollo de este proyecto.

Teresa Elizabeth Mancia Magaña.

"Ama y haz lo que quieras. Porque si es el Amor el que guía tu vida, realizarás grandes empresas."

DEDICATORIA

A Nuestro amoroso Padre Celestial y su Hijo amado Jesucristo, por que siempre han estado a mi lado y ayudado en cada paso que doy, para guiarme y bendecirme cada día.

A mis padres Luisa Elena Medina y Oscar Edgardo Quintanilla, por amarme, apoyarme y dirigirme en los momentos que los necesite, por ser mi guía constante en cada meta en mi vida.

A mis hermanos José Edgardo e Ingrid Elena, por ser mis compañeros y amigos, que me brindan su ayuda y comprensión cada vez que los necesito.

A mis amigos Elsa, Fernando, Los Inque, los hermanitos Mendoza, etc., por que siempre me han demostrado su amistad, cariño, aprecio, respeto y sobre todo apoyo en cada uno de los desafíos y oportunidades que he tenido.

A mi compañera de tesis Teresa Mancia, por su amistad y la oportunidad de haber compartido la culminación de esta meta.

A mis profesores, que a lo largo de mi vida me han ayudado a adquirir conocimiento e impartido consejos, los cuales me han sido y seguirán siendo de gran valor.

A mis amigos y amigas, compañeros de estudio y a quienes he conocido a lo largo de mi vida, los cuales agradezco por tantas alegrías, los excelentes consejos que he recibido y buenos momentos compartidos.

Kristel Maria Quintanilla Medina

"…Podemos ver que el señor en su infinita bondad bendice y hace prosperar a aquellos que en él ponen su confianza" (Helamán 12:1)

INDICE

INTRODUCCION

La era de la información, ofrece un contexto favorable para implementar, cambios tecnológicos, en cualquier área que se requiera un cambio, siempre y cuando los involucrados aprovechen las oportunidades que brindan las nuevas tecnologías de información. Como también pudiendo mejorar la eficiencia y eficacia de los resultados en la obtención de información útil, tratando de hacerlo experimental, innovador y económico; mejorando los procedimientos actuales de seguridad, que es el caso de este proyecto de graduación.

Por tal razón, la creación de un prototipo de sistema de seguridad que automatice e integre todo el proceso realizado por los custodios de la FMO, de modo que alberge una solución integral y completa vendría a ser de suma ayuda e importancia para la labor de resguardo y vigilancia de tecnología, instrumentos e información de alta prioridad.

Por lo tanto, en este documento se presenta lo que es el estudio, análisis y diagnóstico de la situación actual del sistema de seguridad en la FMO, para partir y establecer una alternativa de solución o ayuda a este problema. Dicha alternativa será propuesta como un prototipo de sistema, que se evaluara su funcionalidad, uso de tecnologías informáticas y sensoriales aplicadas en el mercado actual, su uso y confiabilidad, etc. Y sobre todo los beneficios que puede proporcionar a la seguridad institucional, en forma económica e innovador.

CAPITULO I:

"ESTUDIO PRELIMINAR"

ANTECEDENTES

La unidad de Custodios de la FMO, también conocida como Unidad de Vigilancia, tiene como función principal cuidar el patrimonio universitario, siendo esta una Unidad importante para la FMO. En este apartado se expondrá una síntesis de la trayectoria de esta unidad en la Facultad a partir de 1992 hasta la fecha.

En el año de 1992 estaban supervisados y coordinados por el Decanato de la Universidad. En el año de 1993 toma un poco más de forma contratando a 6 elementos más, nombrando como Jefe de personal de Custodios al Sr. Rigoberto Olivares, a partir de este año comienzan a darse pequeños pasos para mejorar esta unidad que en su segundo año en función, aún no contaba con uniformes y mucho menos con un equipo adecuado para realizar su trabajo, debido a esta falta de equipo debían emplear métodos rudimentarios como silbidos, gritos y armas hechizas, para desempeñar las funciones que su cargo les exigía.

Iniciando el año de 1995, después de un par de debates, logran obtener algunos implementos para desempeñar su trabajo, específicamente un CORBO por empleado. Desde este año se comienza a programar de mejor manera las rondas y los sectores para vigilar, dividiendo el campus en zonas que se detallan de la siguiente manera:

- Centro de la Facultad.
- Sector desde Departamento de Física hasta el Auditórium.
- Sector desde Bunker hasta el parqueo.
- Sector cafetería (hoy edificio de Usos Múltiples y Medicina).
- Bosque o Zona Verde.
- Edificio de Agua y Talleres.
- Canchas deportivas y gimnasio.

Se programan los turnos de manera de repartir equitativamente la carga sobre cada uno de los miembros del personal, cada miembro tendrá tres turnos por

semana: 2 de turnos de 12 horas y uno de 16 horas, rotando de manera regular al personal para evitar caer en un cansancio extremo debido a la carga laboral.

A inicios del año 2000 por iniciativa del jefe en función, el señor Rigoberto Olivares, propone un plan de trabajo en el cual se mencionan mejoras específicas entre las cuales podemos mencionar 3 capacitaciones al año (mejoramiento de la seguridad, primeros auxilios y relaciones humanas) las cuales no tuvieron seguimiento. A finales del mismo año, se recibe la primer capacitación, la cual fue impartida por un subinspector de la PNC dicha capacitación fue denominada *"Mejoramiento de Seguridad".*

A mediados del 2001 dentro de la FMO, se realizaron reuniones para formar foros de discusión de carácter informativo, acerca de los desastres ocasionados por los terremotos recién pasados y los mecanismos de solución hacia los mismos, participando diferentes personalidades, que hicieron el llamado de atención a las autoridades sobre la falta de seguridad en el parqueo de la institución. Como iniciativa por parte de algunos participantes en dicho forum, se realiza la donación de 3 silbatos los cuales son utilizados por la Unidad de Custodios desde esa fecha hasta hoy en día.

En el 2003, con el cambio de autoridades en la FMO se hace un reordenamiento de personal en los distintos departamentos incluyendo la Unidad de Custodios quedando como nuevo jefe el Sr. Omar García, quien permanece en el puesto hasta el presente año. Durante su gestión se realizo una capacitación denominada: *"Autoestima y Trato con las personas",* impartida por una psicóloga del ISSS.

A finales del 2004 se inicia la construcción de la fachada de la FMO, con la finalidad de proporcionarle a ésta una apariencia digna de su estatus como institución de educación superior. Esta construcción, produjo ciertas reacciones dentro de la Unidad de Custodios, quienes han manifestado haber sentido

incrementada su carga laboral, debido a que a la fecha no se ha reemplazado a dicho elemento. Siendo hasta principios del 2005 que el jefe en funciones, Sr. Omar García, es notificado que se espera la contratación de nuevos elementos para lograr equiparar la carga laboral de la unidad de Custodios.

En el presente año 2006, el jefe en funciones de la unidad de custodios es el Sr. Luís Alonso Rivera Marroquín, quien manifiesta que el desafío primordial que presenta esta unidad actualmente, es poseer un ente eficiente que llene las necesidades de la FMO; lo cual puede llevarse solo a cabo con el aumento de personal, materiales o herramientas y modernización de aparatos, que ayudan a la custodia del patrimonio universitario.

PLANTEAMIENTO DEL PROBLEMA

Actualmente, la FMO cuenta con un personal encargado de la vigilancia capacitado para brindar un buen servicio; sin embargo, insuficiente para cubrir y satisfacer todas las necesidades concernientes a la seguridad dentro del plantel universitario. El ingenio del hombre y el desarrollo de la ciencia y tecnología han sido capaces de minimizar éstos problemas. Los grandes avances de la electrónica y las computadoras, son muestra increíbles del potencial de la mente humana, que han contribuido enormemente al desarrollo de los sistemas de seguridad.

Al recorrer la historia de éstos, desde sus inicios muy rudimentarios, al actual en donde se puede combinar el recurso humano con el avance tecnológico las personas manifiestan sentirse más seguras, ya que se apoyan de dichas herramientas para mejorar sus servicios.

La FMO no debe ser la excepción en cuanto a modernizar sus sistemas de vigilancia. Debido a que se ha visto amenazada por una serie de eventos delincuenciales como: Hurto de equipo informático, artículos personales, extracción forzada de componentes de vehículos (dentro del parqueo de la Institución), intrusión de personas ajenas a la universidad que causan daños a la misma. Así como en el caso presentado en el Alma Mater en San Salvador, en donde un grupo de manifestantes se introdujo violentamente en la universidad tomando posesión de las instalaciones, por la falta de personal que brinde seguridad en las áreas de ingreso a la universidad.

Desde la creación de los nuevos edificios en el campus universitario, como lo son: el edificio de medicina, el edificio de usos múltiples y la fachada de la facultad, se ha incrementado enormemente la adquisición de nuevo equipo y materiales didácticos de mucho valor; por lo cual aumentaron las necesidades y expectativas de seguridad, como lo es la carga de trabajo para los pocos miembros de la

unidad de Custodios. Así como podemos mencionar, el ejemplo en el edificio de Medicina, ya que con las nuevas instalaciones e infraestructura, adquirieron nuevos equipos, materiales didácticos y de oficina; por lo cual la Sociedad de Padres de Familia de este departamento, hizo la adquisición de un servicio de seguridad de alarmas de una empresa privada, para el resguardo de laboratorios, oficinas y aulas de dicho departamento, que hasta la fecha ya no es utilizado o activado por ninguna persona a cargo, debido a la negligencia y falta de capacitación para su uso.

En lo que refiere a la construcción de la fachada, expresan los custodios, que ha incrementado su carga de trabajo, por la mala distribución de las entradas de acceso a las instalaciones de la facultad y ubicación de la caseta de seguridad, incrementando también la capacidad de vehículos a resguardar, sin poseer un sistema de pluma conveniente.

La poca vigilancia e inseguridad que presenta la institución desde la perspectiva antes mencionada, es sin duda, por la falta de una buena estrategia de vigilancia; ya que, el poco personal de seguridad con que cuenta la institución no permite satisfacer a plenitud cada una de las demandas de un sistema de vigilancia bien estructurado, tomando en consideración el espacio territorial que hay que abarcar. Además la estructura física de las instalaciones se encuentra muy deteriorada (interna y externa), ejemplo: puertas, ventanas y vallado, la cual facilita la intrusión de personas desde las viviendas o lugares aledaños, para la extracción o hurto del patrimonio universitario y artículos personales, creando un ambiente de desconfianza entre los estudiantes y trabajadores. Por lo que dichos sucesos han aumentado su frecuencia en los últimos años.

Por tal razón, no se cumplen todas las expectativas de un buen servicio de seguridad, ya que el equipo de custodios es insuficiente para dar vigilancia total al patrimonio de la facultad, surge la necesidad de proponer alternativas diferentes que permitan brindar un ambiente de seguridad entorno a las instalaciones de la

FMO, en la medida que se logre hacer un plan sin costos elevados, con buenos resultados a corto plazo. Como tal, se propone un sistema de seguridad informática con tecnología sensorial que permite al personal encargado de la seguridad mantener en resguardo los distintos puntos o lugares vulnerables de la institución, lo que proporciona un apoyo sumamente valioso para desempeñar sus funciones, garantizando un menor grado de riesgo de que se lleve a cabo un acto ilícito dentro de las instalaciones.

OBJETIVOS

GENERAL

- ✓ Estudiar la problemática de la inseguridad institucional y resguardo del patrimonio de la FMO, mediante un análisis y estudio científico, generando una propuesta novedosa de seguridad basada en tecnologías sensoriales e informáticas que brinden una solución al fenómeno.

ESPECÍFICOS

- ✓ Investigar las tecnologías mas adecuadas en sensores y dispositivos electrónicos utilizados para este tipo de aplicaciones, y su disponibilidad en el mercado local.

- ✓ Analizar la situación referente al ambiente de seguridad que se posee en la FMO sus fortalezas, debilidades, logros y errores.

- ✓ Ejecutar un prototipo de dicho sistema, aplicado a un área específica de las instalaciones de la FMO, para demostrar su funcionalidad y credibilidad.

- ✓ Diseñar la distribución física de los dispositivos sensoriales y el cableado de las áreas de vigilancia de la institución.

- ✓ Generar una propuesta de fortalecimiento e inversión que demuestre la factibilidad de dicho sistema de seguridad, apegada a la realidad de nuestro medio universitario.

- ✓ Diseñar un software basado en el lenguaje de VisualBasic.Net, para el control y mantenimiento de los dispositivos de seguridad o resguardo utilizados.

- ✓ Diseñar una base de datos, que se adapte a la información que se verá involucrada en los procesos identificados.

ALCANCES

- ✓ El software que se producirá será un prototipo de características seleccionadas, el cual consiste en un modelo funcional que incluya algunas, pero no todas, de las características que tendrá el sistema final; para comprobar su desempeño en controlar dichos dispositivos, si bien no será una implantación completa en todo el campus, pero brindará una funcionalidad absoluta en un área particular propia para el prototipo.

- ✓ El modelo desarrollo de software que se pretende utilizar es el orientado a objetos, con la tecnología conocida como Microsoft.NET se desarrollará la codificación y para el diseño se empleará el lenguaje unificado de modelado.

- ✓ Se realizará un estudio de la seguridad institucional, en el cual se utilizará a la unidad de custodios, administrativos y estudiantes, para el análisis del estado de seguridad de la FMO e identificación de puntos importantes o vulnerables de las instalaciones.

- ✓ En base al punto anterior, se podrá reforzar dicho trabajo con las sugerencias propuestas por ellos mismos.

- ✓ Se investigará las diferentes alternativas existentes en el mercado, sobre dispositivos sensoriales y electrónicos que se adapten a las necesidades y recursos de la institución.

- ✓ Se pretende diseñar una propuesta de distribución física y estructural de los dispositivos sensoriales y electrónicos, en distintas áreas, oficinas y departamentos de la FMO.

- ✓ Se estructurará una propuesta de inversión, para demostrar la factibilidad del proyecto.

- ✓ Se demostrará por medio de una prueba piloto en el edificio de Usos Múltiples de la FMO, la funcionalidad de dicha propuesta.

- ✓ Se documentará adecuadamente la aplicación a desarrollar, al término de la elaboración del trabajo de grado.

LIMITACIONES

- ✓ El factor humano es uno de los mayores obstáculos para implementar un sistema de seguridad de este tipo, ya que las personas o usuarios del mismo, serán los vigilantes, quienes carecen de conocimientos en computación o informática para el manejo de dicho software, y muchas veces tampoco disponen de voluntad o la aptitud para aprender a utilizarlo, pues humanamente se teme al cambio. Por lo cual, se documentará adecuadamente al personal encargado del sistema e incluyendo una capacitación para el manejo de dicho sistema y dispositivos.

- ✓ Debido a la magnitud o dimensiones de las instalaciones de la Universidad y del tiempo que se dispone, no podrá encaminarse hasta la implementación total ya que esta fuera de los alances antes mencionados. Por lo tanto, se elaborara un prototipo con las características principales que demuestre su funcionalidad.

- ✓ El software que se produzca como producto del trabajo de grado será para uso de la Universidad de El Salvador, por lo que no se aplicará a instituciones de carácter privado.

- ✓ El poco interés por las autoridades sobre herramientas tecnológicas de este tipo. En donde se basará una investigación mas avanzada en el mercado del país o empresas privadas.

- ✓ Para la investigación de este trabajo de grado, se carece actualmente de la colaboración en función de equipo por parte de la FMO, por lo que se esta sujeto a las limitantes económicas de los estudiantes que lo llevarán acabo en la adquisición de equipo.

- ✓ Para la implementación de dicho prototipo estará a cargo por parte de la UES, por su responsabilidad en el aspecto económico; que esta sujeto a su burocracia.

JUSTIFICACIÓN

En la situación actual de nuestras sociedades, nos resulta no un lujo sino una necesidad, el proteger todo material intelectual, equipo valioso o bienes materiales, que son utilizados para el desempeño de nuestras actividades cotidianas, las cuales entre mas valiosas están mas propensas al hurto o daño de parte de otras personas, ya sea de empresas privadas o publicas.

Actualmente en la Facultad Multidisciplinaria de Occidente existen muchas deficiencias, y dificultad en la seguridad de las oficinas, materiales y equipo que pertenecen a esta, incidiendo ya en varias ocasiones el robo de equipo valioso, agregando a esto que el personal de seguridad con que cuenta para las dimensiones del lugar son muy pocos y los puntos vulnerables de dichas instalaciones son muchos.

Por tal motivo, se propone el empleo de tecnología sensorial e informática, muy utilizada por el mercado nacional, por su economía, eficiencia y eficacia, para espacios reducidos o amplios, según se presenten las necesidades del caso.

Por lo que en el siguiente prototipo de sistema de seguridad, se utilizará esta tecnología, para evitar la intrusión de personas en los puntos detectados como vulnerables de las instalaciones físicas y evitar el daño y hurto del patrimonio universitario.

Se propone investigar y diseñar un sistema de seguridad informático que venga a solucionar tal problema, de una forma económica, ya que no solo sería innovador, sino necesario, además del hecho de que nuestro propio recurso formado viene a ser solución para necesidades tecnológicas del país.

Toda la información producida o como resultado del empleo de este sistema, podrá ser manejada y almacenada de forma digital, para usos posteriores de parte de la unidad de custodios o autoridades de la FMO.

Dicho sistema de seguridad ofrecerá los siguientes beneficios:

- Hardware de bajo costo.
- Construcción sencilla.
- Calibración sencilla y rápida.
- Independiente de plataforma.
- Innovador.
- Cobertura de puntos vulnerables a robos.
- Red independiente.
- Control restringido solo para personal autorizado.

METODOLOGÍA DE INVESTIGACIÓN

Para el alcance de los objetivos planteados y la recolección de información se recurrirá a utilizar las siguientes metodologías o herramientas de investigación:

- ✓ Visitas u observación directa.
- ✓ Entrevistas.
- ✓ Recolección de información y formatos ya impresos referente al funcionamiento realizado actualmente.
- ✓ Recolección de información referente al funcionamiento de los sistemas de seguridad basados en sensores, implementados actualmente en otros lugares, similar al trabajo de grado.
- ✓ Apoyo bibliográfico.
- ✓ Ciclo de Vida del trabajo de graduación.

Visita u Observación Directa:

Es el registro visual de lo que ocurre en una situación real, clasificando y consignando los acontecimientos pertinentes de acuerdo algún esquema previsto y según los problemas que se estudia. Es un método que permite obtener datos cuantitativos como cualitativos.

Pasos posibles:

1. Elegir un día específico para hacer la visita.
2. Al llegar al lugar proceder a tomar fotografías.
3. Recolectar datos interesantes que nos servirán para realizar nuestro trabajo de campo.
4. Referirnos a las personas involucradas directamente, en los puntos a que nos orienten al entendimiento perfecto del sistema de seguridad actual.

Entrevista:

Una entrevista es una conversación dirigida con un proceso específico, que usa un formato de preguntas y respuestas. En la entrevista se quiere obtener la opinión del entrevistado, objetivos de la organización y procedimientos informales.
Se utilizará esta herramienta para establecer una comunicación más directa con el personal de la unidad de custodios, con la finalidad de conocer la situación actual, profundizar en ciertos aspectos de interés, y obtener información más completa acerca del presente trabajo de grado, de acuerdo a la perspectiva de cada miembro que ahí labora.

Pasos posibles:

1. Diseñar y generar las entrevistas.
2. Escoger las personas idóneas para entrevistar.
3. Tabular los datos obtenidos.
4. Analizar los datos obtenidos.

Cuestionario:

Un cuestionario es una técnica de recopilación de información que permite que los analistas de sistemas estudien actitudes, creencias, comportamientos y características de varias personas principales en la organización que pueden ser afectadas por los sistemas actuales y propuestos.

La utilización de estas herramientas de recolección de información, se realizan con el fin de economizar tiempo a los objetos de estudio.

Tomando en cuenta para el diseño de esta herramienta aspectos tales como:

- ✓ Formular preguntas abiertas, como complemento de las que necesitan ampliación del tema.
- ✓ Formular preguntas dicotómicas (SÍ O NO)
- ✓ Determinar preguntas de selección múltiple para que el cuestionado tenga una gama de alternativas, siempre y cuando no se salga del tema en estudio.

Pasos posibles:

1. Diseñar y generar las encuestas cuestionarios
2. Escoger una muestra idóneas para la encuesta.
3. Tabular los datos obtenidos.
4. Analizar los datos obtenidos.

Apoyo Bibliográfico:

El apoyo bibliográfico o La investigación bibliográfica, es la etapa de investigación científica donde se explora qué se ha escrito en la comunidad científica sobre un determinado tema o problema. A través de la investigación bibliográfica se podrá contar con una fuente que sirva de base para la elaboración del marco teórico del proyecto. La investigación puede ser realizada independiente o como parte de otros tipos de investigación como la de campo o de laboratorio. Busca siempre conocer los acontecimientos científicos o culturales del pasado. En nuestro caso la investigación bibliográfica se efectuó con el propósito de recolectar información sobre el lenguaje de programación, el uso de sensores y los distintos tipos de tecnologías propias de los sistemas de seguridad actual; de manera que los resultados generados sean también congruentes a los que otros sistemas de seguridad que diseña la empresa privada actualmente, también podemos investigar por medio de proyectos similares que se hayan desarrollado tanto en el país como en otros países.

Pasos posibles:

1. Recolectar información de las tecnologías que se utilizan aplicados a este tipo de sistemas de vigilancia, solicitándola a empresas, instituciones o personas especializadas en esta área tecnológica.
2. Utilización de medios como Internet, libros, etc.
3. Sintetizar la información recolectada.
4. Proceder a formar el marco teórico del Proyecto de Tesis.

HERRAMIENTA PARA EL DESARROLLO DEL SISTEMA:

Para el desarrollo del sistema se tomará en cuenta el Ciclo de Vida de los Sistemas:

Ciclo de vida del sistema:

Por ciclo de vida de un producto software se entiende la secuencia de fases y/o actividades que en cada una de ellas, existen controles que sirven para pasar de una fase a otra y los resultados generados en cada una de ellas, permiten el desarrollo de un producto desde su concepción.

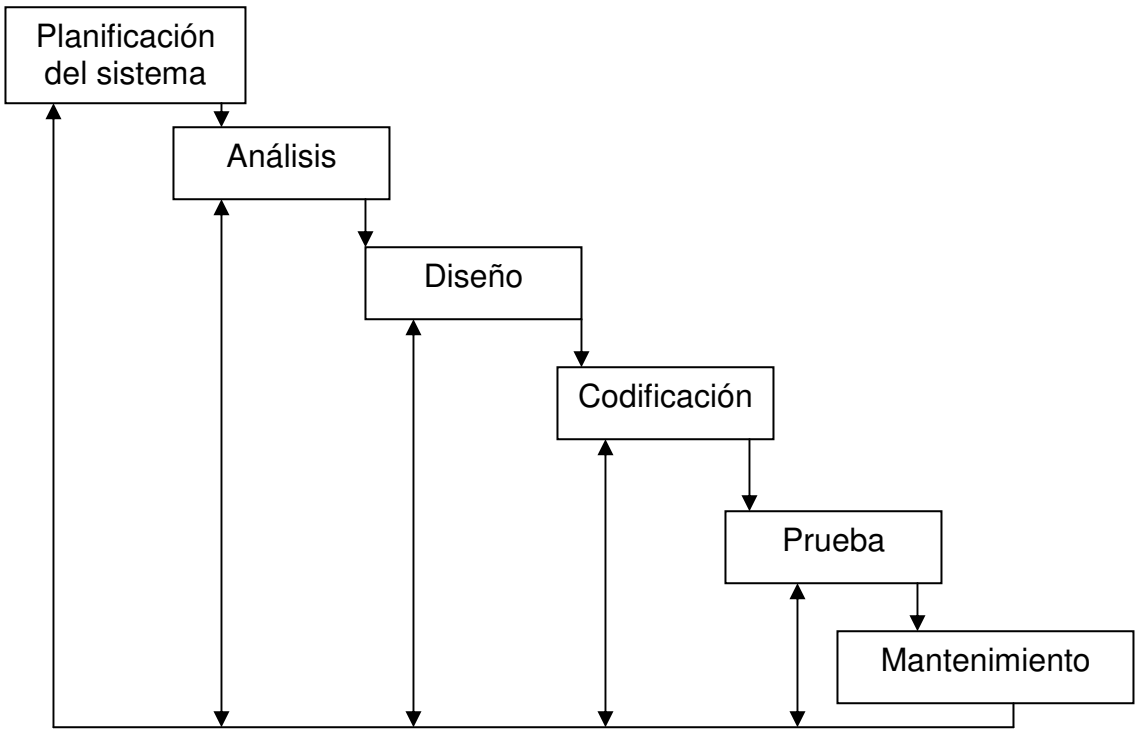
No existe un modelo de ciclo de vida único. Tanto el tipo, orden y actividades, en cada fase, pueden cambiar adaptándose a las necesidades del producto a realizar y a la propia estructura de la organización que lo desarrolla, a partir de las posibilidades que ofrece la tecnología de software empleada.

En la propuesta de desarrollo se ha concebido la implementación del modelo de cascada como modelo de ciclo de vida de software, pues este es conocido como modelo convencional y es el más implementado en el desarrollo de software, este modelo se caracteriza por poseer una serie de fases a través del tiempo, las cuales son secuenciales.

Este modelo se caracteriza por la existencia de un conjunto de fases en secuencia del tiempo. Al finalizar una fase se comienza la siguiente, tomando como datos de entrada los resultados de la anterior. En cada una de las fases se introduce más detalle hasta obtener un código ejecutable y eficiente que incorpore los requisitos necesarios.

A continuación se puede observar gráficamente el modelo de ciclo de vida en cascada:

Ciclo de vida Cascada



Estas etapas no son estrictamente secuenciales, puede repetirse algunas de las etapas varias veces, haciendo lo que se conoce como realimentación.

RESULTADOS ESPERADOS

Los beneficios esperados de este trabajo de grado, para los entes involucrados son:

Dirigido a:	Beneficios
Institución	• Dar una alternativa de sistema de seguridad o custodia al patrimonio Universitario, que pertenece a todos los involucrados en el rol de la enseñanza superior. • Facilidad en el manejo de información, generación de consultas y reportes adecuados de forma instantánea, por medio de un software, económica e innovador, el cual sea flexible a cambios que se produzcan en el fututo (Expansión en otras áreas). • Certeza en los resultados generados por el software.
Docentes y Personal Administrativo	• Resguardo del equipo de oficina y material de importancia académica.
Estudiantes	• Para futuras generaciones de estudiantes se brindara un mejor bienestar y resguardo del patrimonio, tanto intelectual como físico.
Unidad de Custodios	• Obtener un software que brinden una solución eficaz al manejo de la información, de modo que se automaticen y optimicen los procesos de vigilancia o custodia de los equipos e instalaciones de la UES-FMO. • Rapidez en la respuesta de activación de un dispositivo sensorial. • Certeza en los resultados generados por el software. • Seguridad y confiabilidad en la información que proporcione el sistema. • Funcionabilidad optima en el desempeño de las operaciones de seguridad.

CAPITULO II

"ANÁLISIS Y DETERMINACIÓN DEL PROBLEMA"

SITUACIÓN ACTUAL

DESCRIPCIÓN DE LA SITUACIÓN ACTUAL

La FMO cuenta con un personal encargado del resguardo, capacitado para brindar un buen servicio; aunque muchas veces, insuficiente para cubrir y satisfacer todas las expectativas concernientes a la seguridad institucional. Según las diferentes unidades o departamentos que conforman la Facultad, expresan que el papel desempeñado por esta unidad es regular, ya que por la falta de personal, las unidades se conforman con los recursos que actualmente existen para la seguridad institucional. Pero además otros aseguran que su desempeño en el momento de la ronda, es buena, ya que hacen chequeo de las puertas de los edificios, en horas no laborales.

Debido a que se ha visto amenazada por una serie de eventos delincuenciales como: Hurto de equipo informático o multimedia, libros, artículos personales desde los cubículos de los docentes dentro de los diferentes departamentos, extracción forzada de componentes de vehículos (dentro del parqueo de la Institución), intrusión de personas ajenas a la universidad, que causan daños a la misma o hacen asaltos al estudiantado en lugares con poco flujo de personas o poca iluminación, como lo son las canchas. Así como en otros casos, como el presentado en la Alma Mater en San Salvador, en donde un grupo de manifestantes se introdujo violentamente en la universidad tomando posesión de las instalaciones, por la falta de personal que brinde seguridad en las áreas de ingreso a la universidad.

Y ya que la estructura física de las instalaciones no es apropiada o se encuentra muy deteriorada exclusivamente en el interior, como por ejemplo chapas, puertas, ventanas, techos, instalaciones eléctricas, etc., facilita la intrusión de personas desde las viviendas o lugares aledaños, para la extracción o hurto del patrimonio universitario y artículos personales, creando un ambiente de desconfianza entre

los estudiantes y trabajadores, dichos sucesos han aumentado su frecuencia en los últimos años, teniendo mas incidencia en el presento año.

Aunque nunca se han hecho investigaciones profundas en los casos con pérdidas mas elevadas, se presume que pueden ser personas pertenecientes a la comunidad universitaria, porque son personas que conocen la Universidad y las actividades que estas desempeñan cotidianamente dentro del Facultad.
Previamente ya se les han dado alternativas o proyectos, para mejorar o dar solución a dicha problemática. Por sus altos costos, y requerimientos de formación del Recurso Humano, o contratación de nuevo personal a los que ya existen, hasta el momento no se han podido implementar. Por lo que no significa también, una de las prioridades de la Universidad sino que se convierte en 3er o 4º lugar.

DIAGNÓSTICO DE LA SITUACIÓN ACTUAL

La FMO no debe ser la excepción en cuanto a modernizar sus sistemas de vigilancia. Aunque por la falta de una buena estrategia y el poco personal de seguridad con que cuenta la institución, no permite satisfacer a plenitud cada una de las demandas de un sistema de seguridad institucional bien estructurado, tomando en consideración el espacio territorial que hay que abarcar.

Siendo los mayores desafíos que presenta la FMO en el tema de Seguridad, el número bajo de custodios, y la gran demanda de recursos que se deben de cuidar, teniendo en cuenta que hay pocas herramientas apropiadas para desarrollar sus obligaciones.

La poca vigilancia e inseguridad que presenta la institución desde la perspectiva antes mencionada, crea las siguientes necesidades de seguridad dentro de la Universidad:

- La construcción de un mejor muro perimetral.
- La renovación de la infraestructura antigua de los edificios.
- El incremento del personal de custodios.
- Y la incorporación de tecnología, sin invadir la privacidad de las personas.

Además las zonas que requieren mayor seguridad son las que tienen mayor vulnerabilidad, por la falta de personal y donde se encuentra equipo de riesgo o material de valor para la institución. Así como también la ampliación del parqueo trajo consigo mayores obligaciones para los custodios, por lo que presentan un incremento en sus labores.

Por tales razones, no se cumplen todas las expectativas de un buen servicio de seguridad, ya que el equipo de custodios es insuficiente para dar vigilancia total al patrimonio de la facultad, surge la necesidad de proponer alternativas diferentes que permitan brindar un ambiente de seguridad entorno a las instalaciones de la FMO, en la medida que se logre hacer un plan sin costos elevados, con buenos resultados a corto plazo. Como tal, se propone un sistema de seguridad informática con tecnología sensorial que permite al personal encargado de la seguridad mantener en resguardo los distintos puntos o lugares vulnerables de la institución, lo que proporciona un apoyo sumamente valioso para desempeñar sus funciones, garantizando un menor grado de riesgo de que se lleve a cabo un acto ilícito dentro. Todo esto acompañado con el apoyo de las autoridades de la FMO, capacitación del personal para el uso y mantenimiento de estos dispositivos.

INVESTIGACIÓN DE NECESIDADES DE LA UES-FMO

Para el análisis de la investigación se ha tomado en cuenta la opinión del estudiantado universitario de la Facultad Multidisciplinaria de Occidente, a través de encuestas o cuestionarios; sin embargo con estos instrumentos de recolección de datos se pretende medir algunos aspectos relacionados con la custodia y seguridad de la facultad; al hacer dicho contacto con los estudiantes, se ha elegido una muestra calculada al azar para conseguir dicha información imparcial y objetiva de lo que se esta analizando.

POBLACIÓN Y MUESTRA DE LA INVESTIGACIÓN

Dentro de la investigación, es necesario elegir una población o grupo en el que se realizará el estudio, para este caso el proceso de diagnostico se llevara a cabo en la instalaciones de la facultad multidisciplinaria de occidente en el mes de Agosto del presente año, tomando en cuenta parte de la población estudiantil de los diversos departamentos de dicha facultad, así como los diferentes turnos en donde existe mayor concentraron de personas, en las diferentes áreas y lugares de la FMO; para conformar lo que es la muestra.

La FMO esta ubicada en Final Avenida Fray Felipe de Jesús Moraga Sur, Santa Ana, El Salvador, C.A.

La Población, es el conjunto de todos los individuos cuyo conocimiento es objeto de interés desde un punto de vista estadístico. La población es también llamada universo.

El estudio estadístico de una población se puede realizar mediante un análisis exhaustivo de todos sus individuos o bien mediante una inferencia realizada a

partir de una muestra extraída de la población.

En este caso se realiza la investigación por medio de un muestreo (proceso por el cual se seleccionan los individuos que formarán una muestra).
Muestra, es la selección de un conjunto de individuos representativos de la totalidad del universo objeto de estudio, reunidos como una representación válida y de interés para la investigación de su comportamiento. Los criterios que se utilizan para la selección de muestras pretenden garantizar que el conjunto seleccionado represente con la máxima fidelidad a la totalidad de la que se ha extraído, así como hacer posible la medición de su grado de probabilidad.

Se distinguen varios tipos de muestras: la muestra simple, en la que cada individuo del universo considerado tiene las mismas probabilidades de resultar elegido; la muestra estratificada, si la selección se realiza sobre grupos o estratos diferentes; y, finalmente, la muestra por agrupamientos, que se basa en los segmentos o asociaciones organizadas dentro del universo considerado.

Para que se puedan obtener conclusiones fiables para la población a partir de la muestra, es importante tanto su tamaño como el modo en que han sido seleccionados los individuos que la componen. La muestra tiene que estar protegida contra el riesgo de resultar sesgada, manipulada u orientada durante el proceso de selección, con la finalidad de proporcionar una base válida a la que se pueda aplicar la teoría de la distribución estadística.

El tamaño de la muestra depende de la precisión que se quiera conseguir en la estimación que se realice a partir de ella. Para su determinación se requieren técnicas estadísticas superiores, pero resulta sorprendente cómo, con muestras notablemente pequeñas, se pueden conseguir resultados suficientemente precisos. Para seleccionar los individuos de la muestra es fundamental proceder aleatoriamente, es decir, decidir al azar qué individuos de entre toda la población forma parte de la muestra.

Si se procede como si de un sorteo se tratara, eligiendo directamente de la población sin ningún otro condicionante, el muestreo se llama aleatorio simple o irrestrictamente aleatorio.

Las inferencias realizadas mediante muestras seleccionadas aleatoriamente están sujetas a errores, llamados errores de muestreo, que están controlados. Si la muestra está mal elegida (no es significativa) se producen errores sistemáticos no controlados.

Inferencia, es el proceso por el cual se deducen (infieren) propiedades o características de una población a partir de una muestra significativa. Uno de los aspectos principales de la inferencia es la estimación de parámetros estadísticos

La inferencia siempre se realiza en términos aproximados y declarando un cierto nivel de confianza. Si se quiere mejorar el nivel de confianza, se deberá aumentar el tamaño de la muestra, o bien disminuir la precisión de la estimación. Recíprocamente, si se quiere aumentar la precisión en la estimación disminuyendo el tamaño del intervalo, entonces hay que aumentar el tamaño de la muestra o bien consentir un nivel de confianza menor. Finalmente, si se quiere mejorar tanto la precisión como el nivel de confianza, hay que tomar una muestra suficientemente grande.

Al determinar la población estudiantil dentro de la FMO se ha considerado un aproximado de 5000 estudiantes en los diferentes turnos y carreras dentro de la comunidad universitaria. Por consiguiente, el muestreo a realizar es simple e irrestrictamente aleatorio. Por lo tanto, el tamaño de la muestra que se ha tomado es representativa de la facultad, ya que es un 0.1% de la población estudiantil. El modelo matemático que se utiliza para determinar la muestra, es el modelo proporcionado por el texto análisis de sistemas de kendall y kendall. En el que se

deben seguir siete pasos, algunos de los cuales son juicios subjetivos, para determinar el tamaño de la muestra requerido:

1. Determinar el atributo (en este caso el tipo de errores que se buscara).
2. Localizar la base de datos o informes en los cuales se puede encontrar el atributo.
3. Examinar el atributo. Calcular p, la proporción de población que tiene el atributo.
4. Tomar la decisión subjetiva con respecto a la estimación del intervalo aceptable, i.
5. Seleccionar el nivel de confianza y buscar el coeficiente de confianza (valor z) en una tabla.
6. Calcular δ_p, el error estándar de la proporción, de la siguiente manera:

$$\delta_p = \frac{i}{z} = \frac{\text{estimación del intervalo}}{\text{coeficiente de confianza}}$$

7. Determinar el tamaño de la muestra necesario, n, con la formula siguiente:

$$n = \frac{p\,(1-p)}{\delta_p^2} + 1$$

Al aplicar la formula empleada en este modelo, es de la siguiente forma:

p = proporción de la población, en donde no debe se mas $\pm$ 0.10 en una proporción real.

p = 0.10

El Nivel de confianza es del 98%, el cual es el grado de certeza deseado. Una vez que se elige dicho nivel, se busca el coeficiente de confianza en una tabla similar a

la que se presenta a continuación, por lo tanto el coeficiente de confianza es igual a 2.33.

Nivel de confianza	Coeficiente de confianza (valor z)
99%	2.58
98	2.33
97	2.17
96	2.05
95	1.96
90	1.65
80	1.28
50	0.67

$\delta p = \frac{i}{z} = \frac{0.10}{2.33} = 0,04292$

$n = \frac{p\,(1-p)}{\delta p^2} = \frac{(0.10)\,(0.90)}{0.00184} = \frac{0.09}{0.00184} = 48,86010 + 1 = 49,86010$

Respuesta n = 50

Por lo tanto el tamaño de la muestra para la población actual es de 50 personas.

INSTRUMENTOS PARA LA RECOLECCIÓN DE DATOS

ENCUESTA:

Instrumento cuantitativo de investigación social mediante la consulta a un grupo de personas elegidas de forma estadística, realizada con ayuda de un cuestionario.

Un cuestionario es una técnica de recopilación de información que permite que los analistas de sistemas estudien actitudes, creencias, comportamientos y características de varias personas principales en la organización que pueden ser afectadas por los sistemas actuales y propuestos.

La utilización de estas herramientas de recolección de información, se realizan con el fin de economizar tiempo a los objetos de estudio.

Tomando en cuenta para el diseño de esta herramienta aspectos tales como:

- ✓ Formular preguntas abiertas, como complemento de las que necesitan ampliación del tema.
- ✓ Formular preguntas dicotómicas (SÍ O NO)
- ✓ Determinar preguntas de selección múltiple para que el cuestionado tenga una gama de alternativas, siempre y cuando no se salga del tema en estudio.

La encuesta se diferencia de otros métodos de investigación en que la información obtenida ya está de antemano preparada y estructurada. En este sentido, la encuesta presenta notables limitaciones, al restringir las posibilidades de obtener información a validar o refutar hipótesis previamente establecidas en el cuestionario, coartando el discurso del entrevistado, y sin que resulte posible saber si existe información relevante que no se tiene en cuenta. Además, al

tratarse de entrevistas individuales, se pierde la riqueza de fenómenos que resultan de la interacción social.

Según la forma en que se obtienen los datos, las encuestas pueden ser presénciales, telefónicas o postales. En cualquier caso, la utilización masiva de las encuestas en procesos de toma de decisiones, tanto en el ámbito publico como privado, ha supuesto una progresiva sistematización de los procesos de trabajo en este tipo de estudios y la creación de normas metodológicas y códigos deontológico que tratan de asegurar su calidad y consistencia.

ENTREVISTA:

Una entrevista es una conversación dirigida con un proceso específico, que usa un formato de preguntas y respuestas. En la entrevista se quiere obtener la opinión del entrevistado, objetivos de la organización y procedimientos informales. Hay muy diversos tipos de entrevistas: laborales, de investigación, informativas y de personalidad, entre otras.

En una entrevista intervienen el entrevistador y el entrevistado. El primero, además de tomar la iniciativa de la conversación, plantea mediante preguntas específicas cada tema de su interés y decide en qué momento el tema ha cumplido sus objetivos. El entrevistado facilita información sobre sí mismo, su experiencia o el tema en cuestión.

Al planear una entrevista hay cinco pasos principales a seguir, para obtener lo que se necesita de la mejor manera posible:

1. Leer los antecedentes (entrevistado u organización)
2. Establecer los objetivos de la entrevista
3. Decidir a quien entrevistar
4. Preparar al entrevistado

5. Decidir el tipo de preguntan y la estructura

Tomando en cuenta para el diseño y estructura de las preguntas que esta herramienta necesita están:

- ✓ Formular preguntas abiertas, como complemento de las que necesitan ampliación del tema. Ya que describen loas opciones del entrevistado.
- ✓ Formular preguntas cerradas, debido a que el entrevistado solo puede contestar con un número finito.
- ✓ Un tipo especial de pregunta cerrada como lo es la bipolar, esta limita aun mas las opciones del entrevistado (SI o NO, VERDADERO o FALSO, etc.).
- ✓ Formular preguntas de sonseo, es para ampliar, clarificar u obtener mas de la opinión del entrevistado

VISITA U OBSERVACIÓN DIRECTA:

Es el registro visual de lo que ocurre en una situación real, clasificando y consignando los acontecimientos pertinentes de acuerdo algún esquema previsto y según los problemas que se estudia. Es un método que permite obtener datos cuantitativos como cualitativos.

Los instrumentos que se utilizaron para la recolección de datos en la obtención de información se tienen: encuestas a la población estudiantil, entrevistas al personal administrativo y/o custodios y observación directa a la comunidad universitaria de la FMO.

La administración de estos recursos metodológicos se ha empleado para la obtención de información acerca de la seguridad y custodia del patrimonio universitario de la FMO, de la siguiente manera:

Encuestas, estás se realizaron de forma aleatoria en diferentes puntos de la facultad, de manera tal recavar las diferentes opiniones de los estudiantes de diversas carreras. Se tomaron como referencia los puntos más concurridos y de mayor afluencia de personas dentro de la variedad de turnos como por la mañana, tarde y noche.

Entrevistas, se realizó con el objeto de tener una comunicación más directa con el personal de la unidad de custodios, con la finalidad de conocer la situación actual, profundizar en ciertos aspectos de interés, y obtener información más completa acerca del presente trabajo de grado, de acuerdo a la perspectiva de cada miembro que ahí labora. A la vez, al realizar la entrevista con el personal encargado administrativo, se llevo a cabo la recolección de mejor y mayor información referente a la seguridad en su totalidad como facultad y el impacto que tiene la unidad de custodios en la realización de su trabajo.

Visita u observación directa, al desarrollar esta metodología se obtuvo un amplio criterio sobre las necesidades que enfrenta la facultad. Aquí se llevo a cabo la toma de fotografías e involucro a la población estudiantil, en lo referente a lo que es este trabajo de grado.

Para visualizar los instrumentos de recolección de datos se debe referir a los anexos que se encuentra al final de este documento.

TABULACIÓN Y ANÁLISIS DE DATOS

Los pasos que se llevaron a cabo para el análisis y resumen de los datos obtenidos por la encuesta son los siguientes:

- ✓ Revisar cada uno de los cuestionarios, seleccionarlos, revisar si le falta alguna pregunta por contestar, respuestas ilegibles, etc.

- ✓ Tabulación de los datos.

  La estructura en que se presenta es la siguiente:
  - ➢ La pregunta completa en orden correlativo
  - ➢ El objetivo respectivo a cada pregunta.
  - ➢ El dato resultante del conteo en una tabla con su respectivo porcentaje o frecuencia, dependiendo el tipo de pregunta y respectiva respuesta.
- ✓ Resumen de la información por medio de gráficos estadísticos correspondientes a cada sector de la encuesta o cuestionario.

- ✓ Por ultimo pero no menos importante, la interpretación de los datos obtenidos por medio de un análisis de cada pregunta.

A continuación se presenta cada pregunta y su respectivo análisis:

Pregunta 1. Departamento que pertenece
Objetivo: Conocer la fuente de información, según su respectiva área.

Respuesta	Frecuencia	Porcentaje
Ciencias jurídicas	17	34%
Idiomas	5	10%
Ciencias Sociales, Filosofía y Letras	8	16%
Matemática	5	10%
Medicina	4	8%
Ingeniería	4	8%
Humanidades	2	4%
Ciencias Económicas	5	10%
Total	50	100%

Tabla 1. Tabulación de datos. Pregunta #1

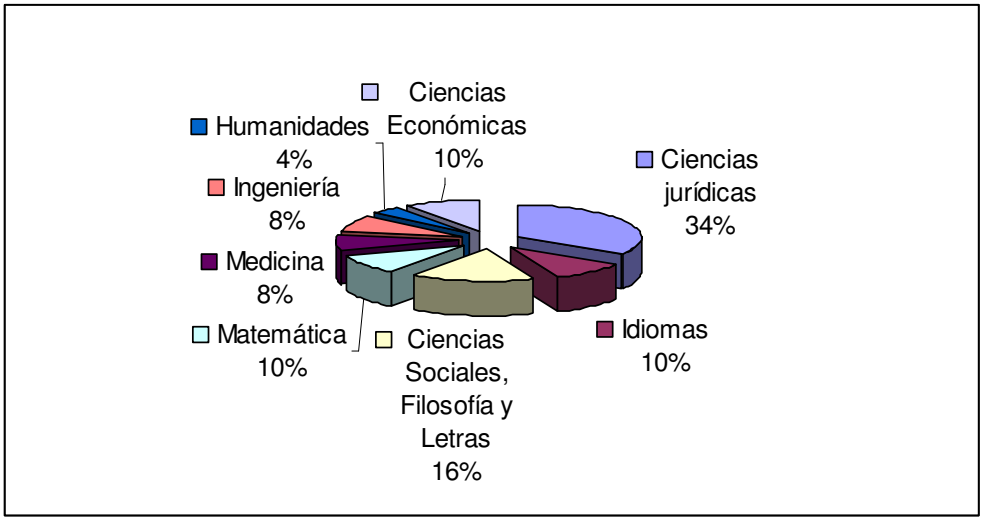


Gráfico 1. Representación grafica de ¿Departamento a que Pertenece?

Según el personal que se tomo como muestra aleatoria en los lugares mas concurridos de personas, el mayor porcentaje de alumnos que expreso su opinión se encuentra los del departamento de derecho y ciencias sociales; sin exceptuar los pocos de los departamentos que hicieron aporte a la investigación.

Pregunta 2. ¿Cómo califica el nivel de seguridad en los lugares que frecuenta en la FMO (aulas, departamento, canchas, etc.)?

Objetivo: Medir el nivel de seguridad de la FMO según la comunidad universitaria.

Respuesta	Frecuencia	Porcentaje
Muy Bueno	3	6%
Bueno	21	42%
Regular	16	32%
Mala	10	20%
Total	50	100%

Tabla 2. Tabulación de datos. Pregunta #2

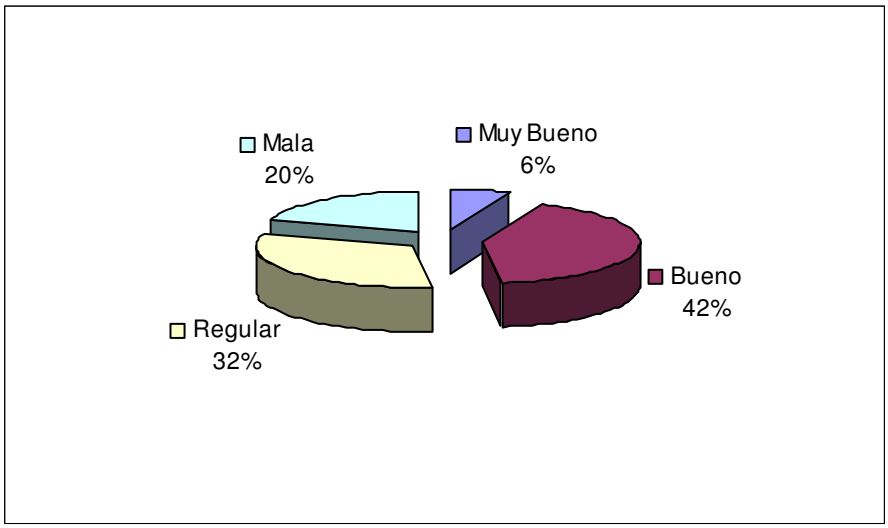


Gráfico 2. Representación grafica de ¿Cómo califica el nivel de seguridad en los lugares que frecuenta en la FMO (aulas, departamento, canchas, etc.)?

La mayor parte de la población estudiantil califica el nivel de seguridad en la FMO como *regular y bueno*, lo cual demuestra que la comunidad no esta satisfecha con el nivel de seguridad que brinda la facultad.

Pregunta 3. ¿Cree que su departamento es un lugar seguro?
Objetivo: Medir la seguridad de los departamentos a nivel de la facultad.

Respuesta	Frecuencia	Porcentaje
Si	26	52%
No	23	46%
No Respondió	1	2%
Total	50	100%

Tabla 3. Tabulación de datos. Pregunta #3

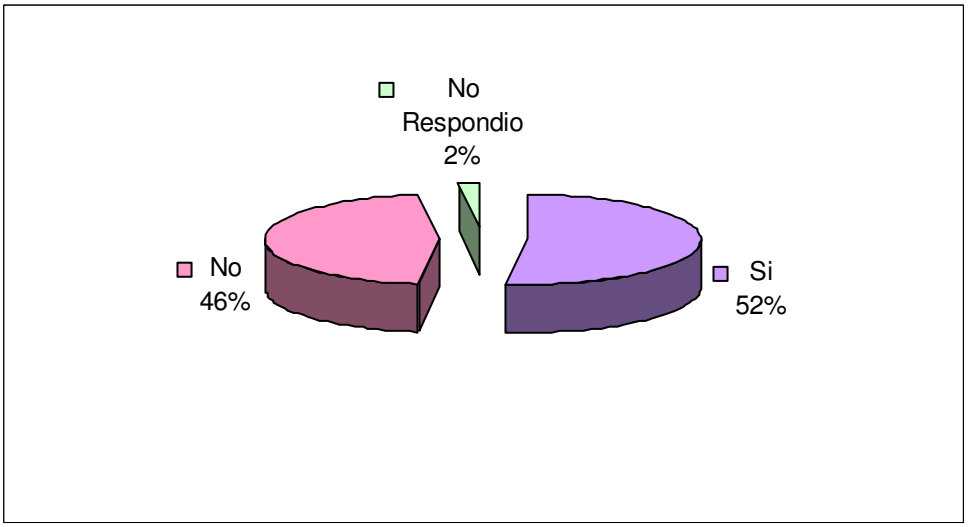


Gráfico 3. Representación grafica de ¿Cree que su departamento es un lugar seguro?

A nivel general se presenta, que si están seguros en sus departamentos respectivos, aunque el demás porcentaje muestra, que a pesar de estar dentro de las instalaciones, no se sienten seguros, ya sea por los distintos casos observados o presentados en la institución.

Pregunta 4. ¿En que zona de la FMO se siente mas seguro (aulas, bunker, edificio de usos múltiples, edificio de medicina, cafetería, etc.)?
Objetivo: Verificar que zona de la FMO es la mas segura.

Respuesta	Frecuencia	Porcentaje
Cafetería	10	19%
No Respondieron	2	4%
Bunker	10	19%
Edificio de Medicina	3	5%
Zona Verde y Canchas	3	5%
Ninguno	4	7%
Todos	5	9%
Edificio de Usos Múltiples	9	17%
Aulas	8	15%
Total	54	100

Tabla 4. Tabulación de datos. Pregunta #4

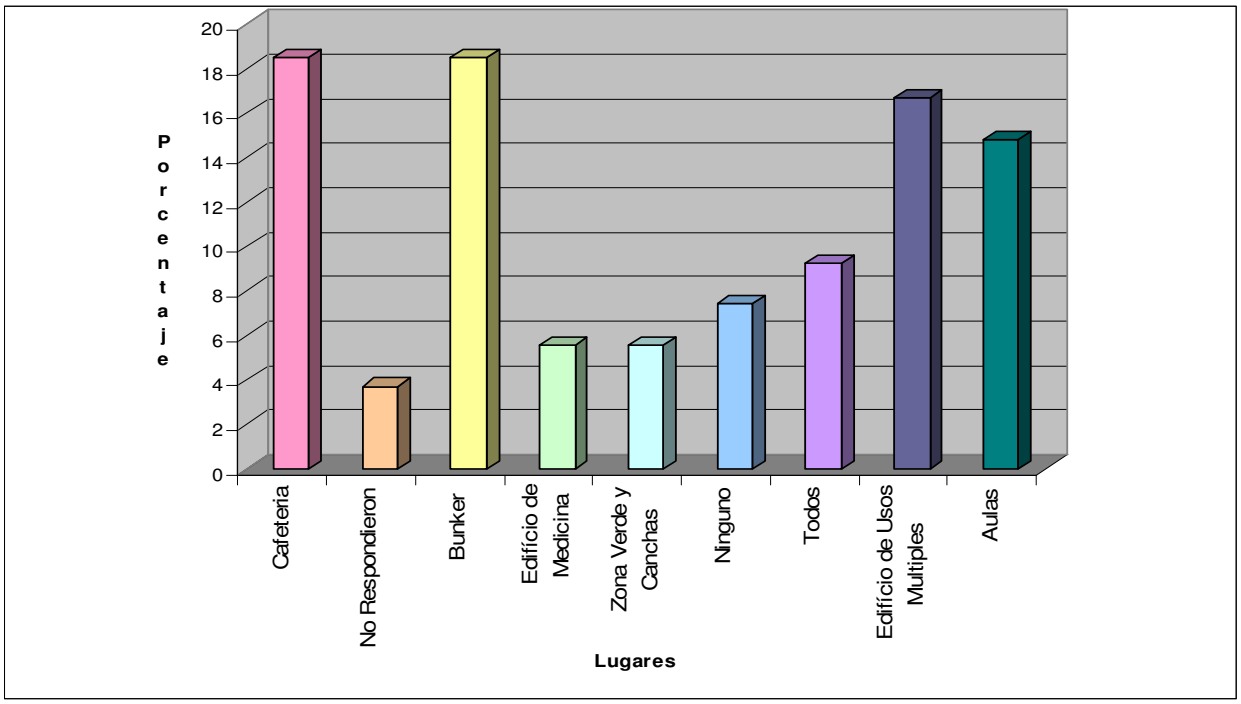


Gráfico 4. Representación grafica de ¿En que zona de la FMO se siente mas seguro (aulas, bunker, edificio de usos múltiples, edificio de medicina, cafetería, etc.)?

Según los resultados obtenidos y los comentarios hechos por los estudiantes, la cafetería es uno de los lugares más seguros, por la afluencia de personas, mientras que el bunker es seguro en su infraestructura ante fenómenos naturales como temblores. Y uno de los lugares más inseguros presentados es el edificio de Medicina, donde se han encontrado más perdidas por robos.

Pregunta 5. ¿Ha sufrido algún percance con su seguridad alguna vez?

Objetivo: Conocer la frecuencia de percances personales sucedidos en la comunidad.

Respuesta	Frecuencia	Porcentaje
Si	2	4
No	48	96
Total	50	100

Tabla 5. Tabulación de datos. Pregunta #5

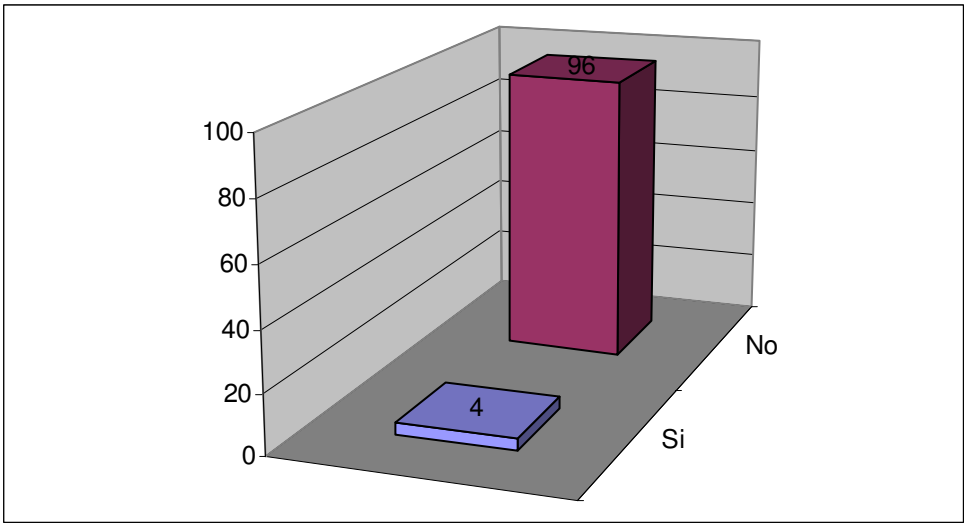


Gráfico 5. Representación grafica de ¿Ha sufrido algún percance con su seguridad alguna vez?

En estos resultados se refleja que la comunidad universitaria no es la afectada directamente en la inseguridad, sino la institución en sí es la más afectada en perdidas ocasionas por la delincuencia.

Pregunta 6. ¿Considera usted que dentro de la FMO existan personas capaces de realizar actos de vandalismo?

Objetivo: Percibir la opinión de la comunidad universitaria acerca de si existen personas que perpetran este tipo de actos vandálicos dentro de la Universidad.

Respuesta	Frecuencia	Porcentaje
Si	39	78
No	11	22
Total	50	100

Tabla 6. Tabulación de datos. Pregunta #6

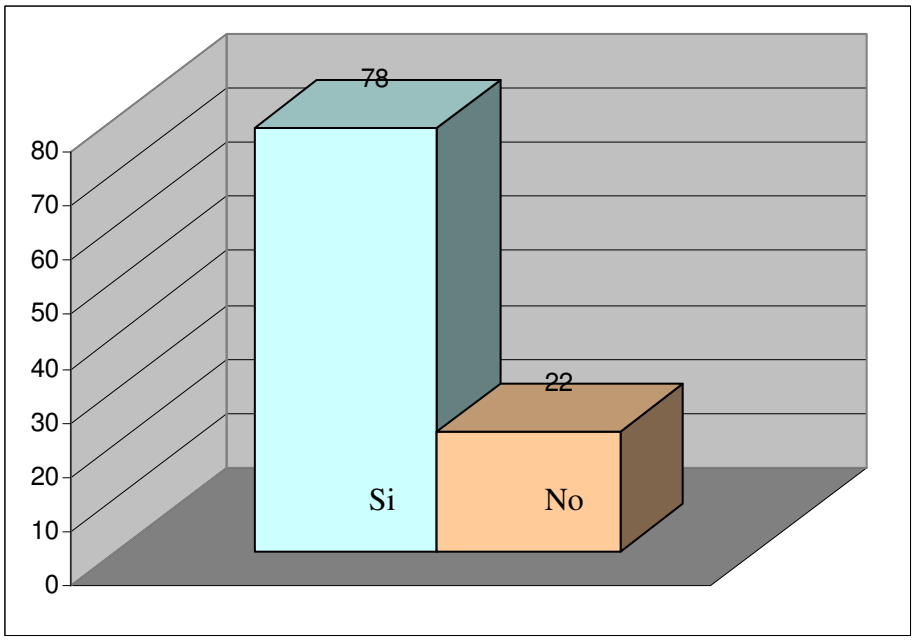


Gráfico 6. Representación grafica de las ¿Considera usted que dentro de la FMO existan personas capaces de realizar actos de vandalismo?

En su mayoría, opina que si pueden ser personas que están dentro de la comunidad universitaria, ya que conocen las actividades cotidianas dentro de la institución, por lo que también pueden ser puentes o contactos para personas externas a la facultad.

Pregunta 7. Según su criterio, ¿Los ataques a la seguridad de la FMO son hechos por personas ajenas a ella?

Objetivo: Percibir la opinión de la comunidad universitaria acerca quienes son las personas que perpetran este tipo de actos vandálicos.

Respuesta	Frecuencia	Porcentaje
Si	40	80%
No	9	18%
No Respondió	1	2%
Total	50	100%

Tabla 7. Tabulación de datos. Pregunta #7

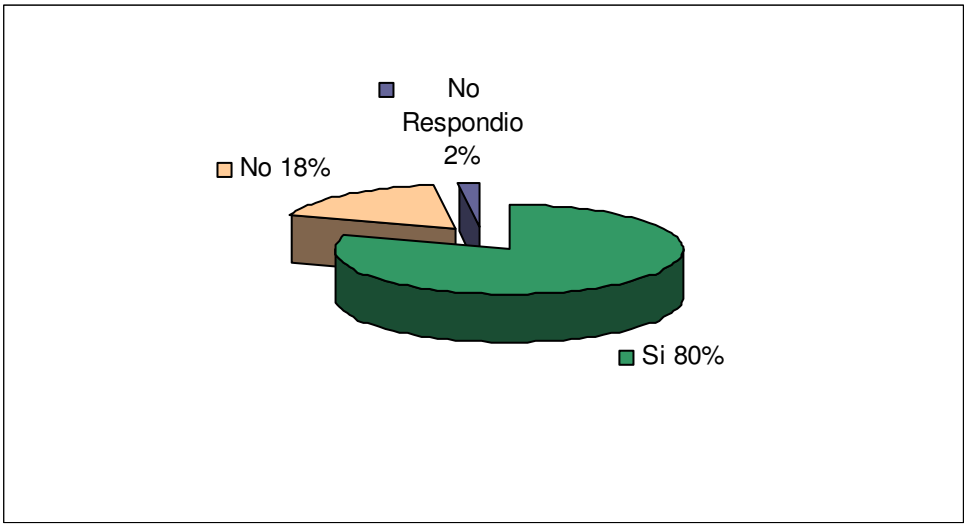


Gráfico 7. Representación grafica de Según su criterio, ¿Los ataques a la seguridad de la FMO son hechos por personas ajenas a ella?

Según se percibe, la comunidad universitaria en su mayoría cree que si pueden ser personas internas a la institución, ya que estás pueden observar más las actividades que se desempeñan cotidianamente en la institución, en cambio alguien externo esta más ajeno a estos detalles observables.

Pregunta 8. ¿Cómo califica el trabajo realizado por los custodios (vigilantes), en sus años dentro de la comunidad universitaria?

Objetivo: Encontrar la calificación de la comunidad, acerca del desempeño de la unidad de custodios.

Respuesta	Frecuencia	Porcentaje
Excelente	0	0%
Muy Bueno	9	18%
Bueno	20	40%
Regular	18	36%
Mala	3	6%
Total	50	100%

Tabla 8. Tabulación de datos. Pregunta #8

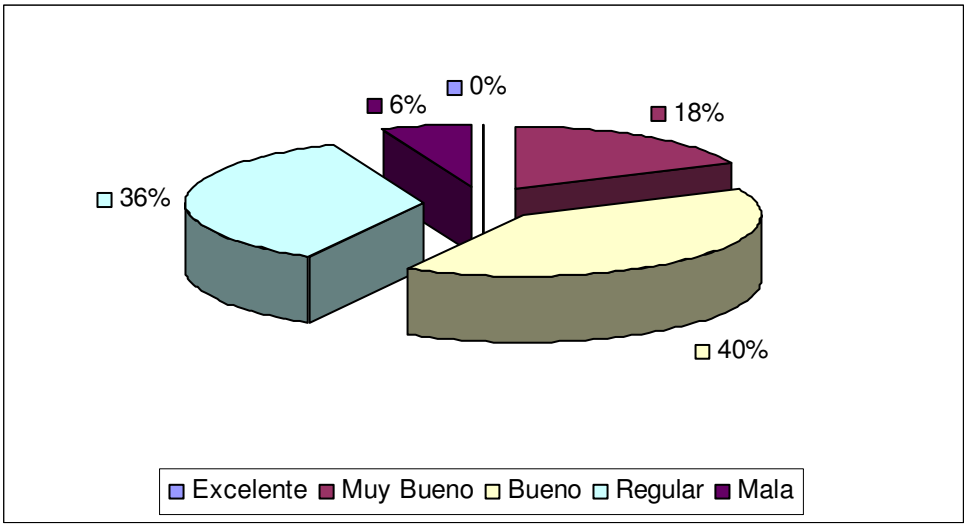


Gráfico 8. Representación grafica de ¿Cómo califica el trabajo realizado por los custodios (vigilantes), en sus años dentro de la comunidad universitaria?

Según se puede apreciar la calificación obtenida dentro del los encuestados, es buena o regular en su mayoría, por lo que pocos consideran que sea muy buena o mala.

Pregunta 9. ¿Ha tenido problemas con la unida de custodios alguna vez?
Objetivo: Conocer la relación de los custodios con respecto al estudiantado.

Respuesta	Frecuencia	Porcentaje
Si	3	6%
No	47	94%
Total	50	100%

Tabla 9. Tabulación de datos. Pregunta #9

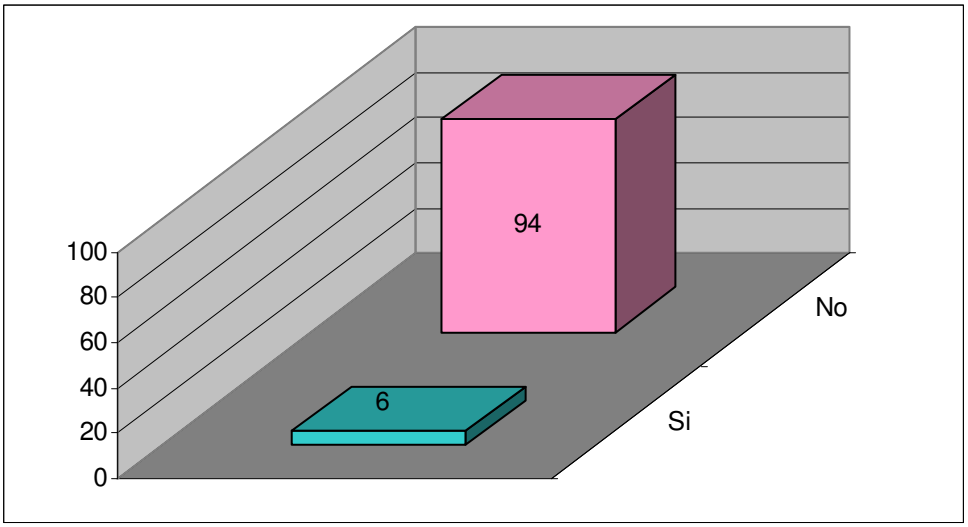


Gráfico 9. Representación grafica de ¿Ha tenido problemas con la unida de custodios alguna vez?

La mayoría de los encuestados afirma que no ha tenido ningún tipo de problema o discrepancia con este personal, pero encontramos una minoría que si ha tenido alguna vez algún percance con este personal.

Pregunta 10. Según su punto de vista, ¿Cuáles son las áreas con mayor seguridad?

Objetivo: Indagar al criterio de la comunidad cuáles son las áreas más protegidas dentro del plantel.

Respuesta	Frecuencia	Porcentaje
Bunker	5	7%
Ninguna	9	12%
Edificio de Medicina	6	8%
Cafetería	11	15%
Entrada	3	4%
Zona Verde	5	7%
No se sabe	3	4%
Parqueo	6	8%
Edificio de Usos Múltiples	11	15%

Respuesta	Frecuencia	Porcentaje
Aulas	2	3%
Edificio Ex Biblioteca	1	1.5%
Área Académica	3	4%
Pasillos	1	1.5%
Departamentos	2	3%
Toda	5	7%
Total	73	100%

Tabla 10. Tabulación de datos. Pregunta #10

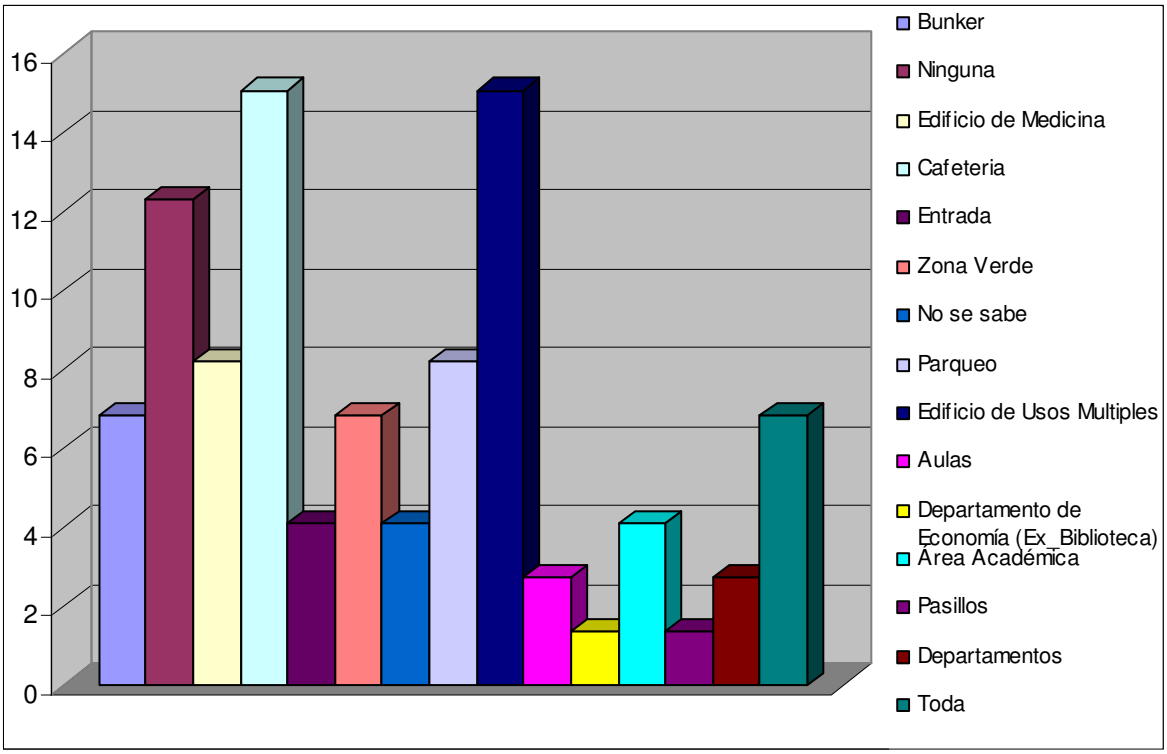


Gráfico 10. Representación grafica de Según su punto de vista, ¿Cuáles son las áreas con mayor seguridad?

En este ítem se obtuvo como resultado que los lugares con mayor seguridad en la Facultad, son la cafetería así como también el edificio de usos múltiples, por su nueva estructura construida.

Pregunta 11. Según su opinión, ¿Cuáles son las áreas con menor seguridad?

Objetivo: Indagar al criterio de la comunidad cuáles son las áreas más desprotegidas dentro del plantel.

Respuesta	Frecuencia	Porcentaje
Canchas	27	39%
Aulas	2	3%
Antes área de Medicina	1	1%
Parqueo	14	20%
Baños	6	9%
Ninguna	1	1%
Entradas	3	4%
No sabe	2	3%
Bunker	5	7%
Edificio de Medicina	4	6%
Toda	2	3%
Auditórium	2	3%
Edificio Ex_Biblioteca	1	1%
Total	70	100%

Tabla 11. Tabulación de datos. Pregunta #11

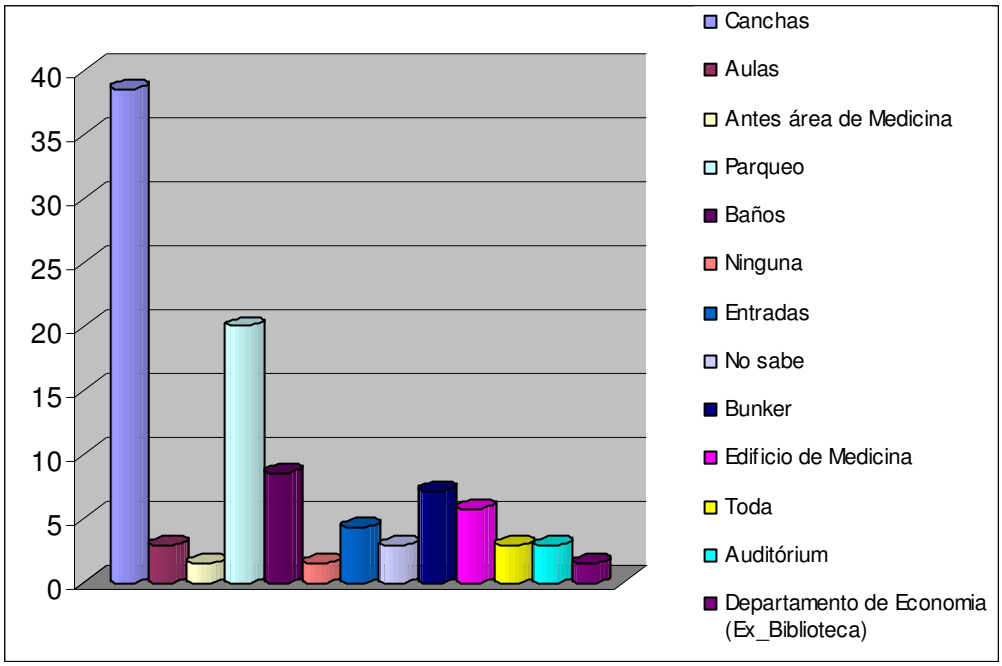


Gráfico 11. Representación grafica de Según su opinión, ¿Cuáles son las áreas con menor seguridad?

Las áreas o zonas mas desprotegidas según los encuestados son las canchas de la Universidad y el parqueo, ya que son los lugares con menos iluminación, con poco flujo de personas, convirtiéndose en un área vulnerable para los asaltos.

Pregunta 12. Según su criterio, ¿Cuáles son las áreas de la FMO que deben tener mayor seguridad?

Objetivo: Indagar al criterio de la comunidad cuáles son las áreas necesitan ser más protegidas dentro de la universidad.

Respuesta	Frecuencia	Porcentaje
Centros de Cómputo	2	2%
Bunker	8	9%
Parqueo	15	18%
Entrada	12	15%
Canchas	12	15%
Aulas	10	11%
Baños	7	8%

Respuesta	Frecuencia	Porcentaje
Cafetería	3	3%
Todas	7	8%
Edificio de Usos Múltiples	2	2%
Biblioteca	1	1%
Edificio de Medicina	3	3%
No respondió	2	2%
Auditórium	1	1%
Administración Académica	1	1%
Ninguna	1	1%
Total	87	100%

Tabla 12. Tabulación de datos. Pregunta #12

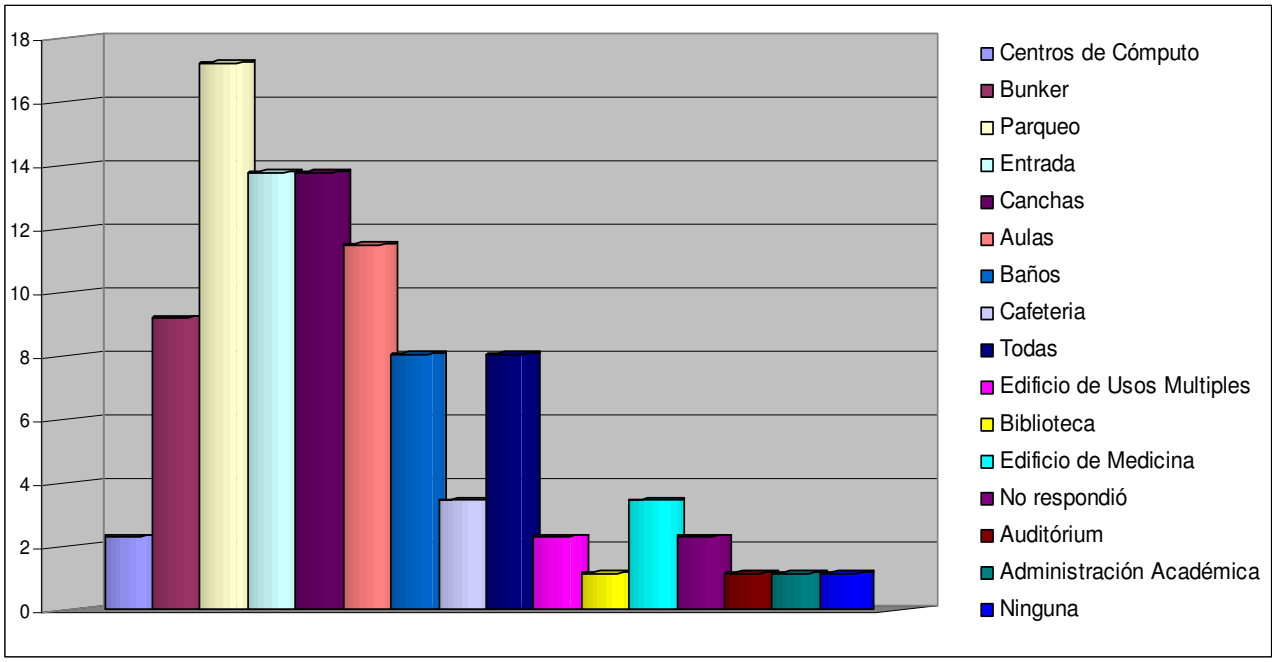


Gráfico 12. Representación grafica de Según su criterio, ¿Cuáles son las áreas de la FMO que deben tener mayor seguridad?

Con este apartado reafirman los encuestados que los lugares que necesitan mayor seguridad son el parqueo, las canchas y los accesos de entrada o salida de la institución, ya que no hay ningún tipo de control para estas.

Pregunta 13. ¿Que sugerencias hace para que halla una mejor seguridad institucional?

Objetivo: Encontrar diferentes sugerencias, que refuercen las opiniones institucionales, así como también este proyecto.

- Capacitar a las personas o hacer un estudio de ellos
- Aumentar el personal
- Mas vigilantes y gente de control
- Más iluminación.
- Tener mas conciencia social para denunciar si alguien cree que otros puedan afectar la institución o a una persona
- Instalación de cámaras de video
- Capacitar al personal en técnicas de seguridad, operativa y legal
- Mayor coordinación y mayor equipo como: armas, lámparas, etc.
- Un ordenamiento
- Mayor Control de sectores
- Vigilancia en los accesos.
- Distribución equitativa del personal
- Solicitar vigilancia de la PNC u invertir en otros vigilantes.
- Usar tecnología: cámaras o sensores
- Rondas continuas del personal de seguridad.
- Mejorar equipo de los vigilantes.
- No permitir acceso a particulares.
- Pedir identificaciones al momento de ingresar a la U.
- Menos burocracia administrativa

Pregunta 14. ¿Cree que al utilizar sensores y tecnología informática aumentaría la calidad de seguridad en la FMO?

Objetivo: Integrar la opinión de la comunidad universitaria, con la propuesta de este proyecto.

Respuesta	Frecuencia	Porcentaje
Si	38	76%
No	12	24%
Total	50	100%

Tabla 13. Tabulación de datos. Pregunta #14

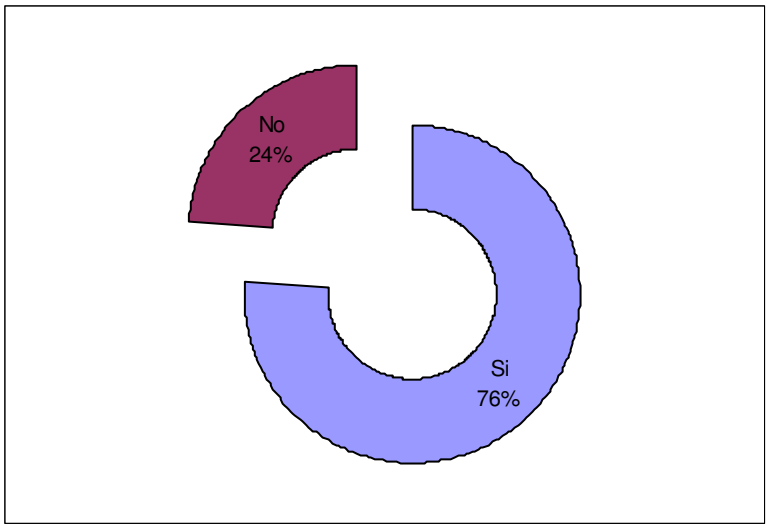


Gráfico 13. Representación grafica de ¿Cree que al utilizar sensores y tecnología informática aumentaría la calidad de seguridad en la FMO?

Como resultado a esta pregunta, se puede constatar que la opinión de los estudiantes es que si se aumentaría la calidad de seguridad en los distintos departamentos y áreas de la facultad.

ALCANCE Y PROYECCIÓN DEL PROBLEMA

UNIDADES QUE EXPERIMENTAN EL PROBLEMA

Actualmente se puede encontrar que en la mayoría de departamentos que poseen mayor equipo didáctico y de oficina, son los que presentan mayor riesgo de robo, al igual que el hurto de vehículos para lo que es el área del parqueo.
Entre estos delitos se pueden mencionar:

- El departamento de Medicina. El hurto de 5 equipos de multimedia en la oficina del área médica. Un microscopio de laboratorio.

- El edificio de usos múltiples que reúne al departamento de Idiomas, Ingeniería y Arquitectura, Matemáticas, Ciencias Sociales, el Decanato, Biblioteca y todos los centros de computo de Ingeniería, así como también salones para defensa de tesis, el hurto de equipo de computo como por ej. cañón digital, computadora portátil, etc.

- Edificio ex Biblioteca, Departamento de Ciencias Económicas, Química y Física; como por ejemplo: en el departamento de Química, el cual asevera que ya se ha dado en el laboratorio de investigación, en la secretaría y en los cubículos de los docentes; y en Ciencias Económicas la extracción de libros y UPS de biblioteca del departamento. Y en la unidad de cultura se extrajeron UPS, teléfono y cañón digital.

- Unidad de recreación y deportes, el cual manifiesta que en varias ocasiones se ha hecho la extracción de equipo de gimnasio y trofeos. Asaltos en el área de las canchas de fútbol, etc.

- Parqueo, sustracción de equipo dentro de los vehículos, robos de los mismos automóviles, rayones y daños a las llantas, etc.

En cuanto a los incidentes con los cuales se ha violentado la seguridad de los departamentos en forma general en la facultad, ha lo largo de los últimos años son: el hurto de libros de gran valor; sustracción de celulares y carteras de los cubículos o escritorios de los docentes; perdida de equipo de los diferentes laboratorios o daños físicos hacia los mismos; forzamiento violento de puertas o cerraduras de algunos departamentos, hurto de equipo informático o multimedia, entre otros.

ZONAS DE RIESGO

En su mayoría los jefes encuestados de cada unidad o departamento, mencionan que los lugares de mayor vulnerabilidad que encuentran a su cargo son las siguientes: Las puertas frágiles o sencillas las cuales poseen chapas inseguras o corrientes, no hay chapas en los escritorios y archivos; en algunos lugares no se encuentran defensas o balcones en las ventanas, la iluminación externa es deficiente, y el diseño de los nuevos edificios es vulnerable, ya que solo existe una entrada o salida para dicho lugar.

Además la infraestructura de los edificios más antiguos, así como sus instalaciones eléctricas, techos, pisos y paredes están en mal estado.
Agregando a esto, el factor humano, que muchas veces no toma las medidas de seguridad necesarias.

En el caso de algunos departamentos, las jefaturas han tomado la iniciativa de asegurar más su lugar de trabajo, ya sea reforzando las puertas o cambiando las cerraduras, el caso mas legible es el del departamento de medicina que cuenta con un sistema de alarmas y sensores de movimiento, para proteger el patrimonio con el que cuenta actualmente. Dicha inversión, fue financiada a través de la sociedad de padres. Pero dicho sistema no es utilizado por el mal manejo de parte del personal encargado. Además, cuentan con un informe de inspección realizada en las instalaciones de sus laboratorios, realizado por el cuerpo de bomberos.
Entre las zonas de riesgo mas observables se encuentran:

Las primeras zonas a considerar como puntos a reforzar son las instancias académicas donde casi o todo tipo de personas tienen acceso como lo son: la biblioteca y hemeroteca, administración académica, el decanato, los departamentos docentes y las zonas externas a los edificios; zonas de acceso restringido como: sala de servidores, colectaría o administración financiera y laboratorios académicos (Química, Biología, Idiomas, Ingeniería, etc.), incluyendo además cada una de las unidades con sus diferentes funcionalidades.

Todo esto es catalogado por su inversión en cuanto a equipo y disposición a la comunidad universitaria, referente a herramientas de diferente índole (informático, multimedia, de investigación, análisis químicos, etc.) y material educativo, como de alta seguridad (exámenes, notas, registros, etc.), por lo tanto existe una responsabilidad fuerte por cada uno de las jefaturas de cuidar este patrimonio universitario.

Entre otras zonas mas vulnerables respecto a infraestructura son las aledañas a la Facultad, donde se encuentran viviendas anexas a los edificios y los muros son muy bajos, además hay zonas donde existen solo una malla de división y no un muro perimetral adecuado, también con poca Iluminación, edificios con infraestructura deteriorada por lo antiguos que están.

PROYECCIONES DEL FENOMENO

Se hace evidente que la violencia es un tema que preocupa a todos sus ciudadanos.

Guardias de seguridad en todas las esquinas, custodiando la entrada a comercios, fábricas, vecindarios y lugares públicos, así como también ventanas con rejas y muros recubiertos con alambres de púa, son algo que despierta la atención de los visitantes, pero que parece estar ya integrado a la vida cotidiana de los lugareños.

El gobierno asegura que, pese a que el número de delitos (homicidios, robos, etc.) continúa siendo elevado, el plan Súper Mano Dura ha tenido éxito dado que antes del plan, se cometían 8.464 delitos, cifra que se redujo a 7.917.

Todas esas manifestaciones de violencia no sólo afectan la vida cotidiana, la calidad de vida de los salvadoreños, e incide negativamente en el desarrollo humano y la consolidación de la gobernabilidad democrática del país. También golpea los bolsillos de todos los salvadoreños.

Según el informe "¿Cuánto cuesta la violencia en El Salvador?", elaborado por el Programa de las Naciones Unidas para el Desarrollo (PNUD) en 2005, el costo económico alcanzó los US $1.717 millones en 2003.

El cálculo, según el PNUD, considera los costos tangibles (fondos del Estado para prevenir y combatir todo tipo de violencia, gastos de salud, costos legales, ausentismo del trabajo y productividad pérdida, etc.), pero deja fuera los intangibles o más humanos.

La cifra representó el 11,5% del Producto Interno Bruto (PIB) de ese año, equiparó el total de la recaudación tributaria, y resultó el doble de los presupuestos en salud y educación.

Terminar con la violencia no es una tarea fácil. Las rejas y los guardias de seguridad pueden ayudar a prevenir la delincuencia y a reducir la inseguridad en las calles, pero no son el instrumento adecuado para eliminarla.

Un mayor fomento de la educación, la equidad de género y mejores herramientas a nivel de justicia, resultan también instrumentos claves para lograr que la violencia -en todas sus manifestaciones- deje de ser algo cotidiano en la vida de los salvadoreños.

La falta de eficacia contra la delincuencia en general y los logros parciales contra el secuestro, se reflejan en un estudio de opinión pública, según el cual el 53.9% de las personas entrevistadas, opinó que la delincuencia ha aumentado con el actual gobierno, el 20.6% sostuvo que sigue igual y solamente el 25.5% dijo que había disminuido; en el mismo sentido, el 56.9% opinó que el plan de combate a la delincuencia ha dado poco o ningún resultado, el 28.1% sostuvo que ha dado algún resultado y solamente el 15% opinó que ha dado mucho resultado. Sin embargo, a la pregunta sobre la eficacia en el combate de los secuestros por parte de la PNC, el 63.8% opinó que se está combatiendo eficazmente, el 34% sostuvo que no es eficaz y el 2.2% no respondió.

El 29% de los salvadoreños sigue teniendo la convicción de que los problemas relacionados a la seguridad pública siguen siendo los más graves del país.

Para profundizar en este tema, LPG Datos indagó, mediante una encuesta de opinión, la incidencia delictiva que han sufrido los hogares salvadoreños este año.

En el estudio se exploró, además, la sensación de inseguridad que tienen los salvadoreños en sus hogares y barrios, al visitar los centros de las ciudades, al viajar en buses o al ir de paseo.

La investigación fue desarrollada entre los días 20 y 28 de noviembre, y fueron visitadas las áreas urbanas de 60 municipios y 49 cantones de todo el país.

La primera pregunta sobre este tema que se hizo a los encuestados fue la siguiente: ¿Usted o algún miembro de su familia ha sido víctima de algún delito en el último año? El 23.1 por ciento de los entrevistados respondió afirmativamente a la pregunta.

Lo anterior significa que uno de cada cinco hogares salvadoreños ha vivido los embates del crimen en el año que está a punto de finalizar.

La información recolectada muestra que la zona urbana del país y la región del área metropolitana de San Salvador son los lugares que más han sufrido los embates de la violencia.

En la zona rural, el 13.9% de los hogares reporta que al menos uno de los miembros de la familia fue victimizado. En la zona urbana, la cantidad de afectados es del doble: 29%. La diferencia entre la victimización en las zonas urbanas y rurales es más clara al observar que el 76% de las familias víctimas de todo el país vive en las ciudades y villas.

Los centros urbanos más afectados son los que forman el Gran San Salvador. En los 14 municipios de esta región, el 35% de las familias reporta haber sido víctima de la delincuencia. En 2º lugar aparece la zona Paracentral (19.5%), en 3° la occidental (18.8%), y finalmente la oriental. En esta última, el 13.3% de las familias fue afectada, una cifra casi tres veces menor a las reportadas en los municipios capitalinos.

La inseguridad que viven los salvadoreños se ha convertido en uno de los problemas más sentidos con mayor intensidad, junto al desempleo y el alto costo de la vida. Todas las encuestas demuestran que los salvadoreños estan a merced de la delincuencia. Los periódicos y noticieros llenan la mayor parte de sus espacios con notas de crímenes, asaltos, violaciones etc. Ya no existe familia salvadoreña que no haya sido víctima directa o indirectamente de la delincuencia.

Lo más trágico es que los tan afamados planes antidelincuenciales han fracasado en forma rotunda. No lograron ni la más leve mejoría. Todo lo contrario, pareciera que después de cada operación antidelincuencia esta surge con más fuerza. El fracaso fue evidente a finales del año pasado cuando se anunció la destitución del Director de la PNC.

Los esfuerzos de la PNC no han contribuido a elevar su eficacia en la reducción general de la delincuencia y la inseguridad en la que vive la población salvadoreña. Muestra de ello es que los propios datos de la PNC indican que la delincuencia aumentó durante el año 2000 en relación a 1999, ya que en este año se registró un total de 54,004 delitos, mientras que en el 2000 la cifra subió a 62,052, produciéndose un incremento de 8,048 delitos, que representan una alza aproximada de 15%.

Al cierre de 2006 los salvadoreños se muestran preocupados por la delincuencia en el país y prevalece una percepción pesimista respecto al futuro del país, reveló una encuesta del Instituto Universitario de Opinión Pública de la Universidad Católica Centroamericana.

Un poco más del 76% de los salvadoreños dijo que la delincuencia aumentó respecto a 2005, mientras que el 15.6% aseguró que siguió igual, frente a 8.2% que opinó que disminuyó.

La encuesta fue realizada entre el 4 y el 12 de noviembre, con una muestra nacional de mil 227 personas adultas y tiene un margen de error de 2.8 puntos porcentuales.

El principal obstáculo que vive El Salvador es la violencia en todas sus manifestaciones. La mayoría de salvadoreños son amantes de la paz y enemigos de la delincuencia.

Actualmente el país vive una de sus peores crisis de seguridad social. Ningún salvadoreño en cualquier sitio del territorio nacional está a salvo. Todos en algún momento pueden ser víctimas de la delincuencia.

La violencia afecta directa e indirectamente a todos los Salvadoreños. Por ahora es el principal obstáculo al desarrollo nacional.

Para lograr que el país crezca a una tasa superior al 7.5%, que es la apuesta del VII Encuentro Nacional de la Empresa Privada (ENADE), el principal reto planteado ayer por el sector privado es superar los altos niveles de inseguridad que vive el país.

De acuerdo con la encuesta empresarial, que ya es tradición en este cónclave, el 98.6% de los empresarios considera que El Salvador es poco o nada seguro, y siete de cada 10 denuncian haber sido víctimas, una o varias veces, de delitos como extorsión, secuestro o robo.

"Desde hace un par de años surgieron problemas de seguridad, y eso ha provocado que se pase a un período de inseguridad, pero la necesidad de inversión depende del ánimo de los empresarios", recordó Federico Colorado, presidente de la Asociación Nacional de la Empresa Privada (ANEP), al dar su discurso.

El Salvador, de escasos 20.742 kilómetros cuadrados, y 6,9 millones de habitantes, afronta una ola delictiva que además de haber generalizado las extorsiones, a diario deja un promedio de 12 homicidios y unos 500 asaltos a mano armada.

Un estudio presentado por el Programa de las Naciones Unidas para el Desarrollo (PNUD) en 2005, en base a cifras de 2003, reveló que el país destina 1.717 millones de dólares anuales (11,5% del PIB) para enfrentar la violencia en distintas áreas.

Toda esta problemática que se puede observar en el país, tiene que verse reflejado de alguna manera en las formas de inseguridad institucional y en las perdidas monetarias de la FMO. Frente a estas circunstancias, la necesidad del cambio en la seguridad representa cada vez más urgente, siendo parte de una inversión para la institución.

Además al indagar las perdidas originadas en los últimos años, según el reflejo en los estatus financieros de la FMO, lo cual significa una información restringida para los estudiantes o solo autorizada por la junta directiva de dicha institución.

Pero se obtuvo esta información indagando algunos departamentos, el promedio en pérdidas los últimos 5 años hasta la fecha:
Departamento de Medicina $ 10000
Departamento de Química $ 1000
Departamento de Ciencias Económicas $ 1500
Departamento de Ciencias Sociales $ 2000
Entre otros departamentos que no especificaron la cantidad.

Además, en dichas circunstancias y naturaleza del caso no se puede hacer un estudio de proyección del comportamiento del caso, ya que es muy improbable el que estas perdidas tengan un desenvolvimiento lineal hacia el futuro, teniendo una frecuencia aleatoria en dicho fenómeno.

ESTUDIO DE TECNOLOGÍA .NET

GENERALIDADES

Los motivos que han llevado a Microsoft al desarrollo de .NET han sido tanto tecnológicos como estratégicos.

Respecto a las motivaciones tecnológicas, la necesidad de poner a disposición del programador una plataforma de desarrollo con plena potencia para abarcar los requerimientos de las nuevas aplicaciones que están a punto de llegar, y que no soporte incómodos lastres derivados de antiguos modelos de programación, ha desembocado en una tecnología totalmente nueva, que no arrastra pesadas incompatibilidades, pero que sin embargo, permite la ejecución de componentes

basados en el anterior modelo de programación. Esto es .NET, una nueva arquitectura para el futuro del desarrollo de aplicaciones, y no, como en un principio pudiera pensarse, que proporciona las herramientas ya conocidas con algunas remodelaciones.

En cuanto a las causas estratégicas, gracias a .NET y a su modelo de distribución de software basado en servicios, Microsoft se sitúa en una posición clave en un mercado que evoluciona hacia la creación de servicios para la Web, que serán utilizados por otras aplicaciones mediante un sistema de suscripción o alquiler. Se espera que en este potencial mercado, comiencen a aparecer empresas dedicadas a la producción y publicación de servicios en Internet. La propia Microsoft, ha expresado en este sentido, su intención de convertirse en proveedor de servicios.

.NET es toda una nueva arquitectura tecnológica, desarrollada por Microsoft para la creación y distribución del software como un servicio. Esto quiere decir, que mediante las herramientas de desarrollo proporcionadas por esta nueva tecnología, los programadores podrán crear aplicaciones basadas en servicios para la Web.

CARACTERÍSTICAS

Las características principales que conforman .NET son las siguientes:

- La plataforma .NET, proporciona la infraestructura para crear aplicaciones y el entorno de ejecución para las mismas.
- Los productos de Microsoft enfocados hacia .NET, entre los que se encuentran Windows .NET Server, como sistema operativo que incluirá de forma nativa la plataforma .NET; Visual Studio .NET, como herramienta integrada para el desarrollo de aplicaciones; Office .NET; b.Central para .NET, etc.

- Servicios para .NET desarrollados por terceros fabricantes, que podrán ser utilizados por otras aplicaciones que se ejecuten en Internet.

Existen adicionalmente un conjunto de productos, que bajo la etiqueta de Servidores Empresariales para .NET (.NET Enterprise Servers) se incluyen dentro de la estrategia .NET. Entre estos productos podemos encontrar a SQL Server 2000, BizTalk Server, Commerce Server 2000, etc. Sin embargo, hemos de hacer una puntualización importante: estos productos no están basados en .NET, pueden funcionar dentro del entorno de ejecución de .NET, pero el único producto actualmente desarrollado bajo el nuevo entorno es Visual Studio .NET.

Gracias a .NET y a su modelo de desarrollo basado en servicios, se flexibiliza y enriquece el modo en el que hasta ahora se construían aplicaciones para Internet. La idea que subyace bajo esta tecnología, es la de poblar Internet con un extenso número de aplicaciones, que basadas en servicios para la Web (Web Services), formen un marco de intercambio global, gracias a que dichos servicios están fundamentados en los estándares SOAP y XML, para el intercambio de información.

En este sentido, un programador puede crear Web Services para que sean utilizados por sus propias aplicaciones a modo de componentes (pero de una forma mucho más avanzada que empleando el modelo COM clásico), siguiendo una estructura de programación ya conocida.

VISUAL STUDIO .NET

Es la nueva versión de la familia de herramientas de desarrollo de software de Microsoft, naturalmente orientadas hacia su nuevo entorno de programación: .NET Framework. Si bien es posible la escritura de programas empleando sólo el SDK

de .NET Framework, este último, al estar compuesto de herramientas independientes, constituye un medio más incómodo de trabajo.

.NET Framework es un componente integral de Windows que admite la creación y la ejecución de la siguiente generación de aplicaciones y servicios Web XML. El diseño de .NET Framework está enfocado a cumplir los objetivos siguientes:

- Proporcionar un entorno coherente de programación orientada a objetos, en el que el código de los objetos se pueda almacenar y ejecutar de forma local, ejecutar de forma local pero distribuida en Internet o ejecutar de forma remota.
- Proporcionar un entorno de ejecución de código que reduzca lo máximo posible la implementación de software y los conflictos de versiones.
- Ofrecer un entorno de ejecución de código que fomente la ejecución segura del mismo, incluso del creado por terceras personas desconocidas o que no son de plena confianza.
- Proporcionar un entorno de ejecución de código que elimine los problemas de rendimiento de los entornos en los que se utilizan secuencias de comandos o intérpretes de comandos.
- Ofrecer al programador una experiencia coherente entre tipos de aplicaciones muy diferentes, como las basadas en Windows o en el Web.
- Basar toda la comunicación en estándares del sector para asegurar que el código de .NET Framework se puede integrar con otros tipos de código.

.NET Framework contiene dos componentes principales: Common Language Runtime y la biblioteca de clases de .NET Framework. Common Language Runtime es el fundamento de la tecnología. El motor en tiempo de ejecución se puede considerar como un agente que administra el código en tiempo de ejecución y proporciona servicios centrales, como la administración de memoria, la administración de subprocesos y la interacción remota, al tiempo que aplica una seguridad estricta a los tipos y otras formas de especificación del código que fomentan su seguridad y solidez. De hecho, el concepto de administración de

código es un principio básico del motor en tiempo de ejecución. El código destinado al motor en tiempo de ejecución se denomina código administrado, a diferencia del resto de código, que se conoce como código no administrado. La biblioteca de clases, el otro componente principal de .NET Framework, es una completa colección orientada a objetos de tipos reutilizables que se pueden emplear para desarrollar aplicaciones que abarcan desde las tradicionales herramientas de interfaz gráfica de usuario (GUI) o de línea de comandos hasta las aplicaciones basadas en las innovaciones más recientes proporcionadas por ASP.NET, como los formularios Web Forms y los servicios Web XML.

.NET Framework puede alojarse en componentes no administrados que cargan Common Language Runtime en sus procesos e inician la ejecución de código administrado, con lo que se crea un entorno de software en el que se pueden utilizar características administradas y no administradas. En .NET Framework no sólo se ofrecen varios hosts de motor en tiempo de ejecución, sino que también se admite el desarrollo de estos hosts por parte de terceros.

Internet Explorer es un ejemplo de aplicación no administrada que aloja el motor en tiempo de ejecución (en forma de una extensión de tipo MIME). Al usar Internet Explorer para alojar el motor en tiempo de ejecución, puede incrustar componentes administrados o controles de Windows Forms en documentos HTML. Al alojar el motor en tiempo de ejecución de esta manera se hace posible el uso de código móvil administrado (similar a los controles de Microsoft® ActiveX®), pero con mejoras significativas que sólo el código administrado puede ofrecer, como la ejecución con confianza parcial y el almacenamiento aislado de archivos.

Common Language Runtime administra la memoria, ejecución de subprocesos, ejecución de código, comprobación de la seguridad del código, compilación y demás servicios del sistema. Estas características son intrínsecas del código administrado que se ejecuta en Common Language Runtime.

La biblioteca de clases de .NET Framework es una colección de tipos reutilizables que se integran estrechamente con Common Language Runtime. La biblioteca de clases está orientada a objetos, lo que proporciona tipos de los que su propio código administrado puede derivar funciones. Esto ocasiona que los tipos de .NET Framework sean sencillos de utilizar y reduce el tiempo asociado con el aprendizaje de las nuevas características de .NET Framework. Además, los componentes de terceros se pueden integrar sin dificultades con las clases de .NET Framework.

.NET Framework proporciona también una colección de clases y herramientas para ayudar al desarrollo y uso de las aplicaciones de servicios Web XML. Los servicios Web XML se basan en estándares como SOAP (un protocolo de llamadas a procedimientos remotos), XML (un formato de datos extensible) y WSDL (el Lenguaje de descripción de servicios Web). En .NET Framework se utilizan estos estándares para fomentar la interoperabilidad con soluciones que no son de Microsoft.

Programación con Visual Basic .NET © Grupo EIDOS 68
Visual Studio .NET (VS.NET a partir de ahora), al tratarse de un entorno de desarrollo integrado (Integrated Development Environment o IDE como también lo denominaremos a lo largo del texto), aúna todas las herramientas del SDK: compiladores, editores, ayuda, etc., facilitando en gran medida la creación de programas. Por este motivo, todas las explicaciones y ejemplos desarrollados a lo largo de este texto se harán basándose en este entorno de programación.

Clases Para Manipulación De Medios De Entrada Y Salida

Desde las primeras versiones del lenguaje, el programador ha dispuesto de un conjunto de instrucciones y funciones para el manejo de las operaciones de lectura/escritura con archivos, y la gestión de los mismos dentro del sistema

operativo, en cuanto a su creación, borrado, copia, etc., entre directorios, unidades y otros medios.

Las instrucciones Open, Input, Write, Put, etc., a pesar de resolver su cometido, no proporcionan un entorno de trabajo cómodo, en un mundo en el que cada vez prima más el trabajo con objetos.

La jerarquía de objetos FileSystemObject, introducida recientemente, vino a paliar en parte esta carencia, aportando un conjunto de clases que ya nos permitían, desde un prisma orienta a objeto, trabajar con todos los aspectos del sistema de archivos, en cuanto a su lectura, escritura, manejo de directorios, unidades, etc. La evolución de este conjunto de objetos se halla en la plataforma .NET.

***System.IO*,** Con la llegada de la tecnología .NET, el acceso al sistema de archivos, es un aspecto que ya no forma parte de un lenguaje determinado, como ocurría en las anteriores versiones de VB, sino que ha integrado dentro de la jerarquía de clases de la plataforma, en el espacio de nombres IO de System.
Con ello, todos los lenguajes compatibles con .NET podrán utilizar este conjunto de objetos.

Las clases incluidas en System.IO, permiten realizar labores de lectura y escritura en archivos de texto, binarios, etc., así como la creación y manipulación de los archivos y directorios que contienen la información.
A continuación se realiza una descripción de algunas de las clases contenidas en IO:

Objetos Stream

Un objeto Stream representa un flujo o corriente de datos, es decir, un conjunto de información guardada en formato de texto o binario, que puede leer y escribir

sobre un soporte físico, también denominado en la plataforma .NET, almacén de respaldo (backing store).

Algunos tipos de Stream, para optimizar el flujo de transferencia de datos entre el objeto y su medio físico de almacenamiento, disponen de una característica denominada almacenamiento intermedio (buffering), que consiste en mantener un búfer intermedio con los datos. En el caso, por ejemplo, de tareas de escritura, todas las operaciones se realizarían en el búfer, mientras este dispusiera de capacidad. Una vez terminado el proceso de escritura, o cuando el búfer estuviera lleno, su contenido pasaría al archivo físico. Podemos también, alterar el comportamiento por defecto del búfer a través de diversas propiedades y métodos del objeto Stream correspondiente.

Las clases TextReader y TextWriter

Estas clases contienen los miembros genéricos para realizar lectura y escritura de caracteres.

Se trata de clases abstractas, por lo que deberemos utilizar las clases derivadas StreamReader, StreamWriter, StringReader y StringWriter, comentadas a continuación:

La clase StreamWriter
Un objeto StreamWriter realiza operaciones de escritura de texto sobre un archivo.
El proceso típico de escritura de datos mediante un StreamWriter, comprende los siguientes pasos:

- **Instanciar un objeto de esta clase mediante alguno de los constructores disponibles**. Aquí se crea un nuevo archivo para escribir datos sobre él, o abrir uno existente.

- **Escritura de texto mediante los métodos WriteLine( ) y Write( )**. El primero escribe el texto pasado como parámetro, y añade los caracteres especiales de retorno de carro y nueva línea. El segundo escribe el texto pasado y deja el puntero de escritura a partir del último carácter escrito, con lo que no produce un cambio automático de línea. Se debe utilizar la propiedad NewLine para introducir manualmente un salto de línea.

- **Cierre del Stream con el método Close( )**. Esta acción vuelca el contenido del búfer del objeto en el archivo.

Algunas de las clases de tipo Stream de escritura disponen del campo compartido Null, que permite realizar una operación de escritura que no será volcada en el medio físico de almacenamiento, con lo que se perderán los datos escritos.

En el caso de que el archivo sobre el que se va a escribir ya exista, se puede utilizar un constructor de StreamWriter que permita especificar si va a añadir texto al archivo o va a sobrescribir, perdiendo el texto que hubiera.

Después de crear un objeto de este tipo, y escribir algunas líneas de texto sin cerrar el Stream, si abre su archivo de texto correspondiente, se encontrara con que no hay texto dentro del archivo. Ello es debido a que todavía no se ha volcado el contenido del búfer del objeto sobre el archivo. Para forzar dicho volcado, se debe llamar al método Flush( ), que se encarga de traspasar el búfer al archivo asociado al Stream.

La clase StreamReader

Un objeto StreamReader realiza operaciones de lectura de texto sobre un archivo.
El proceso que se debe llevar a cabo para leer el contenido de un Stream de lectura es muy similar al de escritura: instanciar el objeto con uno de sus

constructores, abriendo un archivo para leer; ejecutar alguno de los métodos de lectura del StreamReader, y cerrar el objeto con Close( ).

Entre los métodos de lectura de este objeto, se tiene ReadLine( ), que devuelve una línea del archivo; y ReadToEnd( ), que devuelve el resto del contenido del archivo, desde el punto en el que se encontrara el Stream al realizar la última lectura.

Otro de los métodos de lectura es ReadBlock( ), que recibe como parámetro un array de tipo Char, sobre el que se depositarán una cierta cantidad de caracteres leídos del archivo. En el segundo parámetro de este método indicamos la posición del array desde la que se comenzarán a guardar los caracteres. En el tercer parámetro, el número de caracteres a leer.

El método Read( ), también permite realizar una lectura igual que ReadBlock( ), pero en el caso de no utilizar parámetros, devuelve un valor numérico, correspondiente al código del carácter que acaba de leer. Cuando llega al final del Stream, devuelve –1.

Para convertir de nuevo a carácter los valores que devuelve Read( ), se debe pasar estos valores a un array de tipo Byte, y después, utilizando un objeto ASCIIEncoding, mediante su método GetString( ), se pasaría el array a una cadena.

Finalmente, el método Peek( ), al igual que Read( ), devuelve el siguiente valor disponible del Stream, pero sin extraerlo del búfer, con lo que se debere utilizar alguno de los métodos anteriormente descritos para realizar una lectura real.

Las clases StringWriter y StringReader

Estas clases proporcionan la misma funcionalidad que StreamWriter y

StreamReader, con la diferencia de que StringWriter trabaja con un objeto StringBuilder como almacén de datos, mientras que StringReader utiliza un String para leer su contenido.

La clase Stream (flujo de datos)

La clase Stream representa un flujo o corriente de datos, es decir, un conjunto secuencial de bytes, como puede ser un archivo, un dispositivo de entrada/salida, memoria, un conector TCP/IP, etc.

Se trata de una clase abstracta, por lo que si se quiere hacer uso de un stream concreto, se tiene que acudir a alguna de sus clases derivadas como son FileStream, MemoryStream, BufferedStream, etc.

La clase FileStream

Realiza escritura y lectura de bytes sobre un archivo; en el caso de que el archivo no exista, lo crearía al mismo tiempo que se instancia este objeto.

Uno de los constructores de esta clase, permite especificar una cadena con la ruta del archivo a utilizar, mientras que en el segundo parámetro se utiliza un valor de la enumeración FileMode, mediante la que indica el modo de trabajo sobre el archivo: añadir, abrir, crear, etc.

Las propiedades CanRead, CanWrite y CanSeek, devuelven un valor lógico que informa de si en el objeto se puede realizar operaciones de lectura, escritura y desplazamiento por los bytes que contiene.

Para escribir datos, se dispone del método WriteByte( ), que escribe un byte en el archivo; y también se tiene el método Write( ), que escribe de un array de bytes

pasado como parámetro, una cantidad de elementos determinada empezando por una de las posiciones de dicho array.

Para las operaciones de lectura, se tiene ReadByte( ), que devuelve el valor sobre el que esté posicionado el objeto en ese momento. También se dispone del método Read(), que traspasa valores un array de bytes.

Si se quiere desplazar por los elementos del Stream, se puede utilizar el método Seek(), pasando la cantidad de posiciones a mover, y el punto desde el que se quiere realizar dicho desplazamiento, mediante los valores de la enumeración SeekOrigin.

Para averiguar el elemento del Stream en el que se esta situado, se dispone de la propiedad Position.

Las clases BufferedStream y MemoryStream, que también heredan de Stream, disponen de los mismos miembros que FileStream, teniendo como principal diferencia el que utilizan la memoria de la máquina como almacén de respaldo.

Manejo de datos binarios

Las clases BinaryWriter y BinaryReader, tal y como se puede anticipar por sus nombres, permiten escribir y leer respectivamente, datos binarios en archivos, utilizando los métodos ya vistos.

Manipulación de archivos mediante File y FileInfo

Las clases File y FileInfo, proporcionan a través de sus miembros, el conjunto de operaciones comunes que se puede realizar con archivos en cuanto a su creación, copia, borrado, etc.

La diferencia principal entre ambas radica en que los miembros de File son todos compartidos, con lo cual se facilita en gran medida su uso, al no tener que crear una instancia previa de la clase; mientras que en FileInfo se debe crear un objeto para poder utilizarla, ya que sus miembros son de instancia.

FileInfo dispone de algunos métodos adicionales que no se encuentran en File. Comenzando por la clase File, los métodos CreateText( ) y OpenText( ), devuelven respectivamente un objeto StreamWriter y StreamReader, que se utiliza para escribir y leer en el archivo pasado como parámetro a estos métodos. Con el método Exists( ), se comprueba si existe un determinado archivo.

Para obtener los atributos de un archivo, se dispone del método GetAttributes( ), al que se pasa la ruta de un archivo, y devuelve un valor de la enumeración FileAttributes con la información sobre los atributos. En el caso de que al intentar acceder a un archivo, este no exista, se producirá una excepción de tipo FileNotFoundException, que se puede tratar en una estructura de manejo de excepciones.

Además de esta excepción, el espacio de nombres IO proporciona algunas clases de excepción adicionales para tratar otras diversas circunstancias de error.

Los métodos Copy( ), Move( ) y Delete( ), permiten copiar, mover y borrar respectivamente el nombre de archivo que pase como parámetro. El método GetCreationTime( ) devuelve un tipo Date con la fecha de creación del archivo.

Por otro lado, si se quiere obtener información adicional sobre un archivo, como su nombre, extensión, ruta, etc., se instanciará un objeto FileInfo( ), pasando al constructor una cadena con el nombre del archivo, y se utilizará algunas de sus propiedades como Name, Extensión, DirectoryName.

Manipulación de archivos mediante Directory y DirectoryInfo

Las clases Directory y DirectoryInfo contienen métodos y propiedades para crear, borrar, copiar y mover directorios, así como otra serie de tareas para su manejo y obtención de información. Los miembros de Directory son compartidos, mientras que los de DirectoryInfo son de instancia; esta es su principal diferencia.

En el método Exists( ) comprueba la existencia de un directorio, y en caso afirmativo, se obtene su última fecha de uso mediante GetLastAccessTime( ). Seguidamente se obtene un array String con su lista de archivos mediante GetFiles( ), y crea un subdirectorio de respaldo con CreateSubdirectory( ). En caso de que el directorio no exista, se crea con CreateDirectory( ).
Para obtener el directorio actual de ejecución, disponemos del método GetCurrentDirectory( ), mientras que si se quiere subir al directorio de nivel superior, se tiene el método GetParent( ), que devuelve un tipo DirectoryInfo, con el que se puede, por ejemplo, mostrar su nombre completo mediante la propiedad FullName, y fecha de creación con CreationTime.

El método GetDirectories( ) devuelve un array de cadenas, con los nombres de los subdirectorios que se encuentran dentro del directorio pasado como parámetro a este método. A continuación, mediante el método Move( ), se cambia de lugar un directorio; con Delete( ) se borra otro de los directorios. Utilizando de forma combinada CType( ), Directory.GetFiles( ), y un elemento del array que contiene la lista de directorios, se crea una expresión que permite averiguar, si en un determinado directorio hay o no archivos.

La clase Path

Esta clase proporciona un conjunto de campos y métodos compartidos, para la obtención de información y manipulación de rutas de archivos.

Monitorización del sistema de archivos con FileSystemWatcher

Esta clase contiene los mecanismos necesarios, que van a permitir la creación de objetos que actúen como observadores de los sucesos que ocurran en el sistema de archivos de un equipo local o remoto en cuanto a la creación, borrado, modificación, etc., de archivos y directorios.

La creación de este proceso de vigilancia se puede dividir en dos partes: instanciación y configuración del propio objeto FileSystemWatcher; y la escritura de los procedimientos manipuladores de los diversos eventos que pueden ocurrir sobre los archivos y directorios.

Para facilitar la escritura de los manipuladores de evento, se puede declarar una variable de esta clase a nivel de módulo, con la palabra clave WithEvents.

Declarado el objeto FileSystemWatcher, lo instanciará y asignará valor a las propiedades mencionadas a continuación, que permitirán configurar el modo de observación que realizará este objeto sobre los archivos.

- **Path**. Tipo String. Contiene la ruta de la unidad de disco sobre la que se efectuará la monitorización.
- **Filter**. Tipo String. Contiene el tipo de fichero que se va a observar, admitiendo los caracteres comodín; por ejemplo: “*.*”, “*.txt”.
- **IncludeSubdirectories**. Tipo Boolean. Establece si se van a monitorizar los subdirectorios de la ruta especificada en la propiedad Path. El valor True incluye los subdirectorio, mientras que False no los incluye.
- **EnableRaisingEvents**. Tipo Boolean. Activa el proceso de observación sobre el sistema de archivos, teniendo en cuenta la configuración establecida en las demás propiedades mencionadas arriba. El valor True pone en marcha el mecanismo de observación, mientras que el valor False lo detiene.

Para completar este proceso que se esta describiendo, sólo queda escribir los procedimientos que van a ejecutarse cuando se realice la creación, borrado, modificación, etc., de un archivo.

Puesto que se ha declarado la variable FileSystemWatcher a nivel del módulo de código, se seleccionará dicha variable en la lista desplegable *Nombre de clase*, del editor de código.

Seguidamente, se abrirá la lista *Nombre de método*, también del editor; seleccionando el evento a codificar. Las anteriores acciones, crearán el procedimiento de evento correspondiente, pero vacío, por lo que se tendrá que escribir el código que se quiere ejecutar en respuesta a tal evento. La lista de parámetros de este procedimiento consiste en un tipo Object, que contiene la referencia al objeto FileSystemWatcher que originó el evento; y un tipo FileSystemEventArgs, que contiene información adicional sobre el evento ocurrido, como el nombre y ruta del archivo.

Ajuste preciso de filtros para el monitor de archivos

Si se quiere realizar un filtro más puntual, por ejemplo, cuando se hagan cambios sobre los archivos a monitorizar, la clase FileSystemWatcher dispone de la propiedad NotifyFilter, que contiene una enumeración de tipo NotifyFilters, cuyos valores se pueden combinar para que sólo se detecten los cambios al modificar el tamaño y/o la última escritura sobre un archivo.

Manipulación de archivos mediante funciones específicas de Visual Basic

Por cuestiones de compatibilidad y migración de aplicaciones existentes, estas instrucciones han sido transformadas en funciones, para facilitar su manejo.

Funciones como FileOpen( ), para abrir un archivo; FileClose( ), para cerrarlo;

LineInput( ), para leer una línea de texto de un archivo, etc., son las que permiten en la actual versión del lenguaje, realizar las operaciones que anteriormente se efectuában mediante sus correspondientes instrucciones.

A pesar de que estas funciones permiten la manipulación de ficheros, se debe tener muy presente que se trata de elementos fundamentalmente proporcionados para compatibilidad con versiones anteriores, por lo que se recomienda que cuando se tenga que hacer cualquier tipo de operación con archivos en cuanto a su lectura, escritura, manipulación, etc., se utilice las clases del espacio de nombres IO.

Manipulación de puertos seriales

A continuación, establecemos cada uno de los manipuladores mediante este sistema hace la captura de datos.

La simplicidad con la cual se pueden manipular los Puertos Seriales utilizando El .Net está dotado de nuevas y muchas clases; también de un namespace que hacen realmente posible la interacción con los puertos serie del computador de una manera eficiente, fácil y sobre todo dentro del ambiente de código manejado sin necesidad de recurrir a librerías y objetos de terceros.

La utilización de algunas de las nuevas funciones y namespaces que proveen acceso a los recursos del sistema son:

1. El namespace My para acceder a la lista de puertos en el computador:

   Por ejemplo:

```
For Each strPort As String In My.Computer.Ports.SerialPortNames
  tscbPorts.Items.Add(strPort)
Next
```

2. **System.IO.Ports** para crear un objeto que interactúe directamente con los puertos serie, y se abre el puerto directamente desde los recursos del sistema:

   Por ejemplo:

   Private WithEvents RS232 **As New** System.IO.Ports.SerialPort
   RS232 = **My**.Computer.Ports.OpenSerialPort(SelectedPort)

3. La instrucción **For Each** para obtener la lista de puertos disponibles desde la colección **My.Computer.Ports.SerialPortNames**; estos datos son obtenidos en formato **string** y luego son colocados en una lista desplegable:

   Por ejemplo:

```
For Each strPort As String In My.Computer.Ports.SerialPortNames
  tscbPorts.Items.Add(strPort)
Next
If tscbPorts.Items.Count > 0 Then   tscbPorts.SelectedIndex = 0
Else
  'No hay puertos en el sistema
End If
```

El código anterior puebla la lista desplegable **tscbPorts** con los nombres de los puertos, luego determina si se encontró algún puerto y, si esto es así, selecciona por defecto el primero de ellos.

Debido a que la manipulación de puertos es muy volátil, es necesario tomar control sobre las excepciones con el objeto de evitar el mal funcionamiento de la aplicación y a la vez informar al usuario de las causas y fallas en la apertura del puerto. Para esto, se encierra el código en un bloque **Try, Catch, End Try** y se utiliza las excepciones más comunes -**System.IO.IOException** y **System.UnauthorizedAccessException**- para obtener un reporte detallado del error, ya sea porque el puerto esté abierto o no exista, y finalmente se utiliza **System.Exception** para atrapar cualquier otra excepción que pueda surgir:

Por ejemplo:

```
Dim SelectedPort As String = tscbPorts.SelectedItem
Try
```

```
   RS232 = My.Computer.Ports.OpenSerialPort(SelectedPort, 9600,
IO.Ports.Parity.None, 8, IO.Ports.StopBits.One)
   txtConsole.AppendText(RS232.PortName & " abierto a las " &_
   Now.ToString & vbCrLf)
   tscbPorts.Enabled = False
Catch ex As System.IO.IOException
   txtConsole.AppendText("Error abriendo el puerto: " & _
   vbCrLf & ex.Message & vbCrLf)
Catch ex As System.UnauthorizedAccessException
   txtConsole.AppendText("El pueto ya esta abierto: " & vbCrLf & _
   ex.Message & vbCrLf)
Catch ex As System.Exception
   txtConsole.AppendText("Error general accediendo al puerto:" & _
   vbCrLf & ex.Message & vbCrLf)
End Try
```

Como se puede observar, en la tercera línea el puerto es abierto e inicializado con los parámetros de configuración ya preestablecidos dentro del código. Para llevar a cabo una codificación más robusta y flexible al usuario, se recomienda utilizar un archivo .XML para guardar esta configuración.

A menudo se esta interactuando con dispositivos que pueden no requerir una transmisión de datos en ambos sentidos o que pueden genera información de una manera espontánea. Es entonces donde entra en juego el evento **DataReceived**, el cual se dispara en el mismo momento en que se detecta la llegada de datos al puerto previamente abierto. Este evento se dispara en un hilo o **Thread** separado; puedes manipular los datos en ese hilo de modo separado, como por ejemplo enviarlos a una base de datos, pero muchas veces se necesita pasar esta información al hilo principal para desplegar estos datos en algún objeto del formulario; se necesita utilizar un delegado para invocar un método dentro del hilo principal que contiene la Interfaz de Usuario. Como por ejemplo:

```
Delegate Sub WriteDataDelegate(ByVal str As String)
```

A lo largo del desarrollo del proyecto, es decir en capítulos posteriores se darán a conocer las clases funcionales de .NET, en que se basara esta aplicación.

REQUERIMIENTOS

Requisitos hardware

A continuación se muestra una lista con las características mínimas y recomendadas que debe tener el equipo en el que instalemos VS.NET.

Requisitos mínimos

- PC con procesador Pentium de 100 MHz
- Microsoft Windows® 98, Windows® NT 4.0 Service Pack 6 o posterior
- 32 MB de RAM
- Unidad de instalación con 450 MB de espacio mínimo en disco para la instalación típica y 1,8 GB para la instalación completa
- Monitor Super VGA (800 x 600) con 256 colores
- Unidad de CD-ROM de 4X
- Microsoft Internet Explorer 5.5

Requerimientos hardware para instalar Visual Studio .NET.

Sistema operativo:

VS.NET puede ser instalado en un equipo con uno los siguientes sistemas operativos:

- Windows 2000 (se requiere tener instalado el Service Pack 2).
- Windows NT 4.0. (se requiere tener instalado el Service Pack 5).
- En los casos como Windows Me, y versiones anteriores, no es recomendado, por los escasos recursos físicos que estos poseen.

Para aprovechar todo el potencial de desarrollo de la plataforma, es recomendable usar como sistema operativo Windows 2000, ya que ciertos aspectos del entorno (las características avanzadas de gestión gráfica por ejemplo) no están disponibles si instalamos .NET en otro sistema con menos prestaciones.

Requisitos recomendados

- Procesador Pentium II – 450 MHz Pentium III – 733 MHz o superior
- Memoria 128 MB 256 MB
- Espacio en disco duro 3 GB o Unidad de instalación con 450 MB de espacio mínimo en disco para la instalación típica y 1,8 GB para la instalación completa
- Microsoft Windows XP® o Microsoft Windows® 2000
- 1024 x 768 con color de alta densidad de 16 bits
- Unidad de CD-ROM de 8X
- Microsoft Internet Explorer 6

DESCRIPCIÓN SOBRE TECNOLOGÍAS SENSORIALES

INTRODUCCIÓN

Para esta descripción, se llevo a cabo una investigación, con la empresa ALTESA S.A., sobre las tecnologías utilizadas actualmente o en forma estándar para los sistemas de alarmas de nuestro país. Los cuales se definen a lo largo de este capitulo.

Dichos sensores se han tomado en cuenta para la aplicación de este proyecto, en el cual se podrán utilizar algunos ejemplos de ellos, tomando en cuenta que son accesibles para la inversión, además de la fácil instalación y uso al manipularlos en áreas internas de una institución.

CONCEPTOS BÁSICOS DE LOS SENSORES (ACTUADOTES)

SENSORES: CLASIFICACION

Los términos sensor y transductor se suelen aceptar como sinónimos, aunque si

hubiera que hacer alguna distinción, el termino transductor es quizás mas amplio, incluyendo una parte sensible o captador propiamente dicho y algún tipo de circuito de acondicionamiento de la señal detectada. Si nos centramos en el estudio de los transductores cuya salida es una señal eléctrica, podemos dar la siguiente definición: *Un transductor es un dispositivo capaz de convertir el valor de una magnitud física en una señal eléctrica codificada, ya sea en forma analógica o digital.*

No todos los transductores tienen porque dar una salida en forma de señal eléctrica. Como por ejemplo puede valer el caso de un termómetro basado en la diferencia de dilatación de una lamina bimetálica, donde la temperatura se convierte directamente en un desplazamiento de una aguja indicadora.

Sin embargo, el termino transductor suele asociarse bastante a dispositivos cuya salida es alguna magnitud eléctrica o magnética y, por otro lado, nos interesan aquí solo este tipo de transductores, en la medida que son elementos conectables a autómatas programables a través de las interfaces adecuadas.

Limitándonos, pues, a los transductores basados fenómenos eléctricos o magnéticos, estos suelen tener una estructura general, en la cual podemos distinguir las siguientes partes:

- Elemento sensor o captador. Convierte las variaciones de una magnitud física en variaciones de una magnitud eléctrica o magnética, que denominamos habitualmente señal.
- Bloque de tratamiento de señal. Si existe, suele filtrar, amplificar, linealizar y, en general, modificar la señal obtenida en el captador, por regla general utilizando circuitos eléctricos.
- Etapa de salida. Esta etapa comprende los amplificadores, interruptores, conversores de código, transmisores y, en general, todas aquellas partes que adaptan la señal a las necesidades de la carga exterior.

Podemos dar varias clasificaciones de los transductores de tipo eléctrico o magnético, atendiendo a diversos puntos de vista que vamos a repasar a continuación.

CLASIFICACIONES SEGÚN EL TIPO DE SEÑAL DE SALIDA.

Atendiendo a la forma de codificar la magnitud medida podemos establecer una clasificación en:

- Analógicos. Aquellos que dan como salida un valor de tensión o corriente variable en forma continua dentro del campote medida. Es frecuente para este tipo de transductores que incluyan una etapa de salida para suministrar señales normalizadas de 0-10 v o 4-200 mA.
- Digitales. Son aquellos que dan como salida una señal codificada en forma de pulsos o en forma de una palabra digital codificada en binario, BCD u otro sistema cualquiera.
- Todo-nada. Indican únicamente cuando la variable detectada rebasa un cierto umbralo o limite. Puede considerarse como un caso limite de los sensores digitales en el que se codifican solo dos estados.

Otro criterio de clasificación, relacionado con la señal de salida, es el hecho de que el captador propiamente dicho requiera o no una alimentación externa para su funcionamiento. En el primer caso se denominan sensores pasivos y en el segundo casos activos o directos.

Los sensores pasivos se basan, por lo general, en la modificación de la impedancia eléctrica o magnética de un material bajo determinadas condiciones físicas o químicas (resistencia, capacidad, inductancia, reluctancia, etc.). Este tipo de sensores, debidamente alimentados, provoca cambios de tensión o de corriente en un circuito, los cuales son recogidos por el circuito de interfaz.

Los sensores activos son, en realidad, generadores eléctricos, generalmente de pequeña señal. Por ello no necesitan alimentación exterior para funcionar, aunque si suelen necesitarla para amplificar la débil señal del captador.

MAGNITUD DETECTADA	TRANSDUCTOR	CARACTERISTICAS
Posición lineal o angular	Potenciómetro	Analógico
	Encoders	Digital
	Sincro y resolver	Analógicos
Pequeños desplazamientos o deformaciones	Transformador diferencial	Analógico
	Galga extensométrica	Analógico
Velocidad lineal o angular	Dinamo tacométrica	Analógico
	encoders	Digital
	Detector inductivo u óptico	Digitales
Aceleración	Acelerómetro	Analógico
	Sensor de velocidad + calculador	Digital
Fuerza y par	Medición indirecta (galgas o trafos diferenciales)	Analógicos
Presión	Membrana + detector de desplazamiento	Analógicos
	Piezoeléctricos	Analógicos
Caudal	De turbina	Analógico
	Magnético	Analógico
Temperatura	Termopar	Analógico
	Resistencias PT100	Analógico
	Resistencias NTC	Analógico
	Resistencia PTC	Todo-nada
	Bimetalitos	Todo-nada
Sensores de presencia o proximidad	Inductivos	Todo-nada o Analógicos
	Capacitivos	Todo-nada
	Ópticos	Todo-nada o Analógicos
	Ultrasónicos	Analógicos
Sensores táctiles	Matriz de contactos	Todo-nada
	Matriz capacitiva piezoeléctrica u óptica	Todo-nada
	Piel artificial	Analógico
Sistemas de visión artificial	Cámaras de video y tratamiento imagen	Procesamiento digital por puntos o pixels
	Cámaras CCD	

CLASIFICACION SEGÚN MAGNITUD FISICA A DETECTAR.

En cuanto a la naturaleza de la magnitud física a detectar, existe una gran variedad de sensores en la industria, en la tabla se da un resumen de los más

frecuentes utilizado en los automatismos industriales. Obsérvese que en la columna encabeza como transductor aparece a veces el nombre del elemento captador de dicho transductor, sobre todo en casos de medición indirecta. Así, por ejemplo, para fuerza y par se utilizan captadores de deformación unidos a piezas mecánicas elásticas.

En general, los principios físicos en los que suelen estar basados los elementos sensores son los siguientes:

- Cambios de resistividad,
- Electromagnetismo(inducción electromagnética),
- Piezoelectricidad,
- Efecto fotovoltaico,
- Termoelectricidad.

CARACTERISTICAS GENERALES DE LOS SENSORES

El comportamiento de un sistema en lazo cerrado depende muy directamente de los transductores e interfaces empleados en el lazo de retroalimentación. Es mas, tal como se ha visto la relación salida/entrada en régimen permanente depende casi exclusivamente del bucle de retroalimentación. Así pues, dejando a un lado las características constructivas particulares de cada transductor o de cada sistema de medida previsto como lazo de realimentación, es importante conocer diversos aspectos genéricos de su comportamiento a fin de prever o corregir la actuación tanto estática como dinámica del lazo de control.

Un transductor ideal seria aquel en que la relación entre la magnitud de salida y la variable de entrada fuese puramente proporcional y de respuesta instantánea e idéntica para todos los elementos de un mismo tipo. Sin embargo, la respuesta real de los transductores nunca es del todo lineal, tiene un campo limitado de

validez, suele estar afectada por perturbaciones del entorno exterior y tiene un cierto retardo a la respuesta. Todo ello hace que la relación salida/entrada deba expresarse por una curva, o mejor por una familia de curvas, para transductores de un mismo tipo y modelo.

Para definir el comportamiento real de los transductores se suelen comparar estos con un modelo ideal de comportamiento o con un transductor patrón y se definen una serie de características que ponen de manifiesto las desviaciones respecto a dicho modelo. Dichas características pueden agruparse en dos grandes bloques:

- Características estáticas, que describen la actuación del sensor en régimen permanente o con cambios muy lentos de la variable a medir.
- Características dinámicas, que describen la actuación el sensor en régimen transitorio, a base de dar su respuesta temporal ante determinados estímulos estándar o a base de identificar el comportamiento del transductor con sistemas estándar, e indicar las constantes de tiempo relevantes.

A continuación se dan las definiciones de las características estáticas y dinámicas más relevantes que suelen aparecer en la mayoría de especificaciones técnicas de los transductores. Debe tenerse en cuenta que todas las características suelen variar con las condiciones ambientales. Por ello, uno de los parámetros esenciales a comprobar al elegir un transductor es el campo de validez de los parámetros que se indican como nominales del mismo y las máximas desviaciones provocadas por dichas condiciones ambientales.

CARACTERÍSTICAS ESTATICAS

CAMPO DE MEDIDA.

El campo de medida, es el rango de valores de la magnitud de entrada comprendido entre el máximo y el mínimo detectables por un sensor, con una tolerancia de error aceptable.

RESOLUCION.

Indica la capacidad del sensor para discernir entre valores muy próximos de la variable de entrada. Se mide por la mínima diferencia entre dos valores próximos que el sensor es capaz de distinguir. Se puede indicar en términos de valor absoluto de la variable física medida o en porcentaje respecto al fondo de escala de la salida.

PRECISION.

La precisión define la máxima desviación entre la salida real obtenida de un sensor en determinadas condiciones de entorno y el valor teórico de dicha salida que correspondería , en idénticas condiciones, según el modelo ideal especificado como patrón. Se suele indicar en valor absoluto de la variable de entrada o en porcentaje sobre el fondo de escala de la salida.

REPETIBILIDAD.

Característica que indica la máxima desviación entre valores de salida obtenidos al medir varias veces un mismo valor de entrada, con el mismo sensor y en idénticas condiciones ambientales.

Se suele expresar en porcentaje referido al fondo de escala y da una indicación del error aleatorio del sensor. Algunas veces se suministran datos de repetibilidad variando ciertas condiciones ambientales, lo cual permite obtener las derivas ante dichos cambios.

LINEALIDAD.

Se dice que un traductor es lineal, si existe una constante de proporcionalidad única que relaciona los incrementos de señal de salida con los correspondientes incrementos de señal de entrada, en todo el campo de medida.

La no linealidad se mide por la máxima desviación entre las respuestas real y la característica puramente lineal, referida al fondo de escala.

SENSIBILIDAD.

Características que indican la mayor o menor variación de la salida por unidad de la magnitud de la entrada. Un sensor es tanto mas sensible cuanto mayor sea la variación de la salida producida por una determinada variación de entrada. La sensibilidad de mide, pues, por la relación:

$$\text{Sensibilidad} = \frac{\text{magnitud de salida}}{\text{Magnitud de entrada}}$$

Obsérvese que para traductores lineales esta relación es constante en todo el campo de medida, mientras que en un traductor de respuesta no lineal depende del punto en que se mida.

RUIDO.

Se entiende por ruido cualquier perturbación aleatoria del propio transductor o del sistema de medida, que produce una desviación de la salida con respecto al valor teórico.

HISTERESIS.

Se dice que un transductor presenta histéresis cuando, a igualdad de la magnitud de entrada, la salida depende de dicha entrada se alcanzo con aumentos en sentido crecientes o en sentido decrecientes. Se suele medir en términos de valor absolutos de la variable física o en porcentaje sobre el fondo de escala. Obsérvese que la histéresis puede no ser constante en todo el campo de medida.

En el caso de sensores todo-nada se denomina histéresis a la diferencia entre el valor de entrada que provoca el basculamiento de 0 → 1 y aquel que provoca el basculamiento inverso de 1 → 0.

Obsérvese la clara diferencia entre los términos resolución, precisión, repetibilidad y sensibilidad, términos que suelen confundirse muchas veces, incluso en alguna bibliografía.

CARACTERÍSTICAS DINAMICAS.

La mayor parte de transductores tienen un comportamiento dinámico que se puede asimilar a un sistema de primer o segundo oren, es decir, con una o, como máximo, dos constantes de tiempo dominantes (véase el concepto de la constante de tiempo en el capitulo f3). Los principales parámetros que caracterizan el comportamiento dinámico de un transductor serán, pues, los que se definieron para estos tipos de sistemas. Solo cabe destacar que los transductores que responden a modelos de segundo orden suele ser sistemas sobre amortiguados, es decir, sistemas en los que no hay rebasamiento en la respuesta al escalón. A continuación damos un resumen de las características dinámicas más importantes:

VELOCIDAD DE RESPUESTA

La velocidad de respuesta mide la capacidad de un transductor para que la señal de salida siga sin retraso las variaciones de la señal de entrada. La forma de cuantificar este parámetro es a base de una o más constantes de tiempo, que suele obtenerse de la respuesta al escalón. Los parámetros mas relevantes empleados en la definición de la velocidad de respuesta son los siguientes:

Tiempo de retardo:

Es el tiempo transcurrido desde la aplicación del escalón de entrada hasta que la salida alcanza el 10% de su valor permanente.

Tiempo de subida:

Es el tiempo transcurrido desde la salida alcanza el 10% de su valor permanente hasta que llega por primera vez al 90% de dicho valor.

Tiempo de establecimiento al 99%.

Es el tiempo transcurrido desde la aplicación de un escalón de entrada hasta que la respuesta alcanza el régimen permanente, con una tolerancia de $\pm$ 1%.

Constante de tiempo:

Para un transductor con respuesta de primer orden (una sola constante de tiempo dominante) se puede determinar la constante de tiempo a base de medir el tiempo de empleado para que la salida alcance el 63% de su valor de régimen permanente, cuando a la entrada se le aplica un cambio en escalón.

RESPUESTA FRECUENCIAL

Relación entre la sensibilidad y la frecuencia cuando la entrada es una excitación senoidal. Se suele indicar gráficamente mediante un grafico de bode.

Tal como se vio en el capitulo 3, la respuesta frecuencial esta muy directamente relacionada con la velocidad de repuesta.

ESTABILIDAD Y DERIVAS.

Características que indican la desviación de la salida del sensor al variar ciertos parámetros exteriores distintos del que se pretende medir, tales como condiciones ambientales, alimentación, u otras perturbaciones.

TRANSDUCTORES DE POSICION: CONCEPTOS GENERALES.

Los traductores de posición permiten medir la distancia de un objeto respecto a un punto o eje de referencia o simplemente detectar la presencia de un objeto a una cierta distancia. Precisamente, su capacidad de medida o solo indicación de presencia y la capacidad de medir distancias más o menos grandes que permite establecer una división en los grupos que se citan a continuación:

- Detectores de presencia o proximidad. se trata de sensores de posición todo o nada que entregan una señal binaria que informa de la existencia o no de un objeto ante el detector. El más elemental de estos sensores es quizás el conocido interruptor final de carrera por contacto.
- Medidores de distancia o posición. Entregan una señal analógica o digital que permite determinar la posición lineal o angular respecto a un punto o eje de referencia.

- Transductores de pequeñas deformaciones. Se trata de sensores de posición especialmente diseñados para detectar pequeñas deformaciones o movimientos. Muchas veces se emplean adosados a piezas elásticas o con palpadores como transductores indirectos de fuerza o de par.

DETECTORES DE PROXIMIDAD
CONCEPTOS GENERALES

Los detectores de proximidad pueden estar basados en distintos tipos de captadores, siendo los mas frecuentes los siguientes:

- Detectores inductivos
- Detectores capacitivos
- Detectores ópticos
- Detectores ultrasónicos

Por lo general, se trata de sensores con respuesta todo o nada, con una cierta histéresis en la distancia de detección y con salida a base de interruptor estático (transistor, tiristor o triac), pudiendo actuar como interruptores de CC o de CA. Pero, algunos de ellos pueden llegar a dar una salida analógica proporcional a la distancia. En tal caso, los estudiaremos como verdaderos medidores de posición.

Atendiendo al tipo de alimentación (CC o CA), al tipo de salida y a la forma de conexión podemos clasificar los detectores de proximidad en diferentes grupos.

CLASIFICACIÓN SEGÚN EL TIPO DE SALIDA.

- Detectores todo o nada de CA. Se trata de detectores cuya salida es un interruptor estático de CA a base de tiristores o triacs. Por lo general, no

pueden utilizarse más que para CA, ya que para CC una vez cebados no desenganchan.

- Detectores todo o nada de CC. Se trata de detectores cuya salida suele ser un transistor PNP o NPN. Precisamente el tipo de transistor determina la forma de conexión de la carga.
- Detector Namur. Detectores de tipo inductivo, previstos para funcionamiento en atmósferas explosivas, según recomendaciones NAMUR (DIN 19.234). son detectores de os hilos que absorben una intensidad alta o baja dependiendo de la presencia o no del objeto detectado. La actuación puede considerarse todo o nada con una histéresis, igual que los tipos mencionados anteriormente. En general, se usan como captadores en atmósferas explosivas y la señal que generan se conecta a un amplificador externo con relé de salida.
- Detector con salida analógica. Los detectores con salida analógica dan una corriente proporcional a la distancia entre el cabezal detector y el objeto a detectar. La conexión suele ser a dos hilos y permite detectar un rango de distancias limitado. Los de tipo inductivo y capacitivo tienen una linealidad y una resolución bastante pobres, que hace que no puedan emplearse como verdaderos medidores d distancia. Únicamente los de tipo óptico y ultrasónico pueden detectar distancias considerables con una resolución aceptable.

CLASIFICACIÓN SEGÚN EL TIPO DE CONEXIÓN

- Conexión a dos hilos. El sensor se conecta en serie con la carga, como si se tratara de un interruptor electromagnético. Esta conexión es habitual para los detectores de CA. Los sensores NAMUR siguen también una conexión de dos hilos, aunque como se ha dicho no son propiamente interruptores, sino que precisan de un circuito auxiliar externo.
- Conexión a tres hilos. Esta es la mas frecuente para los detectores de CC con salida por transistor. Se tiene un hilo común para alimentación y carga

y los otros dos son diferenciados uno para la alimentación y otro para la carga. El hilo común debe conectarse al terminal negativo de la alimentación para transistores PNP y al terminal positivo para los de tipo NPN.

- Conexión a cuatro o cinco hilos. Se suelen emplear para detectores de CC. –emplean dos hilos para la alimentación, y otros dos (o tres, en montaje conmutado) corresponden al contacto de salida para control de la carga.

CARACTERÍSTICAS DE SALIDA.

Como se ha dicho, los detectores de proximidad suelen tener salida estática a base de tiristores o transistores. Este tipo de conmutadores presentan siempre una caída de tensión residual en el estado cerrado y una corriente de fugas en el estado abierto. Esto implica que no pueden trabajar por debajo de una cierta tensión de alimentación y que requieren una mínima corriente de carga para asegurar una buena conmutación.

Desde el punto de vista de su aptitud para ser usados como elementos de mando en los autómatas, una excesiva corriente de fugas puede ocasionar problemas de interpretación de nivel alto de entrada cuando en realidad el interruptor esta desactivado. Por ello, los detectores con excesiva corriente de fugas no son aptos para accionar las entradas de los autómatas.

DETECTORES INDUCTIVOS

Este tipo de detectores sirven para detectar la proximidad de piezas metálicas en un rango de distancias que va desde 1mm a unos 30mm, con una posible resolución del orden de décimas de milímetro. La ejecución mecánica y eléctrica esta normalizada a nivel europeo por CENELEC (normas EN 50.032, EN 50.036, EN 50.037, EN 50.038)

Mecánicamente las mencionadas normas definen varios tipos:

- Forma A cilíndrica roscada con diámetros normalizados de M8, M12, M18 y

M30. existen, además, otros tipos sin rosca con tamaños de diámetro de 4 y 5 mm. A su vez, todos ellos pueden ser de tipo enrasable o no enrasable, dependiendo de si se puede o no enrasar el cabezal detector en metal.

- Forma C de paralelepípedo con cabezal orientable. Generalmente son utilizados para distancias grandes.

A nivel de bloques están formados por un circuito oscilador L-C con alta frecuencia de resonancia. La bobina esta construida sobre un núcleo de ferrita abierto en forma de "pot-core", de forma que el flujo se cierra en la parte frontal a través de la zona sensible. La presencia del metal dentro de dicha zona sensible altera la reluctancia del circuito magnético, atenúa el circuito oscilante y hace variar la amplitud de oscilación. La detección de dicha amplitud permite obtener una señal de salida todo-nada.

La distancia de detección esta definida según norma para una placa cuadrada de acero ST57 de 1mm de espesor y de dimensiones acordes al diámetro del cabezal sensible. Para otros tipos de metal otras dimensiones la distancia nominal de detección debe corregirse con un factor de valor entre 0,4 a 0,6 y 1.

La variación de amplitud de la oscilación, provocada por la presencia de metal frente al cabezal detector, puede utilizarse para obtener una señal analógica de posición. El detector de proximidad da entonces una señal que es proporcional a la distancia. Sin embargo, la medida es muy imprecisa, depende mucho del tipo de metal y de las condiciones ambientales.

El campo de aplicación más importante de los detectores inductivos es como interruptores final de carrera con algunas ventajas con respecto a los electromecánicos, tales como: ausencia de contacto con el objeto a detectar, robustez mecánica, resistencia a ambientes agresivos y altas temperaturas y bajo precio.

DETECTORES CAPACITIVOS

El principio de funcionamiento y las características son análogos a las descritas para los detectores inductivos, pero en este caso el elemento sensible es el condensador del circuito oscilante, formado por dos aros metálicos concéntricos situados en la cara sensible, y cuyo dieléctrico es el material de la zona sensible.

Este tipo de sensores permiten detectar materiales metálicos o no, pero su sensibilidad se ve muy afectada por el tipo de material y por el grado de humedad ambiental y del cuerpo a detectar. Por ello se utilizan exclusivamente como detectores todo-nada, con una repetibilidad bastante dependiente de las condiciones ambientales. Para paliar el problema de dependencia de la sensibilidad con el tipo de material, se suelen construir con un ajuste de sensibilidad que permite utilizarlos para la detección de algunos materiales entre otros (por ejemplo, aluminio entre cobre o latón). Las aplicaciones típicas son, sin embargo, la detección de materiales no metálicos como vidrio, cerámica, plástico, madera, aceite, agua, cartón, papel, etc.

En cuento a las formas de ejecución mecánica, tipos de alimentación y formas de conexión son idénticas a las de los detectores inductivos.

DETECTORES OPTICOS

Los detectores ópticos emplean fotocélulas como elementos de detección. Algunos tipos disponen de un cabezal que incorpora un emisor de luz y la fotocélula de detección, actuando por reflexión y detección del haz de luz reflejado sobre el objeto que se pretende detectar. Otros tipos trabajan a modo de barrera y están previstos para detección a mayores distancias con fuentes luminosas independientes del cabezal detector. Ambos tipos suelen trabajar con frecuencias luminosas en la gama de los infrarrojos.

DETECTORES ULTRASONICOS

Estos detectores están basados en la emisión- recepción de ondas ultrasónicas.

Cuando un objeto interrumpe el haz, el nivel de recepción varia y el receptor lo detecta.

Como ventaja frente a las fotocélulas, los detectores ultrasónicos pueden detectar con facilidad objetos transparentes, como cristal y plásticos, materiales que ofrecen dificultades para la detección óptica.

Sin embargo, y dado que estos detectores utilizan ondas ultrasónicas que se mueven por el aire, no podrán ser utilizados en lugares donde este circulen con violencia (bocas de aire acondicionamiento, cercanías de puertas, etc.), o en medios de elevada contaminación acústica (prensas, choques entre metales, etc.).

Por lo demás, y en cuanto a características de funcionamiento, estos detectores son semejantes a las células fotoeléctricas.

INTRODUCCION A LOS SISTEMAS DE MEDIDA

- **SISTEMA DE MEDIDA**

Se denomina sistema a la combinación de dos o más elementos, subconjuntos y partes necesarias para realizar una o varias funciones.

Los objetivos de la medida pueden ser: la vigilancia o seguimiento de procesos.

Las medidas de prototipos son además necesarias para verificar los resultados de los modelos desarrollados en un ordenador.

En la siguiente figura se describe la estructura general de un sistema de medida y control.

En un sentido amplio, la realización de una medida implica, pues, además de la adquisición de la información, realizada por un elemento sensor o transductor,

también el procesamiento de dicha información y la presentación de resultados, de forma que puedan ser percibidos por nuestros sentidos.

- **TRANSDUCTORES, SENSORES Y ACCIONAMIENTOS**

Se denomina transductor, en general, a todo dispositivo que convierte una señal de una forma física en una señal correspondiente pero de otra forma física distinta. Es por tanto, un dispositivo que convierte un tipo de energía en otro.

Dado que hay seis tipos de señales: mecánicas, térmicas, magnéticas, eléctricas, ópticas y moleculares (químicas), cualquier dispositivo que convierta una señal de un tipo en una señal de otro tipo debería considerarse un transductor, y la señal de salida podría ser de cualquier forma física "útil". En la práctica, no obstante, se consideran transductores por antonomasia aquellos que ofrecen una señal de salida eléctrica.

Un sensor es un dispositivo que, a partir de la energía del medio donde se mide, da una señal de salida transducible que es función de la variable medida.

Sensor y transductor se emplean a veces como sinónimos, pero sensor sugiere un significado más extenso: la ampliación de los sentidos para adquirir un conocimiento de cantidades físicas que, por su naturaleza o tamaño, no pueden ser percibidas directamente por los sentidos. Transductor, en cambio, sugiere que la señal de entra y la de salida no deben ser homogéneas.

- **ACONDICIONAMIENTO Y PRESENTACIÓN**

Los acondicionadores de señal, adaptadores o amplificadores, en sendo amplio, son los elementos del sistema de medida que ofrecen, a partir de la señal de salida de un sensor electrónico, una señal apta para ser presentada o registrada o que simplemente permita un procesamiento posterior mediante un equipo o

instrumento estándar. Consiste normalmente en circuitos electrónicos que ofrecen, entre otras funciones, las siguientes:
Amplificación, filtrado, adaptación de impedancias y modulación o desmodulación

Se considera, por ejemplo, el caso en que una de las etapas de tratamiento de la señal de medida es digital, si la salida del sensor es analógica, que es lo más frecuente, hará falta un convertidor A/D.

DETECTORES DE HUMO

- **SERIE 400**

CARACTERISTICAS DEL PRODUCTO

- 12 o 24 voltios para operar.
- Cámara óptica sensitiva única.
- Cubierta removible para fácil limpieza de insectos.
- Indicador intermitente para señalar buen funcionamiento, y fijo en alarma activada.
- Sellado contra polvo, insectos o bajas presiones.
- Campo medidor sensitivo del detector para cumplir los 72 requerimientos de la NFPA.
- 3 años de garantía.
- Sensor opcional para interiores de 135° F (57° C).

Este sistema de sensor serie 400 foto electrónico es específicamente diseñado para cumplir los estrictos requerimientos de la industria y comercio de los detectores de humo y fuego. El diseño de estos detectores facilita la instalación y mantenimiento de ellos mismos. La serie 400 ofrece 6 diferentes detectores foto electrónicos con una variedad de voltajes, configuración de conectividad o alambrado y opciones de terminales.

ESPECIFICACIONES DE INGENIERIA:

El detector de humo debe ser un tipo foto electrónico (modelo 2400, 2412, 2424) o una combinación térmica foto electrónica (modelo 2400TH, 2412TH, 2424TH) con sensores térmicos que oscilan en 135° F de su manufacturación. Las conexiones de alambrado debe ser hecho por tornillos SEMS. El detector tendrá un LED visible que parpadeará en el correcto funcionamiento y cerrara cuando se active la alarma. El detector deberá tener una sensibilidad nominal del 3% por pie según la capacidad que muestre en las indicaciones de la caja contenedora. La pantalla del detector y su cobertor son fácilmente removibles para su limpieza. Debe ser posible ejecutar una prueba sensitiva y funcional sin la necesidad de generar humo. Los circuitos del detector pueden ejecutar una prueba de la cámara sensitiva y los circuitos electrónicos internos cada 40 segundos. Si el sistema de circuitos falla, el LED detector parará de dar señal.

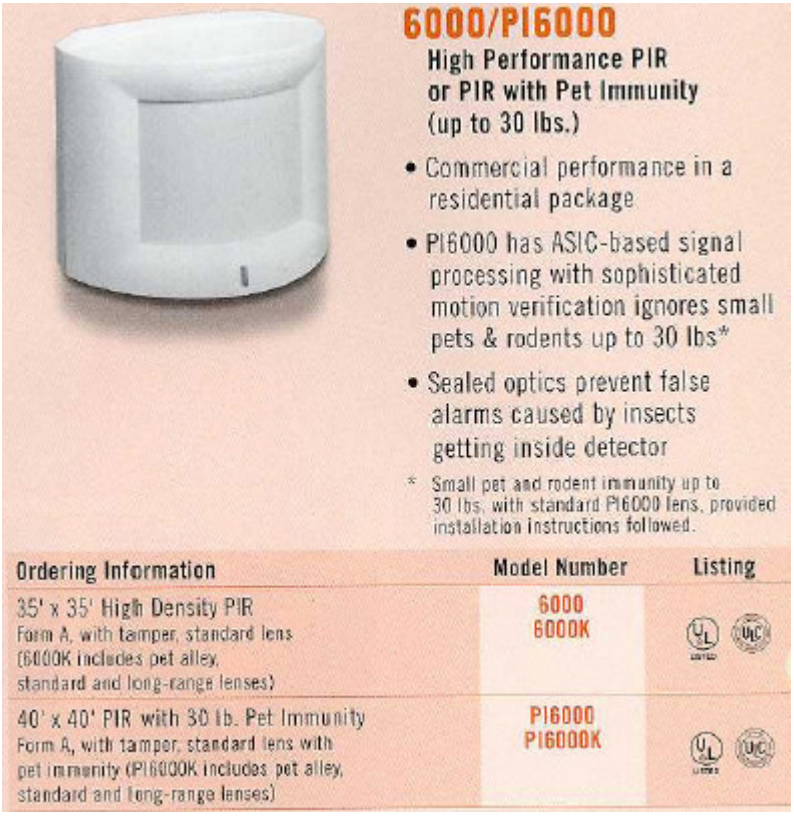
ESPECIFICACIONES

- Tamaño: 3.2” (8.1 cm.) altura, 5.5” (13.9 cm.) diámetro, con el soporte montado, añadir .5” (1.27 cm.) de altura por térmico.
- Peso de embarque: 0.5 lb. (227 g)
- Rango de Temperatura de Operación: Modo térmico 32° F a 100° F (0° a 38° C), Otros modelos 32° F a 120° F (0° a 49° C)
- Rango de Humedad de operación: 10 – 93 % RH no condensable.
- Velocidad del aire: 3000 FPM.
- Variación del sensor térmico: 135° F

Espacio que abarca el detector de humo: En encielados suaves, espacios de 30 pies debe ser usado como un guía. Otros espacios deben usar una altura independiente del encielado.

Sensibilidad: 3.0% ± 7 pies nominales.

DECTECTORES DE MOVIMIENTO



- **INFRARROJO PASIVO SERIE 6000**

  - Elige entre los nuevos lentes de alta densidad (HD) modelos 6000 o los PI6000, cuando existen pequeñas mascotas de hasta 30 libras e inmunidad de roedores.
  - Posee procesos basados en señales ASIC, analiza tamaños, velocidad y formas de señales entrantes.
  - Posee ópticas selladas, que previenen las falsas alarmas causada por insectos que se introducen al detector.
  - En un producto industrial en un paquete residencial.

Las series 6000 ahora incluyen los PI6000, que están diseñadas para satisfacer la demanda de inmunidad animal en establecimientos residenciales. Si el usuario

posee pequeñas mascotas o problemas de roedores, ya no se tienen que utilizar PIR's de 3-4 pies y cambiar a lentes Alley. Sino, simplemente con el uso del PI6000 a una altura estándar de 7-8 pies.

Estos únicos diseños ópticos de los PI6000, optimizan la cantidad de energía infrarroja, pasando a través de los lentes del sensor Piro-eléctrico. Estas señales vuelan a través de series de portales en el acostumbrado ASIC, donde el PIR analiza los tamaños, velocidad y forma de los signos eléctricos, y este podrá determinar pequeñas mascotas o roedores de los intrusos.

Los diseños PI6000 están optimizados para la inmunidad animal, mientras los 6000 incorporan un "alcance fácil", con diseño de lentes con alta densidad. Ambos, ofrecen una fácil instalación para residencias, ópticas selladas, un lente opaco de Fresnel y entre 2 a 3 formas de aplicación de pulsos. Las series 6000 proveen una falsa alarma de inmunidad, típicamente vista mas que todo en PIR's comerciales de alto costo.
Los chips de PI6000 poseen un lente de inmunidad animal 40', mientras que los chips de serie 6000 poseen un lente denso de 35'.

Sentrol 6000 Series

Specifications:

Electrical

Voltage	10–16 V DC
Current	14 mA Typical 20 mA max.
Maximum loop rating	16 V DC, 50 mA
Alarm output	Fail safe contacts–closed loop
Alarm duration	3.2 seconds (±0.5 sec)
Cover Tamper Contacts	Closed loop
Rating	16V, 50 mA

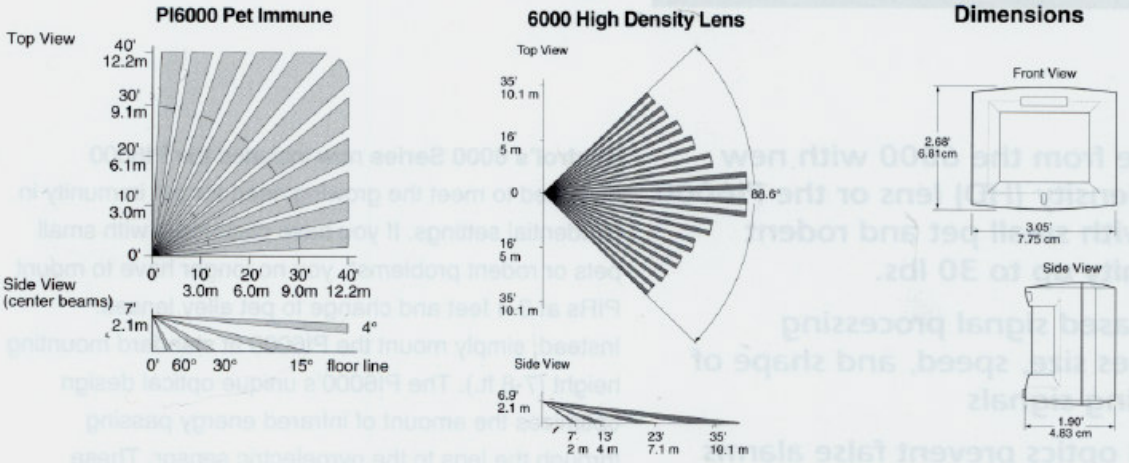
Environmental

Operating temperatures	14°F to 122°F (–10°C to +50°C)
Humidity	5%–95% noncondensing
RFI immunity	Greater than 10 V/meter from 10 to 1000MHz
Static immunity	20 kV
Lightning immunity	2.4 kV, 1.2 joules max. energy impulse, 100 µsec duration on field wiring

Features

Pulse count	2 pulse or 3 pulse
Range	
PI6000	40' x 90°
6000 High Density	35' x 90°
Mounting	7'– 9' (wall or corner)
Dimensions	2.68" (6.81 cm) H 3.05" (7.75 cm) W 1.90" (4.83 cm) D
Color	White

Lens coverage



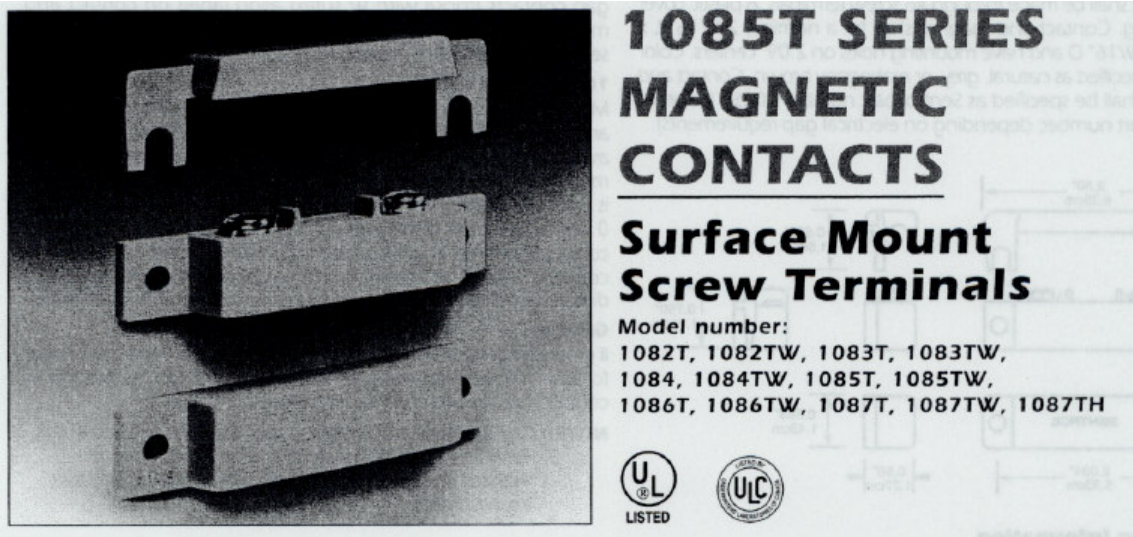
Ordering Information

Model Number	Description
PI6000	40' x 40' PIR with 30 lb Pet Immunity Form A, with tamper, pet immune lens
6000	35' x 35' High Density PIR Form A, with tamper, high density lens

*Pet alley lens option available for pets exceeding 30 lbs.

SENSORES MAGNETICOS

- SERIE 1085T

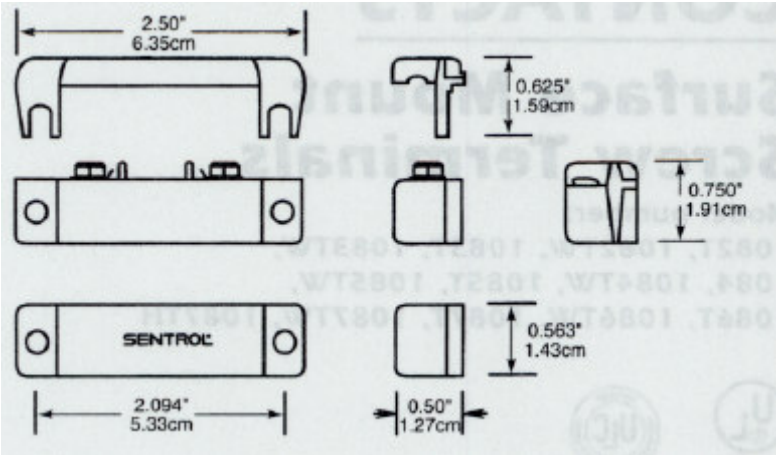


APLICACIONES:

- Fácil instalación de terminales.
- Conveniente superficie para el montaje.
- Disponibilidad de resistores residenciales y para industriales.
- Envolturas, espaciadote y tornillos incluidos, para su instalación.

ESPECIFICACIONES GENERALES

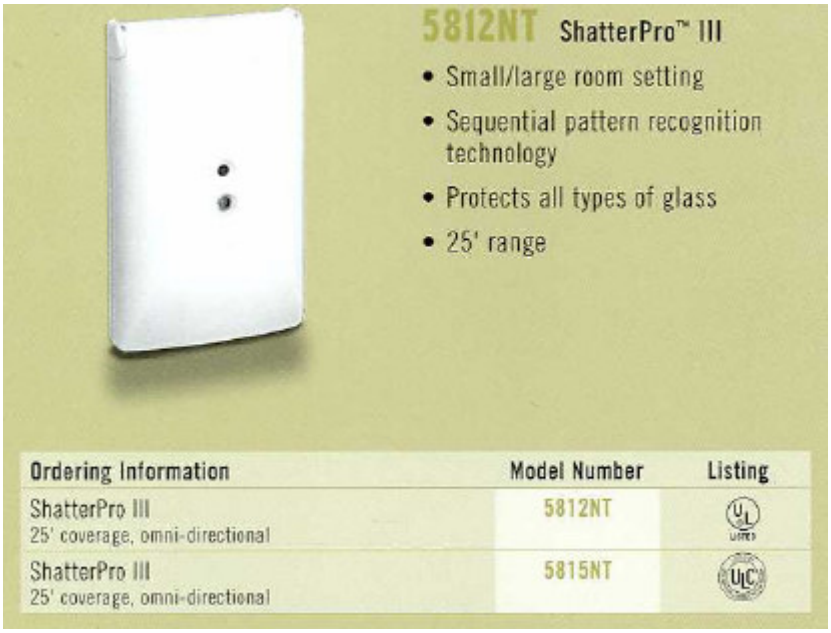
Rango de temperatura: -40° F a 150° F (-40° C a 65° C)
Ambiente: Switch herméticamente cerrado.
Promedio NEMA: 1
Categoría de Protección: IP 62
Tiempo de respuesta: 1 mseg. Máximo
Ciclos de vida: 10,000 bajo Carga total, 10,000 bajo circuito seco
Conexión: Terminal de tornillo #6
Opciones de color: Natural, Mahogany, Gris
UL/ULC listados: todos los modelos



Otra información		Especificaciones eléctricas				
Número de parte	**Configuración del contacto**	**Promedio Descarga (AC/DC)**	**Voltajes de intercambio (AC/DC)**	**Intercambio reciente (AC/DC)**	**Resistencia del contacto**	**Rango nominal de sensibilidad**
1085T-G, M, N	N.O.	7.5W/VA	100V	0.5A	0.2 Ohms	0.8(1.9cm)
1085TW-G, M, N	N.O.	7.5W/VA	100V	0.5A	0.2 Ohms	1.5(3.8cm)
1084TW-N	SPDT	3W/VA	30V	0.25A	0.2 Ohms	2.0(5.1cm)
1086T-N	N.C.	3W/VA	30V	0.25A	0.2 Ohms	0.8(1.9cm)
1087T-M, N	SPDT	3W/VA	30V	0.25A	0.2 Ohms	0.8(1.9cm)
1087TW-M, N	SPDT	3W/VA	30V	0.25A	0.2 Ohms	1.5(3.8cm)
1080T-N	Actuador solo para 1082T, 1083T, 1084T, 1082TW, 1083TW, 1084TW					
1081T-N	Actuador solo para 1085T, 1086T, 1087T, 1085TW, 1086TW, 1087TW					

CATALOGO DE SENSORES RESIDENCIALES QUE OFRECE GENERAL ELECTRIC

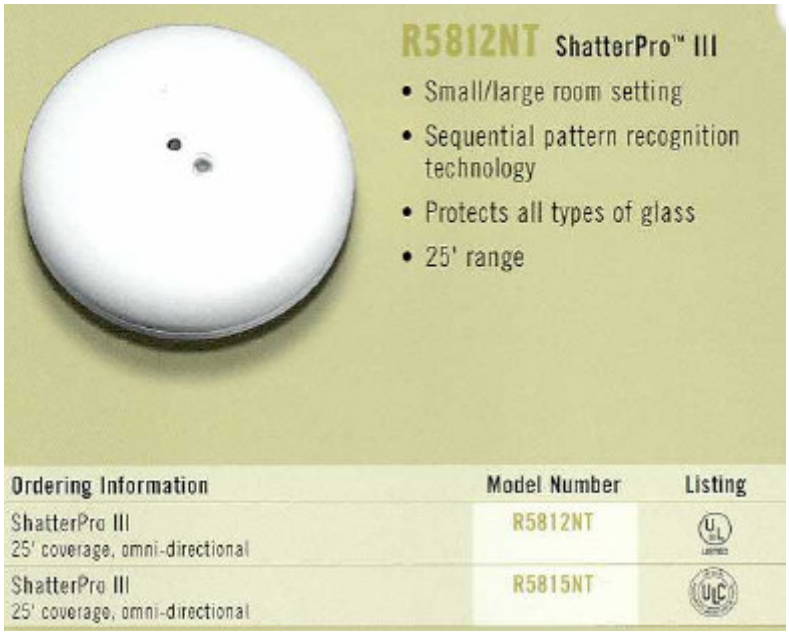
- *SENSORES DE ROPTURA DE VIDRIOS O VENTANAS*



5812NT ShatterPro™ III

- Small/large room setting
- Sequential pattern recognition technology
- Protects all types of glass
- 25' range

Ordering Information	Model Number	Listing
ShatterPro III 25' coverage, omni-directional	5812NT	
ShatterPro III 25' coverage, omni-directional	5815NT	



R5812NT ShatterPro™ III

- Small/large room setting
- Sequential pattern recognition technology
- Protects all types of glass
- 25' range

Ordering Information	Model Number	Listing
ShatterPro III 25' coverage, omni-directional	R5812NT	
ShatterPro III 25' coverage, omni-directional	R5815NT	

5600/5620
ShatterPoint™ Residential

"Tru-Dual" Transducer Sensor

- 10' (3.0m) range on fixed pane, 8' (2.4m) multiple pane, measured to farthest point of glass
- Protects all types of glass
- Minimum glass size 1' by 1'

Ordering Information	Model Number
ShatterPoint Residential	5600/W
ShatterPoint Residential With Form C relay and tamper switch	5605/W
ShatterPoint Residential With magnetic contact	5620/W
ShatterPoint Residential With Form C relay, tamper switch, and magnetic contact	5625/W
ShatterPoint Residential Wireless 3V tamper switch and magnetic contact	564503/W

5650/5655C
ShatterPoint™ Commercial

"Tru-Dual" Transducer Sensor

- 10' (3.0m) range on fixed pane, 8' (2.4m) multiple pane, measured to farthest point of glass
- Protects all types of glass
- Minimum glass size 2' by 3'

Ordering Information	Model Number
ShatterPoint Commercial With Form A relay and tamper switch	5650C/W
ShatterPoint Commercial With Form C relay and tamper switch	5655C/W

GS610 On-the-Glass or Frame Sensor and Analyzer

- Can be installed on walls or roofs for special perimeter detection
- Designed for use with GS614/615/617 analyzer module for sensitivity and pulse count selection
- Up to 20 sensors can be connected to each analyzer

Ordering Information	Model Number
Inertia Shock Sensor With 2M jacketed cable	GS600/W
Inertia Shock Sensor	GS610/W GS610/B

GS611 On-the-Glass or Frame Sensor and Analyzer

- Can be installed on walls or roofs for special perimeter detection
- Designed for use with GS614/615/617 analyzer module for sensitivity and pulse count selection
- Up to 20 sensors can be connected to each analyzer

Ordering Information	Model Number
Inertia Shock Sensor With reed switch	GS611/W GS611/B

5885 ShatterPro™ Plus Motion-Sensitive Acoustic Glassbreak Sensor

- Can function as two sensors in one package (PIR/Glassbreak)
- 24 hour acoustic glassbreak application

Ordering Information	Model Number
ShatterPro Plus Motion-sensitive acoustic glassbreak sensor	5885/W

5820 Recessed ShatterPro™ II

- Recess mounted directly in any ceiling or wall
- 360° coverage pattern
- 25' (7.5m), 360° range measured to the farthest point on all types of glass
- Protects all glass types up to a standard thickness of 1/4" (0.64cm) including plate, wired, tempered and laminated glass

Ordering Information	Model Number	Listing
Recessed ShatterPro II 1" (2.5cm) recessed mounting, latch or non-latch LED	5820A/W	
Recessed ShatterPro II with Tamper 1" (2.5cm) recessed mounting, latch or non-latch LED	5822A/W	UL
ShatterPro II 5820A with Single Gang Box Kit	5825A/W	

5150 Shock Sensor

On-the-Glass Protection

- 10' (3.0m) protection on a single pane of glass
- Detects all glass types without sensitivity adjustment
- On-the-glass mounting is visible from the outside to deter potential intruders
- Field proven with over 3 million installations around the world

Ordering Information	Model Number	Listing
Shock Sensor 10' (3.0m) range, two-wire, adhesive mount	5150/M 5150/W	
Shock Sensor with Coil Cord 10' (3.0m) range, two-wire, 3' (0.91m) coil cord for doors and windows	5150C/M 5150C/W	
Shock Sensor with LED Corner mount with range verification, two-wire, set-up LED, and sensitivity adjustment	5125/M 5125/N	UL

5400 Glassbreak Shock Sensor

- Protects 10' (3.0m) of glass
- Protects all glass types up to a standard thickness of 1/4" (0.64cm) including plate, wired, tempered and laminated glass
- Protects multi-pane windows
- Set-up LED

Ordering Information	Model Number	Listing
Glassbreak Shock Sensor Two-wire, closed loop, lithium battery powered (included)	5414/M 5414/W	UL
Glassbreak Shock Sensor Two-wire, closed loop, lithium battery powered (included), reed switch and magnet	5415A/M 5415A/W	UL
Glassbreak Shock Sensor Four-wire, closed loop, latching LED option	5425/M 5425/W	

GS614/615

- Up to 20 sensors can be connected to each analyzer
- Adjustable sensitivity

Ordering Information	Model Number
Single Zone Analyzer	GS614/W
Single Zone Analyzer With terminal reset	GS615/W

GS617

- Up to 20 sensors can be connected to each analyzer
- Adjustable sensitivity

Ordering Information	Model Number
Four Zone Analyzer With EOL resistor and manual reset	GS617/W

5845 Wireless ShatterPro™

- 20' (6.0m), 360° range to farthest point on all glass
- Protects all glass types
- Compatible with virtually all wireless transmitters

Ordering Information	Model Number
Wireless ShatterPro Compatible with 3-volt transmitters, mounting bracket included	584503/W
Wireless ShatterPro Compatible with 9-volt transmitters, mounting bracket included	584509/W

5402/5422 Metal Enclosure Assault Sensor

- Protects sheet metal enclosures up to 2' x 4' x 1' (61cm x 122cm x 30.5cm)
- Replaces costly lined boxes
- Detects drilling, sawing and other assaults to metal enclosures

Ordering Information	Model Number	Listing
Metal Enclosure Assault Sensor Two-wire, lithium battery powered (included)	5402/W	
Metal Enclosure Assault Sensor Four-wire, external powered	5422/W	

DV1201/1221

Advisor X Vibration Sensor

- Detects structural vibration generated in walls, floors, ceilings and doors
- Sensitivity adjustment discriminates between real attempts and nuisance alarms
- Helps prevent intrusion via hammers, drills, pressure and thermal tools, and explosives

Ordering Information	Model Number	Listing
Structural Vibration Sensor For vaults and safes, mounting plate included	DV1201	
Structural Vibration Sensor For ATMs and night deposit safes, mounting plate included	DV1221	
Self-Contained Test Transmitter Testing Accessory	DV1215	
Test System Panel Eight-Point Remote Annunciator/Test Accessory	DV1208A	

ACCESSORIES

5709C/W
Handheld tester with real glass break sounds

1838/N
Replacement Magnet

5828A/W

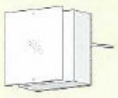
5829A/W

Ordering Information	Model Number
Hand-held Tester For testing effective range	5709C/W0
Replacement Magnet	1838/N

Ordering Information	Model Number
Trim Plate	5828A/W
Single Gang Box Plate	5829A/W

SAFE & VAULT ACCESSORIES

DV1212
Steel Mounting Plate: For mounting the detector on a steel surface by welding



DV1203
Recess Mounting Kit. For the detector. Used for vaults under construction to create a recess mounting cavity in concrete.

DV1204
Weather Protection Box. Mounting box to protect the detector in severe environments.

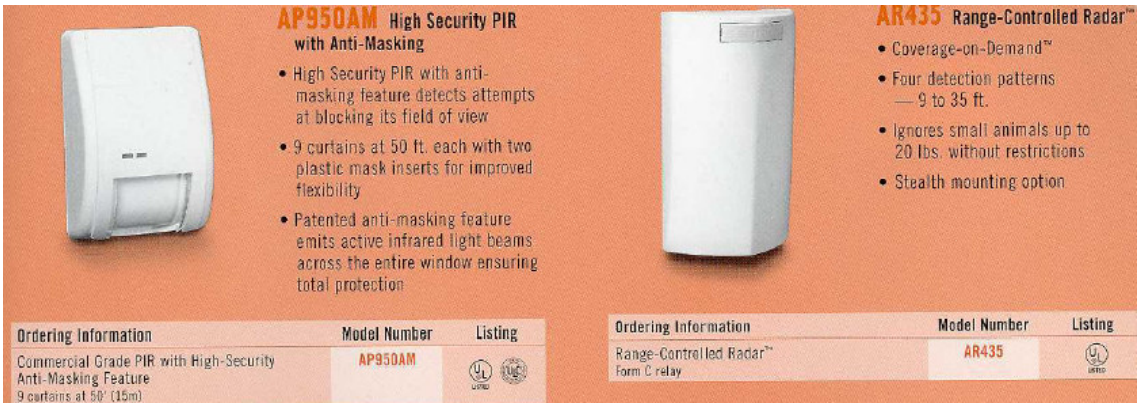
WS300/302
High Security Cable

DV1230
Safe Application Test Sensor

DV1160
DV Seismic Detector

Additional Accessories Available:
Plastic Cable Inserts (DV1216, DV1217, DV1218)
Metal Junction Box (DV1228)
Floor Mounting Box (DV1220)

- ***SENSORES DETECTORES DE ESPACIO O DETECTORES DE MOVIMIENTO***



AP950AM High Security PIR with Anti-Masking

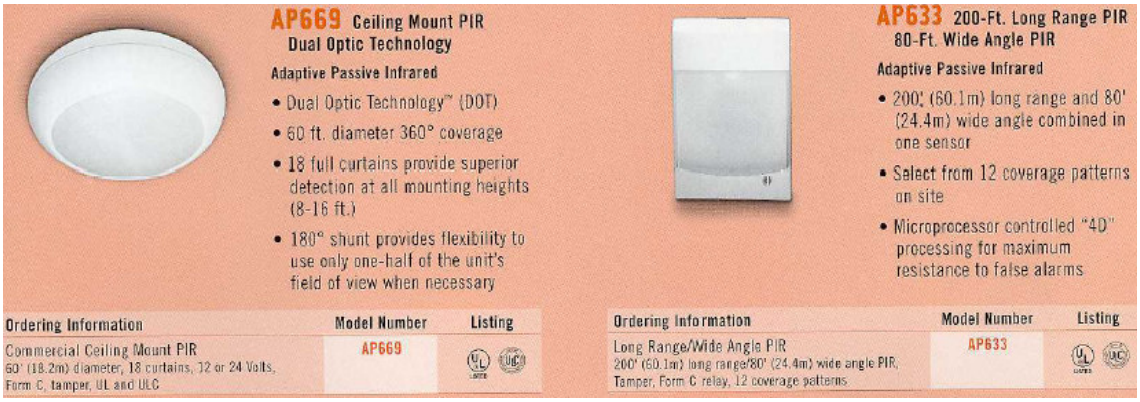
- High Security PIR with anti-masking feature detects attempts at blocking its field of view
- 9 curtains at 50 ft. each with two plastic mask inserts for improved flexibility
- Patented anti-masking feature emits active infrared light beams across the entire window ensuring total protection

Ordering Information	Model Number	Listing
Commercial Grade PIR with High-Security Anti-Masking Feature 9 curtains at 50' (15m)	AP950AM	

AR435 Range-Controlled Radar™

- Coverage-on-Demand™
- Four detection patterns — 9 to 35 ft.
- Ignores small animals up to 20 lbs. without restrictions
- Stealth mounting option

Ordering Information	Model Number	Listing
Range-Controlled Radar™ Form C relay	AR435	



AP669 Ceiling Mount PIR
Dual Optic Technology

Adaptive Passive Infrared

- Dual Optic Technology™ (DOT)
- 60 ft. diameter 360° coverage
- 18 full curtains provide superior detection at all mounting heights (8-16 ft.)
- 180° shunt provides flexibility to use only one-half of the unit's field of view when necessary

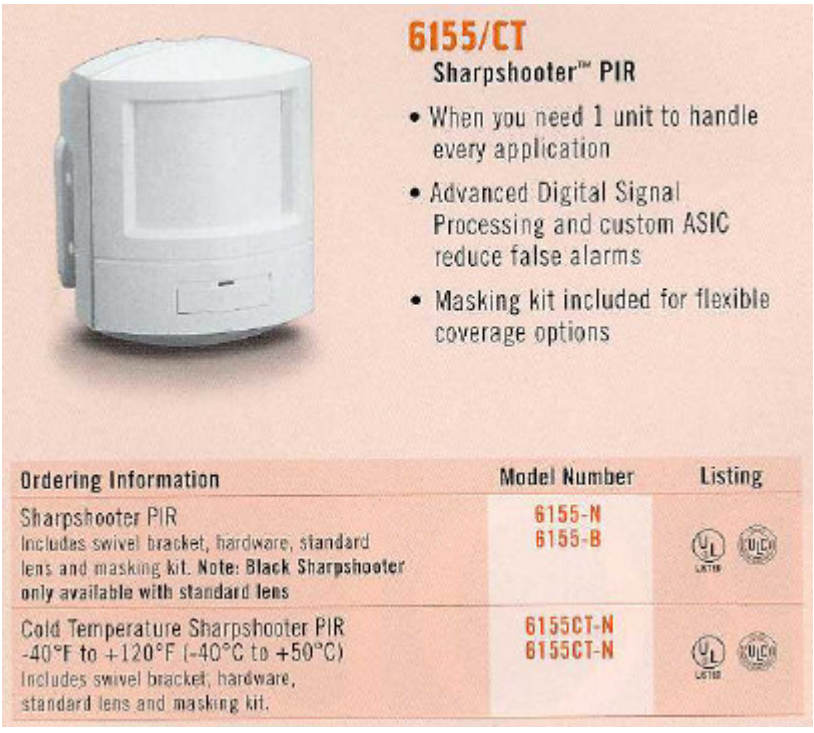
Ordering Information	Model Number	Listing
Commercial Ceiling Mount PIR 60' (18.2m) diameter, 18 curtains, 12 or 24 Volts, Form C, tamper, UL and ULC	AP669	

AP633 200-Ft. Long Range PIR
80-Ft. Wide Angle PIR

Adaptive Passive Infrared

- 200' (60.1m) long range and 80' (24.4m) wide angle combined in one sensor
- Select from 12 coverage patterns on site
- Microprocessor controlled "4D" processing for maximum resistance to false alarms

Ordering Information	Model Number	Listing
Long Range/Wide Angle PIR 200' (60.1m) long range/80' (24.4m) wide angle PIR, Tamper, Form C relay, 12 coverage patterns	AP633	



6155/CT
Sharpshooter™ PIR

- When you need 1 unit to handle every application
- Advanced Digital Signal Processing and custom ASIC reduce false alarms
- Masking kit included for flexible coverage options

Ordering Information	Model Number	Listing
Sharpshooter PIR Includes swivel bracket, hardware, standard lens and masking kit. **Note: Black Sharpshooter only available with standard lens**	6155-N 6155-B	
Cold Temperature Sharpshooter PIR -40°F to +120°F (-40°C to +50°C) Includes swivel bracket, hardware, standard lens and masking kit.	6155CT-N 6155CT-N	

6000/PI6000

High Performance PIR or PIR with Pet Immunity (up to 30 lbs.)

- Commercial performance in a residential package
- PI6000 has ASIC-based signal processing with sophisticated motion verification ignores small pets & rodents up to 30 lbs*
- Sealed optics prevent false alarms caused by insects getting inside detector

* Small pet and rodent immunity up to 30 lbs. with standard PI6000 lens, provided installation instructions followed.

Ordering Information	Model Number	Listing
35' x 35' High Density PIR Form A, with tamper, standard lens (6000K includes pet alley, standard and long-range lenses)	6000 6000K	
40' x 40' PIR with 30 lb. Pet Immunity Form A, with tamper, standard lens with pet immunity (PI6000K includes pet alley, standard and long-range lenses)	PI6000 PI6000K	

6187CTX Industrial/Outdoor Sharpshooter™

- Indoor or outdoor applications
  - Ultraviolet-stabilized protective lens cover
  - Water resistant, dust proof
  - -40°F (-40°C) to +120°F (50°C) operating temperature
- Die-cast aluminum housing with gasket for rugged installations

Ordering Information	Model Number	Listing
Sharpshooter PIR Industrial/Outdoor – Form C Includes hardware, three lenses and masking kit for wireless or special configuration, call factory	6187CTX-N	

6197/6198

Intrinsically Safe PIR System and Non-Incentive PIR

- UL and FM-approved PIR system
- Cast metal PIR, both water resistant and dust proof
- Advanced signal processing for superior intruder detection

Ordering Information	Model Number	Listing
Sharpshooter PIR Intrinsically Safe System Approved for Class I, II, III, Division 1, Groups A, B, C, D, E, F, G	6197-N	
Sharpshooter PIR Non-Incentive Approved for Class I, Division 2, Groups A, B, C, D	6198-N	

AP750 True 3D Curtain Coverage PIR

Adaptive Passive Infrared

- AP750 provides four separate curtain coverage patterns with 50' (15.2m) range
- Mounting height 6'-10' (1.8-3.0m) with AP750 model
- AP475 provides a unique, single curtain coverage pattern that extends to 75' (190.5m)—mounts up to 16 ft.

Ordering Information	Model Number	Listing
Commercial Grade PIR with Range Control 7 curtains at 50' (15m). Form C relay with tamper	AP750	
Commercial Grade PIR Single curtain at 75' (23.8m), mounting height up to 16' (4.8m). Form C relay and tamper	AP475	

RTE1000 Request-to-Exit PIR

- Wall, door frame, or ceiling mount
- Built-in sounder for door position monitoring
- Dual relay outputs
- Adjustable range

Ordering Information	Model Number	Listing
Request to exit PIR with built-in sounder 12 or 24 Volt, dual relay output	RTE1000	
Request to exit PIR with built-in sounder 12 or 24 Volt, dual relay output	RTE1000B	

6255/FM

SureShot™ PIR Flush Mount

- Ideal for prewire and flush mount applications
- The only PIR with all these coverage patterns built in:
  - 360° ceiling mount
  - Wide angle
  - Pet alley
  - Vertical barrier
  - Single spot zone

Ordering Information	Model Number	Listing
SureShot PIR Includes masking templates to create 5 coverage patterns	6255-W	
SureShot Flush Mount PIR Includes masking templates to create 5 coverage patterns	6255FM-W	

Ordering Key

Color	Product # Suffix
White	W
Brown	M
Gray	G

ACCESSORIES

AP669RK
Recessed
Mounting Kit

Ordering Information	Model Number
AP669 Recessed Mounting Kit for semi-recessed mounting in ceiling tiles	AP669RK
AP669 Base Plate	AP669BP

BR601
Ceiling/Wall
Mount Swivel
Bracket

SB01
Ceiling/Wall
Mount Swivel
Bracket

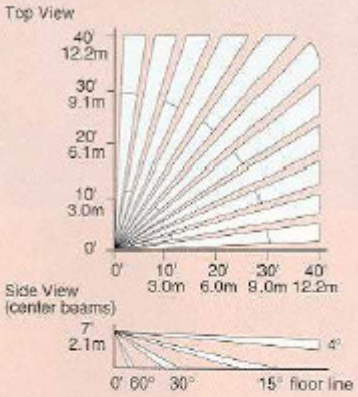
Ordering Information	Model Number
Ceiling/Wall Mount Swivel Bracket For use with AP633/643	BR601
Ceiling/Wall Mount Swivel Bracket For use with AP750, AP475 and AP950AM	SB01

6083-N
Swivel Bracket
Assembly

Ordering Information	Model Number
Swivel Bracket Assembly for 6180/90 Series Includes hardware for mounting to housing (hardware for mounting to surface not included)	6083-N

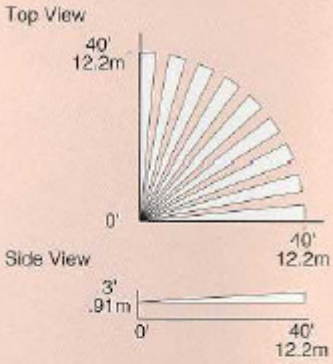
PI6000STD

40' Standard Lenses



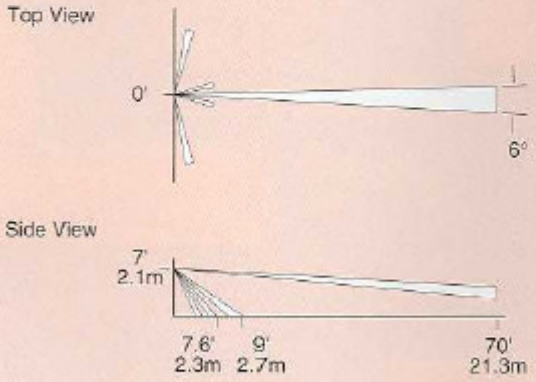
6000PET

40' Pet Alley Lenses



6000LR

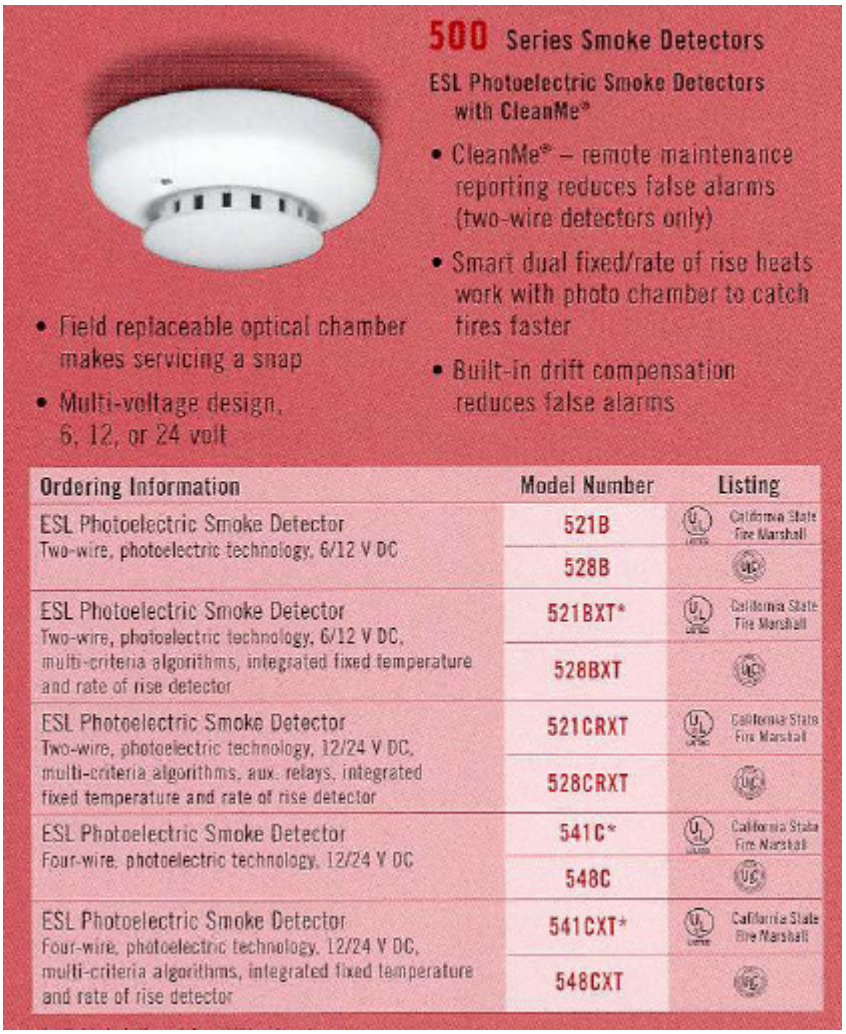
70' Long-Range Lenses



Ordering Information	Model Number
40' Standard Lenses, 5 pack for PI6000	PI6000STD
35' Standard Lenses, 5 pack for 6000	6000STD
40' Pet Alley Lenses, 5 pack for PI6000 & 6000*	6000PET
70' Long-Range Lenses, 5 pack for PI6000 & 6000	5820A-W

*Pet alley lens option available for pets exceeding 30 lbs.

- ***DETECTORES DE HUMO Y SEGURIDAD***



500 Series Smoke Detectors

ESL Photoelectric Smoke Detectors with CleanMe®

- CleanMe® – remote maintenance reporting reduces false alarms (two-wire detectors only)
- Smart dual fixed/rate of rise heats work with photo chamber to catch fires faster
- Built-in drift compensation reduces false alarms
- Field replaceable optical chamber makes servicing a snap
- Multi-voltage design, 6, 12, or 24 volt

Ordering Information	Model Number	Listing
ESL Photoelectric Smoke Detector Two-wire, photoelectric technology, 6/12 V DC	521B	California State Fire Marshall
	528B	
ESL Photoelectric Smoke Detector Two-wire, photoelectric technology, 6/12 V DC, multi-criteria algorithms, integrated fixed temperature and rate of rise detector	521BXT*	California State Fire Marshall
	528BXT	
ESL Photoelectric Smoke Detector Two-wire, photoelectric technology, 12/24 V DC, multi-criteria algorithms, aux. relays, integrated fixed temperature and rate of rise detector	521CRXT	California State Fire Marshall
	528CRXT	
ESL Photoelectric Smoke Detector Four-wire, photoelectric technology, 12/24 V DC	541C*	California State Fire Marshall
	548C	
ESL Photoelectric Smoke Detector Four-wire, photoelectric technology, 12/24 V DC, multi-criteria algorithms, integrated fixed temperature and rate of rise detector	541CXT*	California State Fire Marshall
	548CXT	

505 CleanMe® Interpreter and 2 to 4-Wire Input Converter Module

- Interprets 521 CleanMe® signal for 12/24V DC control panel
- Allows central station to be notified when detector needs to be cleaned.
- Converts any ESL 2-wire smoke detector to 4-wire
- 3 diagnostic LEDs

* CleanMe® Compatible control panels that do not require use of the 505 module:

ESL 2500 24V FACP

Moose ZX300/ZX400/ZX440F 12V Burg/Fire Panel

Ordering Information	Model Number	Listing
505 Module Module that interprets the 521's CleanMe® signal and can convert 2-wire 521 Series detectors to a 4-wire input	505	California State Fire Marshall

240-CO SafeAir™ Carbon Monoxide Alarm

- Reliable industrial-grade electrochemical technology
- Low 8mA current draw eliminates need for additional power supplies
- Attractive design and low cost increase add-on sales
- Listed UL 2075 for commercial or residential applications

Ordering Information	Model Number	Listing
ESL SafeAir™ Carbon Monoxide Alarm 12V DC, electrochemical system carbon monoxide alarm	240-CO	

2500 Modular 1-25 Zone Fire Alarm Control Panel & Power Booster (with CleanMe®)

- Expandable modular design reduces equipment costs
- CleanMe® remote maintenance reporting reduces false alarms
- Up to 7A of usable power saves cost of power booster
- Panel Listings: Local, Remote station, Central station, Auxiliary, Proprietary, Proprietary receiving unit, Household fire warning

Ordering Information	Model Number	Listing
Single Zone Panel; 4.5A power supply	2501	
2-Zone, Class B, expander	2501-ZEM	
4-Zone, Class B, expander	2504-ZEM	
Upload/download digital fire dialer, dual line. For use with 1500 Series and 2500 Series fire control panels	2500-DAC*	
2 Additional bell circuits, 3.5A expander	2500-BELL	
2-Zone, relay module, expander	2500-ZRM2	
4-Zone, relay module, expander	2500-ZRM4	
LCD keypad/remote annunciator and 2500 D AC programmer	2500-KPD	
Single zone panel, replacement circuit board	2500-BMB	
System housing with nameplates	2500-SH	
8 Amp power supply, for replacement	2500-PS	
Local energy module	2500-LRM/LEM	
LED remote annunciator	2500-RA	
LED remote annunciator and LED driver (Driver not supervised)	2500-RADVR	
Printer interface module	ZX-PTR	

DH Series

Electromagnetic Door Holders

- Surface, flush, floor and recess mounted models
- Swivel catch plate and extension rods
- Extremely low current consumption
- Dual voltage AC/DC

Ordering Information	Model Number	
	Chrome	Brass
12 or 24 V DC/AC, recess mount, with 3" extension rod	DHR-1224C	DHR-1224B
24 or 120 V DC/AC, recess mount, with 3" extension rod	DHR-24120C	DHR-24120B
24 or 220 V DC/AC, recess mount, with 3" extension rod	DHR-24220C	DHR-24220B
12 or 24 V DC/AC, semi-flush mount	DHF-1224C	DHF-1224B
24 or 120 V DC/AC, semi-flush mount	DHF-24120C	DHF-24120B
24 or 220 V DC/AC, semi-flush mount	DHF-24220C	DHF-24220B
12 or 24 V DC/AC, surface mount	DHS-1224C	DHS-1224B
24 or 120 V DC/AC, surface mount	DHS-24120C	DHS-24120B
24 or 220 V DC/AC, surface mount	DHS-24220C	DHS-24220B
12 or 24 V DC/AC, floor mount, single coil	DHFM1-1224C	
24 or 120 V DC/AC, floor mount, single coil	DHFM1-24120C	
24 or 220 V DC/AC, floor mount, single coil	DHFM1-24220C	
12 or 24 V DC/AC, floor mount, dual coil	DHFM2-1224C	
24 or 120 V DC/AC, floor mount, dual coil	DHFM2-24120C	
24 or 220 V DC/AC, floor mount, dual coil	DHFM2-24220C	

700 Photoelectric Smoke Detector Heads

Detachable Head Smoke Detectors with Self-Diagnostics

- Intelligent, self-diagnostic
- Built-in drift compensation
- Detachable head and base design

Ordering Information	Model Number	Listing
Photoelectric smoke detector	711U	
Photo/Heat detector with fast response algorithms	711UT	
Photoelectric detector with remote alarm/trouble LED	721U	
Photo/Heat detector with fast response algorithm with remote alarm/trouble LED	721UT	
Photoelectric detector with form C Auxiliary relay	731U	
Four-wire photoelectric detector with remote alarm/trouble LED	741U	
Four-wire photo/heat detector with pst response algorithms	741UT	

12/24 V

400 Two and Four-Wire Smoke Detectors

Self-Diagnostic Photoelectric Smoke Detectors

- Intelligent, self-diagnostic
- Field replaceable optical chamber
- Available 6V DC, 12V DC and 24V DC operation
- Options include sounders, auxiliary relays, and heat sensors, built-in end-of-line relay, low temperature output

Ordering Information	Model Number	Listing
Two-Wire Smoke Detectors		
6V/12V, heat sensor, integral heat sensor	429AT/428AT	California State Fire Marshall
12V/24V	429C/428C	
12V/24V, heat sensor, integral heat sensor	429CT/428CT	
12V/24V, heat sensor, auxiliary relay, integral heat sensor	429CRT	
12V/24V, heat sensor, sounder, integral heat sensor	429CST/428CST	
12V/24V, heat sensor, sounder, integral heat sensor	429CSST	MEA (New York City) approved
Four-Wire Smoke Detectors		
6V/12V, integral heat sensor	449AT/448AT	
12V/24V	449C/448C	
12V/24V, integral heat sensor	449CT/448CT	
12V/24V, integral heat sensor, auxiliary relay	449CRT	
12V/24V, integral heat sensor, auxiliary relay, sounder	449CSRT/448CSRT	
12V/24V, integral heat sensor, sounder	449CST/448CST	California State Fire Marshall
12V/24V, integral heat sensor, auxiliary relay, sounder	449CSST	
12V/24V, integral heat sensor, power supervision/ sensitivity status relay	449CTE/448CTE	
12V/24V, isolated heat sensor, auxiliary relay, sounder	449CSRH/448CSRH	
12V/24V, integral heat sensor, sounder, power supervision/sensitivity status relay	449CSTE	
12V/24V, integral heat sensor, sounder, power supervision/sensitivity status relay	449CSSTE	MEA (New York City) approved
12V/24V, integral heat sensor, low temperature output	449CLT/448CLT	
12V/24V, integral heat sensor, low temperature output, sounder	449CSLT/448CSLT	
12V/24V, isolated heat sensor, sounder	448CSH	

700 Electric Heat Detector Heads

Detachable Head Smoke Detectors with Self-Diagnostics

- Detachable head and base design

Ordering Information	Model Number	Listing
Rate-of-rise/fixed 135°F electronic heat detector	713-5U	
Rate-of-rise electronic heat detector	713-6U	

12/24 V

700 Mounting Bases

- Wire clamp type terminals
- Built-in continuity switch
- Break-off anti-tamper latch

Ordering Information	Model Number	Listing
4" base, 3 terminals for 711U, 712U or 713U heads	701E	
6" base, 3 terminals for 711U, 712U or 713U heads	701U	
4" universal base, 6 terminals for all heads	702E	
6" universal base, 6 terminals for all heads	702U	
4" base, 6 terminals for 731U, 732U	702RE	
6" base, 6 terminals for 731U, 732U	702RU	

770 Projected Beam Smoke Detector

- Low-profile blends in with any environment
- Selectable sensitivity reduces false alarms
- Drift compensation reduces false alarms
- Detects smoke within a 33-330 ft. range

Ordering Information	Model Number	Listing
Four-wire, projected beam type detector, transmitter, receiver and control	771-Kit	

1500 24V DC Fire Alarm Control Panels

- Approved for local, proprietary, central station, elevator recall, waterflow, and sprinkler supervisory
- Modular design–1, 3 & 5 zone models
- Suitable for occupied areas in hotels, motels and dormitories
- Class "A" or "B" initiating and indicating circuits standard
- Alarm verification & walk test built-in

Ordering Information	Model Number	Listing
Single-zone control	1501	
Three-zone control	1503	
Five-zone control	1505	
Programmer for 2500-DAC	2500-KPD	
Digital alarm communicator (Replaces 1500-DAC2)	2500-DAC	
Local energy master box trip module	1500-LEM	
Five zone supervised remote alarm annunciator	1500-RA-5	
Five zone supervised remote alarm annunciator w/trouble indicator	1500-RA-5A	
Semi-flush trim kit	1500-TK	
Zone relay module (one zone requiring relay)	1500-ZRM-1	
Zone relay module (two or three zones requiring relay)	1500-ZRM-3	
Zone relay module (four or five zones requiring relay)	1500-ZRM-5	

700 Series Duct Smoke Detectors

- Integral self-diagnostics provide advanced maintenance notification
- Built-in drift compensation increases service life
- 12V, 24V, 120V or 220V operation

Ordering Information	Model Number	Listing
12-24 V	729C-DUCT	
12-24 V model with relay output	739CR-DUCT	
120-VAC model with voltage transformer on the function card (to reduce the voltage to the detector to 24V), outputs on two relays	749DR-DUCT	

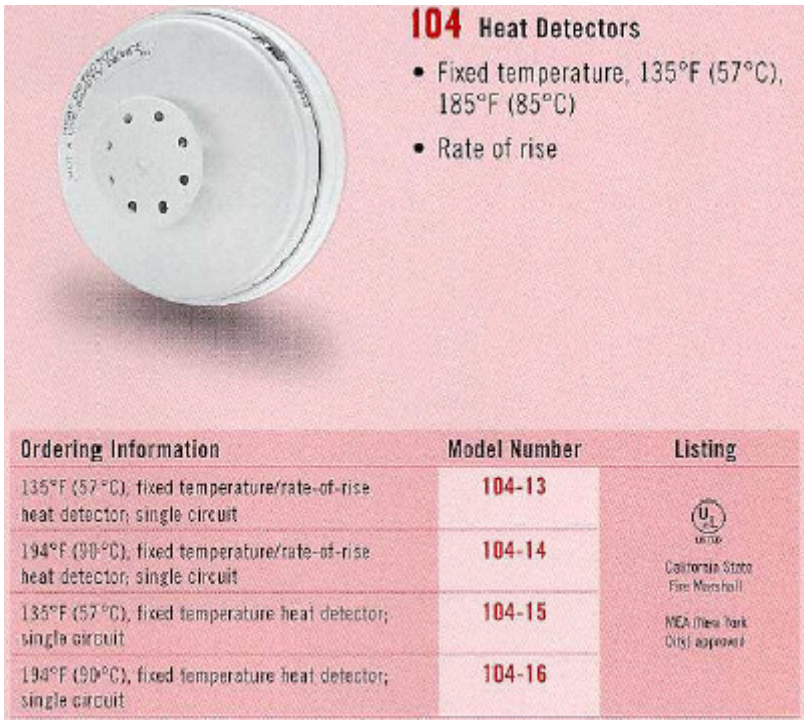
103 Manual Fire Alarm Stations

- Single and dual action pull stations
- Terminal or pigtail connections
- Semi-flush or surface mounting

Ordering Information	Model Number	Listing
Single action (SPST) with hex reset	103-20	
Single action (DPST) with hex reset	103-21	
Dual action (SPST) with hex reset	103-22*	
Dual action (SPST) with hex reset and NY stripe	103-22S*	
Dual action (DPST) with hex reset	103-23	
Dual action, pre-signal (DPST) with hex reset	103-24*	
Single action (SPST) with key reset	103-31	
Single action (DPST) with key reset	103-32	
Dual action (SPST) with key reset	103-42	
Weatherproof, single action (SPST) with key reset	103-60	
Explosion-proof, dual action (DPDT) with key reset	103-80	

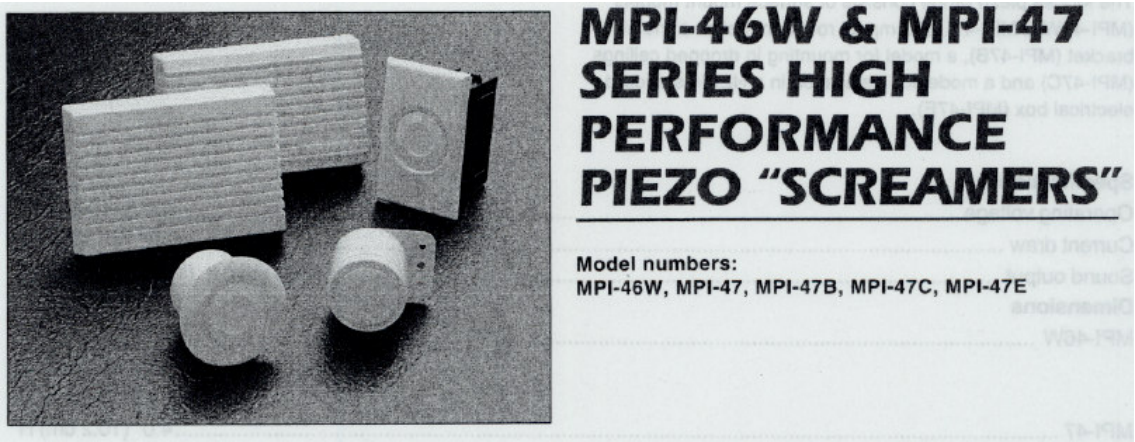
California State Fire Marshall

*NYC BSA



- ***OTROS DISPOSITIVOS UTILIZADOS EN LOS SISTEMAS DE SEGURIDAD***

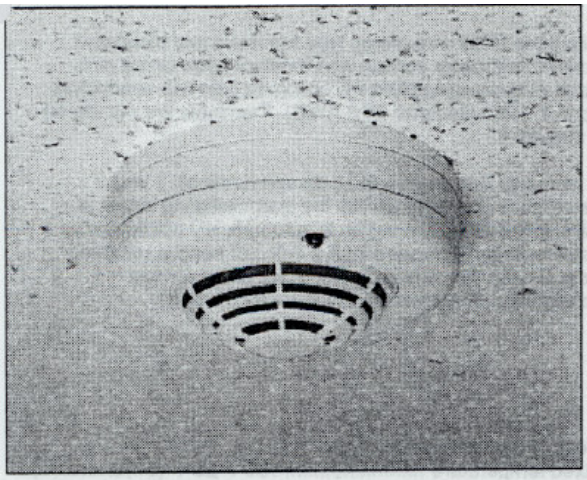
  - **ALARMAS**



Specifications:

Operating voltage	6–13.8 VDC
Current draw	175 mA
Sound output	106dB at 10 ft. (112dB at 1 meter)
Dimensions	
MPI-46W	4.0" (10.2 cm) H
	6.5" (16.5 cm) W
	1.5" (3.8 cm) D
MPI-47	4.0" (10.2 cm) H
	6.5" (16.5 cm) W
	1.5" (3.8 cm) D
MPI-47B	2.0" (5.1 cm) diameter
	bracket 2.0" (5.1 cm) x 2.75" (6.9 cm)
MPI-47C	2.0" (5.1 cm) diameter
	faceplate 3.0" (7.6 cm) diameter
MPI-47E	2.0" (5.1 cm) diameter
	faceplate 4.5" (11.4 cm) H x 2.75" (6.9 cm) W
	Fits 2.0" (5.1 cm) wide plastic box; a metal box should be at least 2.1" (5.3 cm) x 2.1" (5.3 cm)
Color	White
Housing material	ABS plastic

- **DETECTORES DE CALOR O TEMPERATURA**



700 SERIES

FIXED TEMPERATURE AND RATE-OF-RISE ELECTRONIC HEAT DETECTORS

Model numbers:
713-5U, 701U, 702E, 702U

UL LISTED FM APPROVED California State Fire Marshal Approved

Specifications:

Product Data

Voltage	8.5 – 33 VDC, non polarity sensitive
Maximum ripple (peak to peak)	10% (Vp—p)
Typical average standby current (24V)	70µA
Typical alarm current (24V)	up to 60 mA max, if not limited by control panel
Operating temperature	32°F to 120°F (0°C to 49°C)
Operating humidity range	0 to 95% Non-condensing
RFI immunity	20 V/m minimum; 0-1000 MHz
Color	White head and base
Field wiring size	12-24 AWG
Detector packaging	10 detectors per carton
Base packaging	10 bases per carton

Heat detector specifications

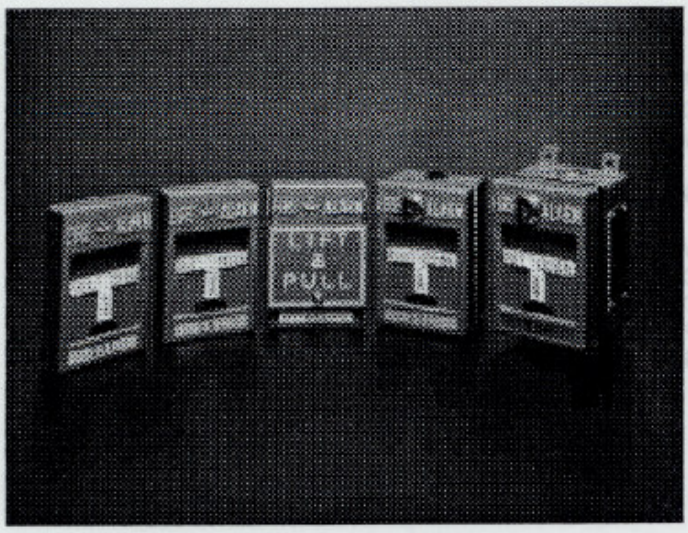
Fixed temperature	135°F ±3°F (57°C ±1.7°C)
Rate of rise	15°F/min and > 105°F (8.3°C/min and > 40.6°C)
UL two-wire compatibility identifier	S10A for heat detector head, S00 for all bases
Detector head dimensions	4" (10 cm) diameter 1.75" (4.44 cm) height
Spacing per UL 521	70' (21.34 m)

Base dimensions

702E	4" (10 cm) diameter, 0.6" (1.3 cm) height
701U, 702U	6" (15 cm) diameter, 0.6" (1.3 cm) height
Total height, head and base together	1.98 (5 cm) height
Reset voltage	2.5 V max
Reset time	1 second max
Listing	UL 521

Ordering Information

Model Number	Description
713-5U	2-wire rate-of-rise and 135°F (57°C) fixed temperature heat detector
Bases	
701U	6" (15 cm) base, 3 terminal connectors for 713-5U heads
701U-10PK	6" (10 cm) base, 3 terminal connectors for 713-5U, 10 packs
702U	Universal 6" (15 cm) base, with 6 terminal connectors
702E-10PK	Universal 4" (10 cm) base, with 6 terminal connectors (10 pack only)



103 SERIES

Manual Fire Alarm Stations

LISTED

Application

The 103 Series manual fire alarm stations are intended for use as fire alarm initiating devices, with any fire alarm initiating circuit to be operated by a contact closure rated not more than 10 amperes, 125 VAC.

Installation

The 103 Series manual fire alarm stations mount to standard single-gang electrical boxes or to a plaster ring. The maximum depth of the equipment is 1 1/4".

Model 103-25 surface mounting box is available. The box is 5" high x 3 1/4" wide x 2 3/4" deep.

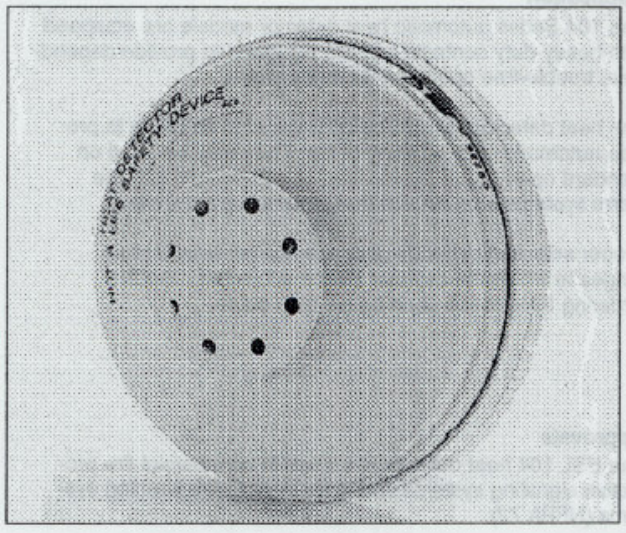
All stations should be installed in compliance with the applicable NFPA standards, local building codes, and plans and specifications.

Product Data

Contact Ratings ..... 10A @ 125 VAC
Mounting ..... Flush Standard Outlet Box with one device cover at least 1 3/4" deep
Surface Model 103-25 ..... Surface Box 5"H x 3 1/4"W x 2 3/4"D
Material ..... Die cast aluminum alloy–enameled finish
Color ..... Red housing–white handle
Operation ..... General alarm–single or dual action
Pre-Signal–Key Operated ..... Removable in either position
Shipping Weight
- Station ..... 20 oz.
- Surface Box ..... 14 oz.

Ordering Information

Model	Description
103-20	Single action manual fire alarm station, hex key reset, non-coded SPST
103-21	Same as above except DPST
103-22 (s)*	Dual action manual fire alarm station, hex key reset, non-coded SPST (s)* with NY stripe
103-23	Same as above except DPST
103-25	Surface mounting box
103-26	Replacement break-glass rod
103-31	Single action (SPST) with key reset
103-42	Dual action manual fire alarm station, cat 30 key reset, non-coded SPST
103-60	Water/weatherproof, single action (SPST) with key reset



104 SERIES

Rate-of-Rise/ Fixed Temperature Heat Detector

Principle of Operation

For Rate-of-Rise: Air expands when heated and contracts when cooled. The ESL 104 Series of heat detectors consist of an air chamber, a flexible metal diaphragm and a carefully calibrated, moisture-proof, trouble-free vent. For normal day-today temperature fluctuations, the "breathing" action of the vent compensates for any expansions or contraction of air in the detector chamber. When a fire occurs, air in the chamber expands faster than it can be vented, causing the diaphragm to distend and close electrical contacts. This feature resets automatically if the fixed temperature element is not activated.

For Fixed Temperature: An actuating spring is held under tension by a eutectic metal. When the detector temperature reaches the melting point of the eutectic metal, the spring releases, causing electrical contacts to close. When activated, the external heat collector drops away to provide quick, visual confirmation that the element has operated and that the detector must be replaced.

Ordering Information

Model	Description	Listings
104-13	135°F (57°C), fixed temperature/rate-of-rise heat detector, single circuit	CSFM, UL 521, MEA
104-14	194°F (90°C), fixed temperature/rate-of-rise heat detector, single circuit	CSFM, UL 521, MEA
104-15	135°F (57°C), fixed temperature heat detector, single circuit	CSFM, UL 521, MEA
104-16	194°F (90°C), fixed temperature heat detector, single circuit	CSFM, UL 521, MEA

EL PUERTO PARALELO

El Puerto paralelo es una herramienta simple y barata para construir dispositivos controlados en la computadora y proyectos.

Su simplicidad y facilidad de programar hace al puerto paralelo popular entre los pasatiempos electrónicos mas frecuentados del mundo. El puerto paralelo es a menudo usado en Robots controlados por computadora, programadores de Atmel/PIC, automatizaciones caseras o del hogar, etc. Las PC's generalmente poseen solo uno de estos puertos (LPT1) pero con muy poco dinero se le puede adicionar una tarjeta con un segundo puerto paralelo (LPT2), comprando tarjetas ISA/PCI del puerto paralelo.

Los puertos de comunicación de la PC son de particular interés para el la electrónica ya que le permiten utilizar una computadora personal para controlar todo tipo circuitos electrónicos utilizados, principalmente, en actividades de automatización de procesos, adquisición de datos, tareas repetitivas y otras actividades que demandan precisión.

El puerto paralelo de una PC es ideal para ser usado como herramienta de control de motores, relés, LED's, etc. El mismo posee un bus de datos de 8 bits (Pin 2 a 9) y muchas señales de control, algunas de salida y otras de entrada que también pueden ser usadas fácilmente.

Conceptos básicos

Puerto: Es un conjunto de líneas (interfaz) que puede utilizar el CPU para intercambiar información con otros dispositivos

Existen dos métodos básicos para transmisión de datos en las computadoras modernas. En un esquema de transmisión de datos en serie un dispositivo envía

datos a otro a razón de un bit a la vez a través de un cable. Por otro lado, en un esquema de transmisión de datos en paralelo un dispositivo envía datos a otro a una tasa de n número de bits a través de n número de cables a un tiempo. Sería fácil pensar que un sistema en paralelo es n veces más rápido que un sistema en serie, sin embargo esto no se cumple, básicamente el impedimento principal es el tipo de cable que se utiliza para interconectar los equipos. Si bien un sistema de comunicación en paralelo puede utilizar cualquier número de cables para transmitir datos, la mayoría de los sistemas paralelos utilizan ocho líneas de datos para transmitir un byte a la vez, como en todo, existen excepciones, por ejemplo el estándar SCSI permite transferencia de datos en esquemas que van desde los ocho bits y hasta los treinta y dos bits en paralelo. En éste caso nos concentraremos en transferencias de ocho bits ya que ésta es la configuración del puerto paralelo de una PC.

En reglas generales la dirección hexadecimal del puerto LPT1 es igual a 0x378 (888 en decimal) y 0x278 (632 en decimal) para el LPT2. Esto se puede verificar fácilmente en el setup de la PC o bien en el cartel que generalmente la PC muestra en el momento del booteo. Puede darse el caso que el LPT1 asuma la dirección 0x3BC (956 en decimal) y el LPT2 0x378, en ese caso habrá que tratar de corregir el setup y/o los jumper de las tarjetas en caso que sea posible

Un típico sistema de comunicación en paralelo puede ser de una dirección (unidireccional) o de dos direcciones (bidireccional). El más simple mecanismo utilizado en un puerto paralelo de una PC es de tipo unidireccional y es el que se analizara en primer lugar. Se distinguen dos elementos: la parte transmisora y la parte receptora. La parte transmisora coloca la información en las líneas de datos e informa a la parte receptora que la información (los datos) está disponible; entonces la parte receptora lee la información en las líneas de datos e informa a la parte transmisora que ha tomado la información (los datos).

Hay que observar que ambas partes sincronizan su respectivo acceso a las líneas

de datos, la parte receptora no leerá las líneas de datos hasta que la parte transmisora se lo indique en tanto que la parte transmisora no colocará nueva información en las líneas de datos hasta que la parte receptora remueva la información y le indique a la parte transmisora que ya ha tomado los datos, a ésta coordinación de operaciones se le llama acuerdo ó entendimiento. Bien, en éstos ámbitos tecnológicos es recomendable utilizar ciertas palabras en inglés que permiten irónicamente un mejor entendimiento de los conceptos tratados. A la coordinación de operaciones entre la parte transmisora y la parte receptora se le llama handshaking, que en español es el acto con el cual dos partes manifiestan estar de acuerdo, es decir, se dan un apretón de manos.

El handshaking

Para implementar el handshaking se requieren dos líneas adicionales. La línea de estroboscopio (en inglés strobe) es la que utiliza la parte transmisora para indicarle a la parte receptora la disponibilidad de información. La línea de admisión (acknowledge) es la que utiliza la parte receptora para indicarle a la parte transmisora que ha tomado la información (los datos) y que está lista para recibir más datos. El puerto paralelo provee de una tercera línea de handshaking llamada en inglés busy (ocupado), ésta la puede utilizar la parte receptora para indicarle a la parte transmisora que está ocupada y por lo tanto la parte transmisora no debe intentar colocar nueva información en las líneas de datos. Una típica sesión de transmisión de datos se parece a lo siguiente:

Parte transmisora:

- La parte transmisora checa la línea busy para ver si la parte receptora está ocupada. Si la línea busy está activa, la parte transmisora espera en un bucle hasta que la línea busy esté inactiva.
- La parte transmisora coloca la información en las líneas de datos.
- La parte transmisora activa la línea de strobe.
- La parte transmisora espera en un bucle hasta que la línea acknowledge está activa.

- La parte transmisora inactiva la línea de strobe.
- La parte transmisora espera en un bucle hasta que la línea acknowledge esté inactiva.
- La parte transmisora repite los pasos anteriores por cada byte a ser transmitido.

Parte receptora:

- La parte receptora inactiva la línea busy (asumiendo que está lista para recibir información).
- La parte receptora espera en un bucle hasta que la línea strobe esté activa.
- La parte receptora lee la información de las líneas de datos (y si es necesario, procesa los datos).
- La parte receptora activa la línea acknowledge.
- La parte receptora espera en un bucle hasta que esté inactiva la línea de strobe.
- La parte receptora inactiva la línea acknowledge.
- La parte receptora repite los pasos anteriores por cada byte que debe recibir.

Se debe ser muy cuidadoso al seguir éstos pasos, tanto la parte transmisora como la receptora coordinan sus acciones de tal manera que la parte transmisora no intentará colocar varios bytes en las líneas de datos, en tanto que la parte receptora no debe leer más datos que los que le envíe la parte transmisora, un byte a la vez.

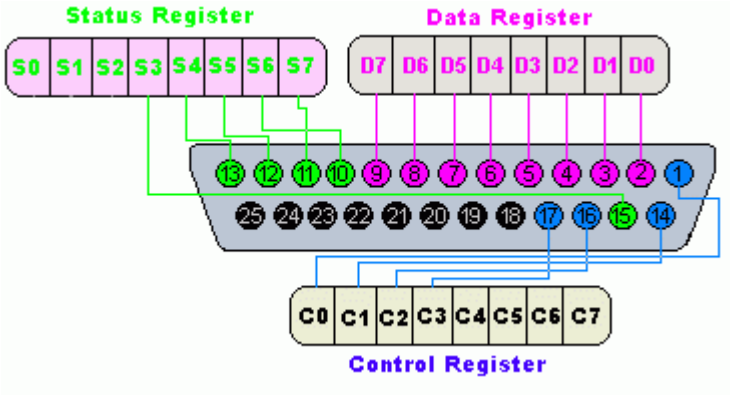
El hardware del puerto paralelo

El puerto paralelo recibe ese nombre debido a la forma en que los datos son leídos y escritos, o sea que se leen en paquetes de 8 bits a la vez, en forma paralela. El verdadero nombre del conector (con el que se va a las tiendas a comprarlo) es DB-25, ya que se trata de un conector de 25 pines (así como el DB-9 y DB-15, puerto serial y gameport respectivamente).

El puerto paralelo de una típica PC utiliza un conector de tipo D de 25 pines (DB-25 S), éste es el caso más común, sin embargo es conveniente mencionar los tres tipos de conectores definidos por el estándar IEEE 1284, el primero, llamado 1284 tipo A es un conector de 25 pines de tipo D, es decir, el que se menciona al principio. El puerto paralelo se presenta de dos formas: Hembra y Macho. Si posee 25 pines se trata de un puerto Macho, pero si posee 25 agujeros entonces se trata de un puerto Hembra. El cable que se utiliza para conectar éste puerto también recibe el nombre de DB-25 y es un cable blindado de 25 líneas; el blindaje por lo general es un vigésimo sexto cable sin forro que se debe de conectar al tierra del circuito.

Los 25 cables para poder ser diferenciados utilizan una codificación de colores, que por lo general se trata de un solo color base o de un color base con una pequeña franja negra o blanca (dependiendo de la tonalidad del color base).

El orden de los pines del conector de salida del tipo DB25, cuyo diagrama y señales utilizadas se puede ver es el siguiente:



La asignación de pines para cada uno de los bits es estándar en todas las computadoras, y lo más notable e importante es que en el caso del puerto de datos se tiene un pin por bit, en cambio en los registros de estado y control, no todos los bits poseen pines, pero sin embargo cuando se trabajen los datos,

siempre se tomarán en cuenta estos bits faltantes. A demás es necesario saber que algunos de los bits están negados.

Pin No (DB25)	Signal name	Direction	Register - bit	Inverted
1	nStrobe	Out	Control-0	Yes
2	Data0	In/Out	Data-0	No
3	Data1	In/Out	Data-1	No
4	Data2	In/Out	Data-2	No
5	Data3	In/Out	Data-3	No
6	Data4	In/Out	Data-4	No
7	Data5	In/Out	Data-5	No
8	Data6	In/Out	Data-6	No
9	Data7	In/Out	Data-7	No
10	nAck	In	Status-6	No
11	Busy	In	Status-7	Yes
12	Paper-Out	In	Status-5	No
13	Select	In	Status-4	No
14	Linefeed	Out	Control-1	Yes
15	nError	In	Status-3	No
16	nInitialize	Out	Control-2	No
17	nSelect-Printer	Out	Control-3	Yes
18-25	Ground	-	-	-

El segundo conector se llama 1284 tipo B que es un conector de 36 pines de tipo centronics y lo encontramos en la mayoría de las impresoras; el tercero se denomina 1284 tipo C, se trata de un conector similar al 1284 tipo B pero más pequeño, además se dice que tiene mejores propiedades eléctricas y mecánicas, éste conector es el recomendado para nuevos diseños.

La siguiente tabla describe la función de cada pin del conector 1284 tipo A:

Pines	E/S	Polaridad activa	Descripción
1	Salida	0	Strobe
2 ~ 9	Salida	-	Líneas de datos (bit 0/pin 2, bit 7/pin 9)
10	Entrada	0	Línea acknowledge (activa cuando el sistema remoto toma datos)
11	Entrada	0	Línea busy (si está activa, el sistema remoto no acepta datos)

Pines	E/S	Polaridad activa	Descripción
12	Entrada	1	Línea Falta de papel (si está activa, falta papel en la impresora)
13	Entrada	1	Línea Select (si está activa, la impresora se ha seleccionado)
14	Salida	0	Línea Autofeed (si está activa, la impresora inserta una nueva línea por cada retorno de carro)
15	Entrada	0	Línea Error (si está activa, hay un error en la impresora)
16	Salida	0	Línea Init (Si se mantiene activa por al menos 50 micro-segundos, ésta señal autoinicializa la impresora)
17	Salida	0	Línea Select input (Cuando está inactiva, obliga a la impresora a salir de línea)
18 ~ 25	-	-	Tierra eléctrica

Tabla 1: Configuración del puerto paralelo estándar

Observe que el puerto paralelo tiene 12 líneas de salida (8 líneas de datos, strobe, autofeed, init, y select input) y 5 de entrada (acknowledge, busy, falta de papel, select y error). El estándar IEEE 1284 define cinco modos de operación:
Modo compatible
Modo nibble
Modo byte
Modo EPP, puerto paralelo ampliado
Modo ECP, puerto de capacidad extendida

El objetivo del estándar es diseñar nuevos dispositivos que sean totalmente compatibles con el puerto paralelo estándar (SPP) definido originalmente por la IBM.

Hay tres direcciones de E/S asociadas con un puerto paralelo de la PC, estas direcciones pertenecen al registro de datos, el registro de estado y el registro de control. El registro de datos es un puerto de lectura-escritura de ocho bits. Leer el registro de datos (en la modalidad unidireccional) retorna el último valor escrito en el registro de datos. Los registros de control y estado proveen la interface a las

otras líneas de E/S. La distribución de las diferentes señales para cada uno de los tres registros de un puerto paralelo esta dada en las siguientes tablas:

Dirección	Nombre	Lectura/Escritura	Bit #	Propiedades	Pin #
Base + 0	Puerto de datos	Escritura	Bit 7	Dato 7	9
			Bit 6	Dato 6	8
			Bit 5	Dato 5	7
			Bit 4	Dato 4	6
			Bit 3	Dato 3	5
			Bit 2	Dato 2	4
			Bit 1	Dato 1	3
			Bit 0	Dato 0	2

Tabla 2: Registro de datos

Dirección	Nombre	Lectura/Escritura	Bit #	Propiedades	Pin #
Base + 1	Puerto de estado	Sólo Lectura	Bit 7	Busy	11
			Bit 6	Acknowledge	10
			Bit 5	Falta de papel	12
			Bit 4	Select In	13
			Bit 3	Error	15
			Bit 2	IRQ (Not)	-
			Bit 1	Reservado	-
			Bit 0	Reservado	-

Tabla 3: Registro de estado

Dirección	Nombre	Lectura/Escritura	Bit #	Propiedades	Pin #
Base + 2	Puerto de control	Lectura/Escritura	Bit 7	No usado	-
			Bit 6	No usado	-
			Bit 5	Permite puerto bidireccional	-
			Bit 4	Permite IRQ a través de la línea acknowledge	IRQ enable
			Bit ~3	Selecciona impresora	17
			Bit 2	Inicializa impresora	16

Dirección	Nombre	Lectura/Escritura	Bit #	Propiedades	Pin #
			Bit ~1	Nueva línea automática	14
			Bit ~0	Strobe	1

Tabla 4: Registro de control

Cada sección es accesada por su propia dirección y actuara independiente del resto. Esto es casi como que si ellos estuvieran en diferentes puertos. Las direcciones son las siguientes:

PUERTO	DIRECCION (DECIMAL)	DIRECCION (HEXADECIMAL)
Líneas de Datos	888	378h
Líneas de Control	890	37Ah
Líneas de Status	889	379h

Se necesita saber la dirección del puerto a usar, y también 2 cosas mas, el comando de acceso al puerto y el numero que se quiere colocar. El comando se estará explicando luego. Los puertos trabajan con números. Estos pueden ser expresados en hexadecimal, binario y decimal, pero para este documento todos los valores se expresaran en decimales para facilitarlo un poco más. El puerto se opera por el envío de un número que represente un patrón binario de las salidas físicas del puerto. Por ejemplo, colocar en la línea 8 datos el 11111111, se enviaría el 255. Para colocarles el 00000000 se colocaría el 0. Nótese que todos son números binarios de 8 bit, y el puerto es también una salida de 8.

Una PC soporta hasta tres puertos paralelo separados, por tanto puede haber hasta tres juegos de registros en un sistema en un momento dado. Existen tres direcciones base para el puerto paralelo asociadas con tres posibles puertos paralelo: 0x3BCh, 0x378h y 0x278h, se refiere a éstas como las direcciones base para el puerto LPT1, LPT2 y LPT3, respectivamente. El registro de datos se localiza siempre en la dirección base de un puerto paralelo, el registro de estado aparece en la dirección base + 1, y el registro de control aparece en la dirección base + 2. Por ejemplo, para un puerto LPT2 localizado en 0x378h, ésta es la dirección del registro de datos, al registro de estado le corresponde la dirección

0x379h y su respectivo registro de control está en la dirección 0x37Ah. Cuando la PC se enciende el BIOS ejecuta una rutina para determinar el número de puertos presentes en el sistema asignando la etiqueta LPT1 al primer puerto localizado, si existen más puertos entonces se asignarán consecutivamente las etiquetas LPT2 y LPT3 de acuerdo a la siguiente tabla:

Dirección inicial	Función
0000:0408	Dirección base para LPT1
0000:040A	Dirección base para LPT2
0000:040C	Dirección base para LPT3
0000:040E	Dirección base para LPT4

Tabla 5: Direcciones base en el BIOS

Si deseamos escribir un dato en el bus de salida de datos (pin 2 a 9) solo debemos escribir el byte correspondiente en la dirección hexadecimal 0X378 (888 en decimal) cuando trabajamos con el LPT1 y 0x278 (632 en decimal) cuando trabajamos con el LPT2. Los distintos pines (bits) de salida correspondientes al bus de datos no pueden ser escritos en forma independiente, por lo que siempre que se desee modificar uno se deberán escribir los ocho bits nuevamente.

Para leer el estado de los pines de entrada (10, 12, 13 y 15) se debe realizar una lectura a la dirección hexadecimal 0x379 (889 en decimal) si trabajamos con el LPT1 o bien leer la dirección 0x279 (633 en decimal) si trabajamos con el LPT2. La lectura será devuelta en un byte en donde el bit 6 corresponde al pin 10, el bit 5 corresponde al pin 12, el bit 4 corresponde al pin 13 y el bit 3 corresponde al pin 15.

En la siguiente tabla se puede ver lo antedicho en una forma más gráfica:

Escritura: Salida de Datos								
Escritura en dirección 0x378 (LPT1) o 0x278 (LPT2)								
DATO	BIT 7	BIT 6	BIT 5	BIT 4	BIT 3	BIT 2	BIT 1	BIT 0

DB25	Pin 9	Pin 8	Pin 7	Pin 6	Pin 5	Pin 4	Pin 3	Pin2
CN5	TTL 7	TTL 6	TTL 5	TTL 4	TTL 3	TTL 2	TTL 1	TTL 0
CN4	No usar	HP 6	HP 5	HP 4	HP 3	HP 2	HP 1	HP 0

Lectura: Entrada de Datos								
Lectura en dirección 0x379 (LPT1) o 0x279 (LPT2)								
DATO	BIT 7	BIT 6	BIT 5	BIT 4	BIT 3	BIT 2	BIT 1	BIT 0
DB 25	No usar	Pin 10	Pin 12	Pin 13	Pin 15	No usar	No usar	No usar
CN6	No usar	Input 3	Input 2	Input 1	Input 0	No usar	No usar	No usar

Características Eléctricas:

El circuito interno del puerto paralelo esta hecho con integrados de la familia TTL, por lo que los datos se manejan de forma lógica de la manera siguiente:

Valor lógico	Rango de voltaje
0	0.0 - 0.8 V
1	2.4 - 5.0 V

Corriente máxima	
Entrada	24 mA
Salida	2.6 mA

Velocidad:

Un puerto paralelo normal toma un ciclo para leer o escribir. Como en muchos sistemas, la velocidad del bus ronda los 1,3 Mhz, podemos decir que la lectura la podemos hacer cada 1 uS (idealmente, ya que siempre se tienen otras instrucciones software, etc.). Algunos puertos soportan un modo “turbo” que elimina los 3 estados de espera de la CPU, con lo que la velocidad de lectura/escritura al puerto se duplica (2,7 MHz).

Detección del tipo de puerto:

En esta sección se indicará los pasos a seguir para detectar si el puerto es SPP

(común), bidireccional compatible PS/2, ECP o EPP. Para una referencia acerca de los puertos EPP y ECP.

Se describe los pasos generales; luego se tendrá que adaptar al lenguaje en uso. Se testea en orden descendente de complejidad, es decir primero ECP, luego EPP, bidireccional y finalmente SPP (esto se debe realiza así para no fallar en la detección).

Detectando ECP

Este es el método que recomienda Microsoft:

Leer el control de registro extendido (ECR) en Base+402h. Base se refiere a la dirección inicial donde se comienza a mapear el puerto (03BCh,0378h y 0278h). Verificar que el bit 0 sea 1 (FIFO vacía) y que el bit 1 sea 0 (FIFO no está llena). Estos bits podrían ser distintos que los bits correspondientes en el puerto de control (Base+2h). Para verificar esto, cambiamos el estado de algún bit del puerto Base+2h y se verifica que no haya cambiado en Base+402h. Una prueba adicional es escribir 34h al ECR y leerlo. Los bits 0 1 y son de sólo lectura, por lo tanto, si se lee 35h, es probable que se tenga un puerto ECP.

Detectando EPP

Además de los tres registros de un puerto SPP, un puerto EPP tiene cinco registros más mapeados desde Base+3h a Base+7h. Estos registros adicionales proveen una manera de testear la presencia de un EPP, escribiendo ciertos valores y leyendo el resultado (de la misma manera que se testea un puerto SPP). Al igual que al detectar puertos SPP, se recomienda escribir un par de valores y leerlos. Dichos valores podrían ser 55h y AAh.

Hay que asegurarse de poner S0 a 0 (EPP timeout), antes de leer o escribir a estas direcciones extendidas.

Otro dato importante es que no existen puertos EPP en la dirección base 03BCh.

Detectando SPP

Para detectar si es SPP, hacemos el procedimiento que realiza la BIOS al arranque.

Se verifica la retención del puerto, que será posible en un puerto SPP. Para ello se escribe un par de valores (55h y AAh) a las direcciones base (03BCh,0378h y 0278h) y se lee el registro verificando que sean los mismo valores escritos.

Detectando puerto bidireccional (PS/2)

Aquí debemos habilitar el puerto bidireccional (con C5=1) y hacer la misma prueba que para un SPP para verificar que no haya retención.

Programar el puerto paralelo

Los software de programación no pueden directamente acceder al hardware en un sistema. Todo hardware requiere ir a través de Windows. Porque de esto, los más cerca que pueden manipularse el puerto paralelo es con el objeto impresor. Mientras este todo bien y bueno cuando pueda imprimir algo, es menos utilizado cuado se quiere directamente un control del hardware. Pueden haber llamadas API alrededor de todo esto, pero por el momento no se a podido encontrar alguna. En orden de controlar el puerto directamente, podemos usar algo externo al programa. El cual es un producto gratis que hace exactamente lo que se quiere. Es un DLL de una compañía llamada SoftCircuits. Se puede bajar este DLL desde su pagina Programming Tools and Libraries. En este caso se utiliza inpout32.dll, para la entrada y salida de bit en el puerto. También se puede utilizar WIN95IO.DLL. No importa cual se escoja, el archivo DLL debe de estar en el directorio windows\system32 en cualquier maquina de software de interfase de control que sea usado o desarrollado. Una nota especial es, que no importa que DLL se use, no funcionaria bajo Windows NT. (Por seguridad).

Antes de usar cualquier función contenida dentro del DLL, se deben declarar. Estas declaraciones están para ser colocadas en cualquier modulo en la sección del programa de las Declaraciones Generales.

Una vez se declaren las funciones, se tendrán 2 nuevos comandos disponibles. Estos son Inp32 y Out32. Out32 que es una declaración y es usada para enviar un bit al Puerto.

Como se puede ver los 2 parámetros requeridos son la dirección del Puerto y el valor que se le desea establecer. La dirección puede ser decimal o hexadecimal. Ya que solo hay 8 líneas de datos, solo se puede enviar un máximo de 255 decimal al puerto o en binario 11111111.

Para utilizar un puerto paralelo como bidireccional (compatible PS/2) se manipulan los bits de control, ya que solo 4 son los que se mantienen activos para las salidas, pero para la manipulación de entradas por medio de este puerto debe activarse el quinto bit de control o pin. El bit C5, está disponible sólo si se trata de un puerto bidireccional; en los puertos comunes actúa como los bits C6 y C7. Si C5=1, el buffer de los datos de salida se pone en alta impedancia, "desconectando" dicho buffer de los pines 2 a 9 del conector del puerto (D0 a D7). Si se escribe al registro de datos, se escribe al buffer pero no a la salida. Esto permite que al leer el puerto, se lea el estado de las entradas y no lo que hay en el buffer.

En otra clase de computadoras como la IBM, para habilitar el puerto paralelo bidireccional, además de lo antes descrito, se debe poner a 1 el bit 7 del registro del puerto 102h (opciones de configuración).

En computadoras que no tengan puerto paralelo bidireccional compatible PS/2 hay que modificar uno o más bits de algún puerto específico correspondiente al chipset de la placa. A veces se habilita por setup o por jumper en la placa del puerto.

Al hacer una interfase con el puerto paralelo se construirá un circuito de muestra en un Protoboard (elemento muy utilizado por los ingenieros electrónicos) esto ofrece mucha comodidad y seguridad al armar un circuito por primera vez.

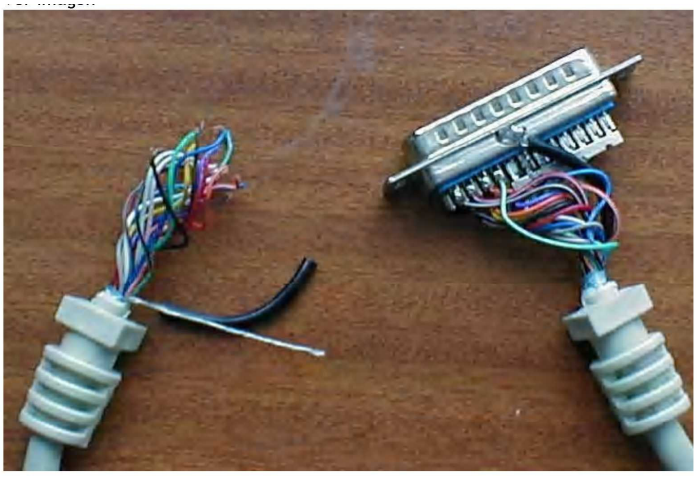
Al manejar el puerto paralelo se tiene que hablar de lo que se llama LOGICA Binaria, la cual solo tiene dos estados que fácilmente se interpretan como 1 y 0, en muchos casos se asocia con encendido y apagado.

Los Pines que tienen una línea superior en su nombre son inversores, o sea en pocas palabras, al indicar encendido el pin niega la acción y queda con el estado contrario.

En este caso solo se usará los pines de salida o sea desde el pin #2(D0) hasta el pin #9(D7) entonces inicialmente se montará un circuito de prueba; los materiales son:

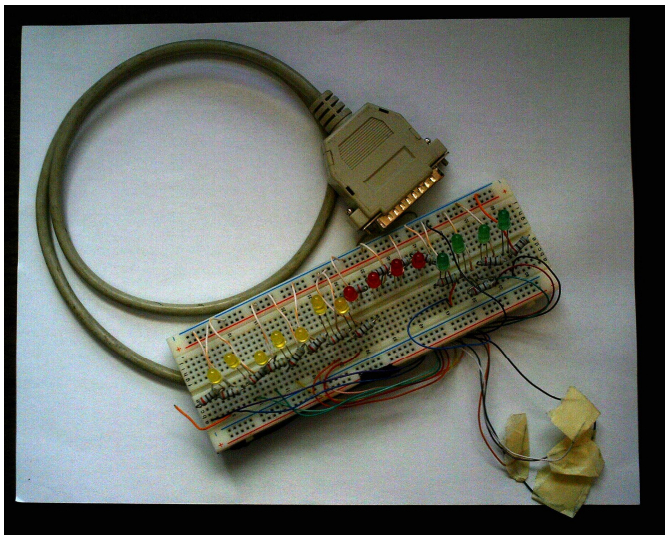
- 1 Cable conector DB-25
- 8 Leds de cualquier color
- 8 Resistencias de 22Oohm a 1/2W
- 1 Protoboard
- Cables conectores

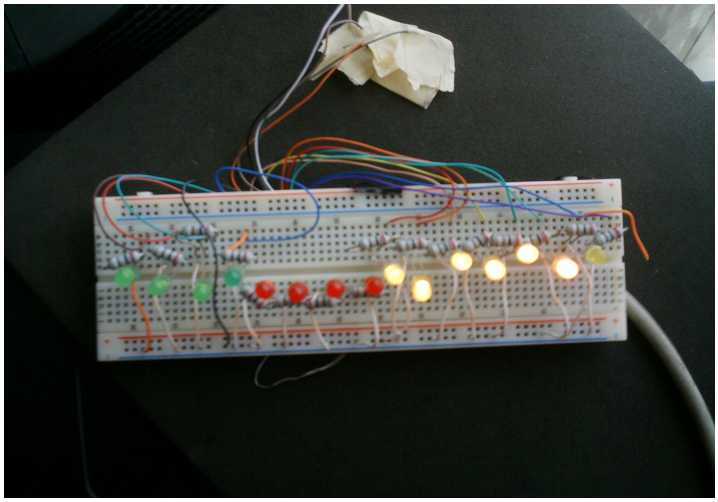
Primero se toma el cable de impresora y se corta el conector que va conectado a la impresora (para reconocerlo es el que no tiene pines, y sobresale una parte rectangular azul)



Luego se desarma el conector del puerto paralelo, hay que observar el conector macho por donde se conecta, el cual posee unos números gravados junto a los pines. Se anotan cada número del pin y el color del cable que esta soldado al conector.

Una vez se tenga la tabla con numero de pin y color del cable se dispondrá a construir el circuito en el protoboard.
Una vez construido el circuito se verá de la siguiente forma:





Ya con el circuito montado se puede pensar en el programa para manejar el puerto. Como el programa es muy sencillo, solo enciende el pin que se quiere encendido y listo.

Luego para la manipulación de los pines ya sea de entrada y salida para no quemar o arruinar el puerto paralelo, ni mucho menos la tarjeta madre (motherboard) se utiliza otra clase de circuito, cuyos materiales son:

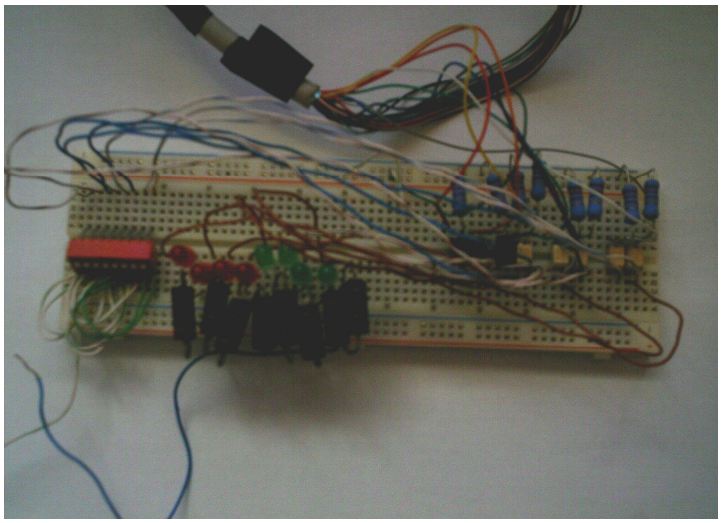
- 1 Cable conector DB-25
- 8 Octocopladores
- 8 Resistencias 1k 1W
- 8 Resistencias 220 ohm 1W
- 8 Diodos leds
- 1 Integrado interruptor
- 1 Protoboard
- 1 Fuente de poder de 5v
- Muchos cables conectores

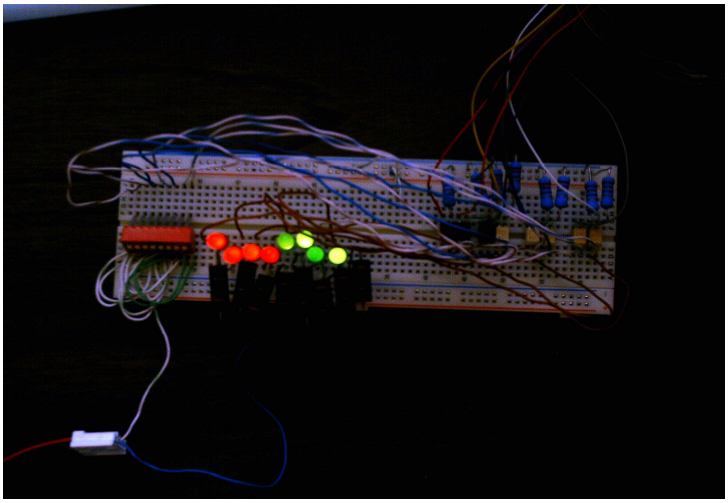
Primero para alimentar el circuito se utilizará una fuente externa de voltaje, el circuito esta calculado para funcionar a 5V, en este caso se utiliza una línea de 5V que brinda la fuente de la PC, hay que recordar que las tierras son comunes en el circuito, ósea que la tierra de la fuente de 5V es la misma que la del puerto paralelo.

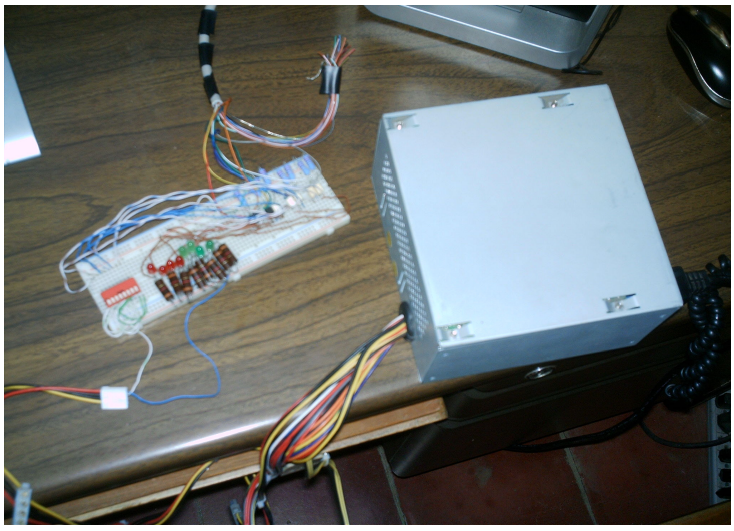
Los octocopladores se conectan, por medio de cada pin que este posee, el pin 4 de este es el que se conecta con una resistencia de 220 ohm, el pin 3 con el pin del puerto, en este caso desde el pin #2(D0) hasta el pin #9(D7), del otro lado del octocoplador el pin 2 se conecta por medio de cables con los diodos leds, que van conectadas paralelamente a las resistencias de 1k, las cuales van conectadas simultáneamente a la línea de 5v. También el integrado del interruptor se conecta con el octocoplador, por medio de cables con el pin 1, para el encendido o

apagado de cada pin. Este interruptor integrado va conectado a la vez, a una línea directa a tierra.

Una vez terminado el circuito se vera de la siguiente forma:







Listo y una vez alimentado el circuito, se comprueba con los leds que la salida funciona, por lo tanto, quiere decir que los octocopladores trabajan correctamente y como resultado el puerto esta protegido.

OCTOCOPLADORES

DEFINICION: son conjuntos integrados de componentes que permiten el acoplamiento de señales desde un circuito a otro por medio de luz visible o infrarroja.

Se les conoce también por el nombre de optoaisladores, debido a que los circuitos en acoplo permanecen en completo aislamiento eléctrico.

VENTAJAS: Además de permitir aislamiento eléctrico entre dos circuitos, los optoacopladores son de reducido tamaño (vienen como Chips), son de reducido tamaño, muy confiables, de bajo precio y tienen total compatibilidad con los circuitos digitales.

DETERMINACIÓN DE NECESIDADES DE LA UES-FMO

DISTRIBUCIÓN DE EDIFICIOS DE LA UES-FMO

Ubicación Geográfica

La Facultad Multidisciplinaria de Occidente se ubica geográficamente al sur de la ciudad de Santa Ana, al final de la colonia San Luís, sobre la Avenida Fray Felipe de Jesús Moraga colindando con la colonia Altos del Palmar al sur y poniente y la colonia Universitaria al Norte y Oriente.

Descripción del terreno

La topografía del terreno es plana, y está definida por dos pendientes que convergen en la parte central.

Los colindantes de la Facultad son la urbanización Altos del Palmar al costado sur, viviendas colindando al norte, viviendas particulares y Asilo San Francisco de Asís, Avenida Fray Felipe de Jesús Moraga Sur de por medio al este, Viviendas Particulares de colonia universitaria.

Climatológicamente, el terreno del campus universitario se encuentra localizado en una zona identificada como bosque húmedo subtropical, con un suelo tipo Andisoles sub-clase lle, la estación del sistema metereológico más cercana es la Sinóptica El Palmar, (latitud Norte 13° 58.6', longitud Oeste 89° 34.2' y 725 m.s.n.m.), con una precipitación promedio anual de 1699 mm, y 138 días de lluvia al año, temperatura promedio anual de 23.8 °C, Evaporación media de 1,812 mm, humedad relativa del 71%, los vientos corren de noreste a 7.8 Km/h, su geología esta compuesta por Piro clásticas ácidas, epiclastítas volcánicas (tobas de color café) de la formación San Salvador.

Áreas del terreno

El centro universitario cuenta con un área total de 86,189.93 m² equivalentes a 123,322.92 vrs², siendo el área construida de 8,129.22 m², equivalentes a 11,631.51 vrs² componiéndose de 28 edificaciones y el espacio por alumnos de 0.61 m².

Distribución de las áreas de la Facultad por uso, área y porcentaje.

Uso	**Área m²**	**Porcentaje (%)**
Aulas	4023.62	4.67%
Cubículos	1545.57	1.79%
Administrativo	1269.28	1.47%
Biblioteca	883.11	1.02%
Laboratorios	2527.88	2.93%
Cómputo	150.87	0.18%
Servicios Sanitarios Docente Administrativo	52.67	0.06%
S.S. Alumnos	249.29	0.29%
Deporte	9487	11.01%
Talleres	405.9	0.47%
Zona Verde	47,331.44	54.92%
Circulación	18,263.30	21.19%
Total	**86,189.93**	**100%**

Plan de desarrollo interno de la facultad

En primer plano entre los proyectos inmediatos se encuentran la remodelación total del Auditórium, incluyendo lo que es: techos, cielo falso, cerraduras, ventanas (solaires), losa, etc.; el cual esta dividido en dos etapas.

Segundo es la construcción de un edificio de 2 niveles ubicado en lo que es actualmente el edificio ex biblioteca, aula 1, 2 y 3, administración académica, financiera, etc., el cual medirá 20 por 80 m² y contendrá 19 aulas además de administración académica, financiera, colecturia, departamento de química, taller

de ingeniería civil, etc.; para la realización de este proyecto se tenia disponible 1.5 millones de dólares, pero como no aprobaron el BID, por circunstancias no esperadas, está como un proyecto a futuro la implementación de dicha construcción.

Luego entre otros proyectos esta la remodelación de calles internas y pavimentación de ellas, al igual que la elaboración de muro perimetral para la facultad.

ANALISIS PARA LA DISTRIBUCIÓN DE SENSORES

Los sistemas de sensores en general ofrecen las siguientes ventajas y desventajas:

Volumétricos	**De vibración**
Ventajas	**Ventajas**
Se activan cuando el delincuente entra en casa y detectan su temperatura corporal.	Se activa cuando alguien intenta violentar la puerta o la ventana.
Desventajas	**Desventajas**
Pueden activarse si se deja la ventana un poco abierta y el aire varía la temperatura del interior.	Están limitados solo a puertas y ventanas. Un forrado en una pared no lo activa.
Magnéticos	**Infrarrojos**
Ventajas	**Ventajas**
Unos imanes crean un campo magnético que se rompe cuando el delincuente entra.	Los sensores detectan el movimiento en una habitación, activando el sistema de alarma.
Desventajas	**Desventajas**
Si se rompe un vidrio, sin abrir puertas y/o ventanas que separen los magnéticos, la alarma no se activa	Se activan con la presencia de animales en casa ya que este sistema se activa en cuanto detecta un movimiento.

UNIDAD DE CUSTODIOS

Actualmente se cuente con 10 miembros, 7 con plaza y 3 por contrato, los cuales deben cumplir 40 horas laborales a la semana en los distintos turnos rotativos y consecutivos en el transcurso de la semana.

La distribución de los turnos es:
Día 7:00 a.m. – 7:00 p.m.
Noche 7:00 p.m. – 7:00 a.m.
Y 3:00 p.m. – 7:00 a.m.

El perfil para la contratación de nuevo personal de custodios es:
Tiempo de contratación: de acuerdo a turno.
Genero: masculino
Edad: 25-50 años
Requisitos académicos: Bachiller
Habilidades requeridas: conocimiento en el manejo de armas, capacidad de trabajar bajo presión, buenas relaciones interpersonales, responsable, dinámico y con iniciativa.

Según el señor Luís Alfonso Rivera Marroquín, actualmente jefe de la unidad de custodios, los desafíos que presenta esta unidad es brindar una vigilancia eficiente

que llene las necesidades y expectativas de la institución. Sugiere que debe tomarse en cuenta en el sistema de trabajo de su unidad aspectos como el aumento de personal para cubrir las 10 manzanas que abarca, proveer herramientas adecuadas al personal y la renovación de los limites perimetrales del terreno y de la infraestructura de los edificios como lo son las puertas , ventanas, techos, etc.

Capacitación

Con el objeto de formar un cuerpo de custodios profesional, estos elementos deben recibir cursos especializados, los cuales se imparten dentro de un período determinado en temas como inteligencia, contrainteligencia, seguridad personal, intervenciones y otros aspectos operativos, de análisis estratégico, táctico, psicología, deontología, gestión y administración de seguridad. Con el cual se pretende evaluar y certificar al personal que presta este tipo de servicios, logrando con ello que la universidad cuente con personal más capacitado que satisfaga los estándares de calidad que la comunidad universitaria requiere.

En el año 2000, se propuso impartirles 3 capacitaciones al año (mejoramiento de la seguridad, primeros auxilios y relaciones humanas) las cuales no tuvieron seguimiento. A finales del mismo año, se recibe la primer capacitación, la cual fue impartida por un subinspector de la PNC dicha capacitación fue denominada *"Mejoramiento de Seguridad".*

Y en el año 2003 se realizo una capacitación denominada: *"Autoestima y Trato con las personas",* impartida por una psicóloga del ISSS. Aunque actualmente, no se recibe ya ninguna clase de capacitación para el personal.

Lugar o espacio

La distribución de rondas y sus sectores, están divididos en el campus en zonas que se detallan de la siguiente manera:

- Centro de la Facultad.

- Sector desde Departamento de Física hasta el Auditórium.
- Sector desde Bunker hasta el parqueo.
- Sector cafetería (hoy edificio de Usos Múltiples y Medicina).
- Bosque o Zona Verde.
- Edificio de Agua y Talleres.
- Canchas deportivas y gimnasio.

En la actualidad, la ubicación de su unidad es al norte de la Universidad, entre el Instituto del Agua y los talleres industriales. Donde guardan sus objetos personales en horas laborales y sus herramientas de trabajo.

Equipo

El equipo que les proporciona la institución o con la cuentan es: bastones de madera, esposas, radios de bajo alcance, uniformes y corbos. Pero las necesidades que este personal presenta, es tener un equipo más adecuado como: gas pimienta, esposas de mejor calidad, bastones de goma, uniformes mas adecuados a su trabajo, pitos y sistemas de choques eléctricos, además de la implementación de aparatos modernos que faciliten el desempeño de su trabajo.

Análisis psicológico

La psicología se hace cada vez más necesaria para el estudio, prevención y atención de los distintos problemas de comportamiento que surgen en este nuevo orden mundial. De ahí que la enseñanza de esta disciplina deba responder a una adecuada generación y transmisión del conocimiento, para entender la naturaleza de los distintos procesos que conforman la problemática psicológica en ambientes naturales y con base en ello, planear y prevenir, investigar, detectar e intervenir en los mismos.

El diagnóstico puede ser individual, grupal o institucional. En el diagnóstico individual se busca lograr un conocimiento de los múltiples y complejos aspectos de la personalidad. Se utilizan por lo regular las siguientes técnicas: Historia

clínica, test de inteligencia, test proyectivos, inventarios de personalidad, test de intereses y actividades, entrevistas focalizadas y abiertas, análisis de casos. Este arsenal psicométrico debe ser utilizado con un criterio profesional y un enfoque objetivo ya que si bien proporciona datos relevantes sobre actitudes e inclinaciones de la persona, no ofrece una certidumbre al 100 % acerca del comportamiento futuro en un momento dado.

En el diagnóstico grupal, se busca conocer las características de un grupo especial. Aquí pueden emplearse las siguientes técnicas: test colectivos de inteligencia, test de personalidad, test proyectivos, dinámicas grupales de simulación, dramatización, juego de papeles, etc. La técnica grupal permite desarrollar programas de sensibilización, lo cual permite clarificar valores, actitudes y comportamientos.

En el diagnóstico institucional se intenta conocer las características psicosociales que presenta la institución o la organización. Las técnicas que se utilizan son las entrevistas individuales, test colectivos, análisis de la comunicación organizacional, formal e informal, diagnóstico de procesos organizacionales, del clima organizacional, grupos informales, liderazgo, etc. El profesional de la psicología puede jugar un rol de agente de cambio interviniendo para realizar programas de capacitación dirigidos al personal administrativo y custodios implementando programas de formación de equipos de trabajo, de liderazgo, motivación laboral y de sensibilización en el trato con los demás.

Lo anterior, nos permitirá mejorar significativamente la calidad del servicio en el desempeño de su trato personal, capacidad y profesionalización en su intervención en situaciones de crisis. De igual forma, se pretende hacer más eficiente el tiempo de respuesta de los operadores o custodios los cuales atienden ha estos llamados.

La aplicación de estos tests o pruebas, esta sujeto a las políticas de cada institución, lo cual en este caso, la Universidad solo aplica el test de personalidad, en la entrevista de trabajo que se le elabora al principio a dicho personal. Lo cual no hay ningún seguimiento de pruebas psicometrías, que mantengan al personal en forma óptima, en el momento del uso de armas.

EQUIPO ESPECÍFICO DE ALTO RIESGO

Según el estudio realizado a las diferentes unidades o departamentos, por medio de una encuesta, se tuvieron los resultados presentes:

Equipos de mayor valor por departamentos o unidades:

- Decanato: equipo de informática, y equipo de oficina.
- Biblioteca: Equipos de informática para servidores y centro de Internet, además equipo de oficina y material didáctico (libros, tesis, revistas, etc.)
- Ingeniería y Arquitectura: equipo de informática, equipo topográfico, equipo de talleres.
- Medicina: equipo de laboratorios y computadoras.
- Ciencias Jurídicas: computadoras, equipo de sonido, libros, etc.
- Ciencias Económicas: computadoras, retro proyector de acetatos, cañón, impresoras.
- Biología: Microscopio, equipo informático, equipo de laboratorio como: autoclave, incubadora, centrífuga, etc.
- Idiomas: Equipos de laboratorio de idiomas, equipos computacionales.
- Física: Equipo de centro de cómputo, equipo investigación y laboratorio.
- Química: Equipos de laboratorio como: absorción atómica, balanzas analíticas, espectrofotómetros, equipo de identificación, equipo de cómputo.

- Ciencias Sociales: computadoras, televisor, retro-proyector, libros, radio grabadoras, VHS, etc.
- Recreación y deportes: computadora, trofeos y material deportivos.
- Librería Universitaria: Dinero colectado de las ventas y los Misceláneos.
- Proyección Social: Equipo de cómputo, expedientes, equipos de sonido e instrumentos musicales.
- Impresiones: Fotocopiadoras, duplicador.
- Odontología: rayos X, lámparas de foto curado, módulos.

En forma general se ha detallado el equipo más indispensable para la facultad multidisciplinaria de occidente, sin dejar a un lado que hay en cada unidad o departamento, equipo específico y pequeño de gran valor, tanto monetario como por funcionalidad.

CAPÍTULO III

"DISEÑO DEL SISTEMA DE SEGURIDAD"

DISEÑO FÍSICO DE SEGURIDAD

PUNTOS VULNERABLES Y DE MAYOR RIESGO

Se consideran aquellas donde no existe una infraestructura inadecuada el cual no impide el acceso de intrusos, los puntos que son menos transitados por la comunidad universitaria o tienen menos vigilancia.

Además, se muestran algunos puntos de mayor riesgo, donde existe mayor equipo o material importante de la institución.

A continuación se muestra lo que son los sitios más vulnerables dentro de la facultad:

Fig. 3.1 Acceso peatonal y vehicular de la UES-FMO

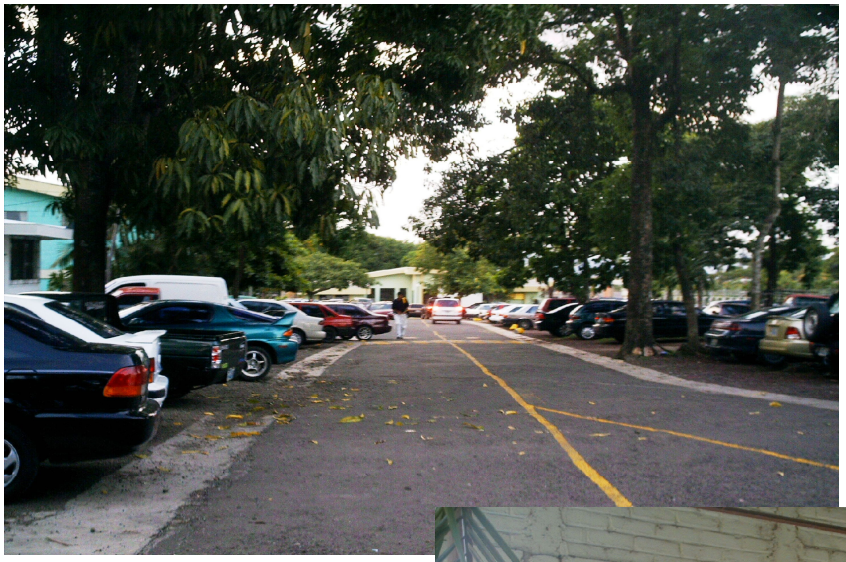


Fig. 3.2 Parqueo

Fig. 3.3 Área Administrativa

Fig. 3.4 Zona de esparcimiento, atrás del aula 10

Fig. 3.5 Pasillos en parte trasera del Edificio Ex - biblioteca

Fig. 3.6 Parte trasera departamento de Física

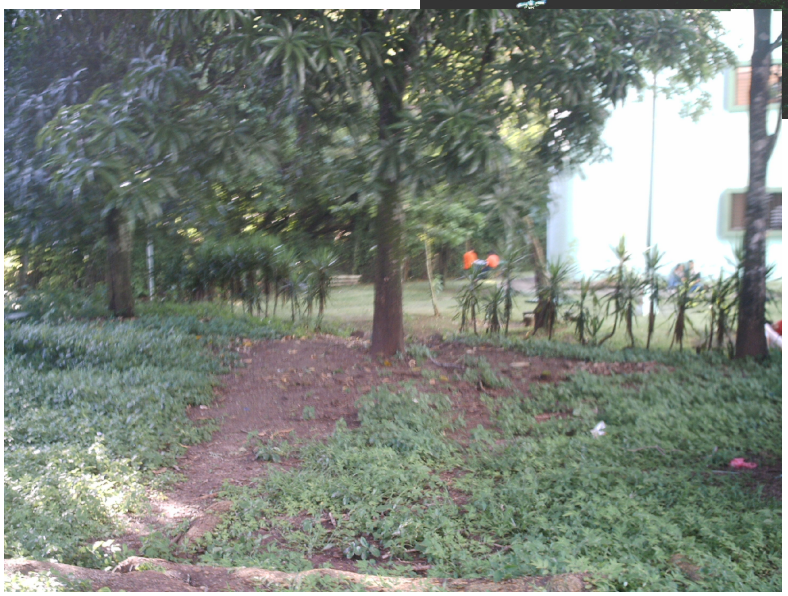


Fig. 3.7 Área verde previa al Bunker

Fig. 3.8 Limite entre Bunker y Edificio de Medicina

Fig. 3.9 Canchas de Fútbol

Fig. 3.10 Cancha de Basketball

Fig. 3.11 Gimnasio, Unidad de Deportes y Recreación

Fig. 3.12 Cafetería de la Facultad

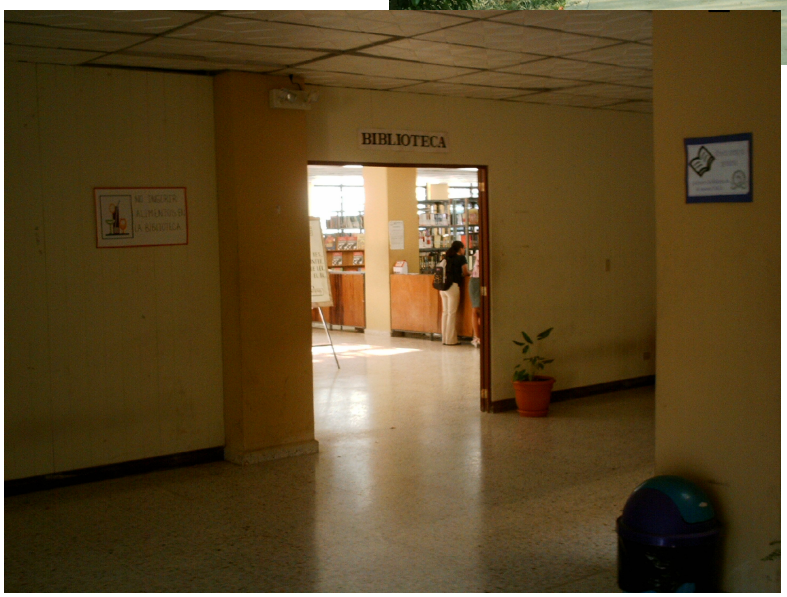


Fig. 3.13 Biblioteca de la Facultad

Fig. 3.14 Tesario y Hemeroteca, Oficina de Biblioteca

Fig. 3.15 Centro de Cómputo de Internet

Fig. 3.16 Decanato de la Facultad y Departamentos del Edificio de Usos Múltiples

ZONAS GEOGRÁFICAS

Considerando las zonas estructurales de la Universidad y según el análisis hecho anteriormente de los puntos más vulnerables y de mayor riesgo, se hace una distribución Geográfica de las zonas donde se pueden instalar los sensores de este proyecto, tomando en cuenta que se discriminan algunas zonas por ser solo aulas de clase o porque se tienen planes de construcción y ampliación en el futuro, además de ser zonas verdes o esparcimiento. Donde no se pueden aplicar el uso de sistemas de intrusión residencial.

Las zonas se dividen de la siguiente manera:

Fig. 3.17Instituto del Agua

Fig. 3.18 Área de Biología

Fig. 3.19 Área Administrativa Académico - Financiera

Fig. 3.20 Edificio Ex - Biblioteca

Fig. 3.21 Edificio de Medicina

Fig. 3.22 Departamento de Derecho

Fig. 3.23 Edificio de Usos Múltiples

DISPOSITIVOS SENSORIALES A UTILIZAR

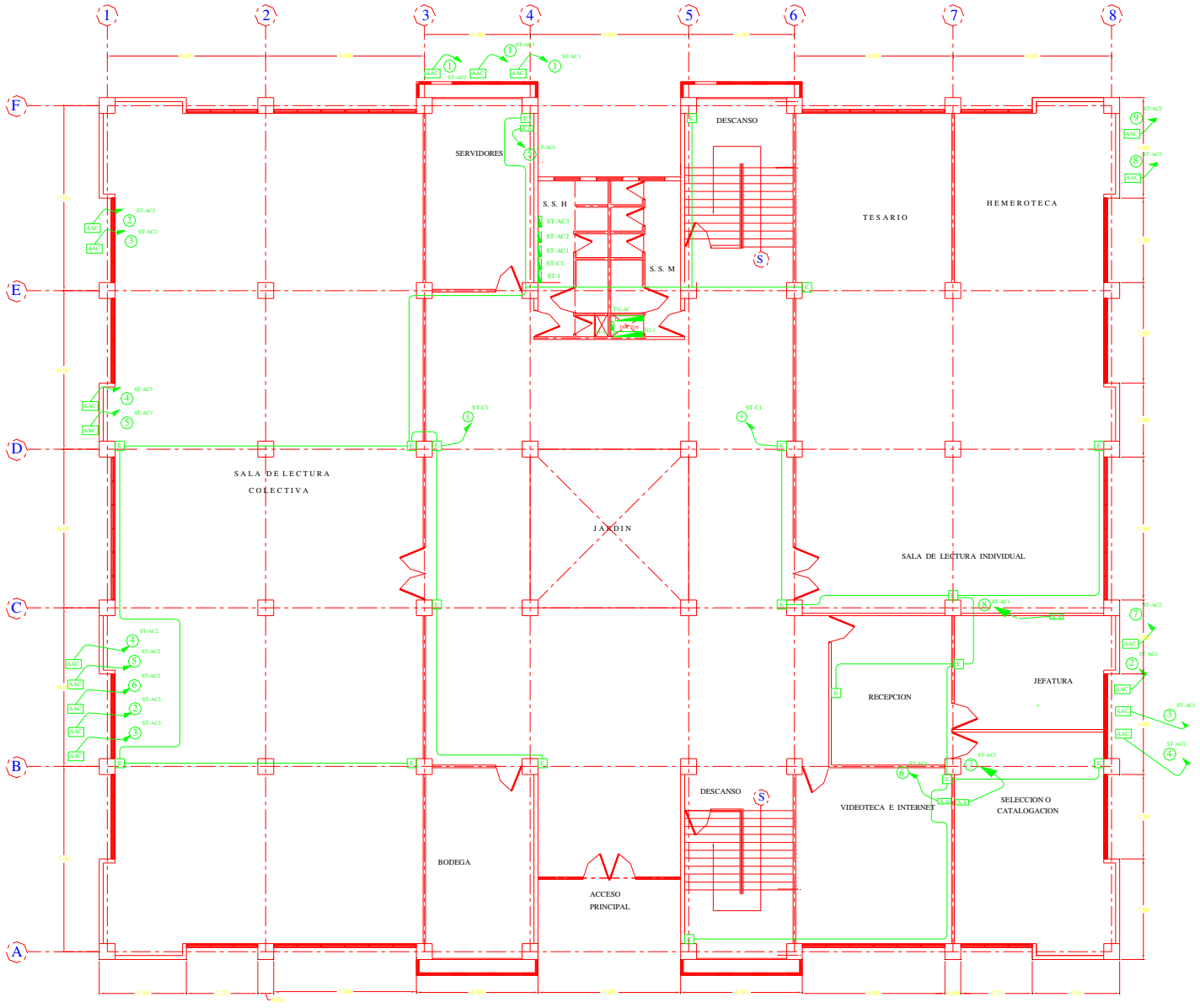
- ✓ Sensores de movimiento SENTROL 6000 SERIES PASSIVE INFRARED. Modelo numero PI 6000
- ✓ Sensores de puerta SCREW TERMINAL 1085T SERIES.
- ✓ Sensores de humo SYSTEM SENSOR. Direct-Wire Photoelectronic, Smoke Detector. Modelo 400 SERIES.
- ✓ Sensores de temperatura Rate-of-Rise/ FiXed temperature Heat Detector 104 SERIES.

RUTA DE CABLEADO

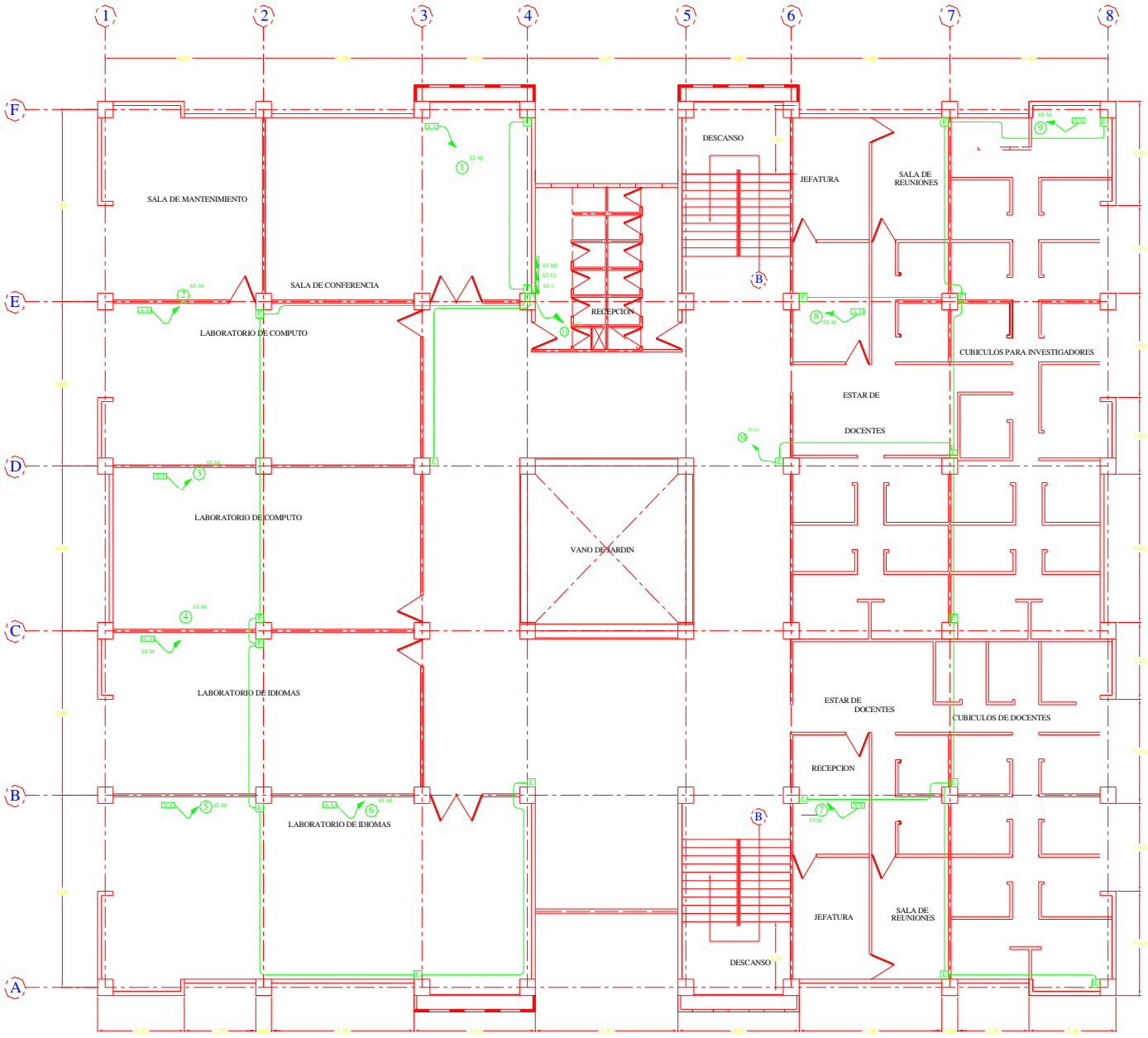
- Cableado eléctrico

EDIFICIO DE USOS MULTIPLES

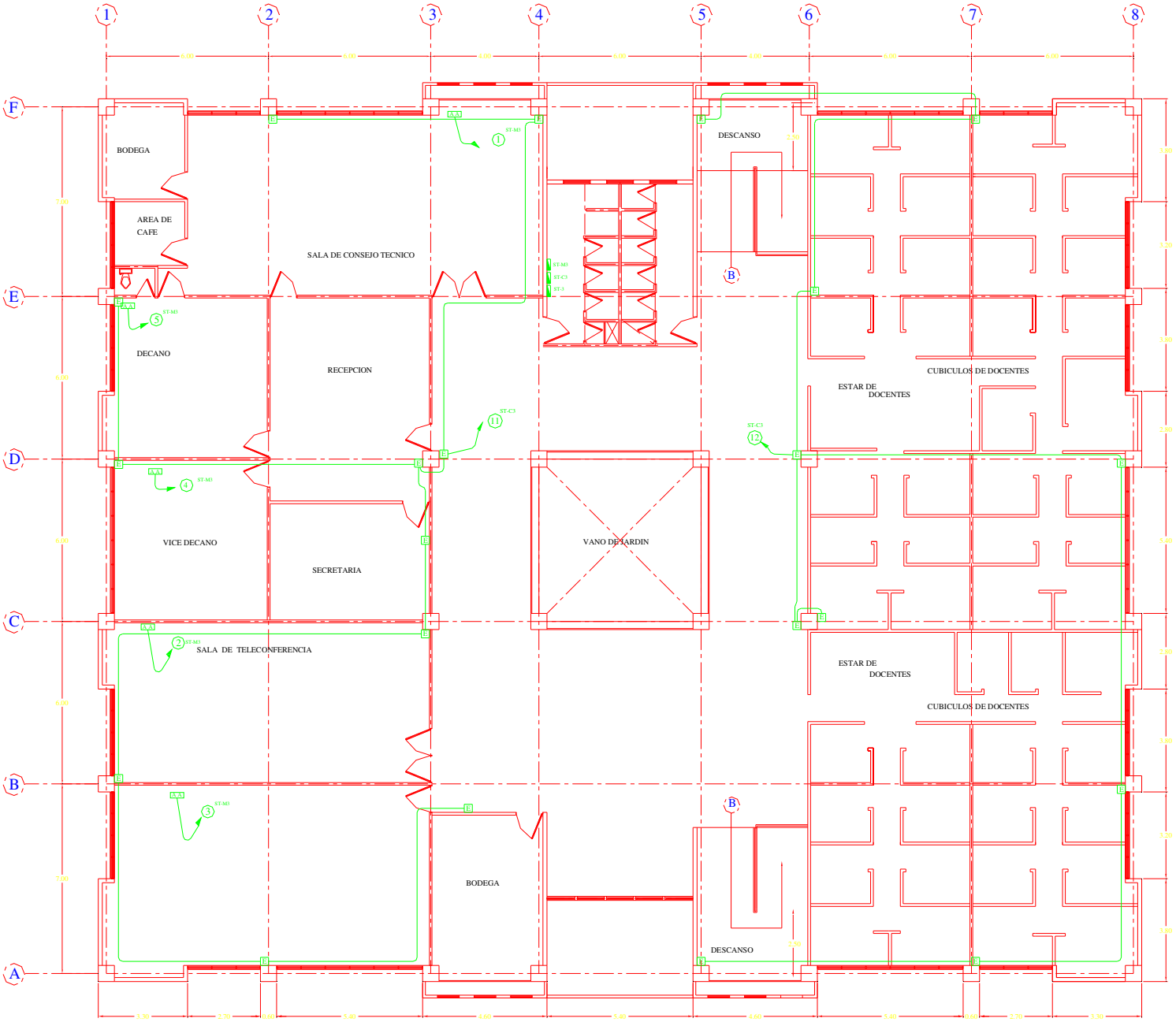
PRIMERA PLANTA



SEGUNDA PLANTA



TERCERA PLANTA



- Cableado de sensores

Referencias:

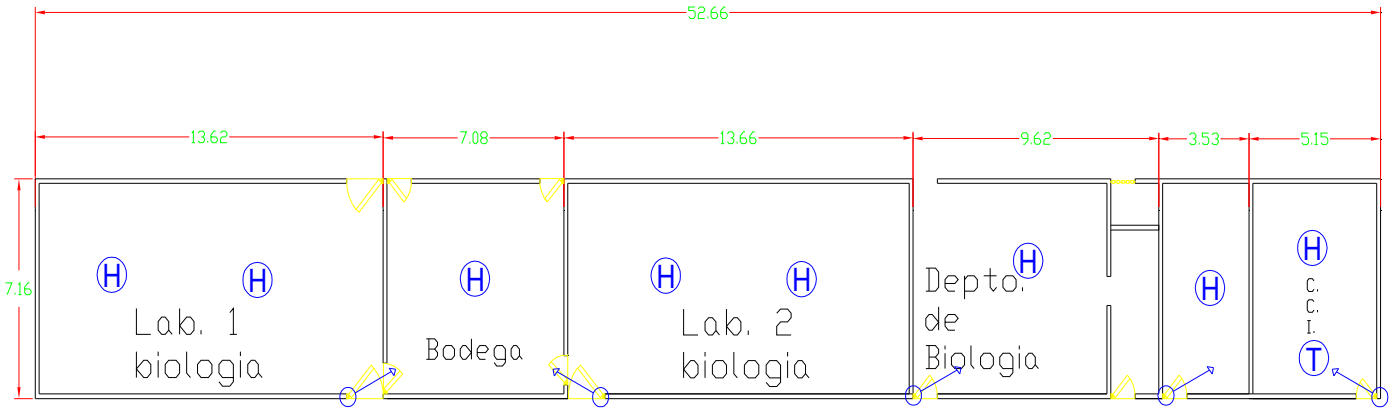
Sensor de Humo

Sensor de Temperatura

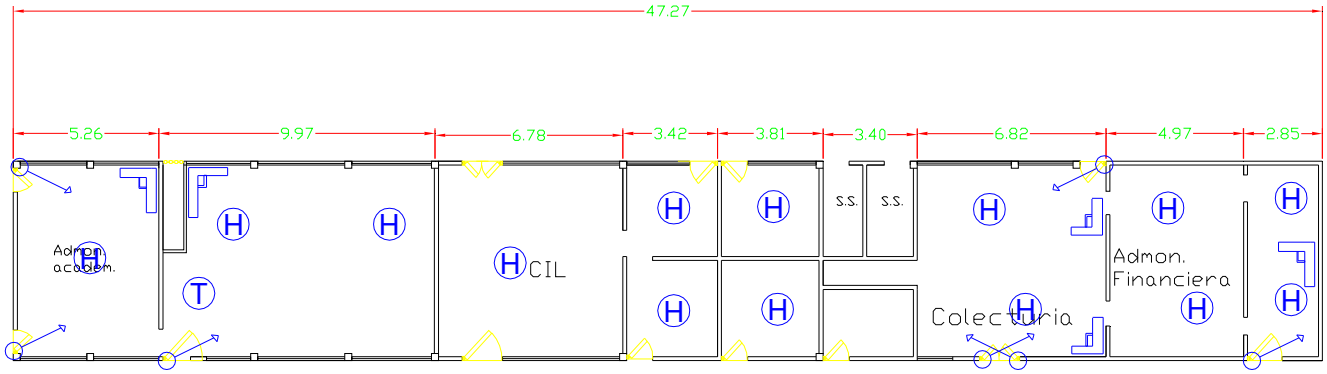
Sensor Magnético de puerta

Sensor de Movimiento

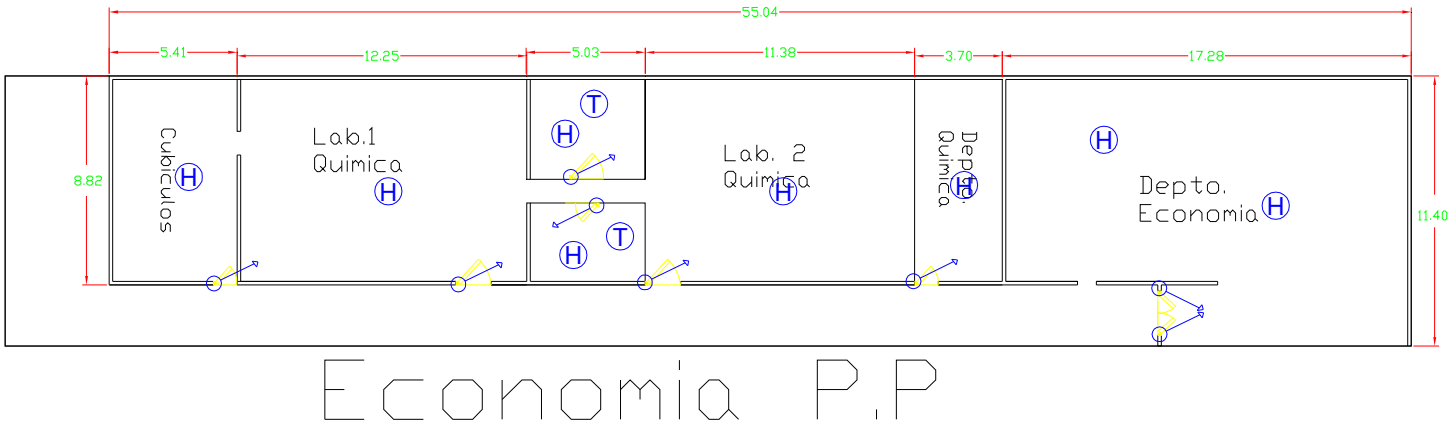
INSTALACIONES DE BIOLOGIA



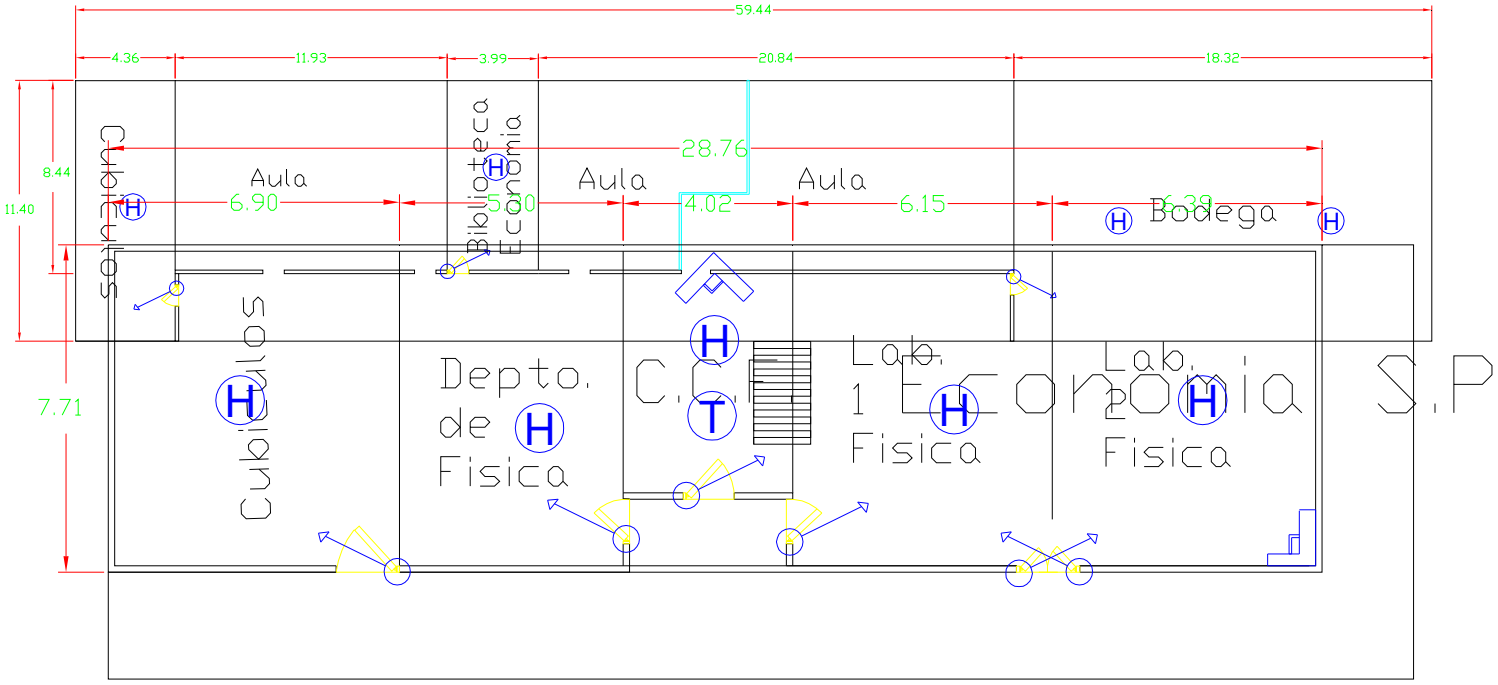
INSTALACIONES ADMINISTRATIVAS



EDIFICIO EXBIBLIOTECA



Economia P.P

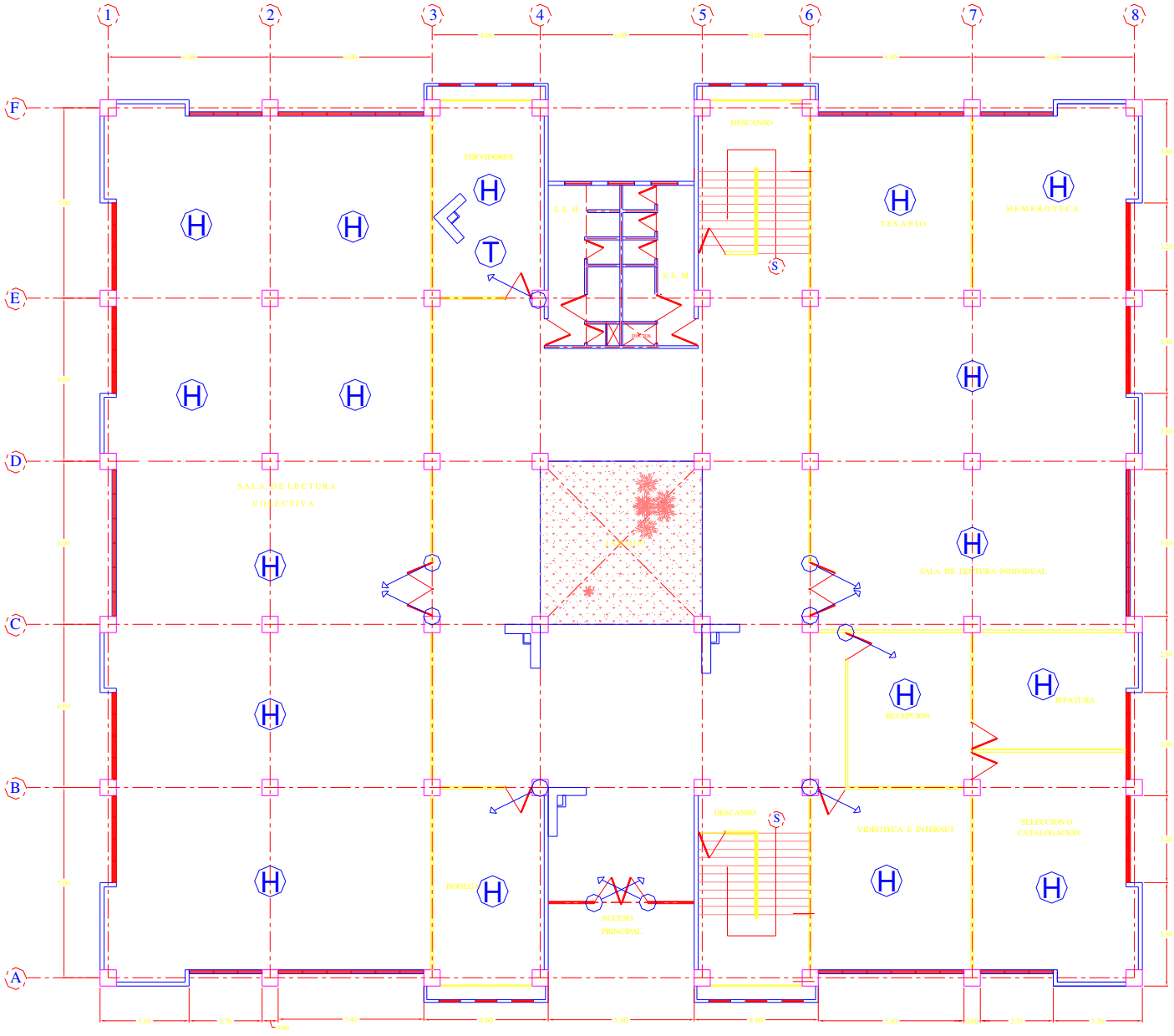


Economia S.P

Fisica

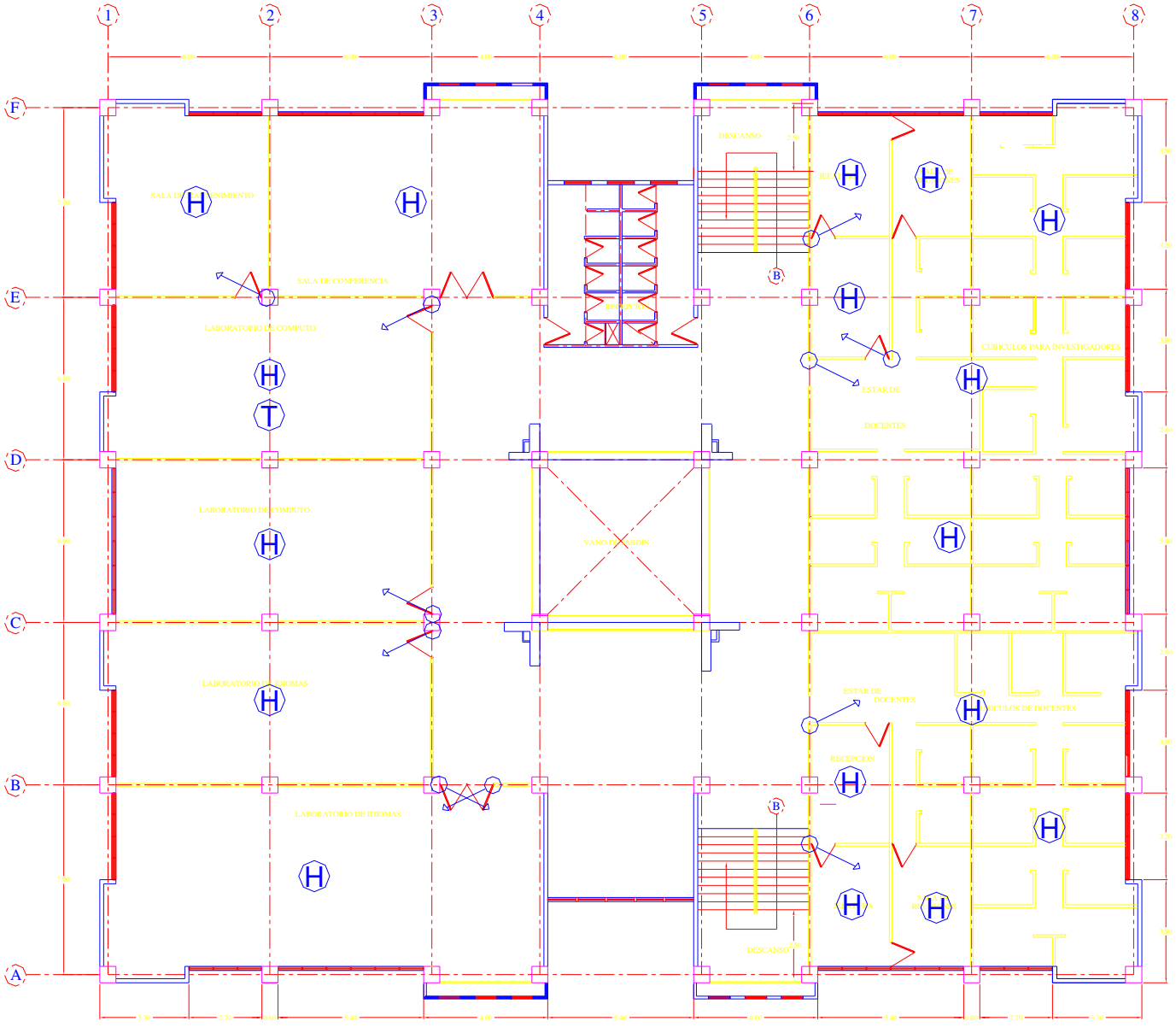
EDIFICIO DE USOS MULTIPLES

PRIMERA PLANTA



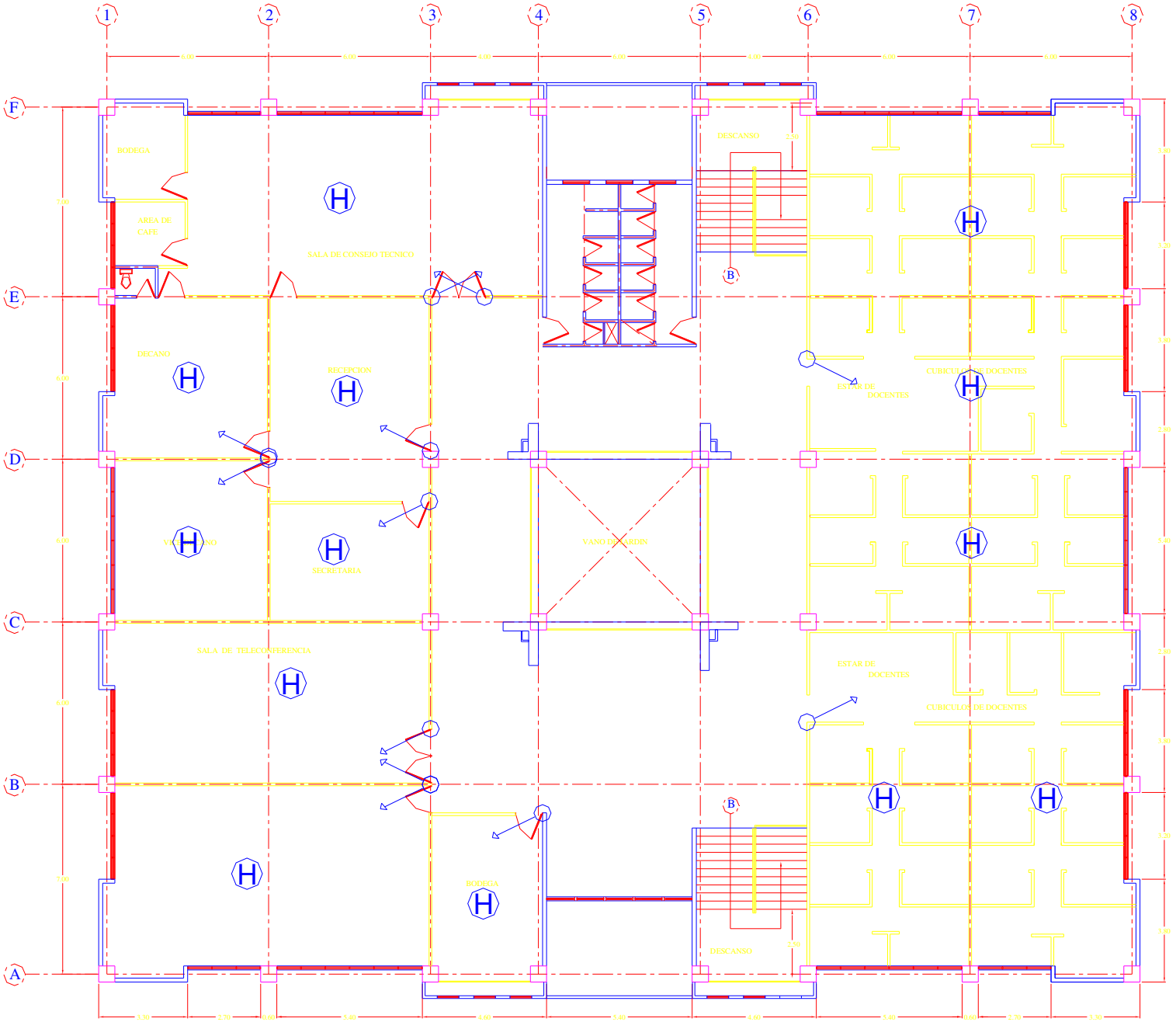
NIVEL 0+0.00

SEGUNDA PLANTA



NIVEL 0+3.60

TERCERA PLANTA



NIVEL 0+7.20

DETERMINACIÓN DE RECURSOS

HUMANO

- ✓ Técnico para la instalación de sensores
- ✓ Especialista para la configuración de sensores en el software
- ✓ Capacitador de usuarios
- ✓ Encargado de mantenimiento de equipo
- ✓ Encargados de utilización de equipo (usuarios o custodios)

EQUIPO Y MATERIALES

Los elementos a tomar de base para la adquisición de los siguientes materiales, es basado en la aplicación y en la cantidad de sensores a utilizar, si dejar de lado la rentabilidad que tendrán conforme pasen los años:

PC

- Pentium IV de 2.33 GHz
- 80 GByte de disco rígido
- 256 Mb de memoria RAM
- Tarjeta de red Intel EthernetExpress(TM) PRO/10+ (PNP Enabled)
- Monitor de 17”
- Accesorios
    - Mouse
    - Teclado
    - Bocinas
    - Impresora
    - Quemador de CD, etc.

- ✓ Software
    - Sistema Operativo Windows XP (Existen otras versiones de sistema operativo que pueden dar soporte a la aplicación, como por ejemplo: WinMe, Windows Server 2003, Windows Vista.)

- Aplicación realizada .net
- Licencia de SQL para la base de datos

✓ Total de 174 Sensores

- 86 de humo
- 7 de temperatura
- 19 de movimiento
- 62 de puerta

✓ Cableado y otros

- Cable Vendel calibre 22
- Baterías de respaldo
- Cable eléctrico
- Dispositivos de transmisión de datos e/s
- 3 Alarmas o sirenas
- Teclado de activación de alarmas por edificio

PROPUESTA DE INVERSIÓN

Al realizar el presupuesto sea hecho lo más apegado a la realidad. Por lo tanto cabe destacar que debido al comportamiento que tienen los dispositivos informáticos en el mercado, tienden a subir o bajar de manera continua, este presupuesto puede estar sujeto a cambios dependiendo del tiempo en que se tarde el implementarlo. Este, esta basado en los precios actuales del mercado y se estima que los precios pueden mantenerse alrededor de 6 meses a partir de la fecha de realización.

FACTIBILIDAD

OPERATIVA

Al hacer el estudio operativo, se logra visualizar la eficiencia y eficacia de las operaciones o funciones que llevara acabo el sistema de vigilancia, por medio de los recursos con que cuenta la institución.

Dicho sistema viene a ser un elemento integral, que no afecta al recurso humano de la institución, sino que es un complemento, que ayudará a la productividad a la unidad de custodios en la realización de su trabajo, haciéndola mas ágil en la realización de su desempeño.

La funcionalidad de este prototipo radica en el enfoque o aprovechamiento de los recursos y herramientas con que cuenta la FMO. Ya que se cuenta con una infraestructura moderna, distribución eléctrica y de redes informáticas, capaz de permitir la incorporación de diferentes servicios o tecnologías. Por otra parte, dicho proyecto puede implementarse de forma modular ya que el diseño de dicho sistema se ha realizado por zonas, permitiendo ejecutarse por etapas, el cual no afectara la funcionalidad del mismo.

La necesidad y deseo de un cambio en el sistema de seguridad actual, expresada por los custodios y el personal involucrado con el mismo, llevó a la conclusión que un sistema moderno es muy aceptable, además que de una manera más sencilla y amigable, cubra todos sus requerimientos, expectativas y proporciona la información en forma oportuna y confiable. Basándose en las entrevistas y conversaciones sostenidas con el personal involucrado se demostró que estos no representan ninguna oposición al cambio, por lo que el sistema es factible operacional mente.

TÉCNICA

La Factibilidad Técnica consiste en realizar una evaluación de la tecnología existente en la institución, este estudio esta destinado a recolectar información sobre los componentes técnicos que posee la organización y la posibilidad de hacer uso de los mismos en el desarrollo e implementación del sistema propuesto y de ser necesario, los requerimientos tecnológicos que deben ser adquiridos para el desarrollo y puesta en marcha del sistema en cuestión.
De acuerdo a la tecnología necesaria para la aplicación del prototipo de seguridad de la Universidad de El Salvador de la FMO, se evaluó bajo dos enfoques: Hardware y Software.

Hardware.

En cuanto a Hardware, específicamente el servidor donde debe estar instalado el sistema propuesto, este debe cubrir con los siguientes requerimientos mínimos:

- ✓ Procesador Pentium IV 2.33 Mhz (como mínimo).
- ✓ 512 MB de Memoria RAM
- ✓ Disco Duro de 80 GB
- ✓ Unidad de CD-ROM
- ✓ Tarjeta de Red
- ✓ Tarjeta de Vídeo
- ✓ Monitor (SVGA)
- ✓ Teclado

- ✓ Mouse
- ✓ Unidad de Protección UPS
- ✓ Puerto Paralelo bidireccional
- ✓ Bocinas

Evaluando el hardware existente y tomando en cuenta la configuración mínima necesaria, la Institución no requerirá realizar inversión inicial para la adquisición de nuevas PC´s, ni tampoco para repotenciar o actualizar los equipos existentes, ya que los mismos satisfacen los requerimientos establecidos tanto para el desarrollo y puesta en funcionamiento del prototipo, como también para instalación futura de un sistema completo y el funcionamiento de la cantidad total de sensores estipulados anteriormente. Además hay que agregar que estos componentes se encuentran en el mercado actualmente a unos precios bajos.

Por características físicas de la Red de Investigación de la Universidad, cada Centro o departamento debe contar con una nueva red interna; que permita la interconexión de todos los sensores, ya que se necesita una red mas segura y aislada para futuros escalamientos de instalación de mas sensores dentro de las instalaciones de la Institución, donde la construcción de estructuras o edificios nuevos incluyen mayor espacio, y por lo tanto mayores demandas de sensores internos a utilizar.

Todas las estaciones de trabajo están conectadas al servidor a través de una red de topología estrella, utilizando cable par trenzado “UTP”, de la categoría número Seis (6), según las normas internacionales del Instituto de Ingenieros Eléctricos y Electrónicos “IEEE”. Lo cual es muy óptimo en la instalación del sistema en algún servidor, como en otros lugares donde podrán acceder en red al sistema.

El servidor cumple las funciones del puerto de enlace y el resto de la red interna de la Universidad y por ende Internet.

Esta configuración permite que el o los equipos instalados para la unidad de custodios (usuario final), puedan interactuar con el sistema. Además cualquier persona que tenga una conexión a Internet, puede desde cualquier punto acceder al servicio de monitorio de los sensores, que el sistema ofrece a los usuarios.

Software

En cuanto al software, la Institución cuenta con todas las aplicaciones que se emplean para el desarrollo del proyecto y funcionamiento del sistema, lo cual no amerita inversión alguna para la adquisición de los mismos. Las estaciones de trabajo, operan bajo ambiente Windows, el servidor requiere el sistema operativo Linux y/o Windows, y como plataforma de desarrollo del software Windows. Para el uso general de las estaciones en actividades diversas se debe poseer las herramientas de escritorio y los navegadores que existen en el mercado actualmente. Así como también se posee en el departamento de Ingeniería una licencia para el uso de Visual Studio. Net

Como resultado de este estudio técnico se determinó que en los actuales momentos, la Institución posee la infraestructura tecnológica (Hardware y Software) necesaria para el desarrollo y puesta en funcionamiento del sistema propuesto, a excepción de la red física para interconexión de sensores que se cuantificaran sus costos a continuación.

ECONÓMICA

Se determina los recursos para desarrollar, implantar, y mantener en operación el proyecto estudiado, haciendo una evaluación donde se puso de manifiesto el equilibrio existente entre los costos intrínsecos del sistema y los beneficios que se derivaron de éste, lo cual permitió observar de una manera más precisa las bondades del proyecto propuesto.

Análisis Costos-Beneficios

Este análisis permite hacer una comparación entre la relación costos del proyecto, y los beneficios que tendría el proyecto, conociendo de antemano los beneficios que la ciencia de la Informática ofrece.

A continuación se presenta un resumen de los costos intrínsecos del proyecto propuesto y una lista de los costos que conlleva implantar el mismo, y los costos de operación. Luego a través de un análisis de valor se determinaron los beneficios que no necesariamente para el nuevo sistema son monetarios o cuantificables.

Costos de equipos y accesorios:

Al momento de adquirir el equipo se ha analizado la calidad de los mismos, los cuales cumplen con los requisitos deseados y los objetivos del diseño sin incurrir en gastos demasiados elevados.

A continuación se detallan los precios y especificaciones:

PC ($0.00)

- Pentium IV de 2.33 GHz
- 80 GByte de disco rígido
- 512 Mb de memoria RAM
- Tarjeta de red Intel EthernetExpress(TM) PRO/10+ (PnP Enabled)
- Indispensable debe de llevar un puerto paralelo bidireccional
- Bocinas

(Requerimientos mínimos)

Software ($0.00)

- Sistema Operativo Windows XP (Ya existe)
- Aplicación realizada en Visual Basic.net $400.00 (solo para efectos de determinación de recursos financieros del proyecto)

- Licencia SQL Express, que se utilizara en este proyecto es gratis, ya que viene incluido en VS.NET

Sensores ($3,424.00)

•	86 de humo	$30.00	$ 2580.00
•	7 de temperatura	$35.00	$245.00
•	19 de movimiento	$25.00	$124.00
•	62 de puerta	$2.00	$475.00
Total	174		$3424.00

Cableado y otros ($ 1,500.00)

- Cable Vendel calibre 22, de 2 o de 4 hilos
- Tubería flexible y canaletas
- Costos de Instalación o mano de obra.
- Dispositivos de transmisión de datos de entrada (caja negra) $35.00

(Puede variar en la cantidad de sensores a conectar)

Capacitación de empleados ($80.00)

- 4 sesiones de 5 horas semanales $20 c/sesión,
- Recursos: manual del usuario, computador con software instalado, otros suplementos y equipo multimedia (proyector de cañón y/o pizarra)

Mantenimiento de equipo ($ 800.00)

- Mantenimiento para los sensores (chequeo de empalmes y conexiones, prueba de elementos, limpieza de sensores, 2 veces al año, $400.00 cada chequeo)
- Mantenimiento de PC correctivo y preventivo (preventivo cada 6 meses y correctivo cada vez que se encuentren errores o problemas de hardware.

El costo total de la instalación es de $2,380.00, que incluye: tendido del cable de telefónico, accesorios, programación y entrenamiento para usuarios, lo que refleja poco más del 41% del presupuesto para realizar el proyecto.

Recogiendo los totales proporcionados en los ítems anteriores se tiene que la inversión inicial viene dada por el detalle siguiente:

Costo de Equipo	$3,424.00
Costo de Instalación	$2,380.00
Total	**$ 5,804.00**

Beneficios Tangibles

Son ventajas que se pueden medir, y que se acreditan a la institución mediante el uso del sistema. Aunque la medición no siempre es fácil, actualmente los beneficios tangibles se pueden medir en términos de ahorros en dólares, recursos o tiempo.

Los beneficios tangibles aportados por el proyecto propuesto están dados por los siguientes aspectos:

- Minimización de nivel de riesgo, en robos y hurtos de equipo informático y didáctico en la FMO $ 3,500.00 anuales en promedio.
- Ahorro en la inversión de esta tecnología, que también es flexible para cambios futuros.
- Optimizar las actividades, aumentando la productividad del personal que labora en el mismo en el ámbito general.
- La flexibilidad al manejar gran volumen y diversidad de información con rapidez, oportunidad y precisión, lo que ofrece una mejor herramienta de trabajo al personal, que facilitará sus labores.
- Mayor y mejor aprovechamiento de los recursos tecnológicos instalados. Capacidad de registrar y almacenar “automáticamente” datos de los

registros y eventos, lo que implica un aumento de la capacidad y seguridad de almacenamiento de registros.

- Rapidez en la respuesta de activación de un dispositivo sensorial.
- Reducción de costos en papelería.
- Limitación del espacio físico interno, en las diferentes rondas o inspecciones de seguridad en los diferentes edificios o infraestructura que cuente con el sistema de seguridad de sensores. Ahorro de tiempo para los custodios que realizan rondas en los diferentes turnos, especialmente los nocturnos.

Beneficios Intangibles

Entre los beneficios intangibles del sistema propuesto se pueden incluir:

- Generación de reportes con información más eficiente y confiable, que sirva de apoyo a la toma de decisiones.
- Funcionalidad optima en el desempeño de las operaciones de seguridad.
- Prestigio en el uso de dispositivos tecnológicos innovadores, para ayudar en el desempeño de las labores de la unidad de custodios. Ya que no existe por el momento en ninguna institución publica, que emplee dicha tecnología.
- Para futuras generaciones de estudiantes se brindara un mejor bienestar y resguardo del patrimonio intelectual.
- Incrementar la satisfacción del trabajo que desempeña la unidad de custodios de la FMO, eliminando procesos tediosos que poseen actualmente.

Análisis Costo-Beneficio

- El Análisis Costo-Beneficio presenta grandes ventajas para la Organización, ya que la misma cuenta con los recursos técnicos necesarios (hardware y software) para el desarrollo e implantación del nuevo sistema.

- De igual manera, el nuevo sistema trae mejoras significativas para el normal desenvolvimiento de las actividades dentro del sistema de seguridad, reduciendo de esta manera el tiempo de procesamiento y generación de la información, de eventos que generen la activación de sensores, ya que la velocidad de procesamiento, veracidad y confiabilidad de los procesos y resultados serán los deseados.

- Es muy importante destacar que en esta nueva era de la informática, mejor conocida como la "Era de la Información", este recurso es la herramienta de competitividad más utilizada por las organizaciones, y en cualquier caso, tenerla al alcance y en forma oportuna, podría significar ahorro, tanto de tiempo como de dinero. Además debe tomarse en cuenta el valor que la información tiene en los actuales momentos, siendo el punto de apoyo en el proceso de la toma de decisiones.

- Con la puesta en marcha de este proyecto se logrará optimizar los procesos que involucra la gestión de la información, permitiendo obtener una información segura y confiable, dirigida a la consecución de los objetivos y agilizar la toma de decisiones dentro de la Unidad de Custodios, y por ende en toda la Universidad.

- Por otra parte un sistema de seguridad como este, contribuye a aumentar la capacidad, el control, la comunicación, disminuir los costos y obtener una ventaja competitiva.

Análisis de la Razón beneficio/costo

Este método de análisis B/C esta basado en la razón de los beneficios a los costos asociados con este proyecto en particular. Se considera que un proyecto es atractivo cuando los beneficios derivados de su implementación y reducidos por los beneficios negativos esperados excede sus costos asociados.

Beneficios (B): Ventajas experimentadas por la institución.
Beneficios negativos (BN): Desventajas para la institución cuando el proyecto es implementado.
Costos (C): Gastos anticipados por el proyecto o inversión, operación, mantenimiento, etc.

Además todos los valores están expresados en las mismas unidades valor presente, valor anual o valor futuro equivalente.

Los beneficios encontrados, están cuantificados en la minimización de riesgos como por ejemplo el hurto de equipo informático y didáctico de los diferentes departamentos, que experimentaron pérdidas en los 5 años anteriores, encontrando un promedio de $3,500.00 anuales, lo cual consideramos que serán los beneficios positivos de este proyecto.

Los beneficios negativos serán considerados como cero, ya que existe un fondo para proyectos especiales en la institución.

Calculo de la razón B/C

Los costos totales lo forman los costos de mantenimiento y operación (M&O), y la inversión inicial del proyecto (VP), pero los siguientes cálculos se estarán trabajando en valor anual, por lo tanto, se necesita obtener el factor A/P (Anualidad dado el valor Presente), por medio de la siguiente formula:

$$VA = VP\ (A/P, i, n)$$

Donde:
VP = $ 5,004.00
i = 43% (tasa de retorno calculada para el sistema de seguridad)
n = 5 años
(A/P, 43%, 5) el factor obtenido es 0.51974

Entonces:

VA = $5,004.00 (A/P, 43%, 5)

VA = $5,004.00 (0.51974)

VA = $ 2,600.78

Por lo tanto:

- **B/C Estándar**

$$B/C = \frac{\text{Beneficios Positivos - Beneficios Negativos}}{\text{Costos}}$$

$$B/C = \frac{\text{Beneficios Positivos – Beneficios Negativos}}{\text{Inversión Inicial + Costos de M\&O}}$$

$$B/C = \frac{\$\ 3{,}500 \quad – \quad 0}{\$\ 2{,}600.78 + \$\ 800}$$

B/C = 1.03

- **B/C Modificado**

$$B/Cm = \frac{\text{Beneficios Positivos – Beneficios Negativos – Costos M\&O}}{\text{Inversión Inicial}}$$

$$B/Cm = \frac{\$\ 3{,}500.00 - \$\ 0.00 - \$\ 800.00}{\$\ 2{,}600.78}$$

B/Cm = 1.04

- **B/C Simple**

Beneficios Totales = Beneficios Positivos

Costos Totales = Inversión Inicial + Costo M&O

B/C = Beneficios Totales – Costos Totales

B/C = $ 3,500.00 – ($ 2,600.78 + $ 800.00)

B/C = $ 3,500.00 - $ 3,400.78

B/C = $ 99.22

Una razón B/C mayor o igual que 1.0 indica que el proyecto evaluado es económicamente ventajoso. Lo que indica una buena rentabilidad para el proyecto, al implementarlo en la FMO.

CAPÍTULO IV

"DISEÑO Y DESARROLLO DEL PROTOTIPO"

DISEÑO DE APLICACIÓN

DESCRIPCIÓN

El software que se producirá será un prototipo de características seleccionadas, el cual consiste en un modelo funcional que incluya algunas, pero no todas, de las características que tendrá el sistema final; para comprobar su desempeño en controlar dichos dispositivos, si bien no será una implantación completa en todo el campus, pero brindará una funcionalidad absoluta en un área particular propia para el prototipo.

Funciones que deberá poseer el prototipo del software:

a. Autentificación de usuarios (Agregar, Modificar o Eliminar Usuarios).
b. Identificar sensor instalado y asignar configuración (ubicación por zona, horario de activación, mantenimiento, casos especiales).
c. Monitoreo de sensores mediante un plano ilustrativo (mapa) de la distribución en las instalaciones de la Universidad.
d. Alerta de activación de sensor o sensores y localización según diagrama o código del sensor.
e. Generación del reporte sobre status de los sensores periódicamente (diario, mensual y anual). El reporte incluiría el custodio que estaba asignado, día y hora que se genero el reporte, cantidad de sensores en funcionamiento, status de los mismos, en caso de activación de alguno de ellos, mostraría la hora en que se activo y el motivo por el cual se activo, cabe mencionar que será registrado por el custodio dicho comentario.
f. Asignación de horarios de vigilancia de los custodios.
g. Conexión remota al estatus de sensores.

Las funciones mencionadas, son las más importantes que se van a desarrollar, pero pueden ampliarse o extenderse, según los criterios o necesidades del cliente o usuario final, adecuándose a las exigencias como horarios o zonas de la

implementación de dichos sensores.

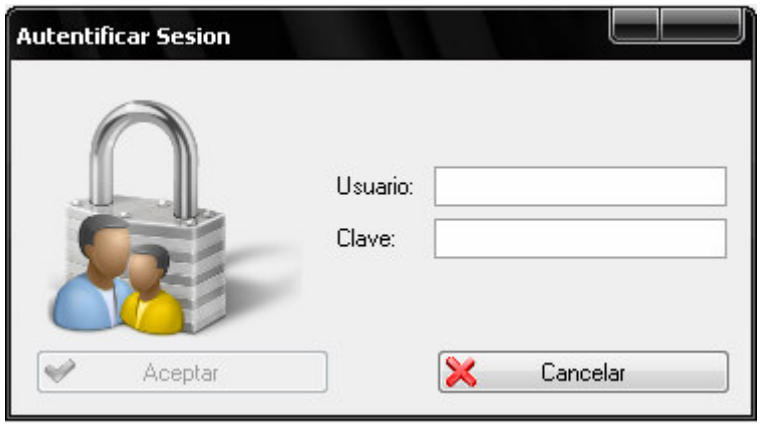
ENTRADAS Y SALIDAS DE INFORMACIÓN

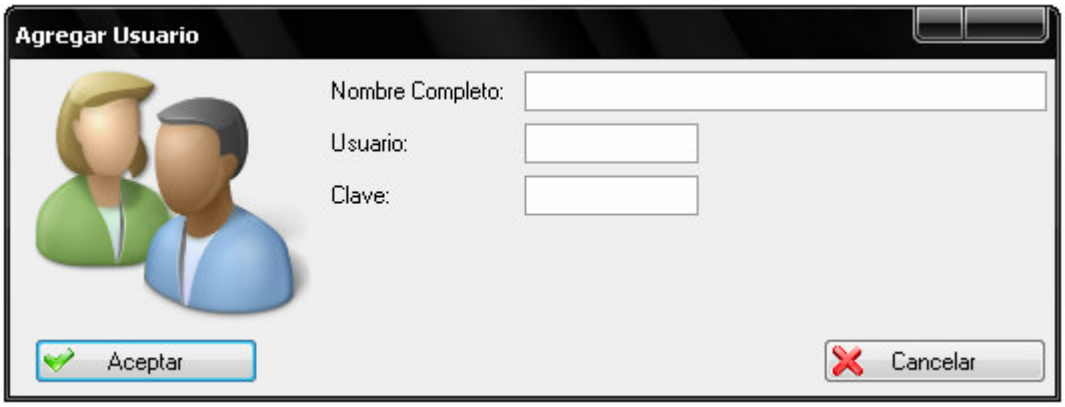
Las entradas de información son los datos que se introducen al sistema, los cuales tienen que ser precisos e indudablemente de calidad, ya que son los que determinan la eficiencia y eficacia del programa o sistema, para este caso se encuentran las entradas por medio del teclado o formularios (pantallas) y también se encuentra el uso de terminales.

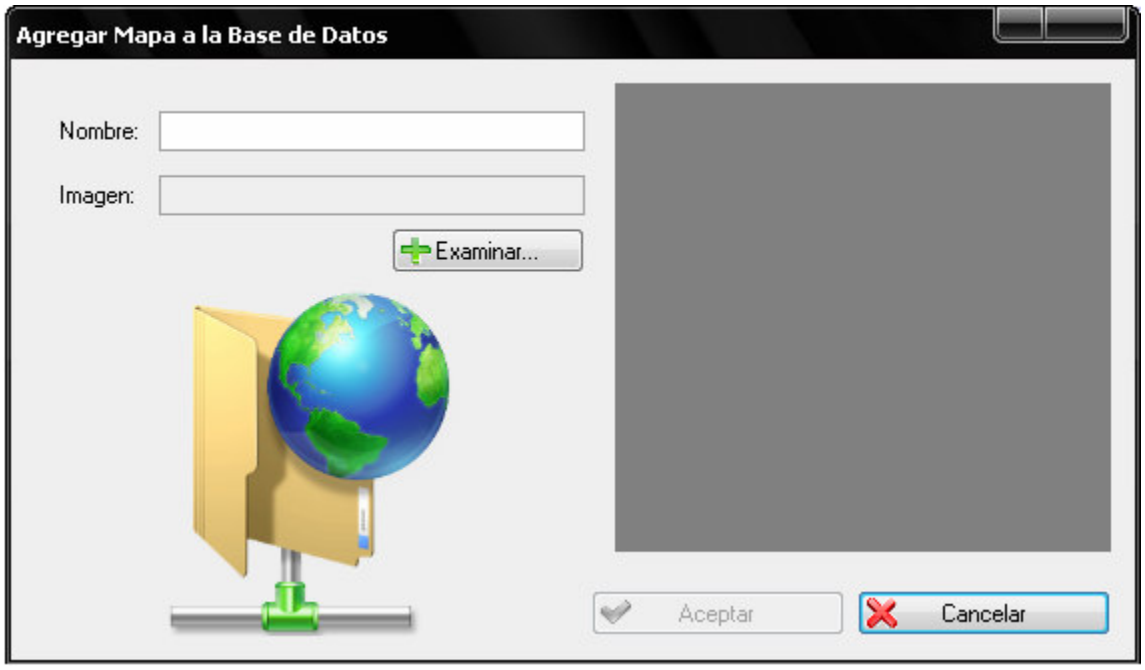
Para el caso de uso de terminales, este se encuentra manejado por el puerto paralelo, el cual emite información al sistema por medio de bits los cuales son generados por cada pin del bus de datos del puerto.

La entrada por teclado o pantallas, es ciertamente el método mas común, es esencial que los datos a introducir al sistema sean validos y coherentes, para que al usuario obtenga lo que necesita.

A continuación se presenta algunas de las pantallas asociadas a la entrada de datos al sistema:







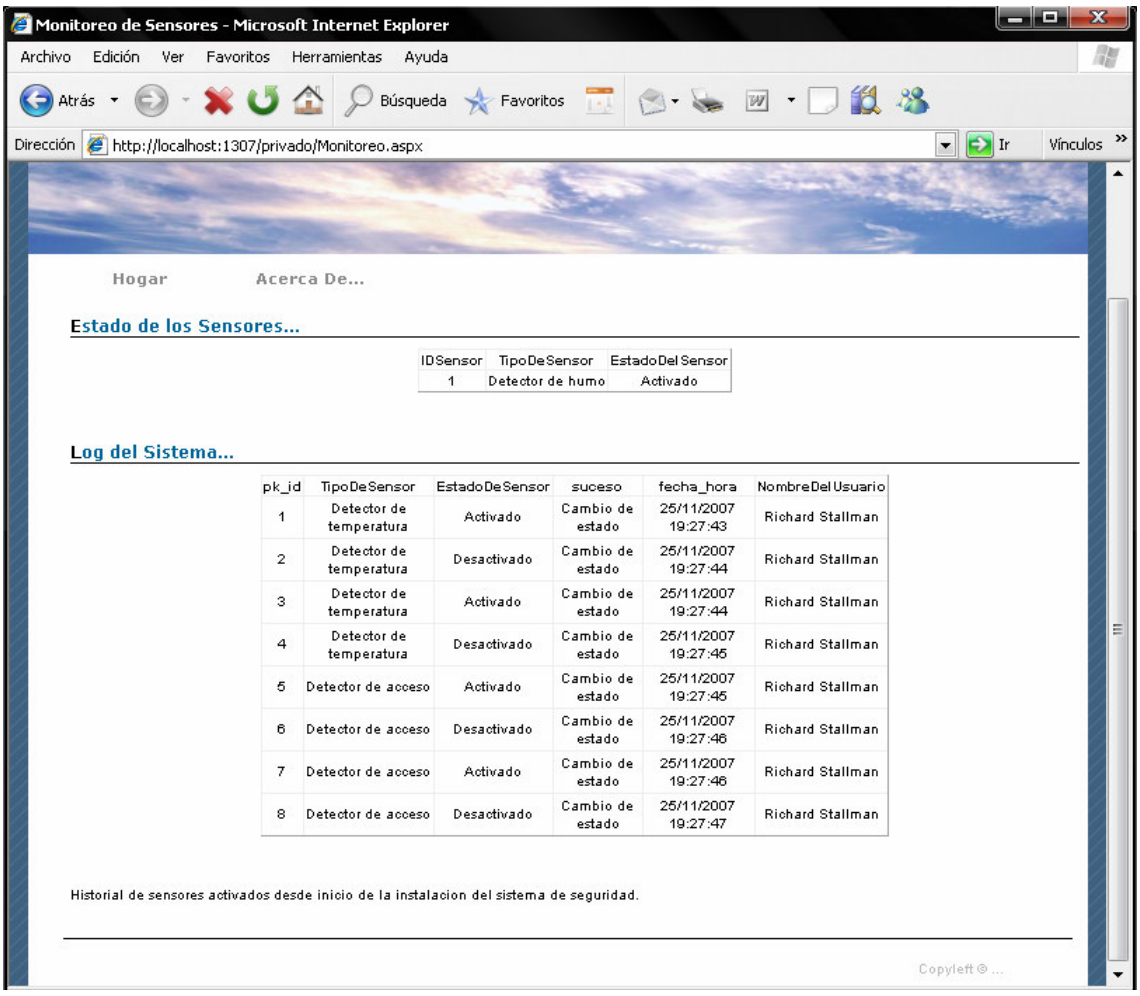
La salida es la información que se entrega a los usuarios, el cual requiere una gran cantidad de procesamiento antes de transformarse en la salida apropiada. En entre los medios de salidas se mencionan los principales:
Las pantallas para el monitoreo, salida de audio, salida impresa y la salida electrónica.

Salidas de audio, esta es precisa para los mensajes transitorios o espontáneos los cuales, permiten al usuario al estar lejos de la pantalla de monitoreo escuchar la emisión de un sonido al momento de activación en este caso de un sensor o el de

varios, este es especial el cual no interfiere con otras tareas, dándole oportunidad al usuario de tener manos libres.

Salida electrónica, es generada en una página Web, es otro tipo de salida entregado mediante la tecnología de actualización automática, que se podrá utilizar dentro o fuera de la institución para que un usuario final analice el estado de los sensores instalados. Dicha difusión de información será filtrada y codificada, para prevenir hurto de información del sistema de seguridad.

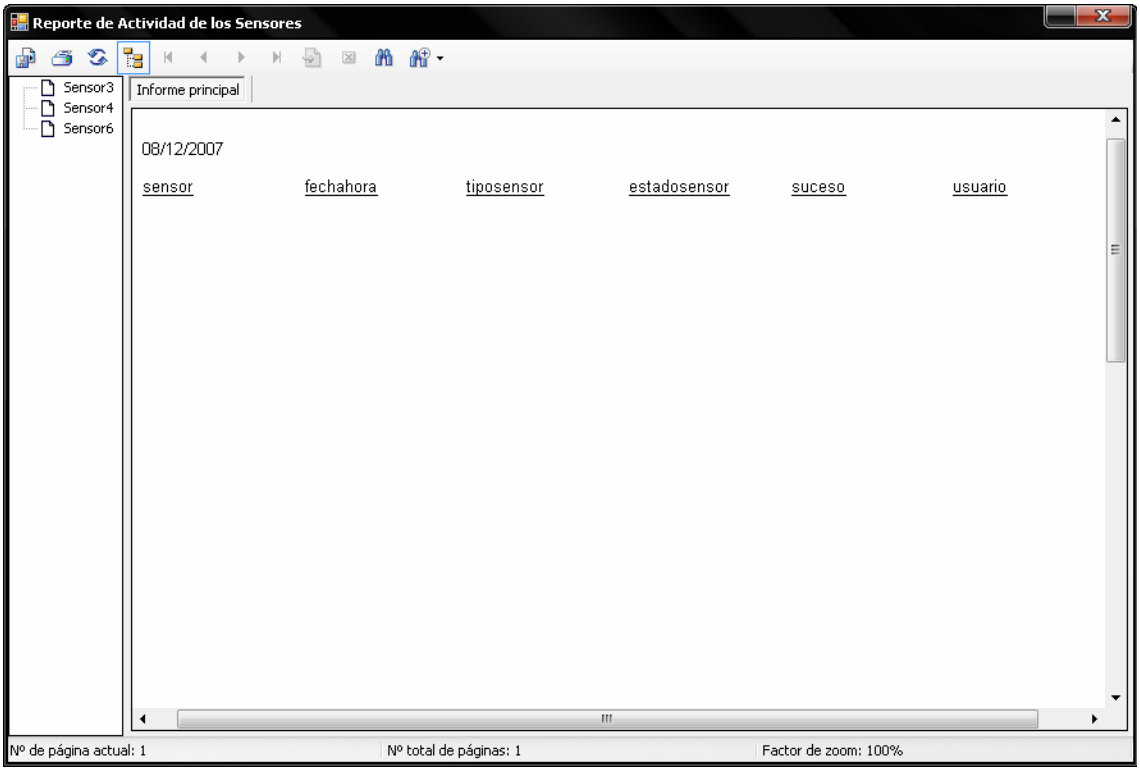
El cual tiene el siguiente aspecto:



Salidas impresas, para esta clase de información impresa o reportes generados del sistema, son información variable ya que es información que puede cambiar cada vez que se imprima el reporte.

Los reportes que se brindan son funcionales al usuario, ya que la información esta organizada de tal manera que proporcione al usuario un análisis detallado y fácil de utilizar para la toma de decisiones.

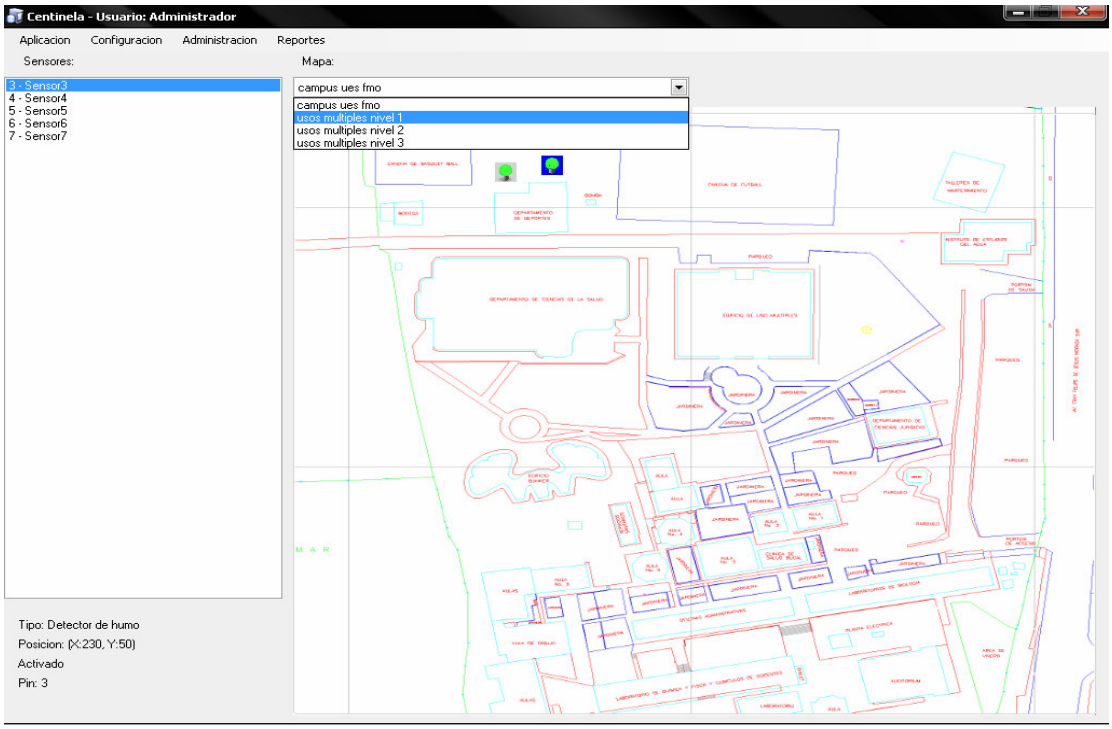
Los reportes que se generaran en el sistema o prototipo son los siguientes:
De actividad de los sensores: diario, mes, año y total. Además por Sensor, asiendo un total de 5 reportes a generar utilizando el mismo esquema o estructura del documento, como se presenta a continuación:



Pantallas, tienen la gran ventaja y el potencial necesario para interactuar con el usuario, esta es flexible ya que puede cambiar su información de salida en tiempo

real y a su vez muestra o despliega los elementos utilizados en la base de datos, lo cual contribuye a la toma de decisiones inmediatas.

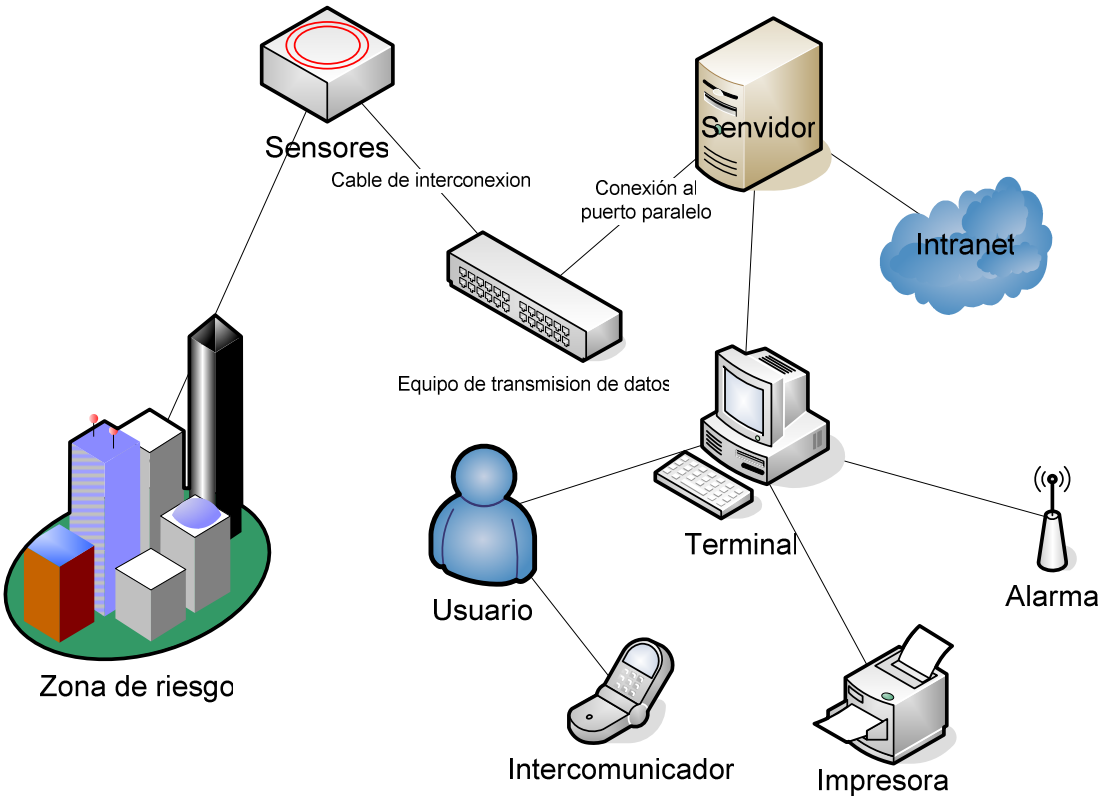
Entre las pantallas principales del sistema se encuentran:



INTERCONEXIÓN DE COMPONENTES

Al visualizar cada uno de los dispositivos y como se relacionan, se logra mejor el objetivo y el desempeño de cada uno de los componentes, para optimizar los resultado del proyecto.

A continuación se ilustra en un plano general los dispositivos o recursos a utilizar en la ejecución de dicho sistema:

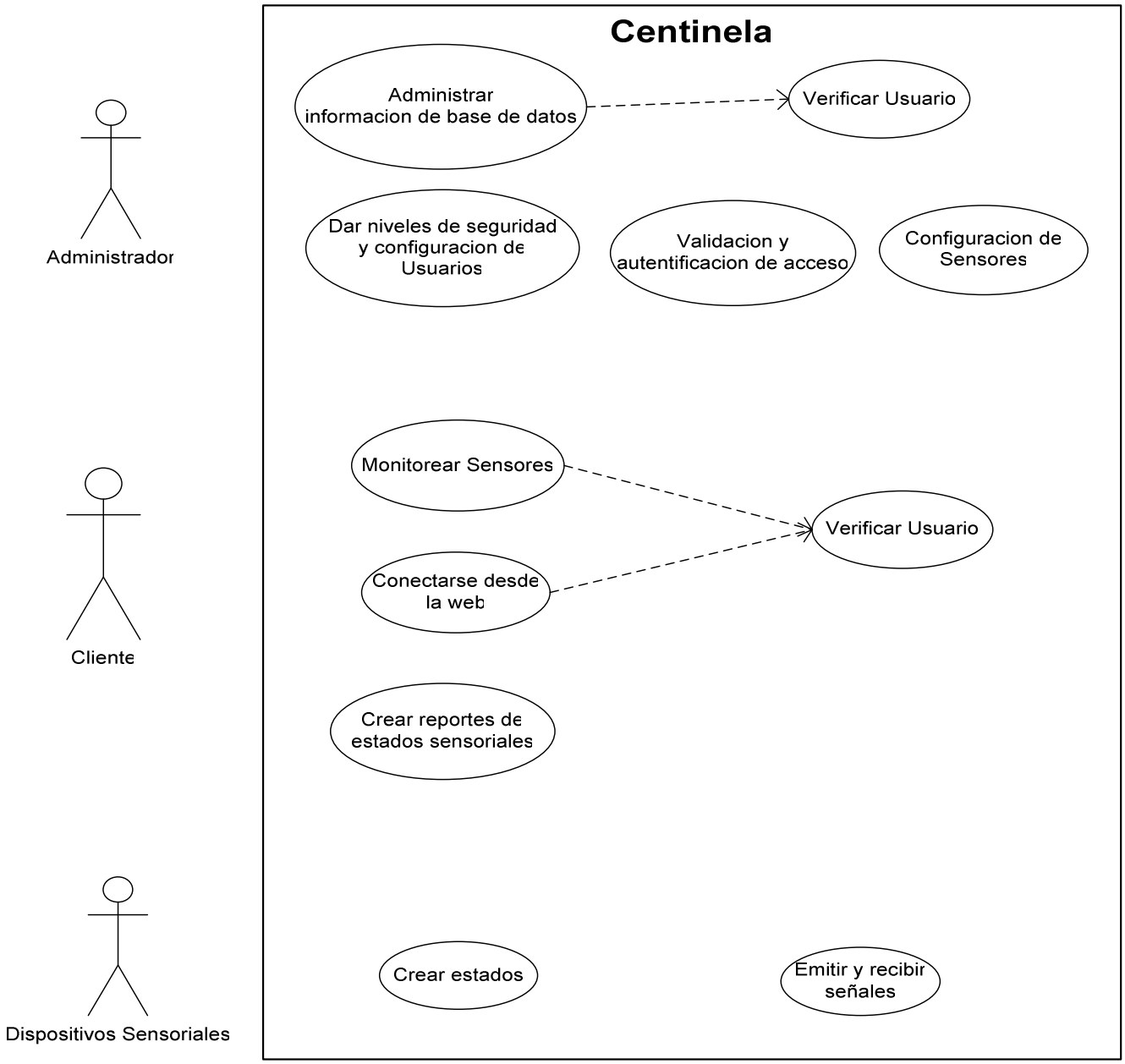

Sensores
Cable de interconexion
Senvidor
Conexión al
puerto paralelo
Intranet
Equipo de transmision de datos
Terminal
Usuario
Alarma
Zona de riesgo
Intercomunicador
Impresora

DISEÑO ORIENTADO A OBJETOS

DIAGRAMAS DE CASOS DE USO

El caso de uso es una estructura para describir la forma en que un sistema lucirá para los usuarios potenciales.

Esta es una poderosa herramienta para obtener los requerimientos funcionales, estos facilitan la comunicación entre los analistas y los usuarios, y entre los analistas y los clientes.

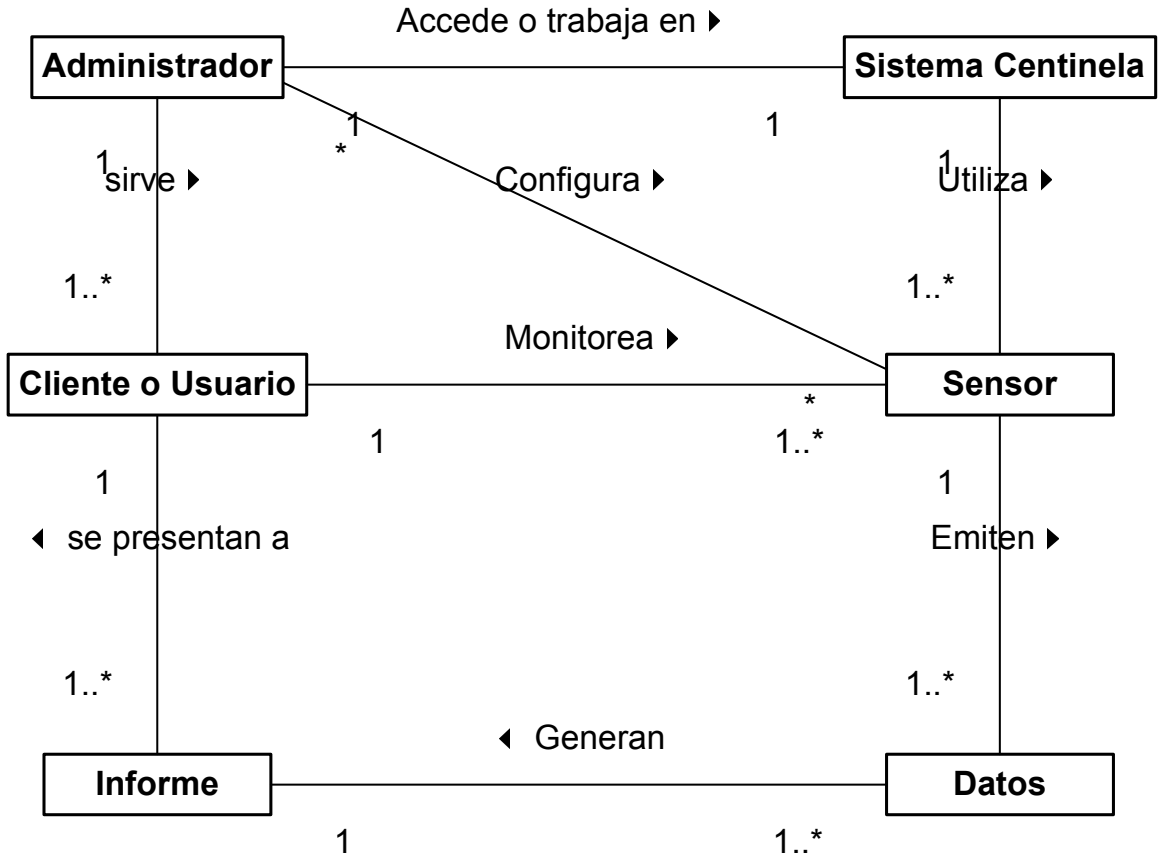


DIAGRAMAS DE CLASES

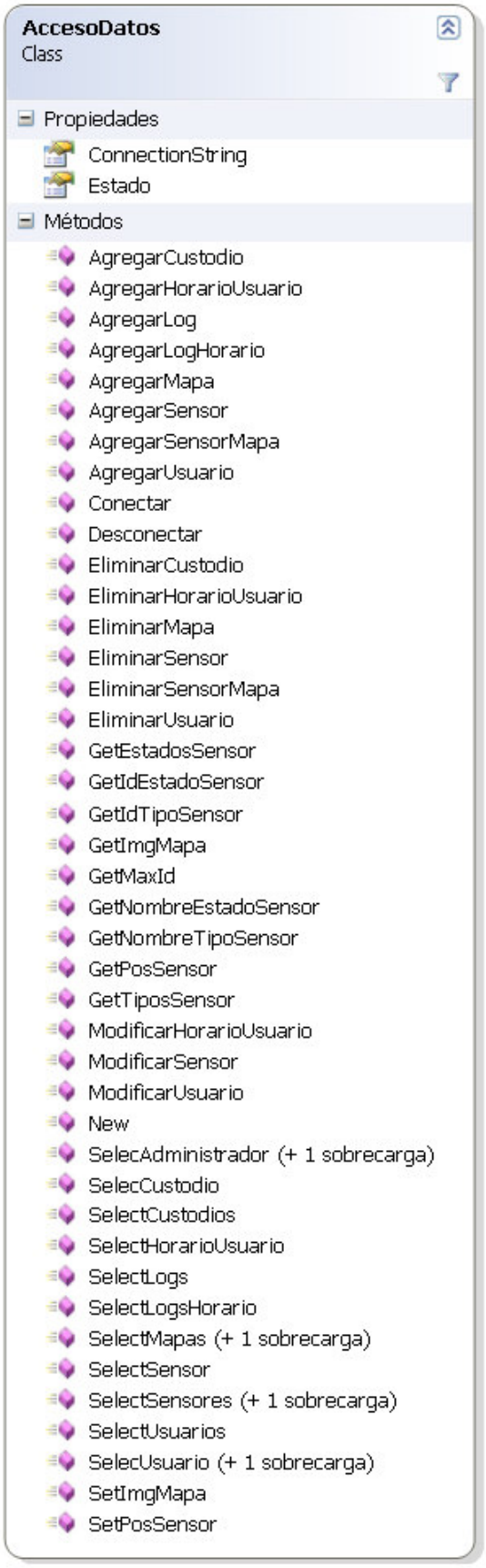
Un diagrama de clases representa un concepto, es decir la abstracción de elementos que confluyen en una categoría.

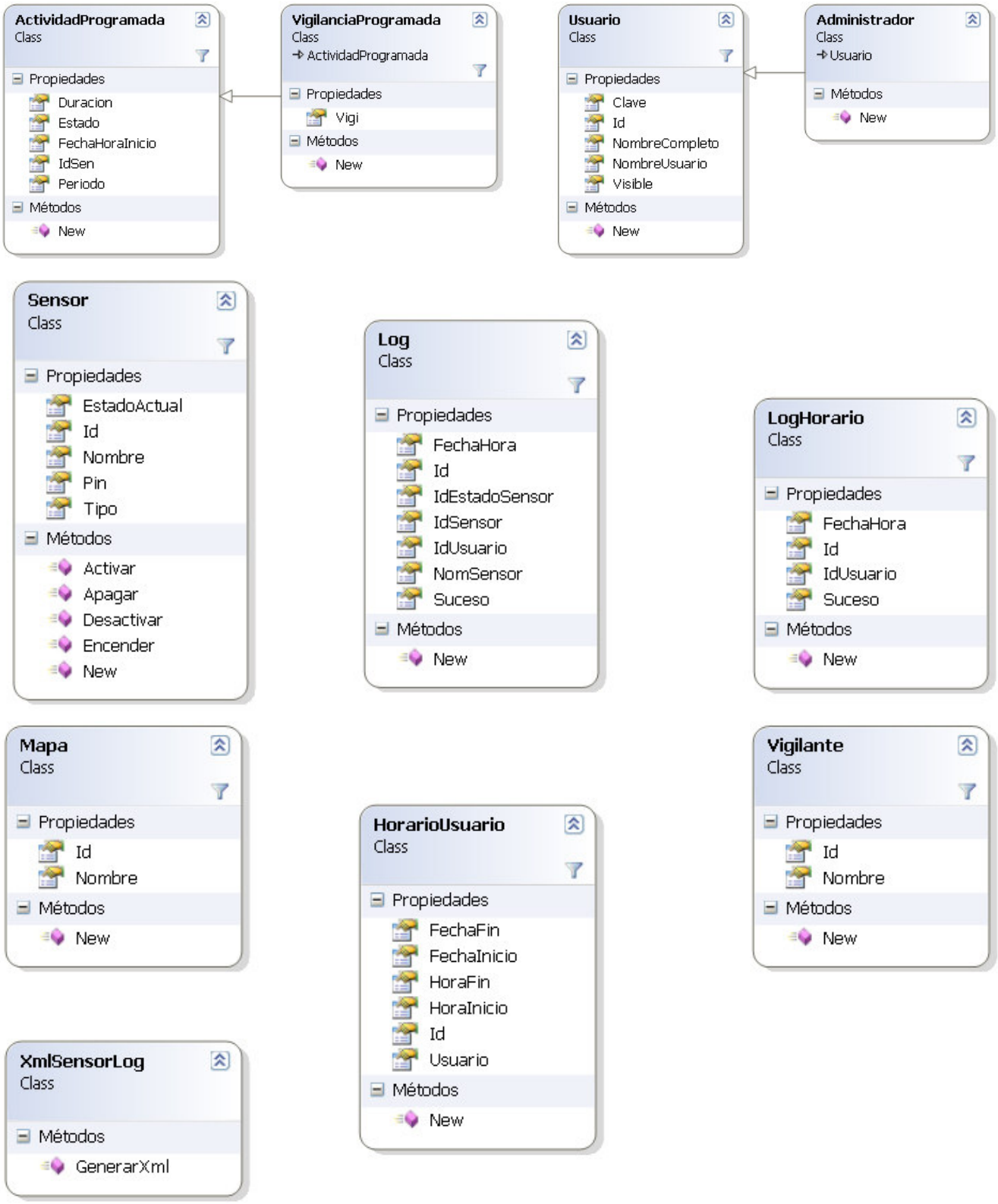
Las clases representan el vocabulario de un área del conocimiento. Ya que el cliente estimula al analista a transformar los verbos en operaciones, con respecto a su área.

El siguiente diagrama de clases representa un modelo relacional o de asociación.



A continuación se demuestra, el diagrama de clases con respecto a sus atributos o propiedades y métodos u operaciones, al igual que las herencias los cuales ameritan.


AccesoDatos
Class
Propiedades
ConnectionString
Estado
Métodos
AgregarCustodio
AgregarHorarioUsuario
AgregarLog
AgregarLogHorario
AgregarMapa
AgregarSensor
AgregarSensorMapa
AgregarUsuario
Conectar
Desconectar
EliminarCustodio
EliminarHorarioUsuario
EliminarMapa
EliminarSensor
EliminarSensorMapa
EliminarUsuario
GetEstadosSensor
GetIdEstadoSensor
GetIdTipoSensor
GetImgMapa
GetMaxId
GetNombreEstadoSensor
GetNombreTipoSensor
GetPosSensor
GetTiposSensor
ModificarHorarioUsuario
ModificarSensor
ModificarUsuario
New
SelecAdministrador (+ 1 sobrecarga)
SelecCustodio
SelectCustodios
SelectHorarioUsuario
SelectLogs
SelectLogsHorario
SelectMapas (+ 1 sobrecarga)
SelectSensor
SelectSensores (+ 1 sobrecarga)
SelectUsuarios
SelecUsuario (+ 1 sobrecarga)
SetImgMapa
SetPosSensor


ActividadProgramada
Class
Propiedades
Duracion
Estado
FechaHoraInicio
IdSen
Periodo
Métodos
New
VigilanciaProgramada
Class
ActividadProgramada
Propiedades
Vigi
Métodos
New
Usuario
Class
Propiedades
Clave
Id
NombreCompleto
NombreUsuario
Visible
Métodos
New
Administrador
Class
Usuario
Métodos
New
Sensor
Class
Propiedades
EstadoActual
Id
Nombre
Pin
Tipo
Métodos
Activar
Apagar
Desactivar
Encender
New
Log
Class
Propiedades
FechaHora
Id
IdEstadoSensor
IdSensor
IdUsuario
NomSensor
Suceso
Métodos
New
LogHorario
Class
Propiedades
FechaHora
Id
IdUsuario
Suceso
Métodos
New
Mapa
Class
Propiedades
Id
Nombre
Métodos
New
HorarioUsuario
Class
Propiedades
FechaFin
FechaInicio
HoraFin
HoraInicio
Id
Usuario
Métodos
New
Vigilante
Class
Propiedades
Id
Nombre
Métodos
New
XmlSensorLog
Class
Métodos
GenerarXml

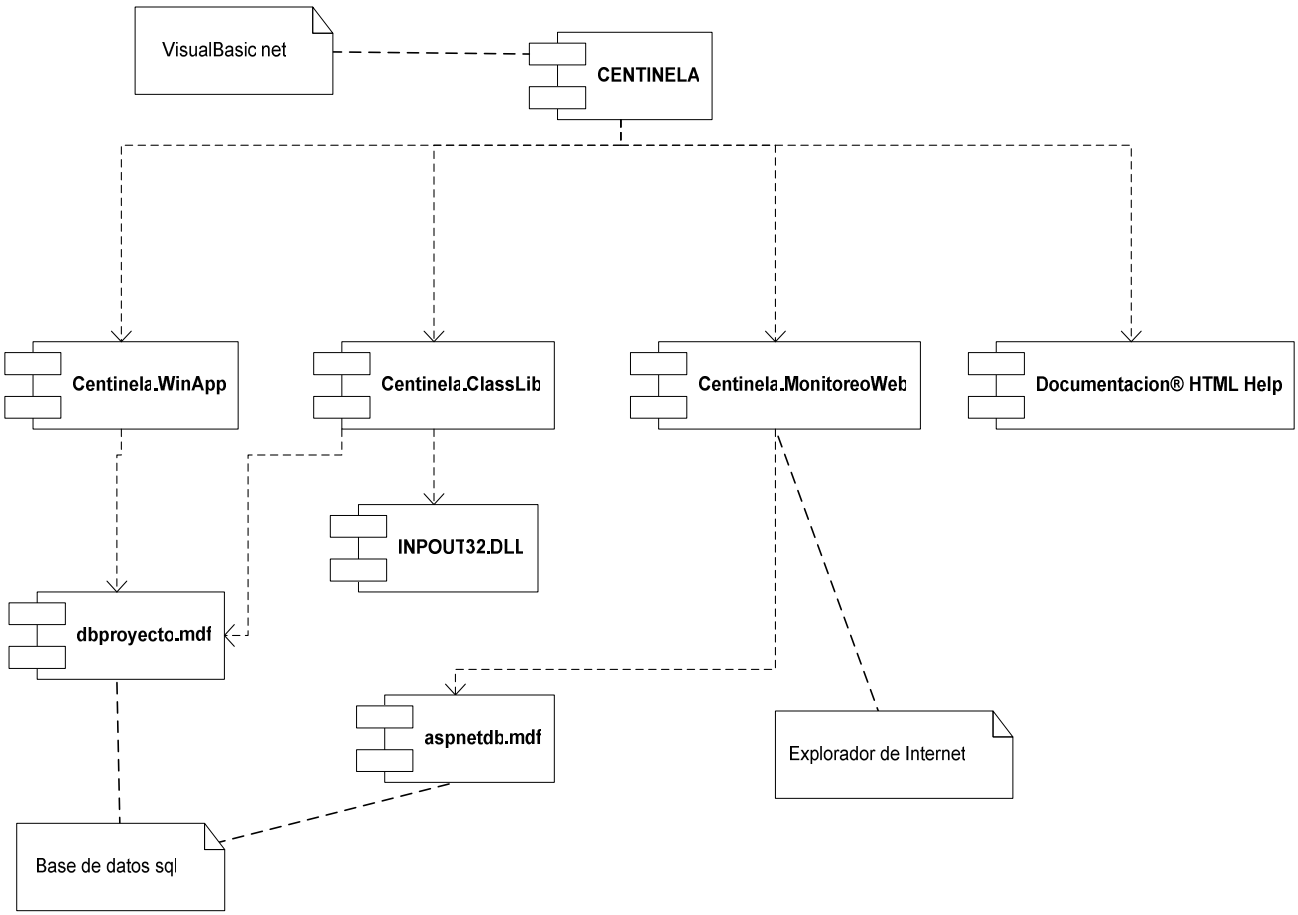
DIAGRAMAS DE COMPONENTES

Un componente de software es una parte física de un sistema, y se encuentra en la computadora, no en la mente del analista.

Hay tres tipos de componentes:

1. Componentes de distribución
2. Componentes para trabajar en el producto
3. Componentes de ejecución

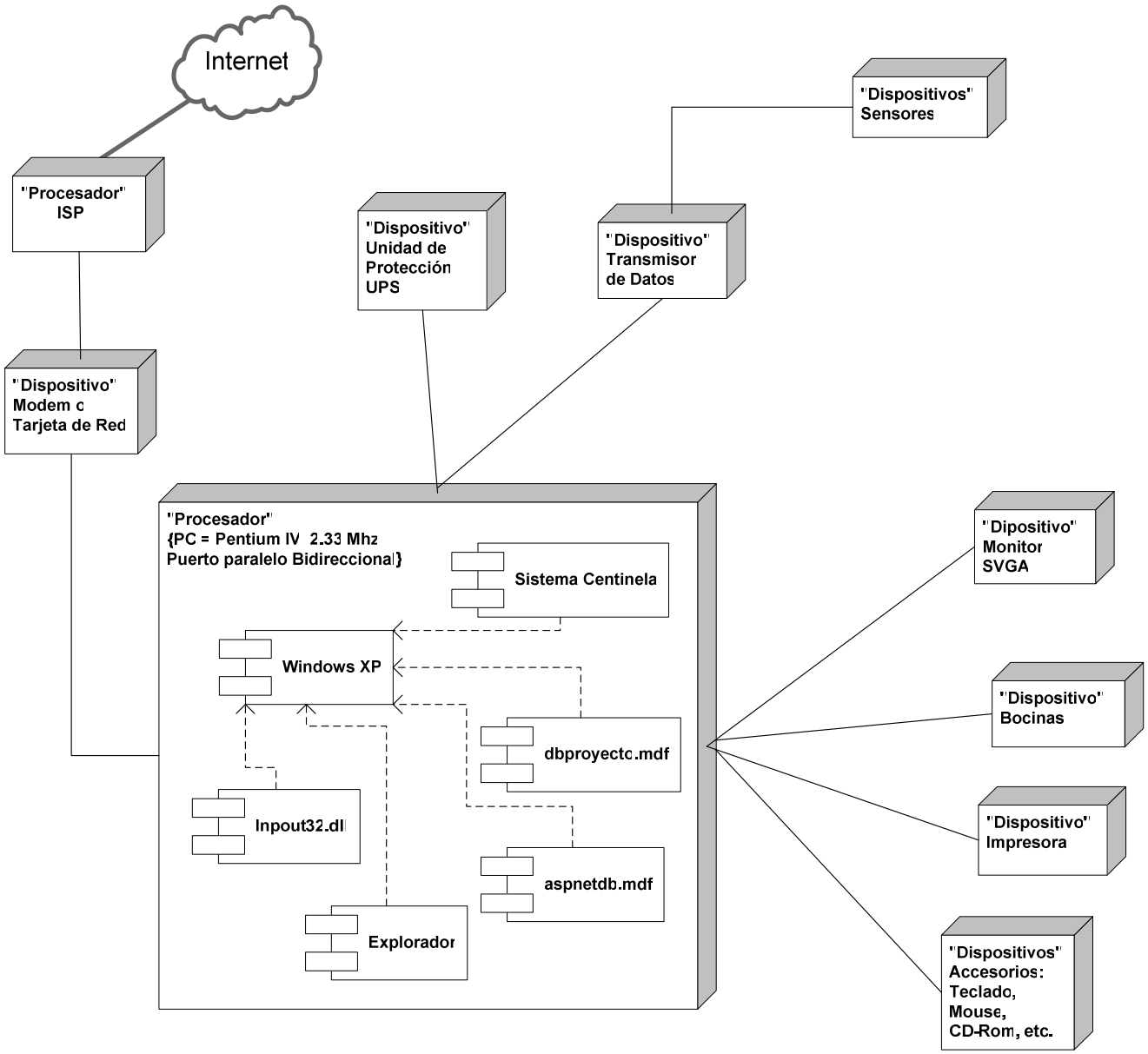
Es decir, en un diagrama de componentes es contenido, obviamente, componentes, interfaces y relaciones.



DIAGRAMAS DE DISTRIBUCIÓN

Claro esta que el software es primordial en un sistema de varios componentes. Pero un diseño sólido de distribución es básico para el diseño del sistema. El cual de una imagen clara de la forma en que debe lucir el hardware final.

El diagrama de distribución ilustra la forma en que luce un sistema físicamente cuando sea conjugado.



DISEÑO DE BASE DE DATOS

DEFINICIÓN

Una base de datos se define como una serie de datos organizados y relacionados entre sí, los cuales son recolectados y explotados por los sistemas de información, ya que son una fuente central de datos destinados a compartirse entre muchos usuarios para una diversidad de aplicaciones. A la base de datos lo constituye el sistema de administración el cual permite la creación, modificación y actualización de la base de datos, la recuperación de datos y la generación de informes y pantallas. Las bases de datos proporcionan la infraestructura requerida para los sistemas de apoyo a la toma de decisiones y para los sistemas de información estratégicos, ya que estos sistemas explotan la información contenida en las bases de datos de la organización para apoyar el proceso de toma de decisiones o para lograr ventajas competitivas. La base de datos creada para este proyecto a sido construida en SQL Express. Cuyos objetivos son la de mantener datos exactos y consistentes; permitir la evolución conforme aumente las necesidades y el tener lo mas actualizada posible la información de los mismos.

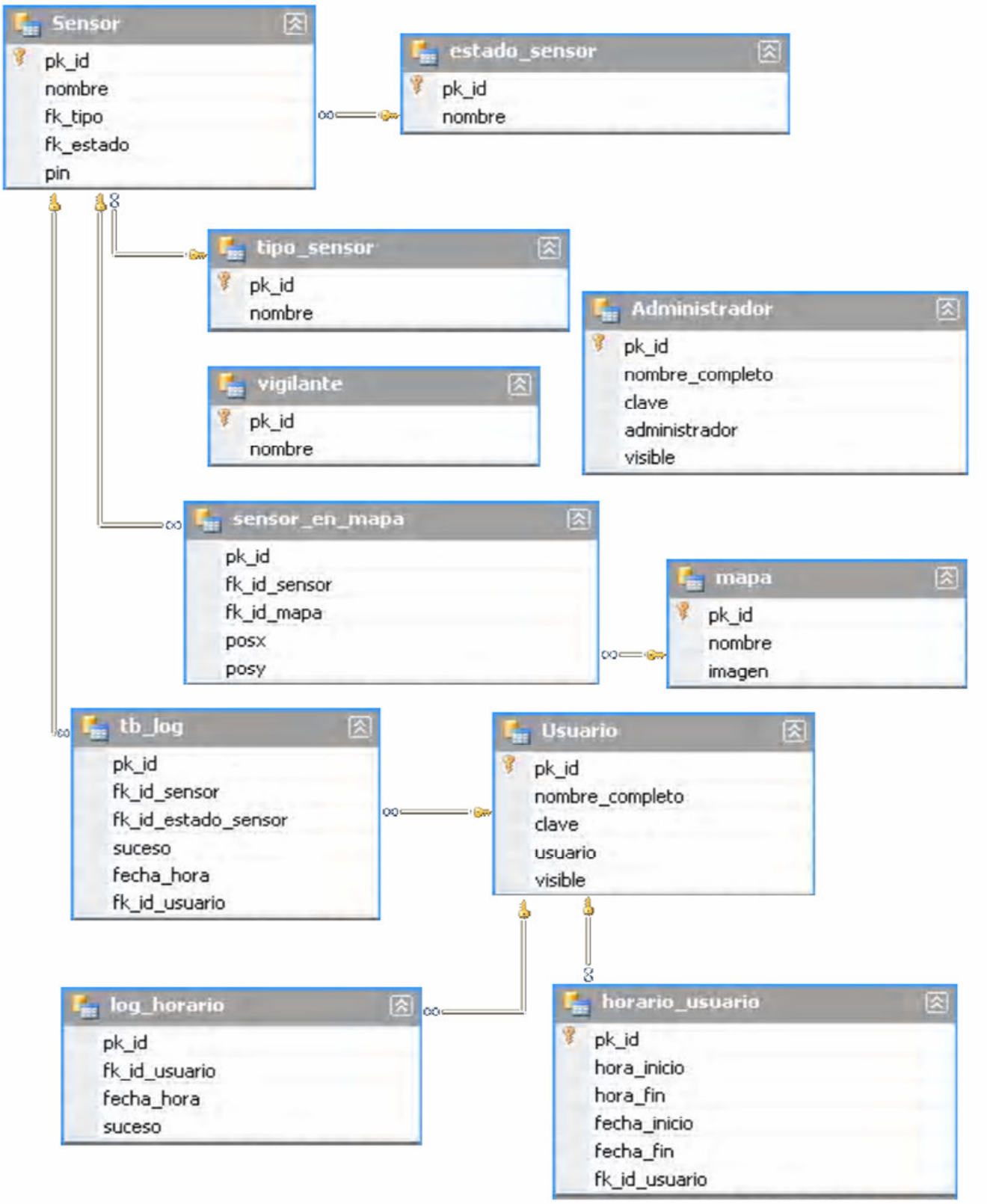
Componentes principales de una base de datos:

Datos: Es toda la información que se encuentra almacenada en la base de datos; es decir, cada uno de los registros que la componen.

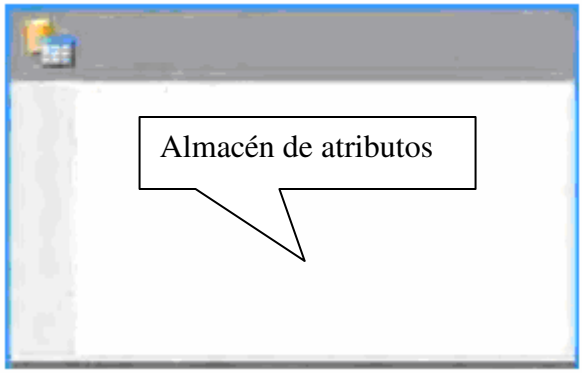
Hardware: Se refiere a los dispositivos de almacenamiento en donde reside la base de datos, así como a los dispositivos periféricos (unidad de control, canales de comunicación, etc.) necesarios para su uso.

Software: Está constituido por un conjunto de programas que se conoce como Sistema Manejador de Base de Datos (DMBS: Data Base Management system). Este sistema maneja todas las solicitudes formuladas por los usuarios a la base de datos.

MODELO FÍSICO RELACIONAL DE LA BASE DE DATOS



SIMBOLOGÍA

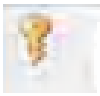


Cuadro contenedor de tablas, identifica una clase de perdonas, lugares o cosas.

 Conector relación a 1, identifica exactamente a uno.

 Conector relación de uno a muchos, identifica uno o mas.

 Conector relación a mas de uno, identifica mayor que uno.

 Llave primaria (pk), es un campo combinación de ellos, que identifican de manera única cada entidad o tabla.

Llave foránea (fk), son campos en una tabla que sirven para establecer las relaciones con otras tablas o entidades, y deben ser las llaves primarias de las otras tablas. (Es decir que los valores contenidos en la llave foránea deben estar en la llave primaria de la otra tabla o en su defecto debe tener valores nulos).

DICCIONARIO DE DATOS

El diccionario de datos es una tabla en el cual se muestra cada uno de los tributos de ella especificando todos los detalles, la razón mas importante del porque de un diccionario de datos es guardar los datos ordenadamente.

Tabla Administrador

Campo	Tipo de Dato	Descripción
Pk_jd	Int (4)	Identificador único
nombre _ completo	Nchar (45)	Nombre del administrador
clave	Nchar (10)	Clave secreta del administrador
administrador	Nchar (15)	Identificador del administrador
visible	Bit (1)	Si esta disponible

Tabla Sensor

Campo	Tipo de Dato	Descripción
Pk_jd	Int (4)	Identificador único
nombre	Nchar (20)	Nombre del sensor
Fk_tipo	Int (4)	Identifica el tipo de sensor
Fk estado	Int (4)	Identifica la clase de estado
Pin	Int (4)	Identifica el numero de pin en que se encuentra ubicado el sensor

Tabla Estado sensor

Campo	Tipo de Dato	Descripción
Pk_jd	Int (4)	Identificador único
nombre	Nchar (20)	Nombre del estado del sensor

Tabla tipo sensor

Campo	Tipo de Dato	Descripción
Pk_jd	Int (4)	Identificador único
nombre	Nchar (25)	Nombre del tipo de sensor

Tabla Sensor en mapa

Campo	Tipo de Dato	Descripción
Pk_jd	Int (4)	Identificador único
Fk_id_sensor	Int (4)	Identifica el sensor
Fk_id_mapa	Int (4)	Identifica el mapa
posx	Int (4)	Identifica la posición en el plano en base a las x (horizontalmente)
posy	Int (4)	Identifica la posición en el plano en base a las y (verticalmente)

Tabla tb _log

Campo	Tipo de Dato	Descripción
Pk_jd	Int (4)	Identificador único
Fk_id_sensor	Int (4)	Identifica el sensor
Fk_id estado sensor	Int (4)	Identifica el estado del sensor
suceso	Ntext (1073741823)	Describe la acción o el cambio que hubo
Fecha _ hora	Datatime (8)	Identifica la fecha y la hora
Fk_id_usuario	Int (4)	Identifica al usuario
Nombre _ sensor	Nchar (25)	Identifica o almacena el nombre del sensor

Tabla Vigilante

Campo	Tipo de Dato	Descripción
Pk_jd	Int (4)	Identificador único
nombre	Ntext (1073741823)	Nombre del vigilante o custodio

Tabla mapa

Campo	Tipo de Dato	Descripción
Pk_jd	Int (4)	Identificador único
nombre	Nchar (30)	Nombre del mapa o imagen
imagen	Image (2147483647)	Almacena e identifica la imagen del mapa a utilizar

Tabla usuario

Campo	Tipo de Dato	Descripción
Pk_jd	Int (4)	Identificador único
nombre _ completo	Nchar (45)	Nombre del usuario
clave	Nchar (10)	Clave secreta del administrador
usuario	Nchar (15)	Identificador del usuario
visible	Bit (1)	Si esta disponible

Tabla horario usuario

Campo	Tipo de Dato	Descripción
Pk_jd	Int (4)	Identificador único
hora_inicio	Smalldatetime (4)	Indica solamente la hora de entrada
hora_Fin	Smalldatetime (4)	Indica solamente la hora de salida
Fecha _ inicio	Smalldatetime (4)	Indica solamente la fecha de entrada
Fecha _ fin	Smalldatetime (4)	Indica solamente la fecha de salida
Fk_id_usuario	Int (4)	Identifica al usuario

Tabla Log usuario (Log _ horario)

Campo	Tipo de Dato	Descripción
Pk_jd	Int (4)	Identificador único
Fk_id_usuario	Int (4)	Identifica al usuario
Fecha _ hora	Datatime (8)	Identifica la fecha y hora del acceso al sistema
suceso	Ntext (1073741823)	Describe la acción o el cambio que hubo

PROGRAMACIÓN Y/O CODIFICACIÓN

El conocer la programación permite al usuario o lector comprender las líneas de código, haciendo el recorrido por las funciones, clases y procedimientos mas utilizables a su conveniencia. A continuación se presentan las clases, los formularios y la aplicación Web respectivamente con sus debidos comentarios:

Clases de: Centinela.ClassLib.vbproj

Clase Acceso a datos

```
Imports System.Windows.Forms
Imports System.Data
Imports System.Data.SqlClient

Namespace ClassLib

Clase que proporciona el acceso a la base de datos para consultar, agregar,
editar, borrar registros de las tablas
   Public Class AccesoDatos


#Region "Campos"
Objeto para realizar conexiones a la base de datos
      Private _con As SqlConnection
Objeto para realizar consultas SQL sobre la base de datos usando la conexion
      Private _cmd As SqlCommand


#End Region

#Region "Propiedades"
Devuelve el estado actual de la conexion a la base de datos
      Public ReadOnly Property Estado() As ConnectionState
         Get
            Return Me._con.State
         End Get
      End Property

Devuelve la cadena de conexion necesaria para establecer una conexion a la base
de datos
      Public ReadOnly Property ConnectionString() As String
         Get
```

```
            Return "Data Source=.\SQLEXPRESS;AttachDbFilename='" +
Application.StartupPath + "\dbproyecto.mdf';Integrated Security=True;Connect
Timeout=30;User Instance=True"
        End Get
    End Property
#End Region

#Region "Metodos"
Crea una nueva instancia de la clase Acceso a Datos
    Public Sub New()
        Me._con = New SqlConnection(Me.ConnectionString)
        Me._cmd = New SqlCommand("", Me._con)
    End Sub
Abre una conexion a la base de datos
    Public Sub Conectar()
        If (Me._con.State <> ConnectionState.Open) Then
            Me._con.Open()
        End If
    End Sub
Cierra una conexion a la base de datos
    Public Sub Desconectar()
        If (Me._con.State <> ConnectionState.Closed) Then
            Me._con.Close()
        End If
    End Sub

#Region "Usuarios"
Agrega un nuevo registro a la tabla usuarios, basado en los datos de un objeto
Usuario
    ''' <param name="usr">Usuario a agregar a la tabla</param>
    Public Sub AgregarUsuario(ByVal usr As Usuario)
        Me._cmd.CommandText = "Insert into usuario values(@id, @nomCompl,
@clave, @usuario, 1)"
        Me._cmd.Parameters.Clear()
        Me._cmd.Parameters.Add(New SqlParameter("@id", usr.Id))
        Me._cmd.Parameters.Add(New SqlParameter("@nomCompl",
usr.NombreCompleto))
        Me._cmd.Parameters.Add(New SqlParameter("@clave", usr.Clave))
        Me._cmd.Parameters.Add(New SqlParameter("@usuario",
usr.NombreUsuario))
        Me._cmd.ExecuteNonQuery()
    End Sub
Modifica los valores de un registro existente en la tabla usuarios, basado en los
datos de un objeto Usuario
    ''' <param name="usr">Usuario a modificar, ya incluidos sus valores
modificados</param>
    Public Sub ModificarUsuario(ByVal usr As Usuario)
```

```
            Me._cmd.CommandText = "Update usuario set nombre_completo =
@nomCompl, clave = @clave, usuario = @usuario where pk_id = @id"
            Me._cmd.Parameters.Clear()
            Me._cmd.Parameters.Add(New SqlParameter("@id", usr.Id))
            Me._cmd.Parameters.Add(New SqlParameter("@nomCompl",
usr.NombreCompleto))
            Me._cmd.Parameters.Add(New SqlParameter("@clave", usr.Clave))
            Me._cmd.Parameters.Add(New SqlParameter("@usuario",
usr.NombreUsuario))
            Me._cmd.ExecuteNonQuery()
        End Sub
Elimina un registro de la tabla usuarios
        ''' <param name="usr">Usuario a eliminar</param>
        Public Sub EliminarUsuario(ByVal usr As Usuario)
            Me._cmd.CommandText = "Update usuario set visible = 0 where pk_id =
@id"
            Me._cmd.Parameters.Clear()
            Me._cmd.Parameters.Add(New SqlParameter("@id", usr.Id))
            Me._cmd.ExecuteNonQuery()
        End Sub
Devuelve un objeto Usuario con los datos de un registro de la tabla usuarios
        ''' <param name="id">Id del registro a seleccionar</param>
        Public Function SelecUsuario(ByVal id As String) As Usuario
            Me._cmd.CommandText = "Select * from usuario where pk_id = @id"
            Me._cmd.Parameters.Clear()
            Me._cmd.Parameters.Add(New SqlParameter("@id", id))
            Dim lector As SqlDataReader = Me._cmd.ExecuteReader()
            If (lector.HasRows) Then
                lector.Read()
                Dim usr As New Usuario(lector("pk_id").ToString(),
lector("nombre_completo").ToString(), lector("clave").ToString(),
lector("usuario").ToString(), CBool(lector("visible")))
                lector.Close()
                Return usr
            End If
            Return Nothing
        End Function
Devuelve un objeto Usuario con los datos de un registro de la tabla usuarios
        ''' <param name="nom">nombre del usuario dentro del registro</param>
        ''' <param name="clave">clave del usuario dentro del registro</param>
        Public Function SelecUsuario(ByVal nom As String, ByVal clave As String) As
Usuario
            Me._cmd.CommandText = "Select * from usuario where usuario = @usr
and clave = @clv"
            Me._cmd.Parameters.Clear()
            Me._cmd.Parameters.Add(New SqlParameter("@usr", nom))
            Me._cmd.Parameters.Add(New SqlParameter("@clv", clave))
```

```
            Dim lector As SqlDataReader = Me._cmd.ExecuteReader()
            If (lector.HasRows) Then
                lector.Read()
                Dim usr As New Usuario(lector("pk_id").ToString(),
lector("nombre_completo").ToString(), lector("clave").ToString(),
lector("usuario").ToString(), CBool(lector("visible")))
                lector.Close()
                Return usr
            End If
            Return Nothing
        End Function
Devuelve una lista de objetos Usuario con los datos de registros de la tabla
usuarios. Si no hay condicion, devuelve todos los usuarios
        ''' <param name="condicion">Condicion para filtrar los registros a
seleccionar</param>
        Public Function SelectUsuarios(Optional ByVal condicion As String = "True")
As List(Of Usuario)
            Me._cmd.CommandText = "Select * from usuario where visible = 1 "
            If (condicion <> "True") Then
                Me._cmd.CommandText += condicion
            End If
            Dim lector As SqlDataReader = Me._cmd.ExecuteReader()
            If (lector.HasRows) Then
                Dim lst As New List(Of Usuario)
                Do While (lector.Read())
                    lst.Add(New Usuario(lector("pk_id").ToString(),
lector("nombre_completo").ToString(), lector("clave").ToString(),
lector("usuario").ToString(), CBool(lector("visible"))))
                Loop
                lector.Close()
                Return lst
            End If
            Return Nothing
        End Function
#End Region
#Region "Administradores"
Devuelve un objeto Administrador con los datos de un registro de la tabla
administradores
        ''' <param name="id">Id del registro a seleccionar</param>
        Public Function SelecAdministrador(ByVal id As String) As Administrador
            Me._cmd.CommandText = "Select * from administrador where pk_id = @id"
            Me._cmd.Parameters.Clear()
            Me._cmd.Parameters.Add(New SqlParameter("@id", id))
            Dim lector As SqlDataReader = Me._cmd.ExecuteReader()
            If (lector.HasRows) Then
                lector.Read()
```

```
            Dim adm As New Administrador(lector("pk_id").ToString(),
lector("nombre_completo").ToString(), lector("clave").ToString(),
lector("administrador").ToString(), CBool(lector("visible")))
            lector.Close()
            Return adm
        End If
        Return Nothing
    End Function
Devuelve un objeto Administrador con los datos de un registro de la tabla
administradores
    ''' <param name="nom">nombre del administrador dentro del
registro</param>
    ''' <param name="clave">clave del administrador dentro del registro</param>
    Public Function SelecAdministrador(ByVal nom As String, ByVal clave As
String) As Administrador
        Me._cmd.CommandText = "Select * from administrador where
administrador = @usr and clave = @clv"
        Me._cmd.Parameters.Clear()
        Me._cmd.Parameters.Add(New SqlParameter("@usr", nom))
        Me._cmd.Parameters.Add(New SqlParameter("@clv", clave))
        Dim lector As SqlDataReader = Me._cmd.ExecuteReader()
        If (lector.HasRows) Then
            lector.Read()
            Dim adm As New Administrador(lector("pk_id").ToString(),
lector("nombre_completo").ToString(), lector("clave").ToString(),
lector("administrador").ToString(), CBool(lector("visible")))
            lector.Close()
            Return adm
        End If
        Return Nothing
    End Function

#End Region

#Region "Sensores"
Agrega un nuevo registro a la tabla sensores
    ''' <param name="sen">Objeto sensor del cual tomar los datos</param>
    Public Sub AgregarSensor(ByVal sen As Sensor)
        Me._cmd.CommandText = "Insert into Sensor values(@id, @nombre,
@tipo, @estado, @pin)"
        Me._cmd.Parameters.Clear()
        Me._cmd.Parameters.Add(New SqlParameter("@id", sen.Id))
        Me._cmd.Parameters.Add(New SqlParameter("@nombre", sen.Nombre))
        Me._cmd.Parameters.Add(New SqlParameter("@tipo", sen.Tipo))
        Me._cmd.Parameters.Add(New SqlParameter("@estado",
sen.EstadoActual))
        Me._cmd.Parameters.Add(New SqlParameter("@pin", sen.Pin))
```

```
            Me._cmd.ExecuteNonQuery()
        End Sub
Modifica los datos de un registro existente dentro de la tabla sensores
        ''' <param name="sen">sensor a modificar, ya incluidos sus valores
modificados</param>
        Public Sub ModificarSensor(ByVal sen As Sensor)
            Me._cmd.CommandText = "update Sensor set nombre=@nombre, fk_tipo
= @tipo, fk_estado = @estado, pin=@pin where pk_id = @id"
            Me._cmd.Parameters.Clear()
            Me._cmd.Parameters.Add(New SqlParameter("@nombre", sen.Nombre))
            Me._cmd.Parameters.Add(New SqlParameter("@tipo", sen.Tipo))
            Me._cmd.Parameters.Add(New SqlParameter("@estado",
sen.EstadoActual))
            Me._cmd.Parameters.Add(New SqlParameter("@id", sen.Id))
            Me._cmd.Parameters.Add(New SqlParameter("@pin", sen.Pin))
            Me._cmd.ExecuteNonQuery()
        End Sub
Elimina un registro de la tabla sensores
        ''' <param name="sen">Sensor a eliminar</param>
        Public Sub EliminarSensor(ByVal sen As Sensor)
            Me._cmd.CommandText = "delete from Sensor where pk_id = @id"
            Me._cmd.Parameters.Clear()
            Me._cmd.Parameters.Add(New SqlParameter("@id", sen.Id))
            Me._cmd.ExecuteNonQuery()
        End Sub

#Region "Estado/Tipo"
Devuelve el Id de un registro en la tabla de estados de sensor, basado en su
nombre
        ''' <param name="nomEstadoSensor">El nombre del estado del
sensor</param>
        Public Function GetIdEstadoSensor(ByVal nomEstadoSensor As String) As
Integer
            Me._cmd.CommandText = "select pk_id from estado_sensor where
nombre = @nom"
            Me._cmd.Parameters.Clear()
            Me._cmd.Parameters.Add(New SqlParameter("@nom",
nomEstadoSensor))
            Return CInt(Me._cmd.ExecuteScalar())
        End Function
Devuelve el nombre de un registro en la tabla estados de sensor, basado en su Id
        ''' <param name="idEstadoSensor">El id del estado de sensor</param>
        Public Function GetNombreEstadoSensor(ByVal idEstadoSensor As Integer)
As String
            Me._cmd.CommandText = "select nombre from estado_sensor where
pk_id = @id"
            Me._cmd.Parameters.Clear()
```

```
            Me._cmd.Parameters.Add(New SqlParameter("@id", idEstadoSensor))
            Return Me._cmd.ExecuteScalar().ToString()
        End Function
Devuelve una lista que contiene todos los nombres de los estados de sensor
existentes en la tabla de estados de sensor
        Public Function GetEstadosSensor() As List(Of String)
            Me._cmd.CommandText = "Select * from estado_sensor"
            Dim lector As SqlDataReader = Me._cmd.ExecuteReader()
            Dim lst As New List(Of String)
            If (lector.HasRows) Then
                Do While (lector.Read())
                    lst.Add(lector("nombre").ToString())
                Loop
                lector.Close()
                Return lst
            End If
            Return Nothing
        End Function

Devuelve una lista que contiene todos los nombres de los tipos de sensor
existentes en la tabla tipos de sensor
        Public Function GetTiposSensor() As List(Of String)
            Me._cmd.CommandText = "Select * from tipo_sensor"
            Dim lector As SqlDataReader = Me._cmd.ExecuteReader()
            Dim lst As New List(Of String)
            If (lector.HasRows) Then
                Do While (lector.Read())
                    lst.Add(lector("nombre").ToString())
                Loop
                lector.Close()
                Return lst
            End If
            Return Nothing

        End Function
Devuelve el Id de un registro en la tabla de tipos de sensor, basado en su nombre
        ''' <param name="nomTipoSensor">El nombre del tipo del sensor</param>
        Public Function GetIdTipoSensor(ByVal nomTipoSensor As String) As Integer
            Me._cmd.CommandText = "select pk_id from tipo_sensor where nombre =
@nom"
            Me._cmd.Parameters.Clear()
            Me._cmd.Parameters.Add(New SqlParameter("@nom", nomTipoSensor))
            Return CInt(Me._cmd.ExecuteScalar())
        End Function
Devuelve el nombre de un registro en la tabla tipos de sensor, basado en su Id
        ''' <param name="idTipoSensor">El id del tipo de sensor</param>
```

```
        Public Function GetNombreTipoSensor(ByVal idTipoSensor As Integer) As
String
            Dim mycmd As New SqlCommand("select nombre from tipo_sensor where
pk_id = @id", Me._con)
            mycmd.Parameters.Clear()
            mycmd.Parameters.Add(New SqlParameter("@id", idTipoSensor))
            Return mycmd.ExecuteScalar().ToString()
        End Function

#End Region
Devuelve un objeto Sensor con los datos de un registro de la tabla sensores
        ''' <param name="id">El id del sensor del cual tomar los datos</param>
        Public Function SelectSensor(ByVal id As String) As Sensor
            Me._cmd.CommandText = "Select * from Sensor where pk_id = @id"
            Me._cmd.Parameters.Clear()
            Me._cmd.Parameters.Add(New SqlParameter("@id", id))
            Dim lector As SqlDataReader = Me._cmd.ExecuteReader()
            If (lector.HasRows) Then
                lector.Read()
                Dim sen As New Sensor(CInt(lector("pk_id")),
lector("nombre").ToString(), CInt(lector("fk_tipo")), CInt(lector("fk_estado")),
CInt(lector("pin")))
                lector.Close()
                Return sen
            End If
            Return Nothing
        End Function
Devuelve una lista de objetos Sensor basados en todos los registros de la tabla
sensores
        Public Function SelectSensores() As List(Of Sensor)
            Me._cmd = New SqlCommand("SELECT * FROM Sensor", Me._con)
            Dim lector As SqlDataReader = Me._cmd.ExecuteReader()
            Dim lst As New List(Of Sensor)
            If (lector.HasRows) Then
                Do While (lector.Read())
                    lst.Add(New Sensor(CInt(lector("pk_id")), lector("nombre").ToString(),
CInt(lector("fk_tipo")), CInt(lector("fk_estado")), CInt(lector("pin"))))
                Loop
                lector.Close()
                Return lst
            End If
            Return Nothing
        End Function
Devuelve una lista de objetos Sensor basados en los registros de la tabla
sensores asociados con un mapa especifico
        ''' <param name="idMapa">el id del Mapa asociado con los sensores a
seleccionar</param>
```

```
    Public Function SelectSensores(ByVal idMapa As Integer) As List(Of Sensor)
        Dim otroCmd As New SqlCommand("select fk_id_sensor from
sensor_en_mapa where fk_id_mapa = @idmapa order by fk_id_sensor", Me._con)
        Dim idSensoresMapa As New List(Of Integer)
        otroCmd.Parameters.Clear()
        otroCmd.Parameters.Add(New SqlParameter("@idmapa", idMapa))
        Dim otroLector As SqlDataReader = otroCmd.ExecuteReader()
        If (otroLector.HasRows) Then
            Do While (otroLector.Read())
                idSensoresMapa.Add(CInt(otroLector("fk_id_sensor")))
            Loop
            otroLector.Close()
        End If

        Dim lst As New List(Of Sensor)
        For Each idSen As Integer In idSensoresMapa
            Me._cmd = New SqlCommand("SELECT * FROM Sensor where pk_id =
@idsen", Me._con)
            Me._cmd.Parameters.Clear()
            Me._cmd.Parameters.Add(New SqlParameter("@idsen", idSen))
            Dim lector As SqlDataReader = Me._cmd.ExecuteReader()
            If (lector.HasRows) Then
                lector.Read()
                lst.Add(New Sensor(CInt(lector("pk_id")), lector("nombre").ToString(),
CInt(lector("fk_tipo")), CInt(lector("fk_estado")), CInt(lector("pin"))))
                lector.Close()
            End If
        Next
        Return lst
    End Function
Devuelve la posicion de un sensor, en un mapa especifico
    ''' <param name="idSen">El id del sensor</param>
    ''' <param name="idMapa">El id del mapa</param>
    Public Function GetPosSensor(ByVal idSen As Integer, ByVal idMapa As
Integer) As System.Drawing.Point
        Me._cmd = New SqlCommand("select posx, posy from sensor_en_mapa
where fk_id_sensor = @idsen and fk_id_mapa = @idmapa", Me._con)
        Me._cmd.Parameters.Clear()
        Me._cmd.Parameters.Add(New SqlParameter("@idsen", idSen))
        Me._cmd.Parameters.Add(New SqlParameter("@idmapa", idMapa))
        Dim lector As SqlDataReader = Me._cmd.ExecuteReader()
        Dim pt As New System.Drawing.Point(0, 0)
        If (lector.HasRows) Then
            lector.Read()
            pt.X = CInt(lector("posx"))
            pt.Y = CInt(lector("posy"))
            Return pt
```

```
            End If
            lector.Close()
            Return Nothing
        End Function
Establece la posicion de un sensor, en un mapa especifico
        ''' <param name="idSen">El id del sensor</param>
        ''' <param name="idMapa">El id del mapa</param>
        ''' <param name="nuevaPos">Un objeto Point que contiene la nueva posicion
del sensor en el mapa</param>
        Public Sub SetPosSensor(ByVal idSen As Integer, ByVal idMapa As Integer,
ByVal nuevaPos As System.Drawing.Point)
            Me._cmd = New SqlCommand("update sensor_en_mapa set posx =
@posx, posy = @posy where fk_id_sensor = @idsen and fk_id_mapa =
@idmapa", Me._con)
            Me._cmd.Parameters.Clear()
            Me._cmd.Parameters.Add(New SqlParameter("@posx", nuevaPos.X))
            Me._cmd.Parameters.Add(New SqlParameter("@posy", nuevaPos.Y))
            Me._cmd.Parameters.Add(New SqlParameter("@idsen", idSen))
            Me._cmd.Parameters.Add(New SqlParameter("@idmapa", idMapa))
            Me._cmd.ExecuteNonQuery()
        End Sub
Agrega una relacion entre un sensor existente y un mapa existente
        ''' <param name="nuevoId">El id de la relacion</param>
        ''' <param name="idSen">El id del sensor a relacionar</param>
        ''' <param name="idMapa">El id del mapa a relacionar</param>
        Public Sub AgregarSensorMapa(ByVal nuevoId As Integer, ByVal idSen As
Integer, ByVal idMapa As Integer)
            Me._cmd = New SqlCommand("insert into sensor_en_mapa
values(@newId, @idsen, @idmapa, 0, 0)", Me._con)
            Me._cmd.Parameters.Clear()
            Me._cmd.Parameters.Add(New SqlParameter("@newId", nuevoId))
            Me._cmd.Parameters.Add(New SqlParameter("@idsen", idSen))
            Me._cmd.Parameters.Add(New SqlParameter("@idmapa", idMapa))
            Me._cmd.ExecuteNonQuery()
        End Sub
Elimina una relacion entre un sensor existente y un mapa existente
        ''' <param name="idSen">El id del sensor relacionado</param>
        ''' <param name="idMapa">El id del mapa relacionado</param>
        Public Sub EliminarSensorMapa(ByVal idSen As Integer, ByVal idMapa As
Integer)
            Me._cmd = New SqlCommand("delete from sensor_en_mapa where
fk_id_sensor = @idsen and fk_id_mapa = @idmapa", Me._con)
            Me._cmd.Parameters.Clear()
            Me._cmd.Parameters.Add(New SqlParameter("@idsen", idSen))
            Me._cmd.Parameters.Add(New SqlParameter("@idmapa", idMapa))
            Me._cmd.ExecuteNonQuery()
        End Sub
```

```
#End Region

#Region "HorarioUsuario"
Agrega un nuevo registro a la tabla horarios de usuario
        ''' <param name="hUsr">El objeto HorarioUsuario del cual tomar los
datos</param>
        Public Sub AgregarHorarioUsuario(ByVal hUsr As HorarioUsuario)
            Me._cmd.CommandText = "insert into horario_usuario values(@id,
@horaini, @horafin, @fechaini, @fechafin, @fkidusuario)"
            Me._cmd.Parameters.Clear()
            Me._cmd.Parameters.Add(New SqlParameter("@id", hUsr.Id))
            Me._cmd.Parameters.Add(New SqlParameter("@horaini",
hUsr.HoraInicio))
            Me._cmd.Parameters.Add(New SqlParameter("@horafin", hUsr.HoraFin))
            Me._cmd.Parameters.Add(New SqlParameter("@fechaini",
hUsr.FechaInicio))
            Me._cmd.Parameters.Add(New SqlParameter("@fechafin",
hUsr.FechaFin))
            Me._cmd.Parameters.Add(New SqlParameter("@fkidusuario",
hUsr.Usuario))
            Me._cmd.ExecuteNonQuery()
        End Sub
Modifica un registro existente en la tabla horarios de usuario
        ''' <param name="hUsr">El objeto HorarioUsuario a modificar, con sus valores
ya modificados</param>
        Public Sub ModificarHorarioUsuario(ByVal hUsr As HorarioUsuario)
            Me._cmd.CommandText = "update horario_usuario set " + _
"hora_inicio = @horaini, hora_fin = @horafin, fecha_inicio = @fechaini, fecha_fin =
@fechafin, " + _
"fk_id_usuario = @fkidusuario where pk_id = @id"
            Me._cmd.Parameters.Clear()
            Me._cmd.Parameters.Add(New SqlParameter("@id", hUsr.Id))
            Me._cmd.Parameters.Add(New SqlParameter("@horaini",
hUsr.HoraInicio))
            Me._cmd.Parameters.Add(New SqlParameter("@horafin", hUsr.HoraFin))
            Me._cmd.Parameters.Add(New SqlParameter("@fechaini",
hUsr.FechaInicio))
            Me._cmd.Parameters.Add(New SqlParameter("@fechafin",
hUsr.FechaFin))
            Me._cmd.Parameters.Add(New SqlParameter("@fkidusuario",
hUsr.Usuario))
            Me._cmd.ExecuteNonQuery()
        End Sub
Elimina un registro de la tabla de horarios de usuario
        ''' <param name="hUsr">El horario a eliminar</param>
        Public Sub EliminarHorarioUsuario(ByVal hUsr As HorarioUsuario)
```

```
        Me._cmd.CommandText = "delete from horario_usuario where pk_id =
@id"
        Me._cmd.Parameters.Clear()
        Me._cmd.Parameters.Add(New SqlParameter("@id", hUsr.Id))
        Me._cmd.ExecuteNonQuery()
    End Sub
Devuelve un objeto HorarioUsuario con los valores de un registro de la tabla
horarios de usuario
    ''' <param name="idusr">el id del usuario asociado con el horario</param>
    Public Function SelectHorarioUsuario(ByVal idusr As Integer) As
HorarioUsuario
        Me._cmd = New SqlCommand("select * from horario_usuario where
fk_id_usuario = @idusr", Me._con)
        Me._cmd.Parameters.Clear()
        Me._cmd.Parameters.Add(New SqlParameter("@idusr", idusr))
        Dim lector As SqlDataReader = Me._cmd.ExecuteReader()
        If (lector.HasRows) Then
            lector.Read()
            'Dim motv As String = ""
            'If Not (TypeOf (lector("motivo")) Is System.DBNull) Then
            '    motv = lector("motivo").ToString()
            'End If
            Dim hUsr As New HorarioUsuario(CInt(lector("pk_id")),
CDate(lector("hora_inicio")), CDate(lector("hora_fin")),
CDate(lector("fecha_inicio")), CDate(lector("fecha_fin")), idusr)
            Return hUsr
        End If
        Return Nothing
    End Function

#End Region


#Region "LogLog"
Agrega un nuevo registro a la tabla log
    ''' <param name="lg">El objeto log del cual tomar los datos</param>
    Public Sub AgregarLog(ByVal lg As Log)
        Me._cmd.CommandText = "Insert into tb_log values(@id, @idsen,
@idestsen, @suceso, @fhHr, @idusr, @nomsensor)"
        Me._cmd.Parameters.Clear()
        Me._cmd.Parameters.Add(New SqlParameter("@id", lg.Id))
        Me._cmd.Parameters.Add(New SqlParameter("@idsen", lg.IdSensor))
        Me._cmd.Parameters.Add(New SqlParameter("@idestsen",
lg.IdEstadoSensor))
        Me._cmd.Parameters.Add(New SqlParameter("@suceso", lg.Suceso))
        Me._cmd.Parameters.Add(New SqlParameter("@fhHr", lg.FechaHora))
        Me._cmd.Parameters.Add(New SqlParameter("@idusr", lg.IdUsuario))
```

```
            Me._cmd.Parameters.Add(New SqlParameter("@nomsensor",
lg.NomSensor))
            Me._cmd.ExecuteNonQuery()
        End Sub
Devuelve una lista de objetos Log en base a los registros de la tabla log
        ''' <param name="periodo">El periodo sobre el cual filtrar los resultados a
devolver</param>Public Function SelectLogs(Optional ByVal periodo As String =
"true") As List(Of Log)
        Public Function SelectLogs(Optional ByVal periodo As String = "true") As
List(Of Log)
            Me._cmd.CommandText = "select * from tb_log"
            Me._cmd.Parameters.Clear()
            If (periodo = "diario") Then
                Dim iniAhora As New DateTime(DateTime.Now.Year,
DateTime.Now.Month, DateTime.Now.Day, 0, 0, 0)
                Dim finAhora As New DateTime(DateTime.Now.Year,
DateTime.Now.Month, DateTime.Now.Day, 23, 59, 59)
                Me._cmd.CommandText += " where fecha_hora between @fini and
@ffin"
                Me._cmd.Parameters.Add(New SqlParameter("@fini", iniAhora))
                Me._cmd.Parameters.Add(New SqlParameter("@ffin", finAhora))
            ElseIf (periodo = "mensual") Then
                Dim iniMes As New DateTime(DateTime.Now.Year,
DateTime.Now.Month, 1, 0, 0, 0)
                Dim finMes As New DateTime(DateTime.Now.Year,
DateTime.Now.Month, DateTime.DaysInMonth(DateTime.Now.Year,
DateTime.Now.Month), 23, 59, 59)
                Me._cmd.CommandText += " where fecha_hora between @fini and
@ffin"
                Me._cmd.Parameters.Add(New SqlParameter("@fini", iniMes))
                Me._cmd.Parameters.Add(New SqlParameter("@ffin", finMes))
            ElseIf (periodo = "anual") Then
                Dim iniAnio As New DateTime(DateTime.Now.Year, 1, 1, 0, 0, 0)
                Dim finAnio As New DateTime(DateTime.Now.Year, 12, 31, 23, 59, 59)
                Me._cmd.CommandText += " where fecha_hora between @fini and
@ffin"
                Me._cmd.Parameters.Add(New SqlParameter("@fini", iniAnio))
                Me._cmd.Parameters.Add(New SqlParameter("@ffin", finAnio))
            End If
            Dim lector As SqlDataReader = Me._cmd.ExecuteReader()
            If (lector.HasRows) Then
                Dim lst As New List(Of Log)
                Do While (lector.Read())
                    lst.Add(New Log(CInt(lector("pk_id")), CInt(lector("fk_id_sensor")),
CInt(lector("fk_id_estado_sensor")), lector("suceso").ToString(),
CDate(lector("fecha_hora")), CInt(lector("fk_id_usuario")),
lector("nombresensor").ToString()))
```

```
            Loop
            lector.Close()
            Return lst
        End If
        Return Nothing
    End Function
#End Region

#Region "LogActividad"
Agrega un nuevo registro a la tabla log de los horarios
    ''' <param name="lg">El objeto LogHorario del cual tomar los datos</param>
    Public Sub AgregarLogHorario(ByVal lg As LogHorario)
        Me._cmd.CommandText = "Insert into log_horario values(@id, @fkidusr,
@fechahora, @suceso)"
        Me._cmd.Parameters.Clear()
        Me._cmd.Parameters.Add(New SqlParameter("@id", lg.Id))
        Me._cmd.Parameters.Add(New SqlParameter("@fkidusr", lg.IdUsuario))
        Me._cmd.Parameters.Add(New SqlParameter("@fechahora",
lg.FechaHora))
        Me._cmd.Parameters.Add(New SqlParameter("@suceso", lg.Suceso))
        Me._cmd.ExecuteNonQuery()
    End Sub
Devuelve una lista de objetos LogHorario basados en los registros de la tabla log
de horarios
    ''' <param name="condicion">La condicion para filtrar los registros a
seleccionar</param>
    Public Function SelectLogsHorario(Optional ByVal condicion As String =
"true") As List(Of LogHorario)
        Me._cmd.CommandText = "select * from log_horario where " + condicion
        Dim lector As SqlDataReader = Me._cmd.ExecuteReader()
        If (lector.HasRows) Then
            Dim lst As New List(Of LogHorario)
            Do While (lector.Read())
                lst.Add(New LogHorario(CInt(lector("pk_id")),
CInt(lector("fk_id_usuario")), CDate(lector("fecha_hora")),
lector("suceso").ToString()))
            Loop
            lector.Close()
            Return lst
        End If
        Return Nothing
    End Function
#End Region

#Region "Custodios"

    Public Sub AgregarCustodio(ByVal v As Vigilante)
```

```
        Me._cmd.CommandText = "Insert into vigilante values(@id, @nombre)"
        Me._cmd.Parameters.Clear()
        Me._cmd.Parameters.Add(New SqlParameter("@id", v.Id))
        Me._cmd.Parameters.Add(New SqlParameter("@nombre", v.Nombre))
        Me._cmd.ExecuteNonQuery()
    End Sub

    Public Sub EliminarCustodio(ByVal v As Vigilante)
        Me._cmd.CommandText = "delete from vigilante where pk_id = @id"
        Me._cmd.Parameters.Clear()
        Me._cmd.Parameters.Add(New SqlParameter("@id", v.Id))
        Me._cmd.ExecuteNonQuery()
    End Sub

    Public Function SelecCustodio(ByVal id As String) As Vigilante
        Me._cmd.CommandText = "Select * from vigilante where pk_id = @id"
        Me._cmd.Parameters.Clear()
        Me._cmd.Parameters.Add(New SqlParameter("@id", id))
        Dim lector As SqlDataReader = Me._cmd.ExecuteReader()
        If (lector.HasRows) Then
            lector.Read()
            Dim v As New Vigilante(lector("pk_id").ToString(),
lector("nombre").ToString())
            lector.Close()
            Return v
        End If
        Return Nothing
    End Function

    Public Function SelectCustodios() As List(Of Vigilante)
        Me._cmd.CommandText = "Select * from vigilante"
        Dim lector As SqlDataReader = Me._cmd.ExecuteReader()
        If (lector.HasRows) Then
            Dim lst As New List(Of Vigilante)
            Do While (lector.Read())
                lst.Add(New Vigilante(lector("pk_id").ToString(),
lector("nombre").ToString()))
            Loop
            lector.Close()
            Return lst
        End If
        Return Nothing
    End Function

#End Region
#Region "Mapas"
```

Devuelve una lista de objetos Mapa basados en los registros de la tabla mapas

```
        Public Function SelectMapas() As List(Of Mapa)
            Me._cmd = New SqlCommand("select * from mapa", Me._con)
            Dim lector As SqlDataReader = Me._cmd.ExecuteReader()
            If (lector.HasRows) Then
                Dim lst As New List(Of Mapa)
                Do While (lector.Read())
                    lst.Add(New Mapa(CInt(lector("pk_id")), lector("nombre").ToString()))
                Loop
                lector.Close()
                Return lst
            End If
            Return Nothing
        End Function
Devuelve una lista de objetos Mapa relacionados con un sensor especifico
        ''' <param name="idSen">El id del sensor relacionado</param>
        Public Function SelectMapas(ByVal idSen As Integer) As List(Of Mapa)
            Dim idmapas As New List(Of Integer)
            Dim lst As New List(Of Mapa)

            Me._cmd = New SqlCommand("select fk_id_mapa from sensor_en_mapa
where fk_id_sensor = @idsen", Me._con)
            Me._cmd.Parameters.Clear()
            Me._cmd.Parameters.Add(New SqlParameter("@idsen", idSen))
            Dim otroLector As SqlDataReader = Me._cmd.ExecuteReader()
            If (otroLector.HasRows) Then
                Do While (otroLector.Read())
                    idmapas.Add(CInt(otroLector("fk_id_mapa")))
                Loop
            End If
            otroLector.Close()

            For Each idmapa As Integer In idmapas
                Me._cmd = New SqlCommand("select * from mapa where pk_id =
@idmapa", Me._con)
                Me._cmd.Parameters.Clear()
                Me._cmd.Parameters.Add(New SqlParameter("@idmapa", idmapa))
                Dim lector As SqlDataReader = Me._cmd.ExecuteReader()
                If (lector.HasRows) Then
                    Do While (lector.Read())
                        lst.Add(New Mapa(CInt(lector("pk_id")),
lector("nombre").ToString()))
                    Loop
                    lector.Close()
                End If
            Next
            Return lst
        End Function
```

```
Agrega un nuevo registro a la tabla mapas
        ''' <param name="mpa">El objeto Mapa del cual tomar los datos</param>
        Public Sub AgregarMapa(ByVal mpa As Mapa)
            Me._cmd = New SqlCommand("insert into mapa(pk_id, nombre)
values(@id, @nom)", Me._con)
            Me._cmd.Parameters.Clear()
            Me._cmd.Parameters.Add(New SqlParameter("@id", mpa.Id))
            Me._cmd.Parameters.Add(New SqlParameter("@nom", mpa.Nombre))
            Me._cmd.ExecuteNonQuery()
        End Sub

Elimina un registro existente en la tabla mapas
        ''' <param name="mpa">El ojbeto Mapa a eliminar</param>
        Public Sub EliminarMapa(ByVal mpa As Mapa)
            Me._cmd = New SqlCommand("delete from mapa where pk_id = @id",
Me._con)
            Me._cmd.Parameters.Clear()
            Me._cmd.Parameters.Add(New SqlParameter("@id", mpa.Id))
            Me._cmd.ExecuteNonQuery()
        End Sub

Devuelve un arreglo de bytes que representan la imagen asociada con un mapa
        ''' <param name="mpa">El mapa asociado con la imagen</param>
        Public Function GetImgMapa(ByVal mpa As Mapa) As Byte()
            Me._cmd = New SqlCommand("select imagen from mapa where pk_id =
@id", Me._con)
            Me._cmd.Parameters.Clear()
            Me._cmd.Parameters.Add(New SqlParameter("@id", mpa.Id))
            Dim lector As SqlDataReader = Me._cmd.ExecuteReader()
            If (lector.HasRows()) Then
                lector.Read()
                Return CType(lector("imagen"), Byte())
            End If
            Return Nothing
        End Function
Establece un arreglo de bytes que representan la imagen asociada con un mapa
        ''' <param name="mpa">El mapa a asociar con la imagen</param>
        ''' <param name="img">El arreglo de bytes a asociar con el mapa</param>
        Public Sub SetImgMapa(ByVal mpa As Mapa, ByVal img As Byte())
            Me._cmd = New SqlCommand("update mapa set imagen = @img where
pk_id = @id", Me._con)
            Me._cmd.Parameters.Clear()
            Me._cmd.Parameters.Add(New SqlParameter("@img", img))
            Me._cmd.Parameters.Add(New SqlParameter("@id", mpa.Id))
            Me._cmd.ExecuteNonQuery()
        End Sub
```

```
#End Region
Devuelve el maximo valor de Id de los registros de una tabla especifica
        ''' <param name="nomTabla">El nombre de la tabla en la cual
buscar</param>
        Public Function GetMaxId(ByVal nomTabla As String) As Integer
            Me._cmd.CommandText = "select max(pk_id) from " + nomTabla
            If (TypeOf (Me._cmd.ExecuteScalar()) Is System.DBNull) Then
                Return 0
            End If
            Return CInt(Me._cmd.ExecuteScalar())
        End Function

#End Region

    End Class

End Namespace
```

Clase ActividadProgramada

```
Namespace ClassLib

    Public Class ActividadProgramada

#Region "Campos"

        Private _fechaHoraInicio As DateTime
        Private _duracion As TimeSpan
        Private _periodo As Integer 'TODO: Usar Integer para # de dias? (sujeto a
cambio)
        Private _idsen As Integer

#End Region

#Region "Propiedades"

        Public Property FechaHoraInicio() As DateTime
            Get
                Return Me._fechaHoraInicio
            End Get
            Set(ByVal value As DateTime)
                Me._fechaHoraInicio = value
            End Set
        End Property

        Public Property Duracion() As TimeSpan
```

```
            Get
                Return Me._duracion
            End Get
            Set(ByVal value As TimeSpan)
                Me._duracion = value
            End Set
        End Property

        Public Property IdSen() As Integer
            Get
                Return Me._idsen
            End Get
            Set(ByVal value As Integer)
                Me._idsen = value
            End Set
        End Property

        Public Property Periodo() As Integer
            Get
                Return Me._periodo
            End Get
            Set(ByVal value As Integer)
                Me._periodo = value
            End Set
        End Property

        Public ReadOnly Property Estado() As EstadoTarea
            Get
                If (Me.FechaHoraInicio > DateTime.Now) Then
                    Return EstadoTarea.Pendiente
                ElseIf ((Me.FechaHoraInicio <= DateTime.Now) And
(Me.FechaHoraInicio.Add(Me.Duracion) >= DateTime.Now)) Then
                    Return EstadoTarea.Realizando
                ElseIf (Me.FechaHoraInicio.Add(Me.Duracion) < DateTime.Now) Then
                    Return EstadoTarea.Realizada
                End If
                Return EstadoTarea.Pendiente
            End Get
        End Property

#End Region

#Region "Metodos"

        Public Sub New(ByVal sen As Integer, ByVal fhini As DateTime, ByVal dur As
TimeSpan, ByVal per As Integer)
            Me._idsen = sen
```

```
            Me._fechaHoraInicio = fhini
            Me._duracion = dur
            Me._periodo = per
        End Sub

#End Region

    End Class

End Namespace
```

Clase Administrador

```
Namespace ClassLib

Representa un objeto usuario con privilegios de administrador
    Public Class Administrador
        Inherits Usuario
Crea una nueva instancia de la clase Administrador
        ''' <param name="id">El id del administrador</param>
        ''' <param name="nomComp">El nombre completo del
administrador</param>
        ''' <param name="clv">La clave de acceso del administrador</param>
        ''' <param name="nomUsr">El nombre de usuario del administrador</param>
        ''' <param name="vis">Indica si el administrador se encuentra visible o
no</param>
        Public Sub New(ByVal id As String, ByVal nomComp As String, ByVal clv As
String, ByVal nomUsr As String, ByVal vis As Boolean)
            MyBase.New(id, nomComp, clv, nomUsr, vis)
        End Sub
    End Class
End Namespace
```

Clase AssemblyInfo

```
Imports System.Reflection
Imports System.Runtime.CompilerServices
Imports System.Runtime.InteropServices

<Assembly: AssemblyTitle("Centinela")>
<assembly: AssemblyDescription("")>
<assembly: AssemblyConfiguration("")>
<assembly: AssemblyCompany("")>
<Assembly: AssemblyProduct("Centinela")>
<assembly: AssemblyCopyright("")>
<assembly: AssemblyTrademark("")>
<assembly: AssemblyCulture("")>
<assembly: ComVisible(False)>
<assembly: AssemblyVersion("1.0.*")>
```

Clase Custodio

```
Namespace ClassLib
   Public Class Vigilante
#Region "Campos"
      Private _id As String
      Private _nombre As String
#End Region
#Region "Propiedades"

      Public Property Id() As String
         Get
            Return Me._id
         End Get
         Set(ByVal value As String)
            Me._id = value
         End Set
      End Property

      Public Property Nombre() As String
         Get
            Return Me._nombre
         End Get
         Set(ByVal value As String)
            Me._nombre = value
         End Set
      End Property

#End Region

#Region "Metodos"

      Public Sub New(ByVal id As String, ByVal nombre As String)
         Me.Id = id
         Me.Nombre = nombre
      End Sub

#End Region

   End Class

End Namespace
```

Clase EstadoTarea

```
Namespace ClassLib
   Public Enum EstadoTarea
      Pendiente
      Realizando
```

```
        Realizada
    End Enum
End Namespace
```

Clase HorarioUsuario

```
Namespace ClassLib
Representa el horario de acceso al sistema por un usuario
    Public Class HorarioUsuario

#Region "Campos"

Identificador unico para cada registro en la base de datos
        Private _id As Integer
Hora  inicial de acceso
        Private _horaInicio As DateTime
Hora final de acceso
        Private _horaFin As DateTime
Fecha inicial de acceso
        Private _fechaInicio As DateTime
Fecha final de acceso
        Private _fechaFin As DateTime
Id del usuario relacionado con el horario
        Private _fkIdUsuario As Integer

#End Region

#Region "Propiedades"

Devuelve o establece el id del horario
        Public Property Id() As Integer
            Get
                Return Me._id
            End Get
            Set(ByVal value As Integer)
                Me._id = value
            End Set
        End Property
Devuelve o establece la hora inicial de acceso
        Public Property HoraInicio() As DateTime
            Get
                Return Me._horaInicio
            End Get
            Set(ByVal value As DateTime)
                Me._horaInicio = value
            End Set
        End Property
Devuelve o establece la hora final de acceso
```

```
        Public Property HoraFin() As DateTime
            Get
                Return Me._horaFin
            End Get
            Set(ByVal value As DateTime)
                Me._horaFin = value
            End Set
        End Property
Devuelve o establece la fecha inicial de acceso
        Public Property FechaInicio() As DateTime
            Get
                Return Me._fechaInicio
            End Get
            Set(ByVal value As DateTime)
                Me._fechaInicio = value
            End Set
        End Property
Devuelve o establece la fecha final de acceso
        Public Property FechaFin() As DateTime
            Get
                Return Me._fechaFin
            End Get
            Set(ByVal value As DateTime)
                Me._fechaFin = value
            End Set
        End Property
Devuelve o establece el Id del usuario asociado con el horario
        Public Property Usuario() As Integer
            Get
                Return Me._fkIdUsuario
            End Get
            Set(ByVal value As Integer)
                Me._fkIdUsuario = value
            End Set
        End Property

#End Region

#Region "Metodos"
Crea una nueva instancia de la clase HorarioUsuario
        ''' <param name="id">El id del horario</param>
        ''' <param name="horaInicio">La hora de inicio del horario</param>
        ''' <param name="horafin">La hora de finalizacion del horario</param>
        ''' <param name="fechainicio">La fecha de inicio del horario</param>
        ''' <param name="fechafin">La fecha de finalizacion del horario</param>
        ''' <param name="fkidusuario">El id del usuario asociado con el
horario</param>
```

```
        Public Sub New(ByVal id As Integer, ByVal horaInicio As DateTime, ByVal
horafin As DateTime, _
        ByVal fechainicio As DateTime, ByVal fechafin As DateTime, ByVal
fkidusuario As Integer)
            Me._id = id
            Me._horaInicio = horaInicio
            Me._horaFin = horafin
            Me._fechaInicio = fechainicio
            Me._fechaFin = fechafin
            Me._fkIdUsuario = fkidusuario
        End Sub

#End Region
    End Class
End Namespace
```

Clase Log

```
Namespace ClassLib
Representa un log de un suceso relacionado con un sensor
    Public Class Log

#Region "Campos"

El identificador unico del log
        Private _id As Integer
El identificador del sensor asociado con el suceso
        Private _idSensor As Integer
El identificador del estado actual del sensor
        Private _idEstadoSensor As Integer
Descripcion del suceso ocurrido
        Private _suceso As String
Fecha y hora de ocurrencia del suceso
        Private _fechaHora As DateTime
Id del usuario logueado al momento de ocurrir el suceso
        Private _IdUsuario As Integer
Nombre del sensor asociado
        Private _nomSensor As String

#End Region

#Region "Propiedades"

Devuelve o establece el identificador del log
        Public Property Id() As Integer
            Get
                Return Me._id
            End Get
```

```
            Set(ByVal value As Integer)
                Me._id = value
            End Set
        End Property
Devuelve o establece el id del sensor asociado
        Public Property IdSensor() As Integer
            Get
                Return Me._idSensor
            End Get
            Set(ByVal value As Integer)
                Me._idSensor = value
            End Set
        End Property
Devuelve o establece el id del estado del sensor
        Public Property IdEstadoSensor() As Integer
            Get
                Return Me._idEstadoSensor
            End Get
            Set(ByVal value As Integer)
                Me._idEstadoSensor = value
            End Set
        End Property
Devuelve o establece la descripcion del suceso ocurrido
        Public Property Suceso() As String
            Get
                Return Me._suceso
            End Get
            Set(ByVal value As String)
                Me._suceso = value
            End Set
        End Property
Devuelve o establece la fecha y la hora a la que ocurrio el suceso
        Public Property FechaHora() As DateTime
            Get
                Return Me._fechaHora
            End Get
            Set(ByVal value As DateTime)
                Me._fechaHora = value
            End Set
        End Property
Devuelve o establece el id del usuario logueado durante el suceso
        Public Property IdUsuario() As Integer
            Get
                Return Me._IdUsuario
            End Get
            Set(ByVal value As Integer)
                Me._IdUsuario = value
```

```
            End Set
        End Property
Devuelve o establece el nombre del sensor asociado
        Public Property NomSensor() As String
            Get
                Return Me._nomSensor
            End Get
            Set(ByVal value As String)
                Me._nomSensor = value
            End Set
        End Property

#End Region

#Region "Metodos"
Crea una nueva instancia de la clase Log
        ''' <param name="id">El id del log</param>
        ''' <param name="idSen">El id del sensor asociado</param>
        ''' <param name="idEstSen">El id del estado del sensor</param>
        ''' <param name="succ">La descripcion del suceso ocurrido</param>
        ''' <param name="fechaHr">La fecha y hora cuando ocurrio el
suceso</param>
        ''' <param name="idUsr">el Id del usuario logueado</param>
        ''' <param name="NomSen">el nombre del sensor</param>
        Public Sub New(ByVal id As Integer, ByVal idSen As Integer, ByVal idEstSen
As Integer, ByVal succ As String, ByVal fechaHr As DateTime, ByVal idUsr As
Integer, ByVal NomSen As String)

            Me.Id = id
            Me.IdSensor = idSen
            Me.IdEstadoSensor = idEstSen
            Me.Suceso = succ
            Me.FechaHora = fechaHr
            Me.IdUsuario = idUsr
            Me.NomSensor = NomSen
        End Sub

#End Region

    End Class
End Namespace
```

Clase logHorario

```
Namespace ClassLib
Representa el log del horario de un usuario
    Public Class LogHorario
```

```
#Region "Campos"
El identificador unico
        Private _id As Integer
El id del usuario logueado
        Private _fkidusuario As Integer
La fecha y la hora de ocurrencia del suceso
        Private _fechaHora As DateTime
La descripcion del suceso
        Private _suceso As String

#End Region

#Region "Propiedades"
Devuelve o establece el identificador del log
        Public Property Id() As Integer
            Get
                Return Me._id
            End Get
            Set(ByVal value As Integer)
                Me._id = value
            End Set
        End Property
Devuelve o establece el id del usuario
        Public Property IdUsuario() As Integer
            Get
                Return Me._fkidusuario
            End Get
            Set(ByVal value As Integer)
                Me._fkidusuario = value
            End Set
        End Property
Devuelve o establece la fecha y la hora de ocurrencia del suceso
        Public Property FechaHora() As DateTime
            Get
                Return Me._fechaHora
            End Get
            Set(ByVal value As DateTime)
                Me._fechaHora = value
            End Set
        End Property
Devuelve o establece la descripcion del suceso
        Public Property Suceso() As String
            Get
                Return Me._suceso
            End Get
            Set(ByVal value As String)
                Me._suceso = value
```

```
            End Set
        End Property

#End Region

#Region "Metodos"
Crea una nueva instancia de la clase LogHorario
        ''' <param name="id">El id del log</param>
        ''' <param name="idUsuario">El id del usuario</param>
        ''' <param name="fechaHr">La fecha de ocurrencia del suceso</param>
        ''' <param name="succ">La descripcion del suceso</param>
        Public Sub New(ByVal id As Integer, ByVal idUsuario As Integer, ByVal
fechaHr As DateTime, ByVal succ As String)
            Me.Id = id
            Me._fkidusuario = idUsuario
            Me.Suceso = succ
            Me.FechaHora = fechaHr
        End Sub

#End Region

    End Class
End Namespace
```

Clase Mapa

```
Namespace ClassLib
Representa un mapa en el cual visualizar sensores
    Public Class Mapa
#Region "Campos"
          Identificador unico del mapa
          Private _id As Integer
          Nombre representativo del mapa
          Private _nombre As String

#End Region

#Region "Propiedades"
Devuelve o establece el identificador del mapa
        Public Property Id() As Integer
            Get
                Return Me._id
            End Get
            Set(ByVal value As Integer)
                Me._id = value
            End Set
        End Property
Devuelve o establece el nombre del mapa
```

```
        Public Property Nombre() As String
            Get
                Return Me._nombre
            End Get
            Set(ByVal value As String)
                Me._nombre = value
            End Set
        End Property

#End Region

#Region "Metodos"

Crea una nueva instancia de la clase Mapa
        ''' <param name="newId">El id del mapa</param>
        ''' <param name="newNombre">El nombre del mapa</param>
        Public Sub New(ByVal newId As Integer, ByVal newNombre As String)
            Me._id = newId
            Me._nombre = newNombre
        End Sub

#End Region

    End Class
End Namespace
```

Clase Sensor

```
Imports System.Drawing
Imports System.Collections.Generic

Namespace ClassLib
Representa un Sensor
    Public Class Sensor

#Region "Campos"

Identificador unico del sensor
        Private _id As Integer
Nombre representativo del sensor
        Private _nombre As String
Id del estado actual del sensor
        Private _estadoActual As Integer
Id del tipo de sensor
        Private _tipo As Integer
Numero de pin asociado con el sensor
        Private _pin As Integer
```

```
#End Region

#Region "Eventos"

Evento disparado cuando se enciende un sensor
    Public Event onEncender(ByVal fhOcur As DateTime)
Evento disparado cuando se apaga un sensor
    Public Event onApagar(ByVal fhOcur As DateTime)
Evento disparado cuando se activa un sensor
    Public Event onActivar(ByVal fhOcur As DateTime)
Evento disparado cuando se desactiva un sensor
    Public Event onDesactivar(ByVal fhOcur As DateTime)

#End Region

#Region "Propiedades"

Devuelve o establece el id del sensor
    Public Property Id() As Integer
        Get
            Return Me._id
        End Get
        Set(ByVal value As Integer)
            Me._id = value
        End Set
    End Property
Devuelve o establece el nombre del sensor
    Public Property Nombre() As String
        Get
            Return Me._nombre
        End Get
        Set(ByVal value As String)
            Me._nombre = value
        End Set
    End Property
Devuelve o establece el id del estado actual del sensor
    Public Property EstadoActual() As Integer
        Get
            Return Me._estadoActual
        End Get
        Set(ByVal value As Integer)
            If (Me._estadoActual <> value) Then
                Me._estadoActual = value
                If (Me._estadoActual = 1) Then RaiseEvent onApagar(DateTime.Now)
                If (Me._estadoActual = 2) Then RaiseEvent
onDesactivar(DateTime.Now)
                If (Me._estadoActual = 3) Then RaiseEvent onActivar(DateTime.Now)
```

```
            End If
        End Set
    End Property
Devuelve o establece el id del tipo de sensor
    Public Property Tipo() As Integer
        Get
            Return Me._tipo
        End Get
        Set(ByVal value As Integer)
            Me._tipo = value
        End Set
    End Property
Devuelve o establece el numero del pin aosciado con el sensor
    Public Property Pin() As Integer
        Get
            Return Me._pin
        End Get
        Set(ByVal value As Integer)
            Me._pin = value
        End Set
    End Property

#End Region

#Region "Metodos"
Crea una nueva instancia de la clase sensor
    ''' <param name="id">El id del sensor</param>
    ''' <param name="nom">El nombre del sensor</param>
    ''' <param name="tipo">El id del tipo de sensor</param>
    ''' <param name="estado">El id del estado de sensor</param>
    ''' <param name="pn">El numero del pin asociado con el sensor</param>
    Public Sub New(ByVal id As Integer, ByVal nom As String, ByVal tipo As
Integer, ByVal estado As Integer, ByVal pn As Integer)
        Me.Id = id
        Me.Nombre = nom
        Me._estadoActual = estado
        Me.Tipo = tipo
        Me.Pin = pn
    End Sub
Cambia el estado del sensor a Encendido
    ''' <param name="autoActivar">Si es verdadero, activa automaticamente el
sensor</param>
    Public Sub Encender(Optional ByVal autoActivar As Boolean = False)
        If (Me.EstadoActual = 1) Then
            If (autoActivar) Then
                Me._estadoActual = 2
                RaiseEvent onActivar(DateTime.Now)
```

```
            Else
               Me._estadoActual = 3
               RaiseEvent onDesactivar(DateTime.Now)
            End If
            RaiseEvent onEncender(DateTime.Now)
         End If
      End Sub
Cambia el estado del sensor a Apagado
      Public Sub Apagar()
         If (Me.EstadoActual = 2) Then
            Me._estadoActual = 1
            RaiseEvent onApagar(DateTime.Now)
         End If
      End Sub
Cambia el estado del sensor a Activo
      Public Sub Activar()
         If (Me.EstadoActual <> 3) Then
            Me._estadoActual = 3
            RaiseEvent onActivar(DateTime.Now)
         End If
      End Sub
Cambia el estado del sensor a Inactivo
      Public Sub Desactivar()
         If (Me.EstadoActual <> 2) Then
            Me._estadoActual = 2
            RaiseEvent onDesactivar(DateTime.Now)
         End If
      End Sub

#End Region
   End Class

End Namespace
```

Clase Usuario

```
Namespace ClassLib

Representa un Usuario de la aplicacion
   Public Class Usuario

#Region "Campos"
Identificador unico del usuario
      Private _id As String
Nombre completo del usuario
      Private _nombreCompleto As String
Clave de acceso del usuario
      Private _clave As String
```

```
Nickname del usuario
        Private _nombreUsuario As String
Si el usuario es visible o no
        Private _visible As Boolean

#End Region

#Region "Propiedades"


Devuelve o establece el identificador del usuario
        Public Property Id() As String
            Get
                Return Me._id
            End Get
            Set(ByVal value As String)
                Me._id = value
            End Set
        End Property
Devuelve o establece el nombre completo del usuario
        Public Property NombreCompleto() As String
            Get
                Return Me._nombreCompleto
            End Get
            Set(ByVal value As String)
                Me._nombreCompleto = value
            End Set
        End Property
Devuelve o establece la clave de acceso del usuario
        Public Property Clave() As String
            Get
                Return Me._clave
            End Get
            Set(ByVal value As String)
                Me._clave = value
            End Set
        End Property
Devuelve o establece el Nickname del usuario
        Public Property NombreUsuario() As String
            Get
                Return Me._nombreUsuario
            End Get
            Set(ByVal value As String)
                Me._nombreUsuario = value
            End Set
        End Property
Devuelve o establece si el usuario es visible o no
```

```
        Public Property Visible() As Boolean
            Get
                Return Me._visible
            End Get
            Set(ByVal value As Boolean)
                Me._visible = value
            End Set
        End Property

#End Region

#Region "Metodos"
Crea una nueva instancia de la clase Usuario
        ''' <param name="id">El id del usuario</param>
        ''' <param name="nomComp">El nombre completo del usuario</param>
        ''' <param name="clv">La clave de acceso del usuario</param>
        ''' <param name="nomUsr">El Nickname del usuario</param>
        ''' <param name="vis">Si el usuario es visible</param>
        Public Sub New(ByVal id As String, ByVal nomComp As String, ByVal clv As
String, ByVal nomUsr As String, ByVal vis As Boolean)
            Me.Id = id
            Me.NombreCompleto = nomComp
            Me.Clave = clv
            Me.NombreUsuario = nomUsr
            Me.Visible = vis
        End Sub

#End Region

    End Class

End Namespace
```

Clase VigilanciaProgramada

```
Namespace ClassLib

    Public Class VigilanciaProgramada
        Inherits ActividadProgramada

#Region "Campos"

        Private _vgl As Vigilante

#End Region
#Region "Propiedades"

        Public Property Vigi() As Vigilante
```

```
            Get
                Return Me._vgl
            End Get
            Set(ByVal value As Vigilante)
                Me._vgl = value
            End Set
        End Property

#End Region

#Region "Metodos"

        Public Sub New(ByVal vgl As Vigilante, ByVal sn As Integer, ByVal fhvigil As
DateTime, ByVal dur As TimeSpan, ByVal per As Integer)
            MyBase.New(sn, fhvigil, dur, per)
            Me._vgl = vgl
        End Sub

#End Region

    End Class

End Namespace
```

Clase XmlSensorLog

```
Imports System.Xml
Imports System.IO

Namespace ClassLib
Representa una clase capaz de almacenar una lista de logs en un archivo XML
    Public Class XmlSensorLog

Genera un archivo XML con la lista de logs encontrados en la tabla log de la base
de datos
        ''' <param name="periodo">Define un periodo para filtrar la lista de logs a
generar</param>
        ''' <param name="nombreArchivoXml">Define el nombre del archivo XML en
el cual guardar la lista de logs. Por defecto es bdlog.xml</param>
        Public Shared Sub GenerarXml(Optional ByVal periodo As String = "",
Optional ByVal nombreArchivoXml As String = "")
            Dim datos As New AccesoDatos()
            If (nombreArchivoXml = "") Then nombreArchivoXml = "bdlog.xml"
            If (File.Exists(nombreArchivoXml)) Then
                File.Delete(nombreArchivoXml)
            End If
            Dim fstream As FileStream = File.Create(nombreArchivoXml)
```

```
        Dim escritor As New System.Xml.XmlTextWriter(fstream, New
System.Text.UTF8Encoding())
        escritor.WriteStartDocument()
        escritor.Formatting = Formatting.Indented
        escritor.WriteStartElement("bdlog")

        datos.Conectar()
        Dim logs As List(Of Log)
        If (periodo = "") Then
           logs = datos.SelectLogs()
        Else
           logs = datos.SelectLogs(periodo)
        End If
        datos.Desconectar()
        If Not (logs Is Nothing) Then
           For Each lg As Log In logs
              datos.Conectar()
              Dim sen As Sensor = datos.SelectSensor(lg.IdSensor.ToString())
              datos.Desconectar()
              datos.Conectar()
              Dim tipoSen As String = datos.GetNombreTipoSensor(sen.Tipo)
              datos.Desconectar()
              datos.Conectar()
              Dim estadoSen As String =
datos.GetNombreEstadoSensor(lg.IdEstadoSensor)
              datos.Desconectar()
              datos.Conectar()
              Dim usr As Usuario = datos.SelecUsuario(lg.IdUsuario.ToString())
              datos.Desconectar()

              escritor.WriteStartElement("log")
              escritor.WriteAttributeString("fechahora",
lg.FechaHora.ToShortDateString() + " " + lg.FechaHora.ToShortTimeString)
              escritor.WriteAttributeString("sensor", sen.Nombre.Trim())
              escritor.WriteAttributeString("tiposensor", tipoSen.Trim())
              escritor.WriteAttributeString("estadosensor", estadoSen.Trim())
              escritor.WriteAttributeString("suceso", lg.Suceso.Trim())
              escritor.WriteAttributeString("usuario", usr.NombreCompleto.Trim())
              escritor.WriteEndElement()
           Next
        Else
           escritor.WriteStartElement("log")
           escritor.WriteAttributeString("fechahora", "")
           escritor.WriteAttributeString("sensor", "")
           escritor.WriteAttributeString("tiposensor", "")
           escritor.WriteAttributeString("estadosensor", "")
           escritor.WriteAttributeString("suceso", "")
```

```
            escritor.WriteAttributeString("usuario", "")
            escritor.WriteEndElement()
        End If
        escritor.WriteEndElement()
        escritor.Flush()
        escritor.Close()
    End Sub

End Class

End Namespace
```

Formularios y clases de: Centinela.WinApp.vbproj

Clase AssemblyInfo

```
Imports System.Reflection
Imports System.Runtime.CompilerServices
Imports System.Runtime.InteropServices

<Assembly: AssemblyTitle("Centinela.WinApp")>
<assembly: AssemblyDescription("")>
<assembly: AssemblyConfiguration("")>
<assembly: AssemblyCompany("")>
<Assembly: AssemblyProduct("Centinela.WinApp")>
<Assembly: AssemblyCopyright("R. S. A. C., R. C. L. M.")>
<assembly: AssemblyTrademark("")>
<assembly: AssemblyCulture("")>

<assembly: ComVisible(False)>
<assembly: AssemblyVersion("1.0.*")>
```

frmAddUsuario.vb

```
Public Class frmAddUsuario

    Private Sub btnCancelar_Click(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles btnCancelar.Click
        Me.Close()
    End Sub

    Private Sub btnAceptar_Click(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles btnAceptar.Click
        Dim datos As New ClassLib.AccesoDatos()
        datos.Conectar()
        Dim esUsr As ClassLib.Usuario = datos.SelecUsuario(Me.txtUsuario.Text,
Me.txtClave.Text)
        datos.Desconectar()
```

```
        datos.Conectar()
        If Not (esUsr Is Nothing) Then
            MsgBox("Usuario existente", MsgBoxStyle.OkOnly, "Error")
        Else
            Dim usr As New ClassLib.Usuario((datos.GetMaxId("usuario") +
1).ToString(), Me.txtNombreCompleto.Text, Me.txtClave.Text, Me.txtUsuario.Text,
True)
            datos.Desconectar()
            datos.Conectar()
            datos.AgregarUsuario(usr)
            Me.Close()
        End If

    End Sub

    Private Sub txtNombreCompleto_TextChanged(ByVal sender As System.Object,
ByVal e As System.EventArgs) Handles txtNombreCompleto.TextChanged
        Me.VerificarBtnAceptar()
    End Sub

    Private Sub VerificarBtnAceptar()
        If (Me.txtNombreCompleto.Text = "" Or Me.txtUsuario.Text = "" Or
Me.txtClave.Text = "") Then
            Me.btnAceptar.Enabled = False
        Else
            Me.btnAceptar.Enabled = True
        End If
    End Sub

    Private Sub txtUsuario_TextChanged(ByVal sender As System.Object, ByVal e
As System.EventArgs) Handles txtUsuario.TextChanged
        Me.VerificarBtnAceptar()
    End Sub

    Private Sub txtClave_TextChanged(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles txtClave.TextChanged
        Me.VerificarBtnAceptar()
    End Sub
End Class
```

frmAgregarMapa.vb

```
Public Class frmAgregarMapa
    Private datos As ClassLib.AccesoDatos
    Private dlg As OpenFileDialog
```

```
    Private Sub Button1_Click(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles Button1.Click
        Me.Close()
    End Sub

    Private Sub Button2_Click(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles btnAceptar.Click
        Me.datos.Conectar()
        Dim nuevoId As Integer = datos.GetMaxId("mapa") + 1
        Me.datos.Desconectar()
        Dim mpa As New ClassLib.Mapa(nuevoId, Me.txtNombre.Text)

        Me.datos.Conectar()
        Me.datos.AgregarMapa(mpa)
        Me.datos.Desconectar()
        Dim stream As System.IO.Stream = dlg.OpenFile()
        Dim bytes(CInt(stream.Length)) As Byte
        stream.Read(bytes, 0, CInt(stream.Length))

        datos.Conectar()
        datos.SetImgMapa(mpa, bytes)
        datos.Desconectar()
        Me.Close()
    End Sub

    Private Sub Button3_Click(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles Button3.Click
        dlg.Filter = "Imagenes(*.jpg;*.bmp;*.gif;*.png)|*.jpg;*.bmp;*.gif;*.png"
        If (dlg.ShowDialog() = DialogResult.OK) Then
            Dim stream As System.IO.Stream = dlg.OpenFile()
            Dim bytes(CInt(stream.Length)) As Byte
            stream.Read(bytes, 0, CInt(stream.Length))
            Me.pbxImagen.Image = New System.Drawing.Bitmap(stream)
            Me.txtImagen.Text = dlg.FileName
        End If
        Me.VerificarBtnAceptar()
    End Sub

    Public Sub VerificarBtnAceptar()
        If (Me.txtNombre.Text <> "" And Me.txtImagen.Text <> "") Then
            Me.btnAceptar.Enabled = True
        Else
            Me.btnAceptar.Enabled = False
        End If
    End Sub
```

```
    Private Sub TextBox1_TextChanged(ByVal sender As System.Object, ByVal e
As System.EventArgs) Handles txtNombre.TextChanged
        Me.VerificarBtnAceptar()
    End Sub

    Private Sub frmAgregarMapa_Load(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles MyBase.Load
        Me.datos = New ClassLib.AccesoDatos()
        Me.dlg = New OpenFileDialog()
    End Sub
End Class
```

frmAgregarSensor.vb

```
Imports Centinela.ClassLib

Public Class frmAgregarSensor
    Private datos As AccesoDatos

    Private Sub btnCerrar_Click(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles btnCerrar.Click
        Me.Close()
    End Sub

    Private Sub txtNombre_TextChanged(ByVal sender As System.Object, ByVal e
As System.EventArgs) Handles txtNombre.TextChanged
        Me.VerificarBtnAceptar()
    End Sub

    Public Sub VerificarBtnAceptar()
        If (Me.txtNombre.Text <> "" And Me.lstPines.Items.Count > 0) Then
            Me.btnAgregar.Enabled = True
        Else
            Me.btnAgregar.Enabled = False
        End If
    End Sub

    Private Sub btnAgregar_Click(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles btnAgregar.Click
        datos.Conectar()
        Dim newId As Integer = datos.GetMaxId("Sensor") + 1
        datos.Desconectar()
        datos.Conectar()
        Dim idTipoSen As Integer = datos.GetIdTipoSensor(Me.cmbTipoSensor.Text)
        datos.Desconectar()
```

```
        Dim sen As New Sensor(newId, Me.txtNombre.Text, idTipoSen, 1,
CInt(Me.lstPines.Items(Me.lstPines.SelectedIndex)))
        datos.Conectar()
        datos.AgregarSensor(sen)
        datos.Desconectar()
        Me.Close()
    End Sub

    Private Sub frmAgregarSensor_Load(ByVal sender As System.Object, ByVal e
As System.EventArgs) Handles MyBase.Load
        Me.datos = New AccesoDatos()
        Me.RellenarTipos()
        Me.CargarPinesDisponibles()
    End Sub

    Public Sub RellenarTipos()
        Me.cmbTipoSensor.Items.Clear()
        datos.Conectar()
        Dim tipos As List(Of String) = datos.GetTiposSensor()
        datos.Desconectar()
        For Each tiposen As String In tipos
            Me.cmbTipoSensor.Items.Add(tiposen)
        Next
        Me.cmbTipoSensor.SelectedIndex = 0
    End Sub

    Public Sub CargarPinesDisponibles()
        Dim pinesOcupados As New List(Of Integer)
        datos.Conectar()
        Dim sensores As List(Of Sensor) = datos.SelectSensores()
        datos.Desconectar()
        For Each sen As Sensor In sensores
            pinesOcupados.Add(sen.Pin)
        Next
        For i As Integer = 1 To 16
            If Not (pinesOcupados.Contains(i)) Then
                Me.lstPines.Items.Add(i)
            End If
        Next
        If (Me.lstPines.Items.Count > 0) Then
            Me.lstPines.SelectedIndex = 0
        End If
    End Sub
End Class
```

frmCambiarClave.vb

```
Imports Centinela.ClassLib

Public Class frmCambiarClave
    Dim datos As New AccesoDatos()
    Dim usr As Usuario

    Public Sub New(ByVal codUsr As String)
        InitializeComponent()
        datos.Conectar()
        Me.usr = datos.SelecUsuario(codUsr)
        datos.Desconectar()
    End Sub
    Private Sub btnCancelar_Click(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles btnCancelar.Click
        Me.Close()
    End Sub

    Private Sub btnAceptar_Click(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles btnAceptar.Click
        If (Me.txtClaveAnterior.Text <> usr.Clave.Trim) Then
            MsgBox("Clave anterior incorrecta", MsgBoxStyle.OkOnly, "Error")
            Return
        End If
        Me.usr.Clave = Me.txtClaveNueva.Text
        datos.Conectar()
        datos.ModificarUsuario(Me.usr)
        datos.Desconectar()
        Me.Close()
    End Sub

    Private Sub txtClaveAnterior_TextChanged(ByVal sender As System.Object,
ByVal e As System.EventArgs) Handles txtClaveAnterior.TextChanged
        Me.VerificarBtnAceptar()
    End Sub

    Private Sub VerificarBtnAceptar()
        If (Me.txtClaveAnterior.Text = "" Or Me.txtClaveNueva.Text = "") Then
            Me.btnAceptar.Enabled = False
        Else
            Me.btnAceptar.Enabled = True
        End If
    End Sub

    Private Sub txtClaveNueva_TextChanged(ByVal sender As System.Object,
ByVal e As System.EventArgs) Handles txtClaveNueva.TextChanged
```

```
        Me.VerificarBtnAceptar()
    End Sub
End Class
```

frmConfigSensores.vb

```
Imports XmlClassLib
Imports System.IO

Public Class frmConfigSensores
    Dim cfg As ConfigManager
    Private dlg, dlg2 As OpenFileDialog

    Private Sub btnCerrar_Click(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles btnCerrar.Click
        Me.cfg.EditValor("intervalo", Me.nudFrecuencia.Value.ToString())
        Me.Close()
    End Sub

    Private Sub frmConfigSensores_Load(ByVal sender As System.Object, ByVal e
As System.EventArgs) Handles MyBase.Load
        Me.cfg = New ConfigManager("app.config")
        Me.dlg = New OpenFileDialog()
        Me.dlg2 = New OpenFileDialog()
        dlg.Filter = "Imagenes(*.jpg;*.bmp;*.gif;*.png)|*.jpg;*.bmp;*.gif;*.png"
        dlg2.Filter = "Archivos wav(*.wav)|*.wav"

        Me.nudFrecuencia.Value = CDec(Me.cfg("intervalo"))
        Me.txtAlarmaSen.Text = Me.cfg("sndalarmasensor")
        Me.txtSenDesconectado.Text = Me.cfg("imgdesconectado")
        Me.txtSenDesactivado.Text = Me.cfg("imgdesactivado")
        Me.txtSenActivado.Text = Me.cfg("imgactivado")
    End Sub

    Private Sub txtSenDesconectado_Enter(ByVal sender As System.Object, ByVal
e As System.EventArgs) Handles txtSenDesconectado.Enter
        Me.pbxImagenSensor.ImageLocation = Application.StartupPath + "\media\" +
Me.txtSenDesconectado.Text
    End Sub

    Private Sub txtSenDesactivado_Enter(ByVal sender As System.Object, ByVal e
As System.EventArgs) Handles txtSenDesactivado.Enter
        Me.pbxImagenSensor.ImageLocation = Application.StartupPath + "\media\" +
Me.txtSenDesactivado.Text
    End Sub
```

```
    Private Sub txtSenActivado_Enter(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles txtSenActivado.Enter
        Me.pbxImagenSensor.ImageLocation = Application.StartupPath + "\media\" +
Me.txtSenActivado.Text
    End Sub

    Private Sub btnSenDesconectado_Click(ByVal sender As System.Object, ByVal
e As System.EventArgs) Handles btnSenDesconectado.Click
        If (dlg.ShowDialog() = DialogResult.OK) Then
            Me.CambiarImagen("SensorDesconectado", "imgdesconectado")
            Me.txtSenDesconectado.Text = Me.cfg("imgdesconectado")
            Me.txtSenDesconectado.Focus()
        End If
    End Sub

    Public Sub CambiarImagen(ByVal imgNombre As String, ByVal
cfgClaveNombre As String)
        Dim mediaDir As String = Application.StartupPath + "\Media"
        File.Delete(Directory.GetFiles(mediaDir, imgNombre + "*")(0))
        Dim origenDir As String = dlg.FileName
        Dim destDir As String = mediaDir + "\" + imgNombre +
dlg.FileName.Substring(dlg.FileName.LastIndexOf("."))
        File.Copy(origenDir, destDir)
        Dim fileName As String = destDir.Substring(destDir.LastIndexOf("\") + 1)
        Me.cfg.DelClave(cfgClaveNombre)
        Me.cfg.AddClave(cfgClaveNombre, fileName)
    End Sub

    Public Sub CambiarSonido(ByVal sndNombre As String, ByVal cfgClaveNombre
As String)
        Dim mediaDir As String = Application.StartupPath + "\Media"
        File.Delete(Directory.GetFiles(mediaDir, sndNombre + "*")(0))
        Dim origenDir As String = dlg2.FileName
        Dim destDir As String = mediaDir + "\" + sndNombre +
dlg2.FileName.Substring(dlg2.FileName.LastIndexOf("."))
        File.Copy(origenDir, destDir)
        Dim fileName As String = destDir.Substring(destDir.LastIndexOf("\") + 1)
        Me.cfg.DelClave(cfgClaveNombre)
        Me.cfg.AddClave(cfgClaveNombre, fileName)
    End Sub

    Private Sub btnSenDesactivado_Click(ByVal sender As System.Object, ByVal e
As System.EventArgs) Handles btnSenDesactivado.Click
        If (dlg.ShowDialog() = DialogResult.OK) Then
            Me.CambiarImagen("SensorDesactivado", "imgdesactivado")
            Me.txtSenDesactivado.Text = Me.cfg("imgdesactivado")
            Me.txtSenDesactivado.Focus()
```

```
        End If
    End Sub

    Private Sub btnSenActivado_Click(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles btnSenActivado.Click
        If (dlg.ShowDialog() = DialogResult.OK) Then
            Me.CambiarImagen("SensorActivado", "imgactivado")
            Me.txtSenActivado.Text = Me.cfg("imgactivado")
            Me.txtSenActivado.Focus()
        End If
    End Sub

    Private Sub btnAlarmaSen_Click(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles btnAlarmaSen.Click
        If (dlg2.ShowDialog() = DialogResult.OK) Then
            Me.CambiarSonido("Alarma", "imgdesconectado")
            Me.txtSenDesconectado.Text = Me.cfg("imgdesconectado")
            Me.txtSenDesconectado.Focus()
        End If
    End Sub
End Class
```

frmCustodios.vb

```
Public Class frmCustodios
    Dim datos As ClassLib.AccesoDatos
    Dim custodios As List(Of ClassLib.Vigilante)
    Private Sub btnCerrar_Click(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles btnCerrar.Click
        Me.Close()
    End Sub
    Private Sub frmCustodios_Load(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles MyBase.Load
        Me.datos = New ClassLib.AccesoDatos()
        Me.CargarCustodios()
    End Sub

    Public Sub CargarCustodios()
        datos.Conectar()
        Me.custodios = datos.SelectCustodios()
        Me.grdCustodios.Rows.Clear()
        For Each cst As ClassLib.Vigilante In Me.custodios
            Me.grdCustodios.Rows.Add()
            Me.grdCustodios.Rows(Me.grdCustodios.Rows.Count -
1).Cells("colNombre").Value = cst.Nombre
        Next
```

```
        datos.Desconectar()

    End Sub

    Private Sub btnEliminar_Click(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles btnEliminar.Click
        If (MsgBox("Esta usted seguro?", MsgBoxStyle.YesNo, "Confirmacion") =
MsgBoxResult.Yes) Then
            datos.Conectar()

datos.EliminarCustodio(Me.custodios(Me.grdCustodios.CurrentCell.RowIndex))
            datos.Desconectar()
            Me.CargarCustodios()
        End If
    End Sub

    Private Sub btnAgregar_Click(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles btnAgregar.Click
        Dim nombre As String = InputBox("Escriba el nombre del custodio", "Agregar
Custodio")
        If (nombre <> "") Then
            datos.Conectar()
            Dim newId As Integer = datos.GetMaxId("vigilante") + 1
            datos.Desconectar()
            Dim cst As New ClassLib.Vigilante(newId.ToString(), nombre)
            datos.Conectar()
            datos.AgregarCustodio(cst)
            datos.Desconectar()
            Me.CargarCustodios()
        End If
    End Sub
End Class
```

frmHorarioUsuario.vb

```
Public Class frmHorarioUsuario

    Private datos As ClassLib.AccesoDatos
    Private HUsr As ClassLib.HorarioUsuario
    Private idusr As Integer

    Public Sub New(ByVal iduser As Integer)
        Me.InitializeComponent()
        datos = New ClassLib.AccesoDatos()
        Me.idusr = iduser
    End Sub
```

```
    Private Sub btnCerrar_Click(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles btnCerrar.Click
        Me.Close()
    End Sub

    Private Sub frmHorarioActividad_Load(ByVal sender As System.Object, ByVal e
As System.EventArgs) Handles MyBase.Load
        Me.RellenarEstadosSensor()
        Me.CargarHorario()
    End Sub

    Public Sub CargarHorario()
        Me.datos.Conectar()
        Me.HUsr = datos.SelectHorarioUsuario(idusr)
        Me.datos.Desconectar()
        If Not (Me.HUsr Is Nothing) Then

            Me.datos.Conectar()
            Me.dtpFechaInicio.Value = Me.HUsr.FechaInicio
            Me.dtpFechaFin.Value = Me.HUsr.FechaFin
            Me.nudHoraE.Value = Me.HUsr.HoraInicio.Hour
            Me.nudMinutoE.Value = Me.HUsr.HoraInicio.Minute
            Me.nudHoraS.Value = Me.HUsr.HoraFin.Hour
            Me.nudMinutoS.Value = Me.HUsr.HoraFin.Minute
        End If
        datos.Desconectar()
    End Sub


    Private Sub btnGuardar_Click(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles btnGuardar.Click

        Dim horaentrada As New DateTime(1900, 12, 1, CInt(Me.nudHoraE.Value),
CInt(Me.nudMinutoE.Value), 0)
        Dim horasalida As New DateTime(1900, 12, 1, CInt(Me.nudHoraS.Value),
CInt(Me.nudMinutoS.Value), 0)
        If (horaentrada >= horasalida) Then
            MsgBox("La hora de entrada debe ser mayor que la hora de salida",
MsgBoxStyle.OkOnly, "Error")
            Exit Sub
        End If
        If (Me.dtpFechaInicio.Value >= Me.dtpFechaFin.Value) Then
            MsgBox("La fecha de inicio debe ser mayor que la fecha de finalizacion",
MsgBoxStyle.OkOnly, "Error")
            Exit Sub
        End If
```

```
        Me.datos.Desconectar()
        Me.datos.Conectar()
        Dim newId As Integer = datos.GetMaxId("horario_usuario") + 1
        datos.Desconectar()
        Me.datos.Conectar()
        'Dim newHAct As New ClassLib.HorarioUsuario(newId, Me.idsen,
Me.dtpFechaInicio.Value, New TimeSpan(CInt(Me.nudHoraS.Value),
CInt(Me.nudMinutoS.Value), CInt(Me.nudSegundoS.Value)), idEstSen,
CInt(Me.nudDias.Value), Me.txtMotivo.Text)
        Dim newHUsr As New ClassLib.HorarioUsuario(newId, horaentrada,
horasalida, Me.dtpFechaInicio.Value, Me.dtpFechaFin.Value, Me.idusr)
        If (Me.HUsr Is Nothing) Then
            datos.AgregarHorarioUsuario(newHUsr)
        Else
            newHUsr.Id = Me.HUsr.Id
            datos.ModificarHorarioUsuario(newHUsr)
        End If

        Me.Close()
    End Sub
End Class
```

frmMain.vb

```
Imports Centinela.ClassLib
Imports System.IO

Public Class frmMain
    Public usr As Usuario
    Private datos As ClassLib.AccesoDatos
    Private mab As MinimizarABandeja
    Private sensores As List(Of SensorVisual)
    Private cargando As Boolean
    Dim uPunto As Point
    Dim mapas As List(Of Mapa)
    Dim lastCmbMapasIndex As Integer = 0

    Public Notificador As NotificarBandeja.ShellNot = New
NotificarBandeja.ShellNot()

#Region "EventHandlers"

#Region "MainForm"

    Sub MainFormLoad(ByVal sender As Object, ByVal e As EventArgs) Handles
MyBase.Load
```

```
        Me.datos = New AccesoDatos()
        Me.PosicionarHijos()
        Dim sesion As New frmSesion()
        sesion.ShowDialog(Me)
        Me.usr = sesion.usr
        If (Me.usr.GetType().Name = "Administrador") Then
            Me.configuracionToolStripMenuItem.Enabled = True
            Me.administracionToolStripMenuItem.Enabled = True
        Else
            Me.configuracionToolStripMenuItem.Enabled = False
            Me.administracionToolStripMenuItem.Enabled = False
        End If
        Me.Text = "Centinela - Usuario: " + usr.NombreCompleto
        Me.CargarMapas()
        Me.CargarSensores()
        Me.mab = New MinimizarABandeja(Me, "Centinela")
        Dim cfg As New XmlClassLib.ConfigManager("app.config")
        Me.srvPuertos.Interval = CInt(cfg("intervalo"))
        Me.srvPuertos.Start()
    End Sub

    Sub MainFormResizeEnd(ByVal sender As Object, ByVal e As EventArgs)
Handles MyBase.Resize
        Me.PosicionarHijos()
    End Sub

    Private Sub frmMain_FormClosing(ByVal sender As System.Object, ByVal e As
System.Windows.Forms.FormClosingEventArgs) Handles MyBase.FormClosing
        Me.GuardarPosSensores()
    End Sub

#End Region

#Region "Sensores"

    Private Sub SensorVisualClick(ByVal sender As Object, ByVal e As EventArgs)
        Dim idSen As Integer = CType(sender, SensorVisual).Sen.Id
        For i As Integer = 0 To Me.sensores.Count - 1
            If (idSen = Me.sensores(i).Sen.Id) Then
                Me.lstSensores.SelectedIndex = i
                Exit For
            End If
        Next
    End Sub

    Private Sub SensorVisualMouseDown(ByVal sender As System.Object, ByVal e
As System.Windows.Forms.MouseEventArgs)
```

```
        If (TypeOf (Me.usr) Is Administrador) Then
            If (e.Button <> Windows.Forms.MouseButtons.Left) Then
                Exit Sub
            Else
                SensorVisualMouseMove(sender, e)
            End If
        End If
    End Sub

    Private Sub SensorVisualMouseMove(ByVal sender As System.Object, ByVal e
As System.Windows.Forms.MouseEventArgs)
        If (TypeOf (Me.usr) Is Administrador) Then
            If (e.Button <> Windows.Forms.MouseButtons.Left) Then
                Exit Sub
            Else
                Dim Izqd As Double = CType(sender, PictureBox).Left
                Dim Tope As Double = CType(sender, PictureBox).Top
                Dim nIzqd As Double = Izqd + e.X - (CType(sender, PictureBox).Height /
2)
                Dim nTope As Double = Tope + e.Y - (CType(sender, PictureBox).Width
/ 2)

                CType(sender, PictureBox).Left = CType(nIzqd, Integer)
                CType(sender, PictureBox).Top = CType(nTope, Integer)
                uPunto = New Point(e.X, e.Y)
                Me.lblPosicion.Text = "Posicion: (X:" +
Me.sensores(Me.lstSensores.SelectedIndex).Left.ToString() + ", Y:" +
Me.sensores(Me.lstSensores.SelectedIndex).Top.ToString() + ")"
            End If
        End If
    End Sub

    Private Sub SensoresToolStripMenuItem_Click(ByVal sender As System.Object,
ByVal e As System.EventArgs) Handles SensoresToolStripMenuItem.Click
        Dim frmSen As New frmSensores()
        frmSen.ShowDialog(Me)
        Me.CargarSensores()
    End Sub

    Private Sub SensoresToolStripMenuItem1_Click(ByVal sender As
System.Object, ByVal e As System.EventArgs) Handles
SensoresToolStripMenuItem1.Click
        Dim frmCfgSen As New frmConfigSensores()
        frmCfgSen.ShowDialog(Me)
        Dim cfg As New XmlClassLib.ConfigManager("app.config")
        Me.srvPuertos.Interval = CInt(cfg("intervalo"))
    End Sub
```

```
#End Region

    Sub SalirToolStripMenuItemClick(ByVal sender As Object, ByVal e As
EventArgs) Handles salirToolStripMenuItem.Click
        End
    End Sub

    Sub UsuariosToolStripMenuItemClick(ByVal sender As Object, ByVal e As
EventArgs) Handles usuariosToolStripMenuItem.Click
        Dim frmUsr As New frmUsuarios()
        frmUsr.ShowDialog(Me)
    End Sub

    Private Sub lstSensores_SelectedIndexChanged(ByVal sender As
System.Object, ByVal e As System.EventArgs) Handles
lstSensores.SelectedIndexChanged
        If Not (Me.cargando) Then
            For Each senvis As SensorVisual In Me.sensores
                senvis.BackColor = Color.LightGray
            Next
            Me.sensores(Me.lstSensores.SelectedIndex).BackColor = Color.Blue
            datos.Desconectar()
            datos.Conectar()
            Me.lblTipo.Text = "Tipo: " +
datos.GetNombreTipoSensor(Me.sensores(Me.lstSensores.SelectedIndex).Sen.Ti
po)
            Me.lblPosicion.Text = "Posicion: (X:" +
Me.sensores(Me.lstSensores.SelectedIndex).Left.ToString() + ", Y:" +
Me.sensores(Me.lstSensores.SelectedIndex).Top.ToString() + ")"
            datos.Desconectar()
            datos.Conectar()
            Dim nomEst As String =
datos.GetNombreEstadoSensor(Me.sensores(Me.lstSensores.SelectedIndex).Sen.
EstadoActual)
            datos.Desconectar()
            Me.lblEstado.Text = "Estado: " + nomEst
            Me.lblPin.Text = "Pin: " +
Me.sensores(Me.lstSensores.SelectedIndex).Sen.Pin.ToString()
        End If
    End Sub

    Private Sub MapasToolStripMenuItem_Click(ByVal sender As System.Object,
ByVal e As System.EventArgs) Handles MapasToolStripMenuItem.Click
        Dim frmmpas As New frmMapas()
        frmmpas.ShowDialog(Me)
        Me.CargarMapas()
```

```
    End Sub

    Private Sub cmbMapas_SelectedIndexChanged(ByVal sender As
System.Object, ByVal e As System.EventArgs) Handles
cmbMapas.SelectedIndexChanged
        Me.GuardarPosSensores()
        datos.Conectar()
        Dim bytes As Byte() =
datos.GetImgMapa(Me.mapas(Me.cmbMapas.SelectedIndex))
        If Not (bytes Is Nothing) Then
            Dim mstream As New System.IO.MemoryStream(bytes.Length)
            mstream.Write(bytes, 0, bytes.Length)
            Dim img As New Bitmap(mstream)
            Me.grpSensores.BackgroundImage = img
        End If
        datos.Desconectar()
        Me.CargarSensores()
        Me.lastCmbMapasIndex = Me.cmbMapas.SelectedIndex
    End Sub

    Private Sub Button1_Click(ByVal sender As System.Object, ByVal e As
System.EventArgs)
        If (Me.sensores(0).Sen.EstadoActual = 3) Then
            Me.sensores(0).Sen.Desactivar()
        ElseIf (Me.sensores(0).Sen.EstadoActual = 1) Then
            Me.sensores(0).Sen.Activar()
        ElseIf (Me.sensores(0).Sen.EstadoActual = 2) Then
            Me.sensores(0).Sen.Activar()
        End If
    End Sub

    Private Sub Button2_Click(ByVal sender As System.Object, ByVal e As
System.EventArgs)
        If (Me.sensores(1).Sen.EstadoActual = 3) Then
            Me.sensores(1).Sen.Desactivar()
        ElseIf (Me.sensores(1).Sen.EstadoActual = 1) Then
            Me.sensores(1).Sen.Activar()
        ElseIf (Me.sensores(1).Sen.EstadoActual = 2) Then
            Me.sensores(1).Sen.Activar()
        End If
    End Sub

    Private Sub tmrVerificarUsuario_Tick(ByVal sender As System.Object, ByVal e
As System.EventArgs) Handles tmrVerificarUsuario.Tick
        If Not (Me.usr Is Nothing) Then
            If Not (TypeOf (Me.usr) Is Administrador) Then
                datos.Conectar()
```

```
            Dim HUsr As HorarioUsuario =
datos.SelectHorarioUsuario(CInt(Me.usr.Id))
            datos.Desconectar()
            Dim finicio As DateTime = New DateTime(HUsr.FechaInicio.Year,
HUsr.FechaInicio.Month, HUsr.FechaInicio.Day, HUsr.HoraInicio.Hour,
HUsr.HoraInicio.Minute, 0)
            Dim ffin As DateTime = New DateTime(HUsr.FechaFin.Year,
HUsr.FechaFin.Month, HUsr.FechaFin.Day, HUsr.HoraFin.Hour,
HUsr.HoraFin.Minute, 59)
            If (DateTime.Now < finicio Or DateTime.Now > ffin) Then
               datos.Conectar()
               Dim nuevoid As Integer = datos.GetMaxId("log_horario") + 1
               datos.Desconectar()
               datos.Conectar()
               Dim lg As New LogHorario(nuevoid, CInt(Me.usr.Id), DateTime.Now,
"Usuario logueado fuera de su horario")
               datos.AgregarLogHorario(lg)
               datos.Desconectar()
            End If
         End If
      End If
   End Sub

   Private Sub CerrarSesionToolStripMenuItem_Click(ByVal sender As
System.Object, ByVal e As System.EventArgs) Handles
CerrarSesionToolStripMenuItem.Click
      Me.usr = Nothing
      Me.lstSensores.Items.Clear()
      Me.grpSensores.Controls.Clear()
      Me.grpSensores.BackgroundImage = Nothing
      Me.Text = "Centinela"
      Me.cmbMapas.Items.Clear()
      Me.lblEstado.Text = "Estado: "
      Me.lblTipo.Text = "Tipo: "
      Me.lblPosicion.Text = "Posicion: "
      Me.lblPin.Text = "PIn: "
      Me.MainFormLoad(Me, New EventArgs())
   End Sub

#End Region

   Public Sub New()
      Me.InitializeComponent()
   End Sub

   Public Sub Restaurar(ByVal mb As System.Windows.Forms.MouseButtons)
      If mb = Windows.Forms.MouseButtons.Left Then
```

```
            Me.WindowState = FormWindowState.Maximized
            Try
                Notificador.DelNotifyBox()
            Catch ex As Exception
                'pass
            End Try
        End If
    End Sub

    Public Sub DispararNotificador(ByVal Mensaje As String)
        If Me.WindowState = FormWindowState.Minimized Then
            Notificador.AddNotifyBox(Me.Icon.Handle, Me.Text, "Aviso!", Mensaje)
            Notificador._delegateOfCallBack = New
NotificarBandeja.ShellNot.delegateOfCallBack(AddressOf Restaurar)
        End If
    End Sub

    Public Sub PosicionarHijos()
        Me.lstSensores.Left = 0
        Me.lstSensores.Top = 50
        Me.lstSensores.Height = CInt(Me.Height * 0.7)
        Me.lstSensores.Width = CInt(Me.Width * 0.25)
        Me.cmbMapas.Left = Me.lstSensores.Width + 10
        Me.cmbMapas.Top = 50
        Me.lblSensores.Left = 15
        Me.lblMapa.Left = Me.lstSensores.Width + 15
        Me.lblTipo.Top = Me.lstSensores.Top + Me.lstSensores.Height + 20
        Me.lblPosicion.Top = Me.lblTipo.Top + 20
        Me.lblEstado.Top = Me.lblPosicion.Top + 20
        Me.lblPin.Top = Me.lblEstado.Top + 20
        Me.grpSensores.Left = Me.lstSensores.Width + 10
        Me.grpSensores.Top = Me.cmbMapas.Top + Me.cmbMapas.Height + 10
        Me.grpSensores.Height = Me.Height - 150
        Me.grpSensores.Width = CInt(Me.Width * 0.75) - 30
    End Sub

    Public Sub CargarMapas()
        Me.cmbMapas.Items.Clear()
        Me.datos.Conectar()
        Me.mapas = Me.datos.SelectMapas()
        Me.cargando = True
        For Each mpa As Mapa In Me.mapas
            Me.cmbMapas.Items.Add(mpa.Nombre)
        Next
        Me.cargando = False
        Me.datos.Desconectar()
        If (Me.cmbMapas.Items.Count > 0) Then
```

```
            Me.cmbMapas.SelectedIndex = 0
        End If
    End Sub

    Public Sub CargarSensores()
        datos.Conectar()
        Me.lstSensores.Items.Clear()

        Me.sensores = New List(Of SensorVisual)

        Dim sens As List(Of Sensor) =
datos.SelectSensores(Me.mapas(Me.cmbMapas.SelectedIndex).Id)
        If (Not sens Is Nothing) Then
            Me.lstSensores.Items.Clear()
            Me.cargando = True
            Me.grpSensores.Controls.Clear()
            For Each sen As Sensor In sens
                Me.sensores.Add(New SensorVisual(sen,
Me.mapas(Me.cmbMapas.SelectedIndex)))
                Me.lstSensores.Items.Add(sen.Id.ToString() + " - " + sen.Nombre)
            Next
            Me.cargando = False
            If (Me.sensores.Count > 0) Then
                Me.lstSensores.SelectedIndex = 0
                Dim myleft As Integer = 10
                For Each senvis As SensorVisual In Me.sensores
                    If (senvis.Left = 0) Then
                        senvis.Left = myleft
                        myleft += 50
                        senvis.Top = 10
                    End If
                    senvis.CargarImagen()
                    senvis.BringToFront()
                    AddHandler senvis.Click, AddressOf Me.SensorVisualClick
                    AddHandler senvis.MouseDown, AddressOf
Me.SensorVisualMouseDown
                    AddHandler senvis.MouseMove, AddressOf
Me.SensorVisualMouseMove
                    Me.grpSensores.Controls.Add(senvis)
                Next
            End If
        End If
        datos.Desconectar()
    End Sub

    Private Sub ActualizarPuertosASensores(ByVal sender As Object, ByVal e As
EventArgs) Handles srvPuertos.Tick
```

```
        If Not (Me.usr Is Nothing) Then
            Me.srvPuertos.ActualizarSensores(Me.sensores)
            datos.Conectar()
            Me.lblEstado.Text =
datos.GetNombreEstadoSensor(Me.sensores(Me.lstSensores.SelectedIndex).Sen.
EstadoActual)
            datos.Desconectar()
        End If
    End Sub

    Private Sub GuardarPosSensores()
        If (Not Me.sensores Is Nothing) Then
            For Each senvis As SensorVisual In Me.sensores
                datos.Desconectar()
                datos.Conectar()
                datos.SetPosSensor(senvis.Sen.Id,
Me.mapas(Me.lastCmbMapasIndex).Id, senvis.Location)
                datos.Desconectar()
            Next
        End If
    End Sub

    Private Sub ReporteDiarioToolStripMenuItem_Click(ByVal sender As
System.Object, ByVal e As System.EventArgs) Handles
ReporteDiarioToolStripMenuItem.Click
        XmlSensorLog.GenerarXml("diario")
        Dim frmRptDiario As New frmReporteDiario()
        frmRptDiario.Text = "Reporte de Actividad de los Sensores - Dia Actual"
        frmRptDiario.ShowDialog(Me)
    End Sub

    Private Sub ReporteMensualToolStripMenuItem_Click(ByVal sender As
System.Object, ByVal e As System.EventArgs) Handles
ReporteMensualToolStripMenuItem.Click
        XmlSensorLog.GenerarXml("mensual")
        Dim frmRptMensual As New frmReporteDiario()
        frmRptMensual.Text = "Reporte de Actividad de los Sensores - Mes Actual"
        frmRptMensual.ShowDialog(Me)
    End Sub

    Private Sub ReporteAnualToolStripMenuItem_Click(ByVal sender As
System.Object, ByVal e As System.EventArgs) Handles
ReporteAnualToolStripMenuItem.Click
        XmlSensorLog.GenerarXml("anual")
        Dim frmRptMensual As New frmReporteDiario()
        frmRptMensual.Text = "Reporte de Actividad de los Sensores - Año Actual"
        frmRptMensual.ShowDialog(Me)
```

```
    End Sub

    Private Sub ReporteTotalToolStripMenuItem_Click(ByVal sender As
System.Object, ByVal e As System.EventArgs) Handles
ReporteTotalToolStripMenuItem.Click
        XmlSensorLog.GenerarXml()
        Dim frmRptMensual As New frmReporteDiario()
        frmRptMensual.Text = "Reporte de Actividad de los Sensores"
        frmRptMensual.ShowDialog(Me)
    End Sub

    Private Sub ReportePorSensorToolStripMenuItem_Click(ByVal sender As
System.Object, ByVal e As System.EventArgs) Handles
ReportePorSensorToolStripMenuItem.Click
        XmlSensorLog.GenerarXml()
        Dim frmReportePorSensor As New frmRptPorSensor()
        frmReportePorSensor.Text = "Reporte de Actividad de los Sensores"
        frmReportePorSensor.ShowDialog(Me)
    End Sub
End Class
```

frmMapas.vb

```
Public Class frmMapas
    Private mapas As List(Of ClassLib.Mapa)
    Private datos As ClassLib.AccesoDatos
    Private cargando As Boolean

    Private Sub frmMapas_Load(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles MyBase.Load
        Me.datos = New ClassLib.AccesoDatos()
        Me.CargarGrdMapas()
    End Sub

    Public Sub CargarGrdMapas()
        Me.grdMapas.Rows.Clear()
        datos.Conectar()
        Me.mapas = datos.SelectMapas()
        datos.Desconectar()
        Me.cargando = True
        For Each mpa As ClassLib.Mapa In Me.mapas
            Me.grdMapas.Rows.Add()
            Me.grdMapas.Rows(Me.grdMapas.Rows.Count -
1).Cells("colNombre").Value = mpa.Nombre
        Next
        Me.cargando = False
```

```
        Me.grdMapas_SelectionChanged(Me, New EventArgs())
    End Sub

    Private Sub grdMapas_SelectionChanged(ByVal sender As System.Object,
ByVal e As System.EventArgs) Handles grdMapas.SelectionChanged
        If Not (Me.cargando) Then
            datos.Conectar()
            Dim bytes As Byte() =
datos.GetImgMapa(Me.mapas(Me.grdMapas.CurrentCell.RowIndex))
            If Not (bytes Is Nothing) Then
                Dim mstream As New System.IO.MemoryStream(bytes.Length)
                mstream.Write(bytes, 0, bytes.Length)
                Dim img As New Bitmap(mstream)
                Me.pbxImagenMapa.Image = img
            End If
            datos.Desconectar()
        End If
    End Sub

    Private Sub Button2_Click(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles Button2.Click
        Dim dlg As New OpenFileDialog()
        dlg.Filter = "Imagenes(*.jpg;*.bmp;*.gif;*.png)|*.jpg;*.bmp;*.gif;*.png"
        If (dlg.ShowDialog() = DialogResult.OK) Then
            Dim stream As System.IO.Stream = dlg.OpenFile()
            Dim bytes(CInt(stream.Length)) As Byte
            stream.Read(bytes, 0, CInt(stream.Length))
            datos.Conectar()
            datos.SetImgMapa(Me.mapas(Me.grdMapas.CurrentCell.RowIndex),
bytes)
            datos.Desconectar()
            Me.grdMapas_SelectionChanged(Me, New EventArgs())
        End If
    End Sub

    Private Sub Button3_Click(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles Button3.Click
        Me.Close()
    End Sub

    Private Sub Button1_Click(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles Button1.Click
        If (Me.grdMapas.Rows.Count > 0) Then
            If (MsgBox("Esta usted seguro?", MsgBoxStyle.YesNo, "Confirmacion") =
MsgBoxResult.Yes) Then
                datos.Conectar()
                datos.EliminarMapa(Me.mapas(Me.grdMapas.CurrentCell.RowIndex))
```

```
            datos.Desconectar()
            Me.grdMapas_SelectionChanged(Me, New EventArgs())
        End If
    End If
    Me.CargarGrdMapas()
End Sub

Private Sub btnAceptar_Click(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles btnAceptar.Click
    Dim frmAddMpa As New frmAgregarMapa()
    frmAddMpa.ShowDialog(Me)
    Me.CargarGrdMapas()
End Sub
End Class
```

frmSelectMapa.vb

```
Public Class frmSelectMapa
    Private mapas As List(Of ClassLib.Mapa)
    Private mapasExcluidos As List(Of ClassLib.Mapa)
    Private datos As ClassLib.AccesoDatos
    Private cargando As Boolean
    Private _acepted As Boolean
    Private idsen As Integer

    Public Property Acepted() As Boolean
        Get
            Return Me._acepted
        End Get
        Set(ByVal value As Boolean)
            Me._acepted = value
        End Set
    End Property

    Public ReadOnly Property MapaSeleccionado() As ClassLib.Mapa
        Get
            Return Me.mapas(Me.grdMapas.CurrentCell.RowIndex)
        End Get
    End Property

    Public Sub New(ByVal idsensor As Integer)
        MyBase.New()
        Me.InitializeComponent()
        Me.idsen = idsensor
    End Sub
```

```
    Private Sub frmMapas_Load(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles MyBase.Load
        Me.datos = New ClassLib.AccesoDatos()
        Me.CargarGrdMapas()
    End Sub

    Public Sub CargarGrdMapas()
        Me.grdMapas.Rows.Clear()
        datos.Conectar()
        Me.mapas = datos.SelectMapas()
        datos.Desconectar()
        datos.Conectar()
        Me.mapasExcluidos = datos.SelectMapas(Me.idsen)
        datos.Desconectar()
        Do While (Me.mapasExcluidos.Count > 0)
            For i As Integer = 0 To Me.mapas.Count - 1
                If (Me.mapasExcluidos(0).Id = Me.mapas(i).Id) Then
                    Me.mapas.RemoveAt(i)
                    Me.mapasExcluidos.RemoveAt(0)
                    Exit For
                End If
            Next
        Loop
        Me.cargando = True
        For Each mpa As ClassLib.Mapa In Me.mapas
            Me.grdMapas.Rows.Add()
            Me.grdMapas.Rows(Me.grdMapas.Rows.Count -
1).Cells("colNombre").Value = mpa.Nombre
        Next
        Me.cargando = False
        If (Me.mapas.Count > 0) Then
            Me.grdMapas_SelectionChanged(Me, New EventArgs())
            Me.btnAceptar.Enabled = True
        End If
    End Sub

    Private Sub grdMapas_SelectionChanged(ByVal sender As System.Object,
ByVal e As System.EventArgs) Handles grdMapas.SelectionChanged
        If Not (Me.cargando) Then
            datos.Conectar()
            Dim bytes As Byte() =
datos.GetImgMapa(Me.mapas(Me.grdMapas.CurrentCell.RowIndex))
            If Not (bytes Is Nothing) Then
                Dim mstream As New System.IO.MemoryStream(bytes.Length)
                mstream.Write(bytes, 0, bytes.Length)
                Dim img As New Bitmap(mstream)
                Me.pbxImagenMapa.Image = img
```

```
            End If
            datos.Desconectar()
        End If
    End Sub

    Private Sub Button3_Click(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles btnCancelar.Click
        Me.Acepted = False
        Me.Close()
    End Sub

    Private Sub btnAceptar_Click_1(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles btnAceptar.Click
        Me.Acepted = True
        Me.Close()
    End Sub
End Class
```

frmSensores.vb

```
Imports Centinela.ClassLib

Public Class frmSensores
    Private datos As AccesoDatos
    Private sensores As List(Of Sensor)
    Private mapas As List(Of Mapa)
    Private cargando As Boolean
    Private nombreAnterior As String = ""

    Private Sub btnCerrar_Click(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles btnCerrar.Click
        Me.Close()
    End Sub

    Private Sub frmSensores_Load(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles MyBase.Load
        Me.datos = New AccesoDatos()
        Me.CargarCmbTiposSensores()
        Me.CargarSensores()
    End Sub

    Public Sub CargarSensores()
        datos.Conectar()
        Me.lstSensores.Items.Clear()

        Me.sensores = New List(Of Sensor)
```

```
        Me.sensores = datos.SelectSensores()
        If (Not Me.sensores Is Nothing) Then
            Me.lstSensores.Items.Clear()
            Me.cargando = True
            For Each sen As Sensor In Me.sensores
                Me.lstSensores.Items.Add(sen.Id.ToString() + " - " + sen.Nombre)
            Next
            Me.cargando = False
            If (Me.sensores.Count > 0) Then
                Me.lstSensores.SelectedIndex = 0
            End If
        End If
        datos.Desconectar()

    End Sub

    Private Sub lstSensores_SelectedIndexChanged(ByVal sender As
System.Object, ByVal e As System.EventArgs) Handles
lstSensores.SelectedIndexChanged
        If Not (Me.cargando) Then
            datos.Desconectar()
            datos.Conectar()
            Dim tiposen As String =
datos.GetNombreTipoSensor(Me.sensores(Me.lstSensores.SelectedIndex).Tipo)
            datos.Desconectar()
            For i As Integer = 0 To Me.cmbTipo.Items.Count - 1
                If (Me.cmbTipo.Items(i).ToString() = tiposen) Then
                    Me.cmbTipo.SelectedIndex = i
                End If
            Next
            Me.txtNombre.Text = Me.sensores(Me.lstSensores.SelectedIndex).Nombre
            Me.nombreAnterior = Me.txtNombre.Text
            Me.CargarMapas(Me.sensores(Me.lstSensores.SelectedIndex).Id)
        End If
    End Sub

    Public Sub CargarMapas(ByVal idSen As Integer)
        Me.lstMapas.Items.Clear()
        Me.datos.Conectar()
        Me.mapas = Me.datos.SelectMapas(idSen)
        Me.datos.Desconectar()
        Me.cargando = True
        For Each mpa As Mapa In mapas
            Me.datos.Conectar()
            Dim posSen As System.Drawing.Point = datos.GetPosSensor(idSen,
mpa.Id)
```

```
            Me.datos.Desconectar()
            Dim strPos As String = ", en la posicion (X:" + posSen.X.ToString() + " , Y:"
+ posSen.Y.ToString() + ")"
            Me.lstMapas.Items.Add(mpa.Nombre.Trim() + strPos)
        Next
        Me.cargando = False
        If (Me.lstMapas.Items.Count > 0) Then
            Me.lstMapas.SelectedIndex = 0
        End If
    End Sub

    Public Sub CargarCmbTiposSensores()
        Me.datos.Conectar()
        Dim lst As List(Of String) = Me.datos.GetTiposSensor()
        Me.cmbTipo.Items.Clear()
        For Each tiposen As String In lst
            Me.cmbTipo.Items.Add(tiposen)
        Next
        Me.datos.Desconectar()
    End Sub

    Private Sub cmbTipo_SelectedIndexChanged(ByVal sender As System.Object,
ByVal e As System.EventArgs) Handles cmbTipo.SelectedIndexChanged
        datos.Conectar()
        Me.sensores(Me.lstSensores.SelectedIndex).Tipo =
datos.GetIdTipoSensor(Me.cmbTipo.Text)
        datos.Desconectar()
        datos.Conectar()
        datos.ModificarSensor(Me.sensores(Me.lstSensores.SelectedIndex))
        datos.Desconectar()
    End Sub

    Private Sub txtNombre_Leave(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles txtNombre.Leave
        If (Me.txtNombre.Text <> Me.nombreAnterior) Then
            Me.sensores(Me.lstSensores.SelectedIndex).Nombre = Me.txtNombre.Text
            datos.Conectar()
            datos.ModificarSensor(Me.sensores(Me.lstSensores.SelectedIndex))
            datos.Desconectar()
            Me.nombreAnterior = Me.txtNombre.Text
            Me.CargarSensores()
        End If
    End Sub

    Private Sub btnAgregarMpa_Click(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles btnAgregarMpa.Click
```

```
        Dim frmSelMpa As New
frmSelectMapa(Me.sensores(Me.lstSensores.SelectedIndex).Id)
        frmSelMpa.ShowDialog(Me)
        If (frmSelMpa.Acepted) Then
            Dim mpa As Mapa = frmSelMpa.MapaSeleccionado
            datos.Conectar()
            Dim nuevoId As Integer = datos.GetMaxId("sensor_en_mapa") + 1
            datos.Desconectar()
            datos.Conectar()
            datos.AgregarSensorMapa(nuevoId,
Me.sensores(Me.lstSensores.SelectedIndex).Id, mpa.Id)
            datos.Desconectar()
            Me.CargarMapas(Me.sensores(Me.lstSensores.SelectedIndex).Id)
        End If
    End Sub

    Private Sub btnEliminarMpa_Click(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles btnEliminarMpa.Click
        If (Me.lstMapas.Items.Count > 0) Then
            If (MsgBox("Esta usted seguro?", MsgBoxStyle.YesNo, "Confirmacion") =
MsgBoxResult.Yes) Then
                datos.Conectar()

datos.EliminarSensorMapa(Me.sensores(Me.lstSensores.SelectedIndex).Id,
Me.mapas(Me.lstMapas.SelectedIndex).Id)
                datos.Desconectar()
                Me.CargarMapas(Me.sensores(Me.lstSensores.SelectedIndex).Id)
            End If
        End If
    End Sub

    Private Sub btnAgregarSen_Click(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles btnAgregarSen.Click
        Dim frmaddsen As New frmAgregarSensor()
        frmaddsen.ShowDialog(Me)
        Me.CargarSensores()
    End Sub

    Private Sub btnEliminarSen_Click(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles btnEliminarSen.Click
        If (Me.lstSensores.Items.Count > 0) Then
            If (MsgBox("Esta usted seguro?", MsgBoxStyle.YesNo, "Confirmacion") =
MsgBoxResult.Yes) Then
                datos.Conectar()
                datos.EliminarSensor(Me.sensores(Me.lstSensores.SelectedIndex))
                datos.Desconectar()
                Me.CargarSensores()
```

```
            End If
        End If
    End Sub
End Class
```

frmSesion.vb

```
Imports Centinela.ClassLib

Public Partial Class frmSesion
    Public usr As Usuario
    Private husr As HorarioUsuario
        private datos as New AccesoDatos()

        Public Sub New()
                Me.InitializeComponent()
        End Sub

    Sub BtnCancelarClick(ByVal sender As Object, ByVal e As EventArgs) Handles
btnCancelar.Click
        End
    End Sub

    Sub BtnAceptarClick(ByVal sender As Object, ByVal e As EventArgs) Handles
btnAceptar.Click
        Me.datos.Conectar()
        usr = Me.datos.SelecUsuario(Me.txtUsuario.Text, Me.txtClave.Text)
        Me.datos.Desconectar()
        If Not (usr Is Nothing) Then
            Me.datos.Conectar()
            Me.husr = Me.datos.SelectHorarioUsuario(CInt(Me.usr.Id))
            Me.datos.Desconectar()
            If Not (Me.husr Is Nothing) Then
                Dim finicio As DateTime = New DateTime(husr.FechaInicio.Year,
husr.FechaInicio.Month, husr.FechaInicio.Day, husr.HoraInicio.Hour,
husr.HoraInicio.Minute, 0)
                Dim ffin As DateTime = New DateTime(husr.FechaFin.Year,
husr.FechaFin.Month, husr.FechaFin.Day, husr.HoraFin.Hour,
husr.HoraFin.Minute, 59)
                If (DateTime.Now < finicio Or DateTime.Now > ffin) Then
                    MsgBox("Su horario de acceso no coincide con la fecha y hora
actual", MsgBoxStyle.OkOnly, "Sesion")
                Else
                    Me.Dispose()
                End If
            Else
```

```
            MsgBox("Su horario de acceso no coincide con la fecha y hora actual",
MsgBoxStyle.OkOnly, "Sesion")
         End If
      Else
         Me.datos.Conectar()
         usr = Me.datos.SelecAdministrador(Me.txtUsuario.Text, Me.txtClave.Text)
         Me.datos.Desconectar()
         If Not (usr Is Nothing) Then
            Me.Dispose()
         Else
            MsgBox("Usuario y / o Clave invalidos", MsgBoxStyle.OkOnly, "Sesion")
         End If
      End If
   End Sub

   Sub TextBox1TextChanged(ByVal sender As Object, ByVal e As EventArgs)
Handles txtUsuario.TextChanged
      Me.RevisarBtnAceptar()
   End Sub

   Sub TxtClaveTextChanged(ByVal sender As Object, ByVal e As EventArgs)
Handles txtClave.TextChanged
      Me.RevisarBtnAceptar()
   End Sub
        Sub RevisarBtnAceptar()
                If(Me.txtUsuario.Text = "" Or Me.txtClave.Text = "") Then
                        Me.btnAceptar.Enabled = False
                Else
                        me.btnAceptar.Enabled = true
                End If
        End Sub
End Class
```

frmUsuarios.vb

```
Imports Centinela.ClassLib
Imports System.Collections.Generic

Public Partial Class frmUsuarios
   Dim usrs As List(Of Usuario)
   Dim datos As New AccesoDatos()
   Public Sub New()
      Me.InitializeComponent()
   End Sub

   Sub BtnCerrarClick(ByVal sender As Object, ByVal e As EventArgs) Handles
btnCerrar.Click
```

```
        Me.Close()
    End Sub

    Private Sub frmUsuarios_Load(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles MyBase.Load
        Me.CargarUsuarios()
    End Sub

    Private Sub CargarUsuarios()
        datos.Conectar()
        Me.usrs = datos.SelectUsuarios()
        datos.Desconectar()
        Me.grdUsuarios.Rows.Clear()
        For Each usr As Usuario In usrs
            If (usr.Visible) Then
                Me.grdUsuarios.Rows.Add()
                Me.grdUsuarios.Rows(Me.grdUsuarios.Rows.Count -
1).Cells("colNombreCompleto").Value = usr.NombreCompleto
            End If
        Next
    End Sub

    Private Sub Button1_Click(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles Button1.Click
        Dim cclave As New
frmCambiarClave(Me.usrs(Me.grdUsuarios.CurrentCell.RowIndex).Id)
        cclave.ShowDialog(Me)
    End Sub

    Private Sub btnEliminar_Click(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles btnEliminar.Click
        If (Me.grdUsuarios.Rows.Count > 0) Then
            If (MsgBox("Esta usted seguro?", MsgBoxStyle.YesNo, "Confirmacion") =
MsgBoxResult.Yes) Then
                datos.Conectar()
                datos.EliminarUsuario(Me.usrs(Me.grdUsuarios.CurrentCell.RowIndex))
                datos.Desconectar()
                Me.CargarUsuarios()
            End If
        End If
    End Sub

    Private Sub btnAgregar_Click(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles btnAgregar.Click
        Dim frmaddusr As New frmAddUsuario()
        frmaddusr.ShowDialog(Me)
        Me.CargarUsuarios()
```

```
    End Sub

    Private Sub btnHorario_Click(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles btnHorario.Click
        Dim frmHUsr As New
frmHorarioUsuario(CInt(Me.usrs(Me.grdUsuarios.CurrentCell.RowIndex).Id))
        frmHUsr.ShowDialog(Me)
    End Sub
End Class
```

MinimizarABandeja.vb

```
Imports System.Windows.Forms
Imports System.Drawing

Public Class MinimizarABandeja
    Public WithEvents menuBandeja As New ContextMenu
Estado por defecto Normal
    Public restaurarEstadoVentana As FormWindowState =
FormWindowState.Maximized
    Private WithEvents iconoBandeja As New NotifyIcon
    Private WithEvents frm As New Form
La Aplicacion es visible?
    Private visible As Boolean = True

    'Constructor
    'Toma 3 parametros
    '1)El Main Form de la aplicacion (Me)          --mandatorio
    '2)Texto que se muestra en la bandeja          --optional
    '3)Icono de la app que se muestra en la bandeja --optional
    Sub New(ByVal elFormulario As Form, Optional ByVal textoIcono As String = "",
_
    Optional ByVal icono As Icon = Nothing)
        'assign passed form to global var
        frm = elFormulario
        'add two main menu items to traymenu and event handlers
        menuBandeja.MenuItems.Add("Restaurar", AddressOf restaurar)
        menuBandeja.MenuItems.Add("Cerrar", AddressOf cerrar)
        'add event handler for passed main form to execute
        AddHandler frm.SizeChanged, AddressOf ejecutar
        'add event handler for double clicking the icon in the tray (same as menu item)
        AddHandler iconoBandeja.DoubleClick, AddressOf restaurar
        'properties for trayicon
        '-hide
        '-if icon passed, assign - otherwise assign default icon
        '-assign passed text - if none then will be blank
        '-assign context menu
```

```
        With iconoBandeja
            .Visible = False
            If IsNothing(icono) Then
                .Icon = frm.Icon
            Else
                .Icon = icono
            End If
            .Text = textoIcono
            .ContextMenu = menuBandeja
        End With
    End Sub

    Private Sub restaurar(ByVal sender As System.Object, ByVal e As
System.EventArgs)
        frm.ShowInTaskbar = True
        visible = True
        iconoBandeja.Visible = False
        frm.WindowState = restaurarEstadoVentana
    End Sub

    Private Sub cerrar(ByVal sender As System.Object, ByVal e As
System.EventArgs)
        iconoBandeja.Visible = False
        frm.Close()
    End Sub

    Private Sub ejecutar(ByVal sender As System.Object, ByVal e As
System.EventArgs)
        'This fires on form size changine so...
        'If form is visible and is being minimized, then
        '-hide app in taskbar
        '-show icon in system tray
        If visible And frm.WindowState = FormWindowState.Minimized Then
            visible = False
            frm.ShowInTaskbar = False
            iconoBandeja.Visible = True
        End If
    End Sub

End Class
```

NotificarEnBandeja.vb

```
Imports System.IO
Imports System.Diagnostics
Imports System
Imports System.Windows.Forms
```

```
Imports System.Runtime.InteropServices

Namespace NotificarBandeja

#Region "FormTmp"
    Friend Class FormTmp
        Inherits System.Windows.Forms.Form

        Private servicesClass As ShellNot = Nothing
        Friend Sub New(ByVal _servicesClass As ShellNot)
            servicesClass = _servicesClass
        End Sub

        Private Const WM_LBUTTONDOWN As Integer = 513
        Private Const WM_RBUTTONDOWN As Integer = 516
        Private Const WM_MBUTTONDOWN As Integer = 519

        ' for popup the menu
        <DllImport("user32.dll", EntryPoint:="TrackPopupMenu")> _
        Private Shared Function TrackPopupMenu(ByVal hMenu As IntPtr, ByVal
wFlags As Integer, ByVal x As Integer, ByVal y As Integer, ByVal nReserved As
Integer, ByVal hwnd As IntPtr, _
        ByRef lprc As RECT) As Integer
        End Function

        <StructLayout(LayoutKind.Sequential)> _
        Private Structure RECT
            Friend Left As Integer
            Friend Top As Integer
            Friend Right As Integer
            Friend Bottom As Integer
        End Structure

        Protected Overloads Overrides Sub WndProc(ByRef msg As Message)
            If msg.Msg = servicesClass.WM_NOTIFY_TRAY Then
                If CInt(msg.WParam) = servicesClass.uID Then
                    Dim mb As System.Windows.Forms.MouseButtons =
System.Windows.Forms.MouseButtons.None
                    If CInt(msg.LParam) = WM_LBUTTONDOWN Then
                        ' left click
                        mb = System.Windows.Forms.MouseButtons.Left
                    ElseIf CInt(msg.LParam) = WM_MBUTTONDOWN Then
                        ' middle click
                        mb = System.Windows.Forms.MouseButtons.Middle
                    ElseIf CInt(msg.LParam) = WM_RBUTTONDOWN Then
                        ' right click
                        If servicesClass.contextMenuHwnd <> IntPtr.Zero Then
```

```
                    ' if connect my menu exist.
                    Dim r As New RECT()
                    r.Left =
System.Windows.Forms.Screen.PrimaryScreen.WorkingArea.Left
                    r.Right =
System.Windows.Forms.Screen.PrimaryScreen.WorkingArea.Right
                    r.Top =
System.Windows.Forms.Screen.PrimaryScreen.WorkingArea.Top
                    r.Bottom =
System.Windows.Forms.Screen.PrimaryScreen.WorkingArea.Right

                    TrackPopupMenu(servicesClass.contextMenuHwnd, 2,
System.Windows.Forms.Cursor.Position.X,
System.Windows.Forms.Cursor.Position.Y, 0, servicesClass.formHwnd, _
                    r)
                Else
                    ' else callback mousebuttons.right
                    mb = System.Windows.Forms.MouseButtons.Right
                End If
            End If

            If mb <> System.Windows.Forms.MouseButtons.None AndAlso
servicesClass._delegateOfCallBack IsNot Nothing Then
                servicesClass._delegateOfCallBack(mb)
                ' run delegate
                Return
            End If
        End If
    End If

    MyBase.WndProc(msg)
  End Sub
 End Class
#End Region

ShellNot Class.
 Public Class ShellNot
  Public Shared ReadOnly myVersion As New System.Version(1, 2)
  'version const
  Private ReadOnly formTmp As FormTmp = Nothing
  ' must here, create message loop
  Private ReadOnly formTmpHwnd As IntPtr = IntPtr.Zero
  ' formtmp.Handle
  Private ReadOnly VersionOk As Boolean = False
  ' version check pass, if false u can use notifyicon with .net framework
  Private forgetDelNotifyBox As Boolean = False
  ' if forget DelNotifyBox we can DelNotifyBox automation
```

```
        Friend formHwnd As IntPtr = IntPtr.Zero
        ' for multi-notifyicon
        Friend contextMenuHwnd As IntPtr = IntPtr.Zero
        ' for multi-notifyicon
        Friend Delegate Sub delegateOfCallBack(ByVal mb As
System.Windows.Forms.MouseButtons)
        Friend _delegateOfCallBack As delegateOfCallBack = Nothing

        Public Sub New()
            WM_NOTIFY_TRAY += 1
            ' id+1 for message, support multi-notifyicon
            uID += 1
            formTmp = New FormTmp(Me)
            ' create new message loop
            formTmpHwnd = formTmp.Handle
            ' handle
            ' check shell32.dll version, must 5.0 (ie 5.0)
            VersionOk = Me.GetShell32VersionInfo() >= 5
        End Sub
        Protected Overrides Sub Finalize()
            Try

                If forgetDelNotifyBox Then
                    Me.DelNotifyBox()
                    ' forget?
                End If
            Finally
                MyBase.Finalize()
            End Try
        End Sub

#Region "API_Consts"
        Friend ReadOnly WM_NOTIFY_TRAY As Integer = 1024 + 2001
        Friend ReadOnly uID As Integer = 5000

        ' see shellapi.h
        Private Const NIIF_NONE As Integer = 0
        Private Const NIIF_INFO As Integer = 1
        Private Const NIIF_WARNING As Integer = 2
        Private Const NIIF_ERROR As Integer = 3

        Private Const NIF_MESSAGE As Integer = 1
        Private Const NIF_ICON As Integer = 2
        Private Const NIF_TIP As Integer = 4
        Private Const NIF_STATE As Integer = 8
        Private Const NIF_INFO As Integer = 16
```

```
        Private Const NIM_ADD As Integer = 0
        Private Const NIM_MODIFY As Integer = 1
        Private Const NIM_DELETE As Integer = 2
        Private Const NIM_SETFOCUS As Integer = 3
        Private Const NIM_SETVERSION As Integer = 4

        Private Const NIS_HIDDEN As Integer = 1
        Private Const NIS_SHAREDICON As Integer = 2

        Private Const NOTIFYICON_OLDVERSION As Integer = 0
        Private Const NOTIFYICON_VERSION As Integer = 3

        ' o, this's master
        <DllImport("shell32.dll", EntryPoint:="Shell_NotifyIcon")> _
        Private Shared Function Shell_NotifyIcon(ByVal dwMessage As Integer,
ByRef lpData As NOTIFYICONDATA) As Boolean
        End Function

if "this.focus()" can't work, u can try this
        ''' <param name="hwnd">this.Handle</param>
        <DllImport("user32.dll", EntryPoint:="SetForegroundWindow")> _
        Public Shared Function SetForegroundWindow(ByVal hwnd As IntPtr) As
Integer
        End Function

        <StructLayout(LayoutKind.Sequential)> _
        Private Structure NOTIFYICONDATA
            Friend cbSize As Integer
            Friend hwnd As IntPtr
            Friend uID As Integer
            Friend uFlags As Integer
            Friend uCallbackMessage As Integer
            Friend hIcon As IntPtr
            <MarshalAs(UnmanagedType.ByValTStr, SizeConst:=128)> _
            Friend szTip As String
            Friend dwState As Integer
            Friend dwStateMask As Integer
            <MarshalAs(UnmanagedType.ByValTStr, SizeConst:=255)> _
            Friend szInfo As String
            Friend uTimeoutAndVersion As Integer
            <MarshalAs(UnmanagedType.ByValTStr, SizeConst:=64)> _
            Friend szInfoTitle As String
            Friend dwInfoFlags As Integer
        End Structure
#End Region
```

Nueva estructura de NOTIFYICONDATA

```
    Private Function GetNOTIFYICONDATA(ByVal iconHwnd As IntPtr, ByVal
sTip As String, ByVal boxTitle As String, ByVal boxText As String) As
NOTIFYICONDATA
        Dim nData As New NOTIFYICONDATA()

        nData.cbSize = System.Runtime.InteropServices.Marshal.SizeOf(nData)
        ' struct size
        nData.hwnd = formTmpHwnd
        ' WndProc form
        nData.uID = uID
        ' message WParam, for callback
        nData.uFlags = NIF_MESSAGE Or NIF_ICON Or NIF_TIP Or NIF_INFO
        ' Flags
        nData.uCallbackMessage = WM_NOTIFY_TRAY
        ' message ID, for call back
        nData.hIcon = iconHwnd
        ' icon handle, for animation icon if u need
        nData.uTimeoutAndVersion = 10 * 1000 Or NOTIFYICON_VERSION
        ' "Balloon Tooltip" timeout
        nData.dwInfoFlags = NIIF_INFO
        ' info type Flag, u can try waring, error, info
        nData.szTip = sTip
        ' tooltip message
        nData.szInfoTitle = boxTitle
        ' "Balloon Tooltip" title
        nData.szInfo = boxText
        ' "Balloon Tooltip" body
        Return nData
    End Function

    Private Function GetShell32VersionInfo() As Integer
        ' getversion of shell32
        Dim fi As New
FileInfo(Path.Combine(System.Environment.SystemDirectory, "shell32.dll"))
        If fi.Exists Then
            Dim theVersion As FileVersionInfo =
FileVersionInfo.GetVersionInfo(fi.FullName)
            Dim i As Integer = theVersion.FileVersion.IndexOf("."c)
            If i > 0 Then
                Try
                    Return Integer.Parse(theVersion.FileVersion.Substring(0, i))
                Catch
                End Try
            End If
        End If
        Return 0
    End Function
```

```
Agrega una Nueva Notificacion
        ''' <param name="iconHwnd">this.icon.handle</param>
        ''' <param name="sTip">tooltip, 5.0 max: 128 char</param>
        ''' <param name="boxTitle">"Balloon Tooltip" title, max: 64 char</param>
        ''' <param name="boxText">"Balloon Tooltip" body, max: 256 char</param>
        ''' <returns>true,false,error(-1)</returns>
        Public Function AddNotifyBox(ByVal iconHwnd As IntPtr, ByVal sTip As
String, ByVal boxTitle As String, ByVal boxText As String) As Integer
            If Not Me.VersionOk Then
                Return -1
            End If

            Dim nData As NOTIFYICONDATA = GetNOTIFYICONDATA(iconHwnd,
sTip, boxTitle, boxText)
            If Shell_NotifyIcon(NIM_ADD, nData) Then
                Me.forgetDelNotifyBox = True
                Return 1
            Else
                Return 0
            End If
        End Function

Elimina el icono de notificacion
        Public Function DelNotifyBox() As Integer
            If Not Me.VersionOk Then
                Return -1
            End If

            Dim nData As NOTIFYICONDATA = GetNOTIFYICONDATA(IntPtr.Zero,
Nothing, Nothing, Nothing)
            If Shell_NotifyIcon(NIM_DELETE, nData) Then
                Me.forgetDelNotifyBox = False
                Return 1
            Else
                Return 0
            End If
        End Function

        Public Function ModiNotifyBox(ByVal iconHwnd As IntPtr, ByVal sTip As
String, ByVal boxTitle As String, ByVal boxText As String) As Integer
            If Not Me.VersionOk Then
                Return -1
            End If
            Dim nData As NOTIFYICONDATA = GetNOTIFYICONDATA(iconHwnd,
sTip, boxTitle, boxText)
            Return CInt(IIf(Shell_NotifyIcon(NIM_MODIFY, nData), 1, 0))
```

```
        End Function

#Region "Optional Module"
        ''' <param name="_formHwnd">main form handle, try: this.handle</param>
        ''' <param name="_contextMenuHwnd">contextMenu.handle</param>
        Public Sub ConnectMyMenu(ByVal _formHwnd As IntPtr, ByVal
_contextMenuHwnd As IntPtr)
            formHwnd = _formHwnd
            contextMenuHwnd = _contextMenuHwnd
        End Sub

        Public Sub Dispose()
            _delegateOfCallBack = Nothing
            Me.formTmp.Dispose()
        End Sub

        Public ReadOnly Property VersionPass() As Boolean
            Get
                Return Me.VersionOk
            End Get
        End Property
#End Region
    End Class
End Namespace
```

Program.vb

```
Imports Microsoft.VisualBasic.ApplicationServices

Namespace My
Este archivo controla el comportamiento de la aplicacion
        Partial Class MyApplication
                Public Sub New()
                        MyBase.New(AuthenticationMode.Windows)
                        Me.IsSingleInstance = False
                        Me.EnableVisualStyles = True
                        Me.SaveMySettingsOnExit = False
        Me.ShutDownStyle = ShutdownMode.AfterMainFormCloses
                End Sub

                Protected Overrides Sub OnCreateMainForm()
         Me.MainForm = My.Forms.frmMain
                End Sub
        End Class
End Namespace
```

SensorVisual.vb

```
Imports Centinela.ClassLib
Imports XmlClassLib

Representa un sensor visualizable en un formulario sobre un mapa

Public Class SensorVisual
    Inherits PictureBox

#Region "Campos"
Acceso a los datos de la base de datos
    Private datos As AccesoDatos
Ruta donde buscar las imagenes
    Private _rutaImgs As String
Clase que permite leer el archivo de configuracion
    Private cfg As ConfigManager
El nombre lo dice todo
    Private WithEvents ttp As ToolTip
Para reproducir el sonido de alarma
    Private snd As Microsoft.VisualBasic.Devices.Audio
Ruta del archivo de sonido de alarma
    Private sndFile As String
El objeto sensor
    Private WithEvents _sen As ClassLib.Sensor

#End Region

#Region "Propiedades"

Devuelve o establece el sensor contenido
    Public Property Sen() As ClassLib.Sensor
        Get
            Return Me._sen
        End Get
        Set(ByVal value As ClassLib.Sensor)
            Me._sen = value
        End Set
    End Property

#End Region

#Region "Metodos"

Crea una instancia de la clase SensorVisual
    ''' <param name="sen">El sensor asociado</param>
```

```
    ''' <param name="mpaActual">el mapa en el actualmente se muestra el
sensor</param>
    Public Sub New(ByVal sen As ClassLib.Sensor, ByVal mpaActual As Mapa)
        MyBase.New()
        Me._rutaImgs = Application.StartupPath + "\Media\"
        Me.cfg = New ConfigManager("app.config")
        Me.datos = New AccesoDatos()
        Me._sen = sen
        Me.Height = 20
        Me.Width = 20
        Me.CambiarMapa(mpaActual)
        Me.BackColor = Color.LightGray
        Me.SizeMode = PictureBoxSizeMode.StretchImage
        Me.ttp = New ToolTip()
        Me.ttp.IsBalloon = True
        Me.ttp.ShowAlways = True
        Me.ttp.InitialDelay = 0
        Me.ttp.ToolTipTitle = Me.Sen.Nombre
        Me.ttp.ToolTipIcon = ToolTipIcon.Info
        Me.ttp.SetToolTip(Me, "Posicion: (X:" + Me.Location.X.ToString() + ", Y:" +
Me.Location.Y.ToString() + ")")
        Me.snd = New Microsoft.VisualBasic.Devices.Audio()
        If (Me._sen.EstadoActual = 3) Then
            Me.sndFile = Me._rutaImgs + Me.cfg("sndalarmasensor")
            snd.Play(sndFile, AudioPlayMode.BackgroundLoop)
        End If
    End Sub

Cambia de mapa y toma la nueva posicion relativa
    ''' <param name="mpaActual">El nuevo mapa en el cual se muestra</param>
    Public Sub CambiarMapa(ByVal mpaActual As Mapa)
        datos.Conectar()
        Me.Location = datos.GetPosSensor(Sen.Id, mpaActual.Id)
        datos.Desconectar()
    End Sub

Cambia la imagen del sensor de acuerdo a su estado actual
    Public Sub CargarImagen()
        If (Me._sen.EstadoActual = 1) Then
            Me.ImageLocation = Me._rutaImgs + Me.cfg("imgdesconectado")
        ElseIf (Me._sen.EstadoActual = 2) Then
            Me.ImageLocation = Me._rutaImgs + Me.cfg("imgdesactivado")
        ElseIf (Me._sen.EstadoActual = 3) Then
            Me.ImageLocation = Me._rutaImgs + Me.cfg("imgactivado")
        End If
    End Sub
```

```
Actualiza el sensor visual cuando el sensor se apaga
    ''' <param name="focur">La fecha cuando ocurre el suceso</param>
    Public Sub Apagado(ByVal focur As Date) Handles _sen.onApagar
        Me.ttp.Show("Nuevo estado: Desconectado", Me, 10000)
        Me.ActualizarSensor()
    End Sub

Actualiza el sensor visual cuando el sensor se enciende
    ''' <param name="focur">La fecha cuando ocurre el suceso</param>
    Public Sub Encendido(ByVal focur As Date) Handles _sen.onEncender
        Me.ttp.Show("Nuevo estado: Encendido", Me, 10000)
        Me.ActualizarSensor()
    End Sub

Actualiza el sensor visual cuando el sensor se activa
    ''' <param name="focur">La fecha cuando ocurre el suceso</param>
    Public Sub Activado(ByVal focur As Date) Handles _sen.onActivar
        Me.ttp.Show("Nuevo estado: Activado", Me, 10000)
        Me.sndFile = Me._rutaImgs + Me.cfg("sndalarmasensor")
        snd.Play(sndFile, AudioPlayMode.BackgroundLoop)
        Me.ActualizarSensor()
    End Sub

Actualiza el sensor visual cuando el sensor se desactiva
    ''' <param name="focur">La fecha cuando ocurre el suceso</param>
    Public Sub Desactivado(ByVal focur As Date) Handles _sen.onDesactivar
        Me.ttp.Show("Nuevo estado: Desactivado", Me, 10000)
        snd.Stop()
        Me.ActualizarSensor()
    End Sub

Actualiza el sensor visual cuando ocurre un suceso en  su sensor interno
    Public Sub ActualizarSensor()
        Me.CargarImagen()
        datos.Conectar()
        datos.ModificarSensor(Me.Sen)
        datos.Desconectar()
        If Not (TypeOf (frmMain.usr) Is Administrador) Then
            datos.Conectar()
            Dim nuevoid As Integer = datos.GetMaxId("tb_log") + 1
            datos.Desconectar()
            Dim lg As New Log(nuevoid, Me.Sen.Id, Me.Sen.EstadoActual, "Cambio de
estado", DateTime.Now, CInt(frmMain.usr.Id), Me.Sen.Nombre)
            datos.Conectar()
            datos.AgregarLog(lg)
            datos.Desconectar()
        End If
```

```
    End Sub

#End Region
End Class
```

Servicio.vb

```
Representa un servicio de revision del puerto paralelo y actualizacion de sensores
Public Class Servicio
    Inherits Timer

Lee datos del puerto paralelo
    ''' <param name="PortAddress">Direccion del puerto</param>
    Public Declare Function Inp Lib "inpout32.dll" Alias "Inp32" (ByVal PortAddress
As Short) As Short
Escribe datos en el puerto paralelo
    ''' <param name="PortAddress">Direccion del puerto</param>
    ''' <param name="Value">Valor decimal a escribir</param>
    Public Declare Sub Outp Lib "inpout32.dll" Alias "Out32" (ByVal PortAddress As
Short, ByVal Value As Short)

#Region "Campos"

Arreglo que contiene los bits leidos del puerto de datos
    Private _datos As List(Of Short)
Arreglo que contiene los bits leidos del puerto de control
    Private _control As List(Of Short)
Arreglo que contiene los bits leidos del puerto de status
    Private _status As List(Of Short)
Representa la direccion del puerto de datos
    Public Const PUERTODATOS As Integer = 888 'Activado/Desactivado
Representa la direccion del puerto de status
    Public Const PUERTOSTATUS As Integer = 889
Representa la direccion del puerto de control
    Public Const PUERTOCONTROL As Integer = 890
Constante utilizada para representar la lectura de los 1ros 8 pines
    Public Const PRIMEROSOCHOPINES As Boolean = True
Constante utilizada para representar la lectura de los 2dos 8 pines
    Public Const SEGUNDOSOCHOPINES As Boolean = False
Evento disparado cuando cambian los valores en el puerto de datos
    Public Event CambioPuertoDatos(ByVal nuevoVal As List(Of Short))
Evento disparado cuando cambian los valores en el puerto de control
    Public Event CambioPuertoControl(ByVal nuevoVal As List(Of Short))
Evento disparado cuando cambian los valores en el puerto de status
    Public Event CambioPuertoStatus(ByVal nuevoVal As List(Of Short))

#End Region
```

```
#Region "Propiedades"

Devuelve o establece la lista de bits leidos del puerto de datos
    Public Property Datos() As List(Of Short)
        Get
            Return Me._datos
        End Get
        Set(ByVal value As List(Of Short))
            Me._datos = value
        End Set
    End Property

Devuelve o establece la lista de bits leidos del puerto de control
   Public Property Control() As List(Of Short)
        Get
            Return Me._control
        End Get
        Set(ByVal value As List(Of Short))
            Me._control = value
        End Set
    End Property

Devuelve o establece la lista de bits leidos del puerto de status
    Public Property Status() As List(Of Short)
        Get
            Return Me._status
        End Get
        Set(ByVal value As List(Of Short))
            Me._status = value
        End Set
    End Property

#End Region

#Region "Metodos"

Crea una instancia de la clase servicio
    Public Sub New()
        MyBase.New()
        Me.Init()
    End Sub

Crea una instancia de la clase servicio
    Public Sub New(ByVal contanierrr As System.ComponentModel.IContainer)
        MyBase.New(contanierrr)
        Me.Init()
```

```
    End Sub
Inicializa los miembros de la clase
    Public Sub Init()
        Me._datos = New List(Of Short)(8)
        Me._control = New List(Of Short)(8)
        Me._status = New List(Of Short)(8)
    End Sub

Verifica el estado de los puertos
''' <param name="leerPrimerosOcho">Define si se leen los primeros ocho o los
segundos</param>
    Public Sub VerificarPuertos(ByVal leerPrimerosOcho As Boolean)
        If (leerPrimerosOcho) Then
            Outp(Servicio.PUERTOCONTROL, 240) 'Leer los 1ros 8 bits
        Else
            Outp(Servicio.PUERTOCONTROL, 242) 'Leer los 2dos 8 bits
        End If
        Me.Init()
        Dim entradaDatos As Short = Inp(Servicio.PUERTODATOS)
        Dim pDatos As Short() = DecToBin(entradaDatos)
        Dim entradaControl As Short = Inp(Servicio.PUERTOCONTROL)
        Dim pControl As Short() = DecToBin(entradaControl)
        Dim entradaStatus As Short = Inp(Servicio.PUERTOSTATUS)
        Dim pStatus As Short() = DecToBin(entradaControl)

        Me._datos.Clear()
        Me._control.Clear()
        Me._status.Clear()
        For i As Integer = 0 To 7
            Me._datos.Add(pDatos(i))
            Me._control.Add(pControl(i))
            Me._status.Add(pStatus(i))
        Next
    End Sub

Actualiza el estado del puerto de encendido
    ''' <param name="bites">los bites a escribir</param>
    Public Sub ActualizarPuertoEncendido(ByVal bites As Short())
        Outp(Servicio.PUERTOSTATUS, Me.BinToDec(bites))
    End Sub
Actualiza el estado del puerto de activacion
    ''' <param name="bites">los bites a escribir</param>
    Public Sub ActualizarPuertoActivacion(ByVal bites As Short())
        Outp(Servicio.PUERTOCONTROL, Me.BinToDec(bites))
    End Sub
Convierte un arreglo de valores que representan un numero binario a su
equivalente decimal
```

```
    ''' <param name="bites">los bites a convertir</param>
    Public Function BinToDec(ByVal bites As Short()) As Short
        Dim valor As Short = 0
        For i As Integer = 0 To bites.Length - 1
            valor += CType(((2 ^ i) * bites(bites.Length - 1 - i)), Short)
        Next
    End Function
Convierte un numero decimal su equivalente en un arreglo de bits
    ''' <param name="NDecimal">el numero decimal a convertir</param>
    Public Function DecToBin(ByVal NDecimal As Short) As Short()

        Dim Binario(7) As Short
        Binario(0) = 0
        Binario(1) = 0
        Binario(2) = 0
        Binario(3) = 0
        Binario(4) = 0
        Binario(5) = 0
        Binario(6) = 0
        Binario(7) = 0
        If NDecimal = 0 Then
            Return Binario
        End If


        Dim bin As String = ""
        Do While Not NDecimal = 1
            bin = bin + (NDecimal Mod 2).ToString.Trim
            NDecimal = CShort(NDecimal \ 2)
        Loop
        bin = bin + "1"
        Try
            Binario(0) = CShort(bin.Substring(0, 1))
        Catch ex As Exception
            Binario(0) = 0
        End Try
        Try
            Binario(1) = CShort(bin.Substring(1, 1))
        Catch ex As Exception
            Binario(1) = 0
        End Try
        Try
            Binario(2) = CShort(bin.Substring(2, 1))
        Catch ex As Exception
            Binario(2) = 0
        End Try
        Try
```

```
            Binario(3) = CShort(bin.Substring(3, 1))
        Catch ex As Exception
            Binario(3) = 0
        End Try
        Try
            Binario(4) = CShort(bin.Substring(4, 1))
        Catch ex As Exception
            Binario(4) = 0
        End Try
        Try
            Binario(5) = CShort(bin.Substring(5, 1))
        Catch ex As Exception
            Binario(5) = 0
        End Try
        Try
            Binario(6) = CShort(bin.Substring(6, 1))
        Catch ex As Exception
            Binario(6) = 0
        End Try
        Try
            Binario(7) = CShort(bin.Substring(7, 1))
        Catch ex As Exception
            Binario(7) = 0
        End Try


        Return Binario
    End Function

Actualiza el estado de los sensores
    ''' <param name="sensores">La lista de sensores a actualizar</param>
    Public Sub ActualizarSensores(ByRef sensores As List(Of SensorVisual))
        'Verificando los sensores asociados con los "1ros 8 pines" del puerto de
datos...
        Me.VerificarPuertos(Servicio.PRIMEROSOCHOPINES)
        For Each senvis As SensorVisual In sensores
            If (senvis.Sen.Pin <= 8) Then
                Dim estadoSen As Integer = CInt(Me.Datos(senvis.Sen.Pin - 1))
                If (estadoSen = 1) Then
                    senvis.Sen.Desactivar()
                ElseIf (estadoSen = 0) Then
                    senvis.Sen.Activar()
                    frmMain.DispararNotificador("Sensor Activado")
                End If
            End If
        Next
```

```
        'Verificando los sensores asociados con los "2dos 8 pines" del puerto de
datos...
        Me.VerificarPuertos(Servicio.SEGUNDOSOCHOPINES)
        For Each senvis As SensorVisual In sensores
            If (senvis.Sen.Pin > 8) Then
                Dim estadoSen As Integer = CInt(Me.Datos(senvis.Sen.Pin - 9))
                If (estadoSen = 1) Then
                    senvis.Sen.Desactivar()
                ElseIf (estadoSen = 0) Then
                    senvis.Sen.Activar()
                    frmMain.DispararNotificador("Sensor Activado")
                End If
            End If
        Next
    End Sub

#End Region

End Class
```

Aplicación Web: MonitoreoWeb.vbproj

Acercade.aspx

```
<%@ Page Language="VB" AutoEventWireup="true"%>

<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">

<html xmlns="http://www.w3.org/1999/xhtml" >
<head runat="server">
    <title>ACERCA DE...</title>
    <link href="estilo.css" rel="stylesheet" type="text/css" />
</head>
<body>
    <form id="frmAcercaDe" runat="server"><br />
<div id="container">
<div id="core_header">
     <div id="header_text">
         ACERCA DE...</div></div>
          <div id="core_container">
          <div id="core_container2">
                    <div id="core_left">
                        <div id="navcontainer">
                    <ul>
                   <li><a href="Default.aspx" title="Hogar">Hogar</a></li>
```

```
                <!-- <li><a href="#" title="generico">generico</a></li> -->
</ul>
                    </div>
                 </div>
        <div id="core_right">
          <div class="content-box" style="text-align: center">
                <h3 style="font-weight: bold; text-align: left">
               <h3>
                  Proyecto...</h3>
             </h3>
             <strong>Nombre 1<br />
                Nombre 2<br />
                <br />
                Universidad de El Salvador, Facultad multidisciplinaria de
Occidente.<br />
                Fecha</strong></div>
        <!-- El contenido va aqui... -->
          <%--<div class="content-box">
          </div>--%>
        <!-- Final de el contenido de la pagina -->
        </div>
        </div>
        <div id="footer2"><div id="footer_line"></div>
            <div id='footer_text'>Copyleft &copy; ...</div>
           </div>
        </div>
</div>
<div id="footer"></div>
</form>
</body>
</html>
```

Default.aspx

```
<%@ Page Language="VB" AutoEventWireup="true" %>

<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">

<script runat="server">
    Protected Sub lgn1_Authenticate(ByVal sender As Object, ByVal e As
System.Web.UI.WebControls.AuthenticateEventArgs)
        Try
            Dim ds As New Data.DataSet
            Dim con As New
Data.SqlClient.SqlConnection(Me.SqlDataSource1.ConnectionString)
```

```
            con.Close()
            con.Open()
            Dim cmd As New Data.SqlClient.SqlCommand("Validar_Login", con)
            Dim PrmUsuario As Data.SqlClient.SqlParameter =
cmd.Parameters.AddWithValue("@Usuario", lgn1.UserName.Trim())
            Dim PrmClave As Data.SqlClient.SqlParameter =
cmd.Parameters.AddWithValue("@Clave", lgn1.Password.Trim())
            cmd.CommandType = Data.CommandType.StoredProcedure
            Dim da As New Data.SqlClient.SqlDataAdapter(cmd)
            da.Fill(ds)
            con.Close()
            If ds.Tables(0).Rows.Count > 0 Then
               e.Authenticated = True
               Response.Redirect(".\privado\Monitoreo.aspx")
            Else
               Response.Redirect(".\error\error.htm")
               e.Authenticated = False
            End If
            Exit Sub
         Catch ex As Exception
            'MessageBox.Show("Error While Executing the Stored Procedures", _
            '    "Executing Stored Procedure Error", _
            '    MessageBoxButtons.OK, MessageBoxIcon.Exclamation)
         End Try
      End Sub
</script>

<html xmlns="http://www.w3.org/1999/xhtml" >
<head runat="server">
   <title>Proyecto</title>
   <link href="estilo.css" rel="stylesheet" type="text/css" />
</head>
<body>
   <form id="frm_hogar" runat="server"><br />
<div id="container">
<div id="core_header">
    <div id="header_text">
       Proyecto
    </div>
</div>
         <div id="core_container">
         <div id="core_container2">
                  <div id="core_left">
                      <div id="navcontainer">
                <ul>
                <li><a href="Default.aspx" title="Hogar">Hogar</a></li>
                <li><a href="Acercade.aspx" title="Acercade">Acerca de...</a></li>
```

```
                </ul>
                <asp:SqlDataSource ID="SqlDataSource1" runat="server"
ConnectionString="<%$ ConnectionStrings:dbproyectoConnectionString %>"
                    SelectCommand="SELECT [usuario], [clave], [visible] FROM
[Usuario]" DataSourceMode="DataReader"></asp:SqlDataSource>
                    </div>
                </div>
        <div id="core_right">
          <div class="content-box" style="text-align: center">
                <h3 style="font-weight: bold; text-align: left">
                  Regístrate</h3>
              <strong>Regístrate para accesar al servicio de monitoreo que se brinda
esta página.<br />
              </strong>
          </div>
        <!-- El contenido va aqui... -->
          <div class="content-box">
                <h3>
                  Login</h3>
              <table width="100%">
                <tr>
                  <td style="width: 60%; height: 100px;">
                    <div class="dropcap">
                      Acceso a prototipo de monitoreo de sensores....
                    </div>
                  </td>
                  <td style="width: 41%; height: 100px;">
                     <asp:Login ID="lgn1" runat="server" BackColor="#F7F6F3"
BorderColor="#E6E2D8" BorderPadding="4" BorderStyle="Solid"
BorderWidth="1px" Font-Names="Verdana" Font-Size="Small"
ForeColor="#333333" Height="111px" Width="239px"
OnAuthenticate="lgn1_Authenticate">
                      <TitleTextStyle BackColor="#5D7B9D" Font-Bold="True" Font-
Size="0.9em" ForeColor="White" />
                      <InstructionTextStyle Font-Italic="True" ForeColor="Black" />
                      <TextBoxStyle Font-Size="0.8em" />
                      <LoginButtonStyle BackColor="#FFFBFF"
BorderColor="#CCCCCC" BorderStyle="Solid" BorderWidth="1px"
                        Font-Names="Verdana" Font-Size="0.8em"
ForeColor="#284775" />
                    </asp:Login>
                  </td>
                </tr>
              </table>
          </div>
        <!-- Final de el contenido de la pagina -->
        </div>
```

```
        </div>
        <div id="footer2"><div id="footer_line"></div>
            <div id='footer_text'>Copyleft &copy; ...</div>
           </div>
        </div>
</div>
<div id="footer"></div>
</form>
</body>
</html>
```

PRUEBAS Y DEPURACIÓN

PRUEBAS DEL SISTEMA

La prueba del sistema consiste en la instalación de cada uno de los tipos de sensores existentes y útiles para la institución.

Estos sensores van conectados al dispositivo de transmisión de datos creado anteriormente, con la capacidad de controlar 16 sensores, ubicado en uno de los salones de laboratorio de ingeniería que se encuentra localizado en el edificio de usos múltiples, segunda planta.

El Prototipo de software estará instalado en una de las maquinas de dicho laboratorio, la cual esta maquina deberá estar conectada a la red interna de la facultad, ya que este es el medio por el cual se podrá visualizar la aplicación Web.

La aplicación es la que realizara la labor de administración y captura de datos sobre el estado de cada uno de los sensores debidamente conectado. Lo cual cada operación quedara almacenada en la base de datos como la fecha, suceso, al igual que el tipo de origen; para su debida utilización posteriormente.

Como primer paso se debe proceder a la adecuación de las instalaciones donde el equipo será instalado, es decir preparar adecuadamente los puntos en los cuales se instalaran tanto los sensores, dispositivo de transmisión, como el equipo de monitoreo con su respectivo software ya instalado, etc.

Inmediatamente después de la instalación del equipo debe iniciarse la interconexión de los dispositivos y sus respectivas pruebas de conexión, asegurando la buena comunicación entre los componentes (Sensores, dispositivo de transmisión, PC).

Luego se realizan las operaciones básicas de configuración y administración de tareas que ejecutara el administrador del equipo, como se hará en cualquier ocasión necesaria o periódica, ya sea el caso.

Ya con el equipo debidamente instalado y probado , se hará su respectivo periodo de prueba, en el cual funcionara como normalmente se espera que funcione como en una implementación real, claro esta no contara con todas las funciones especificas ya que para el caso solo es un prototipo, pero este cumplirá los requisitos primordiales de captura de información de los estados del sensor resolviendo las interrogantes de ¿Dónde esta localizado?,¿Cuándo ocurrió el suceso? y ¿Por qué se dio el evento? y a su ves el monitoreo constante con su debido aviso al custodio.

Las conclusiones luego de tener la oportunidad de prueba y fallo con los dispositivos ya instalados, se ha logrado la satisfacción de tener un sistema de seguridad fiable, amigable al usuario, y sencillo de utilizar; haciendo esto mas factible el trabajo de los custodios, debido que responde adecuadamente, cumpliendo con cada de las funciones requeridas.

Además, cada una de las pruebas realizadas a llevado a cabo la función de mejorar el rendimiento y adecuar mejor el formato y el monitoreo que requerirá el custodio o quien lo administre.

Una de las observaciones mas claras en el periodo de prueba, es la forma de respuesta que se debe dar para el logro de las expectativas que se tienen con el proyecto, contando mucho con el recurso humano, para respaldar las actividades de monitoreo, también se reconoce que para las actividades requeridas la cantidad de recursos que consume no son elevadas.

IDENTIFICACIÓN Y MANEJO DE ERRORES

Para la identificación de cada error posible y el manejo de los mismos, se encuentra lo que es una parte muy importante en la creación y utilización de los sistemas, el cual es el mantenimiento de software, ya que es una de las actividades en la ingeniería de software y es el proceso de mejorar y optimizar el software desplegado (revisión del programa), así como también remediar los defectos.

El mantenimiento de software es también una de las fases en el ciclo de vida de desarrollo de sistemas, que se aplica a la desarrollo de software. La fase de mantenimiento es la fase que viene después del despliegue (implementación) del software en el campo.

La fase de mantenimiento de software involucra cambios al software en orden de corregir defectos y dependencias encontradas durante su uso tanto como la adición de nueva funcionalidad para mejorar la utilización y aplicabilidad del software.

Tipos de mantenimiento:
Correctivo
Perfectivo
Preventivo
Adaptativo

A continuación se señalan los tipos de mantenimientos existentes con su respectivo concepto, y entre paréntesis el porcentaje aproximado respecto al total de operaciones de mantenimiento:

Perfectivo (60%): Mejora del software (rendimiento, flexibilidad, reutilización) o implementación de nuevos requisitos.

El mantenimiento perfectivo se realizara únicamente si es requerido por la Facultad, ya sea por adquisición de nuevo equipo o que la administración desea incorporar o modificar la aplicación con la finalidad de optimizar el software.

Correctivo (17%): Son aquellos cambios precisos para corregir errores detectados durante la implantación.
Este tipo de mantenimiento se efectuara durante la prueba, ya sea, modificando el código o agregando elementos al sistema.

Preventivo (5%): Facilitar el mantenimiento futuro del sistema (verificar precondiciones, mejorar legibilidad).
Referente al mantenimiento preventivo es concerniente al mantenimiento de la base de datos, se realiza periódicamente por parte del administrador, a través de la verificación de la configuración de los sensores y el mantenimiento de los usuarios que este conlleva (agregar, eliminar y contraseña).

Adaptativo (18%): Adaptación del software a cambios en su entorno tecnológico (nuevo hardware, otro sistema de gestión de base de datos, otro sistema operativo).

CAPÍTULO V

"DOCUMENTACIÓN Y RESULTADOS DEL PROTOTIPO"

MANUAL DE USUARIO DEL SOFTWARE

Un manual de usuario es el que permite explicar el proceso a llevarse a cabo para iniciar el programa: además de explicar la pantalla de inicio. No olvidando comentar los diferentes tipos de menús empleados en el sistema.

INTRODUCCION

El sistema de monitoreo de sensores, permite resguardar las zonas de mayor riesgo, en este caso, en el que hay un mayor flujo de personas en las áreas internas de los edificios.

La implementación del software tiene como sus objetivos principales, el apoyo a la unidad de custodios, para la optimización del trabajo realizado en el interior de las instalaciones de la facultad. Logrando de esta manera la modernización de los métodos de control y vigilancia utilizados por dicha unidad, otorgándoles una herramienta tecnológica poderosa, la cual permite realizar un monitoreo constante de los sectores o zonas de riesgo, ya mencionadas anteriormente.

Al iniciar la ejecución del prototipo, lo básico es poseer ya un usuario y su clave respectiva, el administrador es el encargado de supervisar y asignar los usuarios y sus claves respectivas a las personas que utilizaran dicho sistema, las cuales están sujetas a cambios en el futuro por los propios usuarios, para la seguridad del sistema en el futuro.

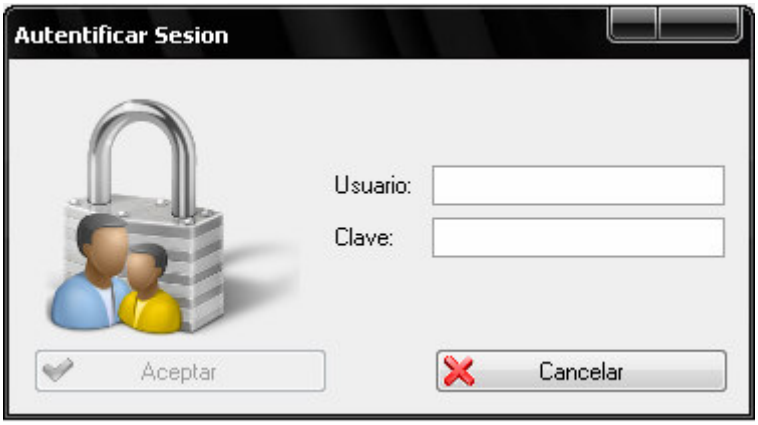


Figure 5.1, cuadro de dialogo autentificar sesión

Como muestra la figura 5.1, es el cuadro de dialogo para autentificar o iniciar la sesión al programa, en dicho cuadro se introducirá el usuario y clave de cada persona que utilizara el sistema, en este caso el administrador y los custodios.

Por lo tanto, en el momento que se introduzca el nombre del usuario y su clave se activara el botón aceptar, del lado inferior izquierdo, el cual debe de darse clic para procesar la información y acceder al programa.

En el momento que el sistema busque en la base de datos y encuentre los datos introducidos respectivos de su usuario le permitirá entrar al modulo o consola principal que muestra la figura 5.3, en caso contrario se introdujera algún dato erróneo, le aparecerá el siguiente mensaje emergente, que muestra la figura 5.2.

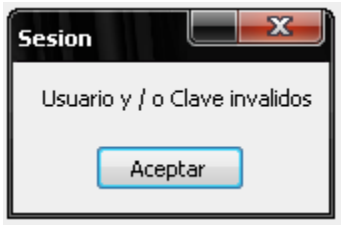


Figure 5.2, ventana emergente datos erroneos

La consola o modulo principal del sistema, que se muestra en la figura 5.3, se observa en la barra superior, la barra del menú principal con las siguientes opciones:

- Aplicación
- Configuración
- Administración
- Reportes

Dichas opciones contienen submenús, que se explicaran en el desarrollo de este manual.

Siguiendo con la consola principal, esta muestra en el lado derecho de la pantalla, el listado desplegable de los diferentes mapas cargados en el sistema, y en el lado izquierdo, se muestra el listado de sensores respectivamente por cada mapa. Es de mencionar que nunca puede aparecer uno o muchos sensores en diferentes mapas, con el objetivo de no crear confusiones futuras o dualidad.

Los mapas son figuras o dibujos de plantas arquitectónicas, que muestran las diferentes áreas de un edificio sean internas o externas, donde se ubicara gráficamente con una figura en proporciones adecuadas, el sensor respectivo que esta situado en una área especifica del mapa o edificio.

Y en el área inferior izquierda se muestra, el tipo de sensor que se ha seleccionado, previamente en alguna de las figuras ubicadas en el mapa, la ubicación del sensor en el plano según la posición del plano X,Y, el estado del sensor (Desactivado o Activado) y el pin del puerto paralelo, que se le asigno en el momento de crearlo o agregarlo al mapa respectivo.

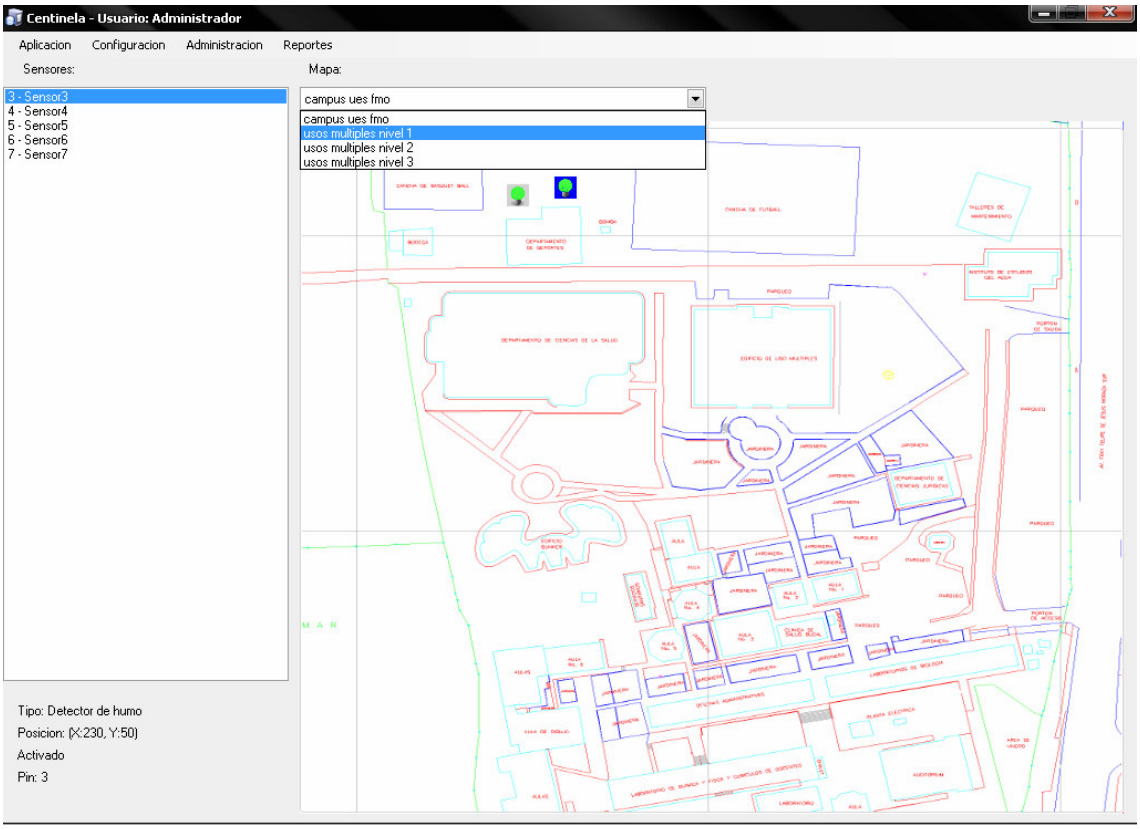
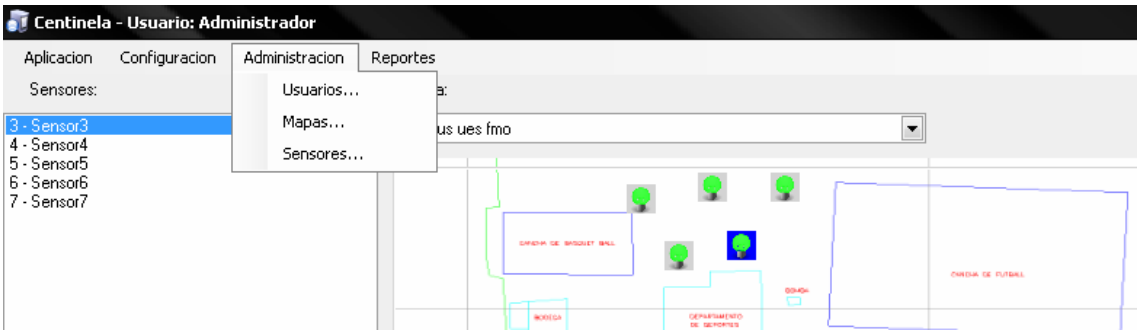


Figure 5.3, pantalla o modulo principal del sistema

COMO AGREGAR UN MAPA

Para agregar un mapa predeterminado, el sistema puede cargar archivos .jpg .bmp .gif .png. Con los siguientes pasos:

- ✓ Se abre el menú Administración.
- ✓ Seleccione la opción Mapas.



- ✓ Da clic en agregar.
- ✓ Le aparecerá el siguiente modulo que muestra la figura 5.4
- ✓ Aparecerá el listado de mapas ya creados, con sus respectivos nombres, y en este caso el usuario agrega el nombre del nuevo mapa.
- ✓ En el lado derecho aparecerá la vista previa del plano.

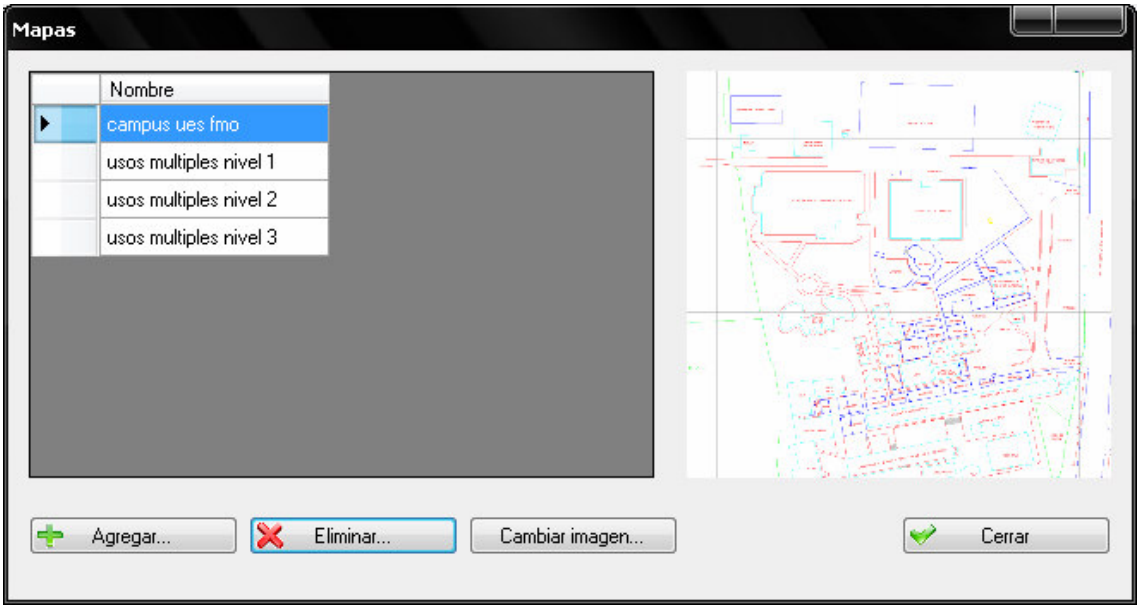
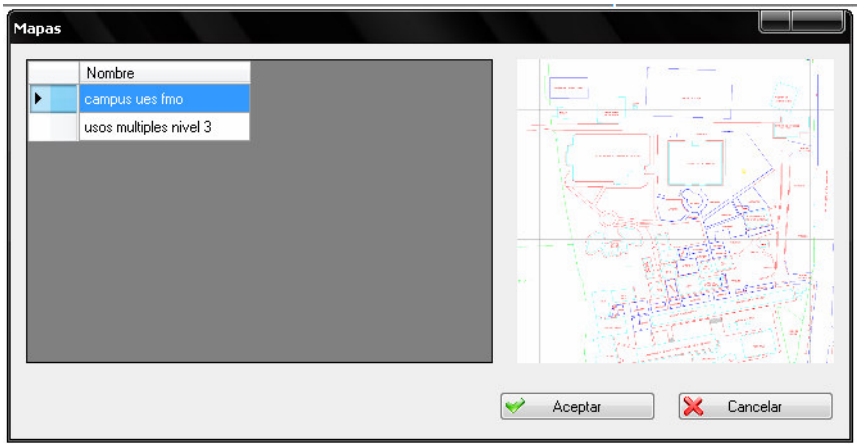


Figure 5.4, modulo para agregar o eliminar mapas

A la misma vez, puede el usuario por medio de esta misma forma u opción eliminar o cambiar la imagen de algún mapa si el usuario lo desea.



COMO AGREGAR SENSORES

Una vez ya agregado el mapa, puede el usuario agregar los respectivos sensores, con los siguientes pasos:

- ✓ Se abre el menú Administración.
- ✓ Seleccione la opción Sensores, aparecerá el modulo que muestra la figura 5.5, dicho modulo muestra en el lado izquierdo la lista de sensores existentes, y en el lado derecho la lista de mapas existentes en la base de datos, el cual le mostrara la ubicación de los sensores existentes en dicho mapa. Y en la parte inferior aparece el nombre y tipo de sensor al seleccionarlo.
- ✓ Da clic en agregar (+).
- ✓ Le aparecerá el siguiente modulo que muestra la figura 5.6 en donde le puede asignar el nombre, el tipo de sensor y el pin relacionado en el puerto paralelo.

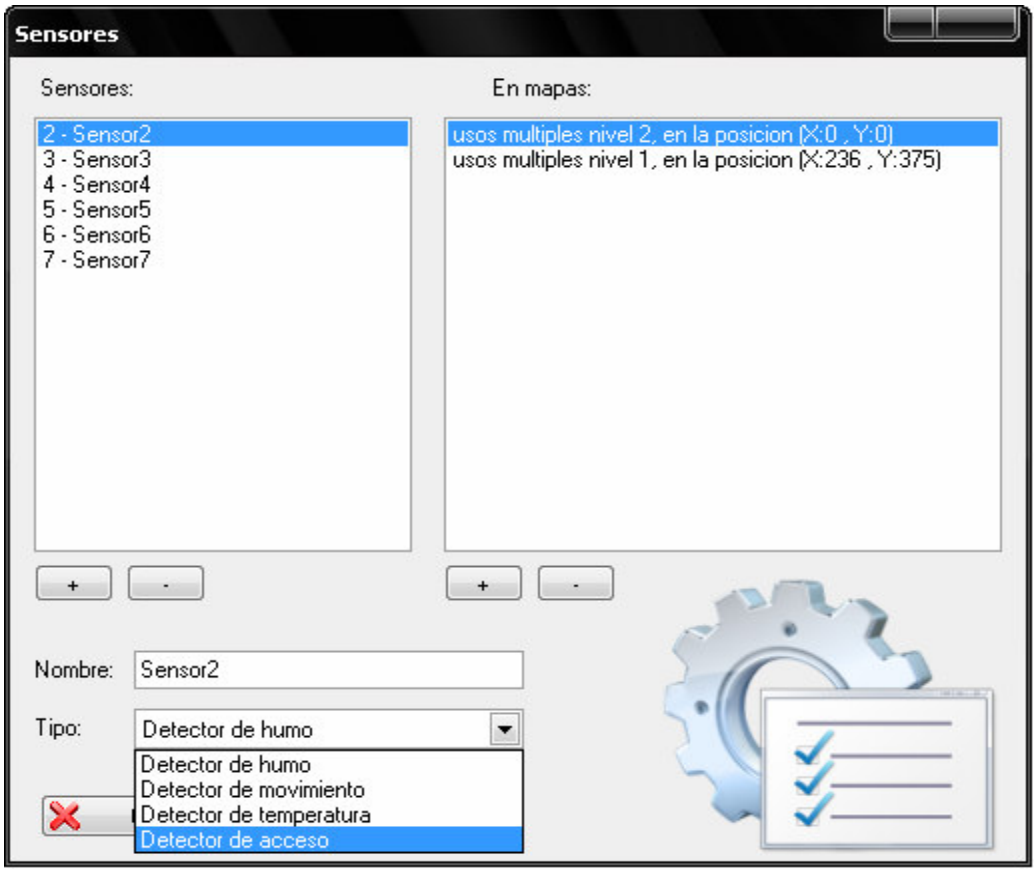


Figure 5.5, modulo administrador de sensores

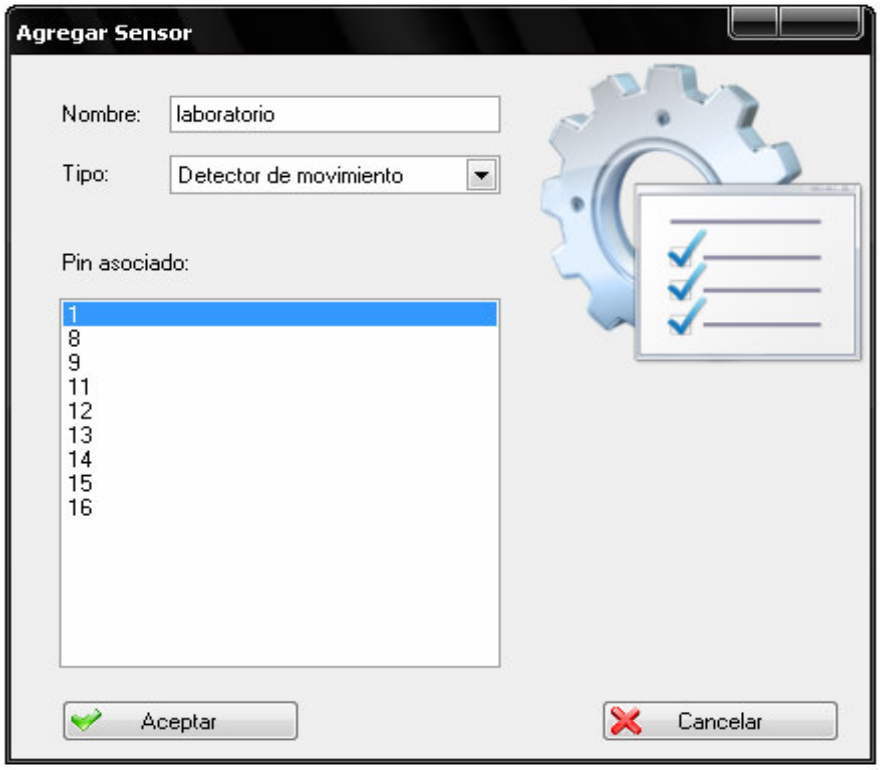
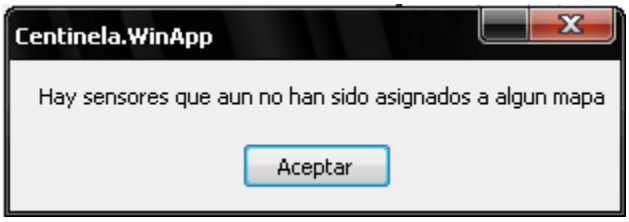


Figure 5.6, modulo agregar sensor

- ✓ Se da aceptar para validar la información introducida.
- ✓ Luego selecciona en la parte izquierda el mapa asociado a este pin, de la misma forma, dando clic en agregar y le desplegara el listado de mapas para asociar, selecciona uno y finaliza con clic en aceptar.

Si en algún caso no asocio ningún mapa al sensor, en el momento que quiera cerrar le mostrara el siguiente mensaje.



CONFIGURACIÓN AVANZADA DE LOS SENSORES

Esta opción permite al usuario configurar en otras opciones a los sensores, como lo es: la imagen o presentación que tendrán los sensores en los mapas, el sonido o alarma que pueden tener cuando se activan y la frecuencia de lectura del puerto paralelo.

Los pasos a seguir son los siguientes:

- ✓ Se abre el menú Configuración.

- ✓ Seleccione la opción sensores, aparecerá el modulo que muestra la figura 5.7.
- ✓ Seleccione cada una de las opciones que mejor parezca.
- ✓ Y al momento de cerrar la ventana todas esas opciones seleccionadas se guardaran en la base de datos del sistema.

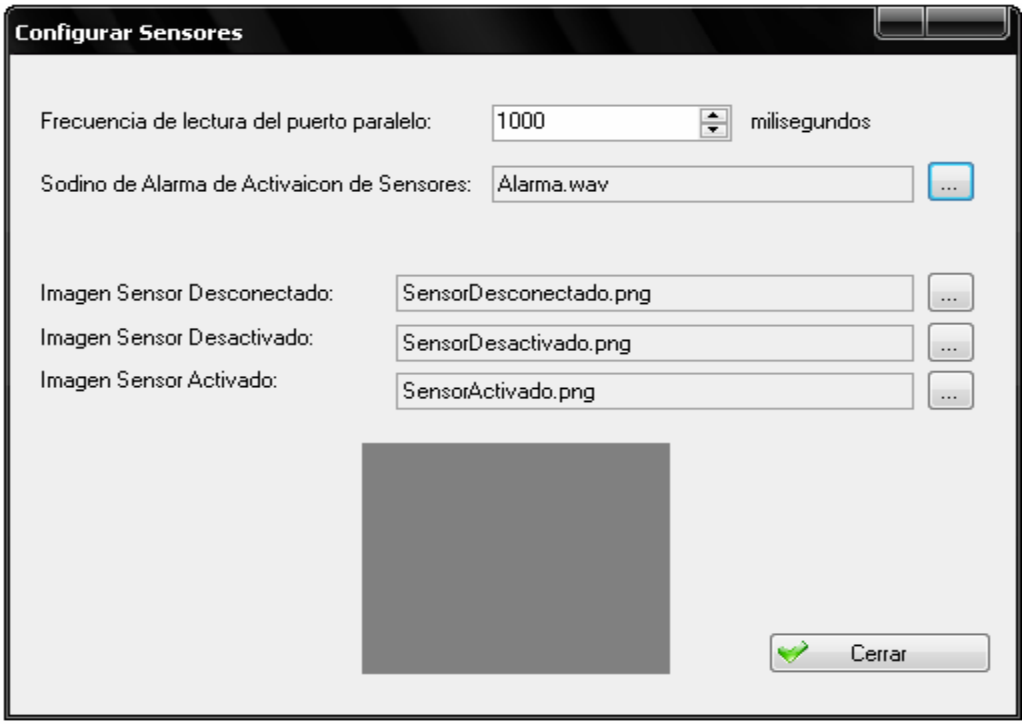


Figure 5.7, configuración avanzada de sensores

COMO CREAR UN USUARIO

La siguiente opción es exclusiva para los administradores del sistema que pueden crear usuarios finales, que utilizaran de una forma restringida el sistema, y se procede de la siguiente forma:

- ✓ Entre en el menú Administración.
- ✓ Seleccione Usuarios. Figura 5.8

Figure 5.8, modulo agregar usuario

- ✓ Clic en el botón agregar, y aparecerá el modulo de la figura 5.9, donde introducirá el nombre completo de la persona, su usuario o alias y la contraseña o clave con la que podrá acceder al sistema de monitoreo.
- ✓ Concluye al dar clic en aceptar para guardar la información en la base de datos.

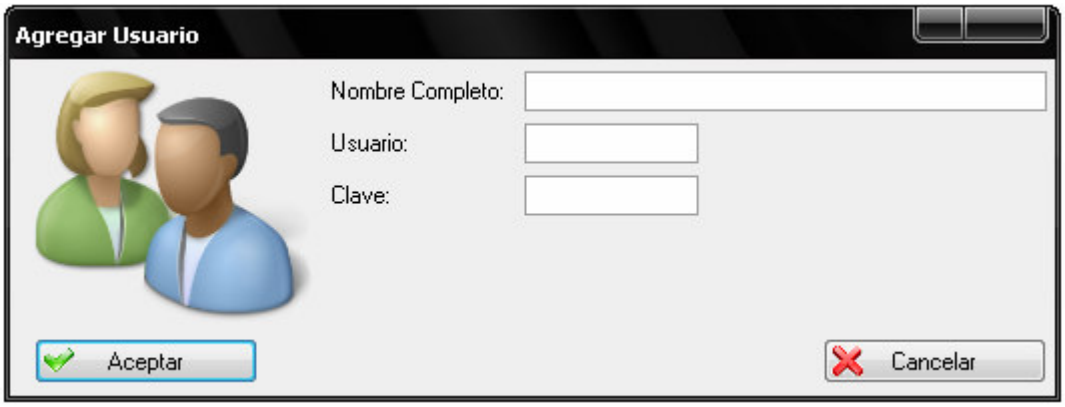


Figure 5.9, modulo agregar usuario (exclusivo para el administrador del sistema)

COMO CAMBIAR CLAVE A SU USUARIO

Dicho usuario debe entrar al sistema, y podrá cambiar la clave solo dentro del mismo, en una forma periódica, para optimizar la seguridad del sistema, y se siguen los siguientes pasos:

- ✓ Entre en el menú Administración.
- ✓ Clic en Usuarios.
- ✓ Selecciones cambiar clave, de los botones del lado derecho de la ventana.
- ✓ Y aparecerá el modulo de la figura 5.10, donde introducirá la contraseña o clave anterior y la nueva que desea asignar.
- ✓ Y concluye dando clic en aceptar para validar y guardar en la base de datos su nueva contraseña.

Figure 5.10, modulo cambiar contraseña del usuario

Pero si en algún caso no introduce bien la información, le aparecerá la siguiente ventana emergente para que vuelva a introducir nuevamente la información. (Fig. 5.11)

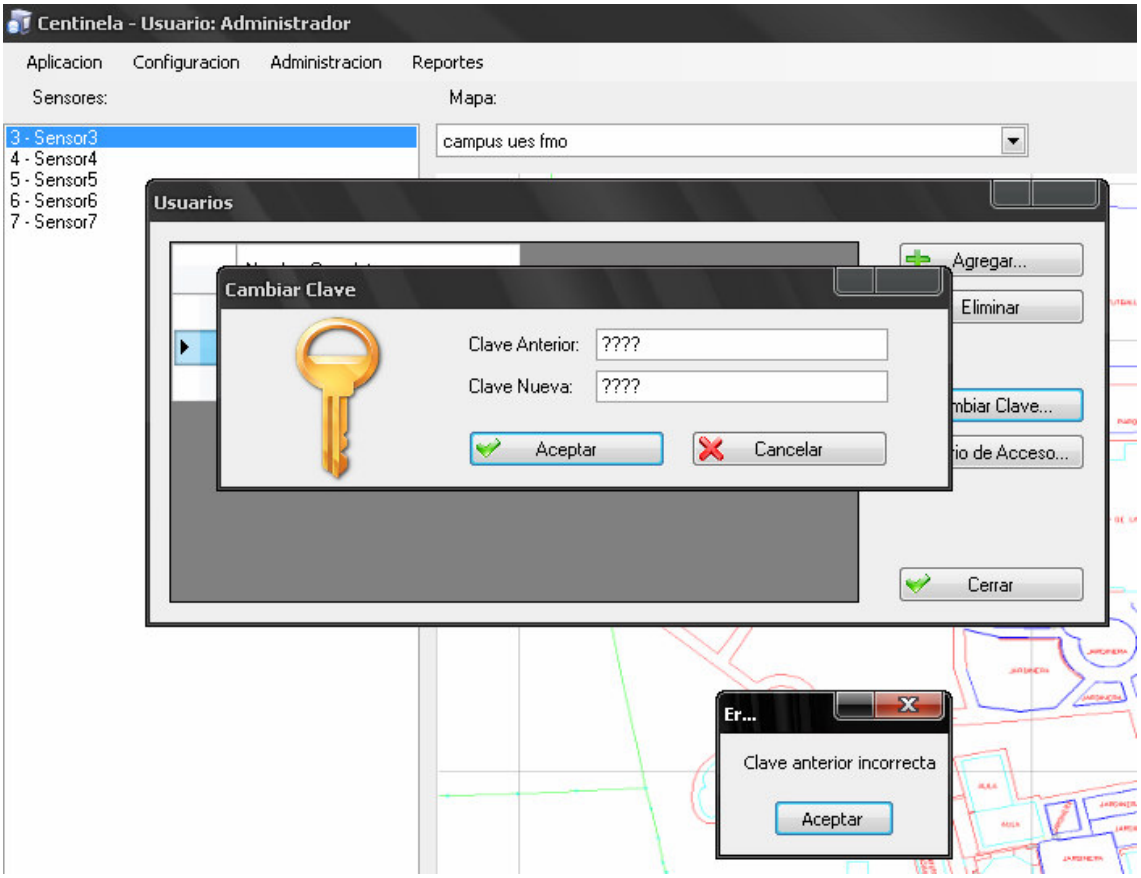


Figure 5.11, cuadro emergente al presentarse un error para cambiar la clave

COMO ASIGNAR HORARIO DE TRABAJO A UN USUARIO

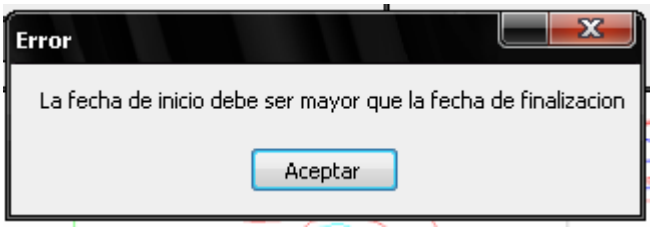
Esta opción es exclusiva para los custodios o usuarios finales del sistema, ya que se debe de controlar el horario o turno en que este estará trabajando, y el al mismo tiempo será el horario en que podrá acceder al sistema. También es de mencionar que solo lo puede hacer el administrador del sistema.

Y se asignara de la siguiente manera:

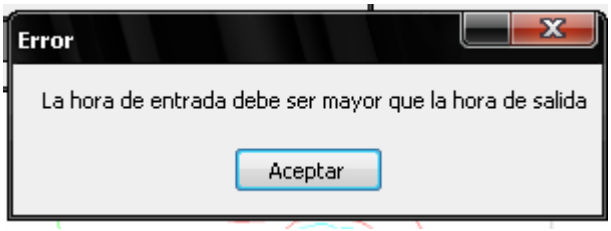
- ✓ Entre en el menú Administración.
- ✓ Clic en Usuarios.
- ✓ Debe posicionarse en el usuario que desea.
- ✓ Da clic en Horario de Acceso, de los botones del lado derecho de la ventana.
- ✓ Le aparecerá el modulo de la figura 5.12, el cual muestra la fecha de inicio y fin de la semana, y las horas correspondientes al turno.
- ✓ Y para finalizar clic en Aceptar para guardar los datos.

Figure 5.12, modulo asignar horario a usuario.

También en este caso pueden introducirse información incorrecta, la cual generaran ventanas emergentes, que le señalaran el error que se cometió en el modulo. Por ejemplo:



La fecha de finalización debe de ser posterior a la inicialización. Sino emergerá esta ventana de error.



La hora de salida debe de ser posterior a la de entrada. Sino emergerá la siguiente ventana de error.

Al guardar toda esta información en la base de datos, el sistema no permitirá que el usuario entre al sistema en horas que no han sido asignadas, según su turno de trabajo. Como lo muestra la pantalla siguiente en el modulo de inicio de sesión en la figura 5.13

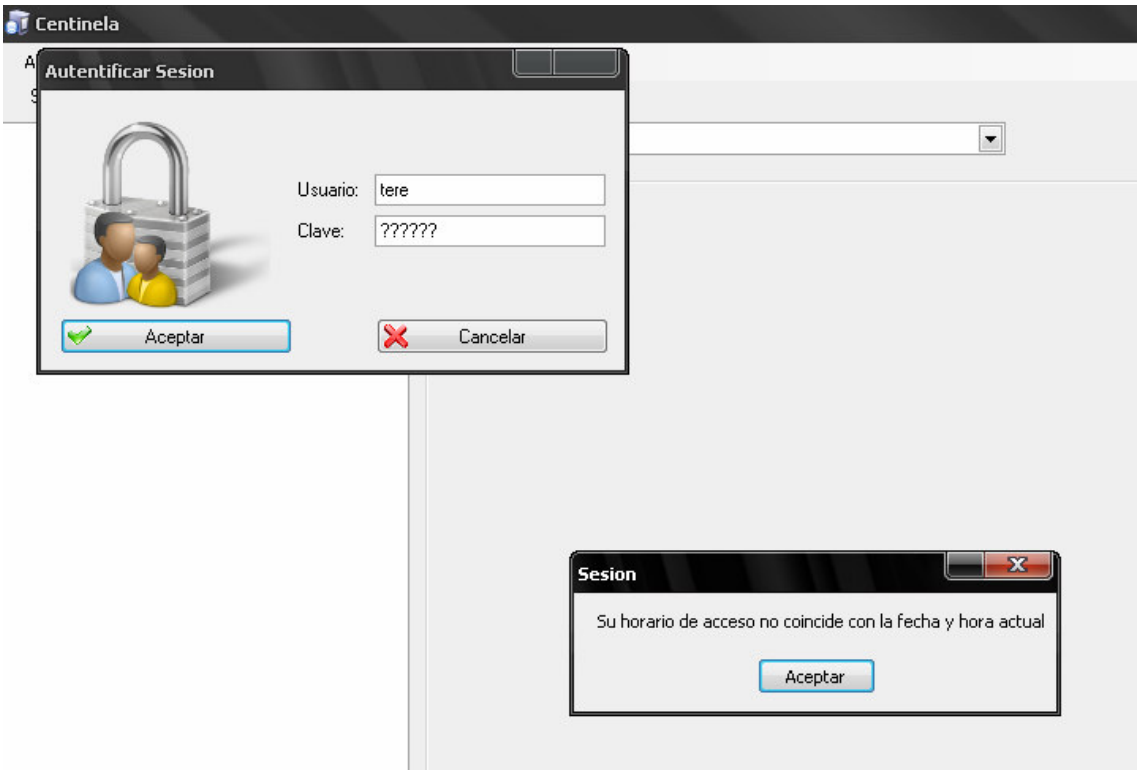
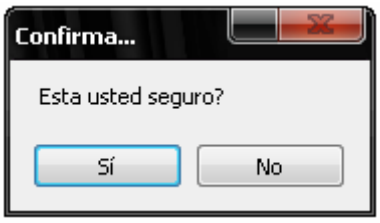


Figure 5.13, sesión de inicio, al introducir usuario con horario no permitido

COMO ELIMINAR UN SENSOR

De la misma forma que se procede para agregar un sensor, según se aprecia en la figura 5.5

- ✓ Se abre el menú Administración.
- ✓ Se selecciona la opción Sensores, aparecerá en el modulo del lado izquierdo la lista de sensores existentes.
- ✓ Se selecciona con el cursor el sensor a eliminar.
- ✓ Da clic en eliminar (-).
- ✓ Y le aparecerá la ventana emergente que preguntara si esta seguro de realizar esta operación.
- ✓ En donde el usuario opta por reafirmar si llevara a cabo esta operación.



COMO ELIMINAR UN MAPA

Un mapa puede eliminarse con los siguientes pasos, tomando en cuenta que si existen sensores asignados a ese mapa deben de moverse o emigrarlos a otro mapa o borrarlos también, si ya no se desean utilizar.

- ✓ Se abre el menú Administración.
- ✓ Seleccione la opción Mapas. Figura 5.4
- ✓ Selecciona según el nombre y figura previa del mapa a eliminar.
- ✓ Da clic en eliminar.
- ✓ Le aparecerá la ventana emergente para afirmar su operación.
- ✓ Según la decisión del usuario da clic en aceptar o cancelar.

Como se menciono previamente si el mapa tiene sensores asociados aparecerá la siguiente ventana emergente o de advertencia del error, según se muestra en la figura 5.14

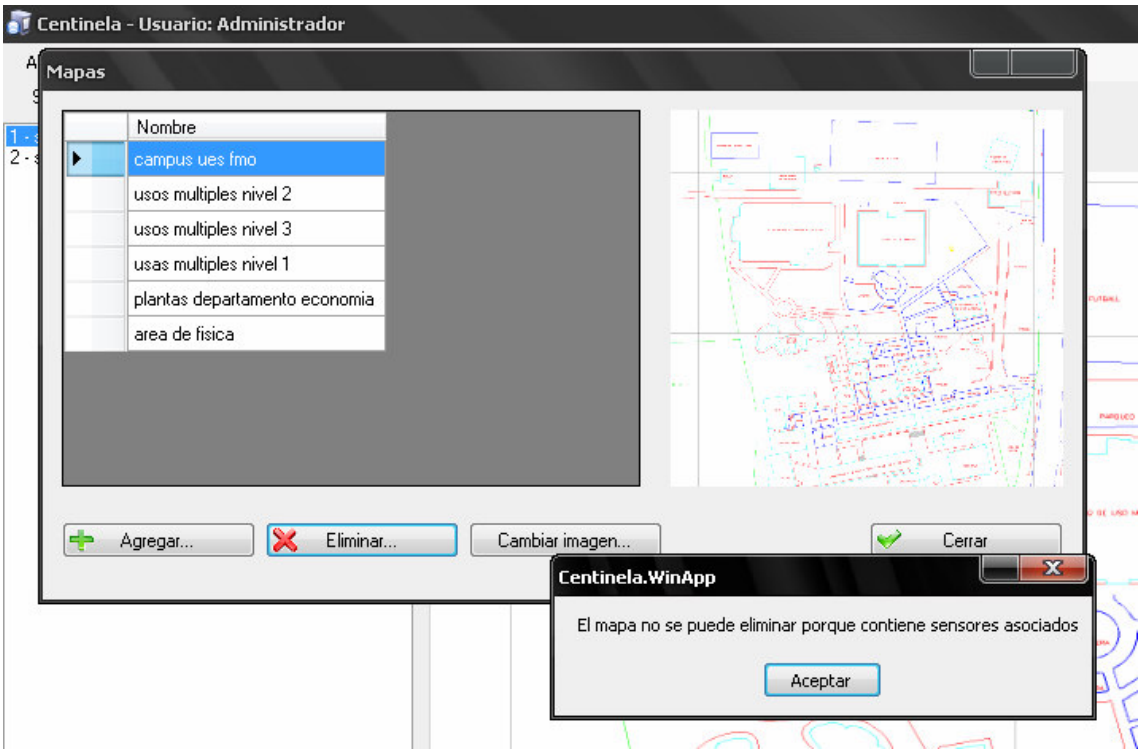


Figure 5.14 error al eliminar mapa con sensor (es) asignados

COMO ELIMINAR UN USUARIO

Se hace referencia los mismos pasos del procedimiento de agregar un usuario

- ✓ Entre en el menú Administración.
- ✓ Seleccione Usuarios. Figura 5.8
- ✓ Seleccionar el usuario a eliminar.
- ✓ Clic en el botón eliminar.
- ✓ Aparecerá la ventana emergente para reafirmar la operación de eliminación del usuario/
- ✓ Concluye al dar clic en aceptar o cancelar.

COMO PRODUCIR REPORTES EN EL SISTEMA DE SUCESOS OCURRIDOS PREVIAMENTE

Para esta operación existen varias categorías o tipos de Reportes que se pueden producir, como a continuación se muestra.

- ✓ Entre en el menú Reportes.
- ✓ Aparecerá el listado siguiente que muestra la Figura 5.15

Figure 5.15, menú reportes y sus opciones

El usuario puede seleccionar la opción que mejor le parezca:

Reporte de hoy: Se genera el reporte de sucesos ocurridos a los sensores del día presente.
Reporte de este mes: Se genera el reporte de sensores activados en el mes vigente.
Reporte de este año: Se genera el reporte de sensores activados en el año correspondiente.
Reporte Total: Se genera un reporte de todos los sensores.
Reporte por sensor: Se genera el reporte de algún sensor en específico.

COMO SALIR DEL SISTEMA O CERRAR SESIÓN

Para cerrar la Sesión, en el caso que el usuario del siguiente turno utilizara el sistema, o definitivamente se desea cerrar se siguen los siguientes pasos:

- ✓ Entre en el menú Aplicación.
- ✓ Seleccione Salir o Cerrar sesión. Figura 5.16

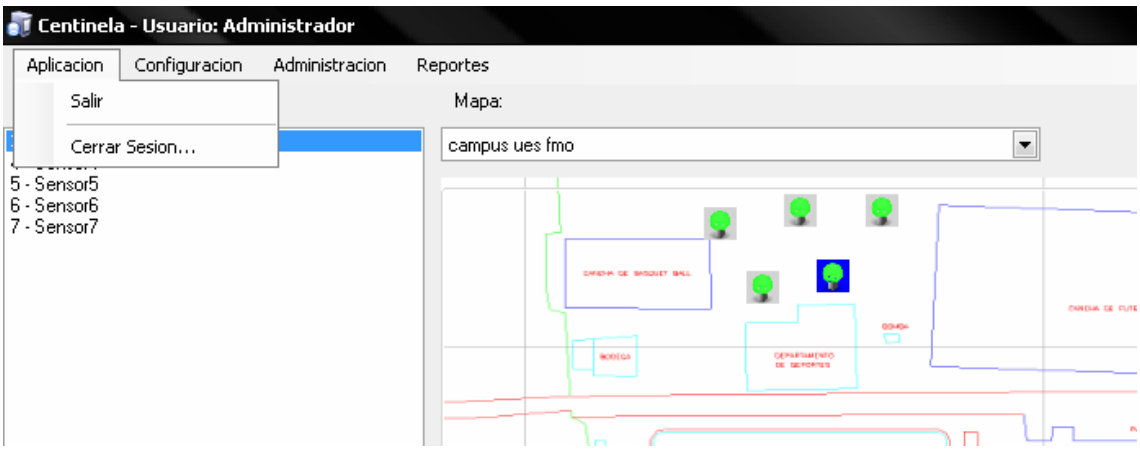
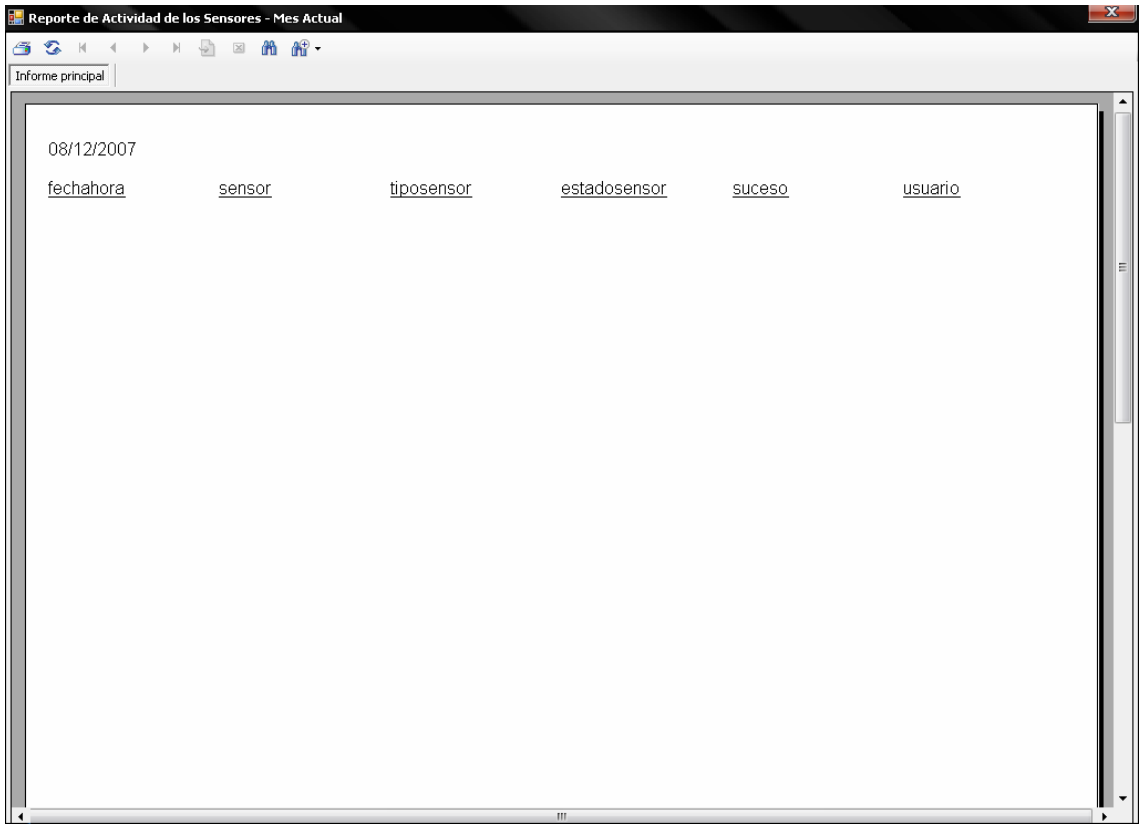
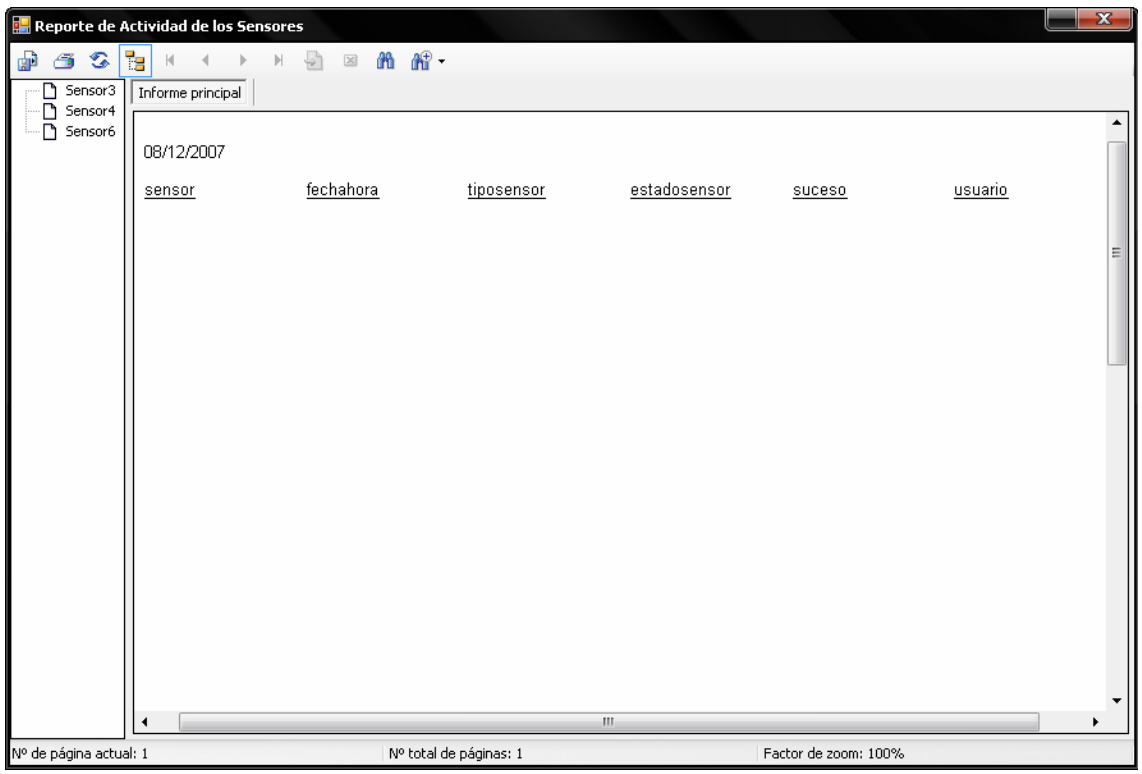


Figure 5.16, Pantalla cerrar sesión o salir del sistema

Y se podrá observar Reportes con similar formato o campos presentes de información.


Reporte de Actividad de los Sensores - Mes Actual
Informe principal
08/12/2007
fechahora
sensor
tiposensor
estadosensor
suceso
usuario


Reporte de Actividad de los Sensores
Sensor3
Sensor4
Sensor6
Informe principal
08/12/2007
sensor
fechahora
tiposensor
estadosensor
suceso
usuario
Nº de página actual: 1
Nº total de páginas: 1
Factor de zoom: 100%

GUÍA RÁPIDA O DE AYUDA

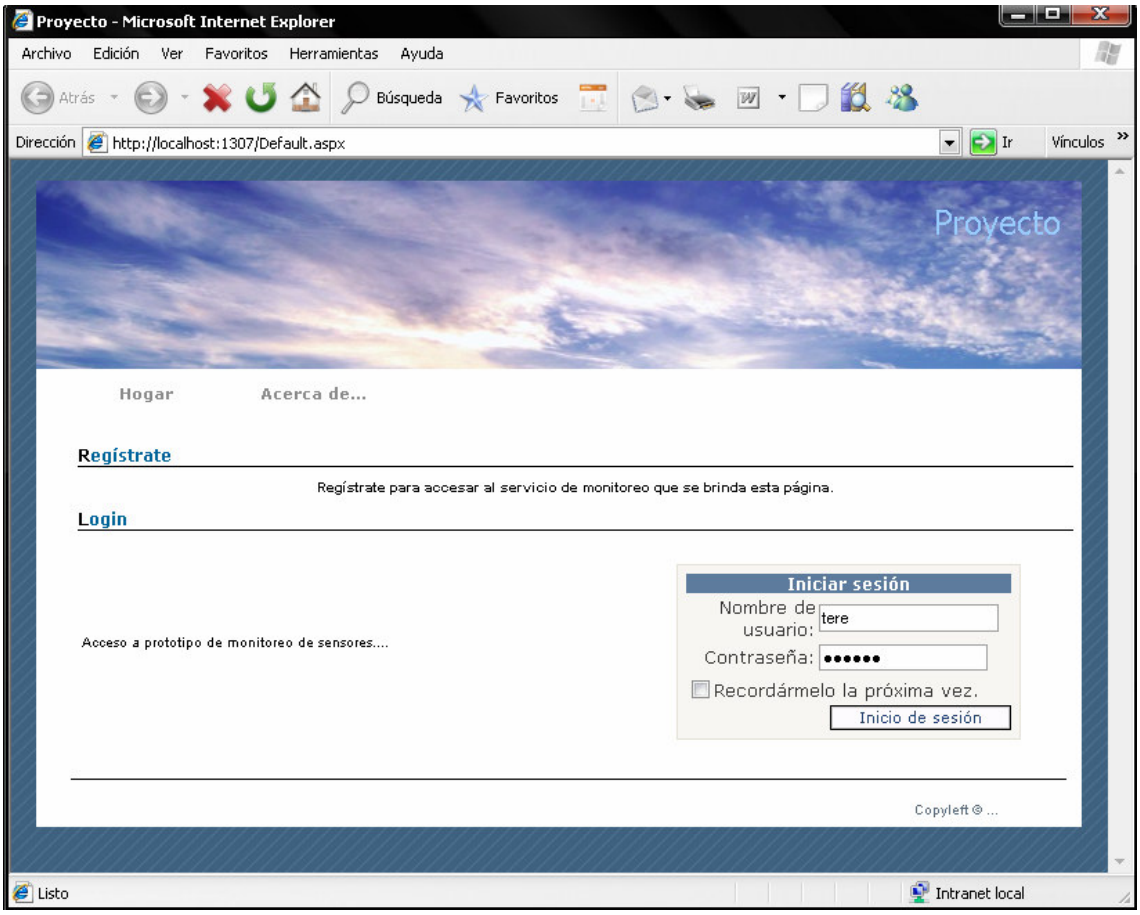
La guía rápida es un instructivo donde se detalla de forma breve y concisa las diferentes operaciones que se pueden realizar en una determinada aplicación. Donde su objetivo principal es que el usuario pueda utilizar dicha aplicación.
Para acceder a ella, se hará directamente desde C:\centinela.

MONITOREO WEB

Como opción alternativa para el chequeo de los sensores, se puede hacer remotamente, pero solo dentro de la red interna de la FMO, pero cabe mencionar que es un prototipo también, ya que la forma en que se presenta la información puede ser modificada, según las exigencias de los clientes

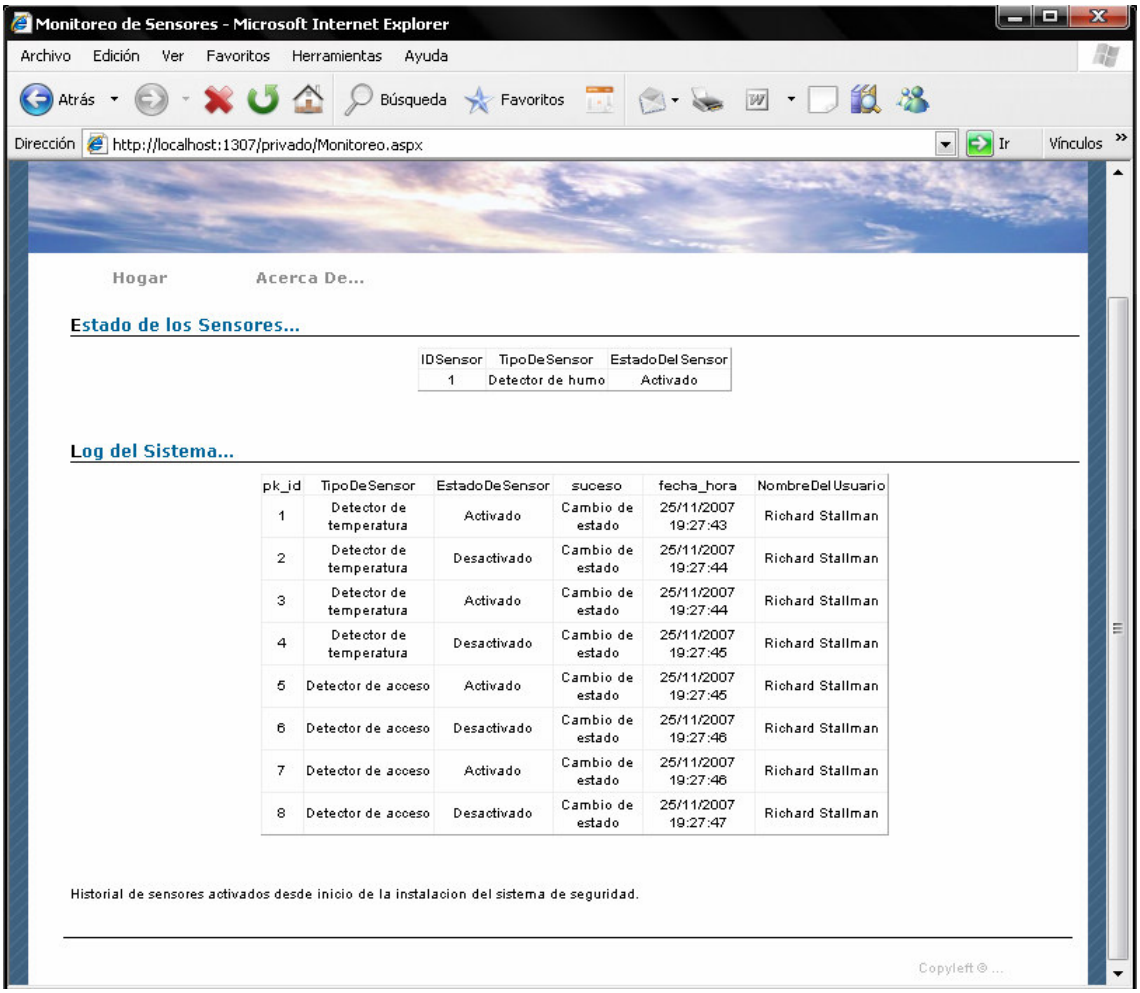
¿Como acceder a ella?

Por medio de la siguiente dirección http://localhost:1307/Default.aspx, se accesa a la pagina de inicio, donde puede introducir el usuario su nombre y clave ya registrado previamente en la base de datos del sistema. Pero en todo caso, la dirección cambiara o dependerá del servidor o PC donde sea instalado. Y tiene el siguiente aspecto amigable para el usuario.

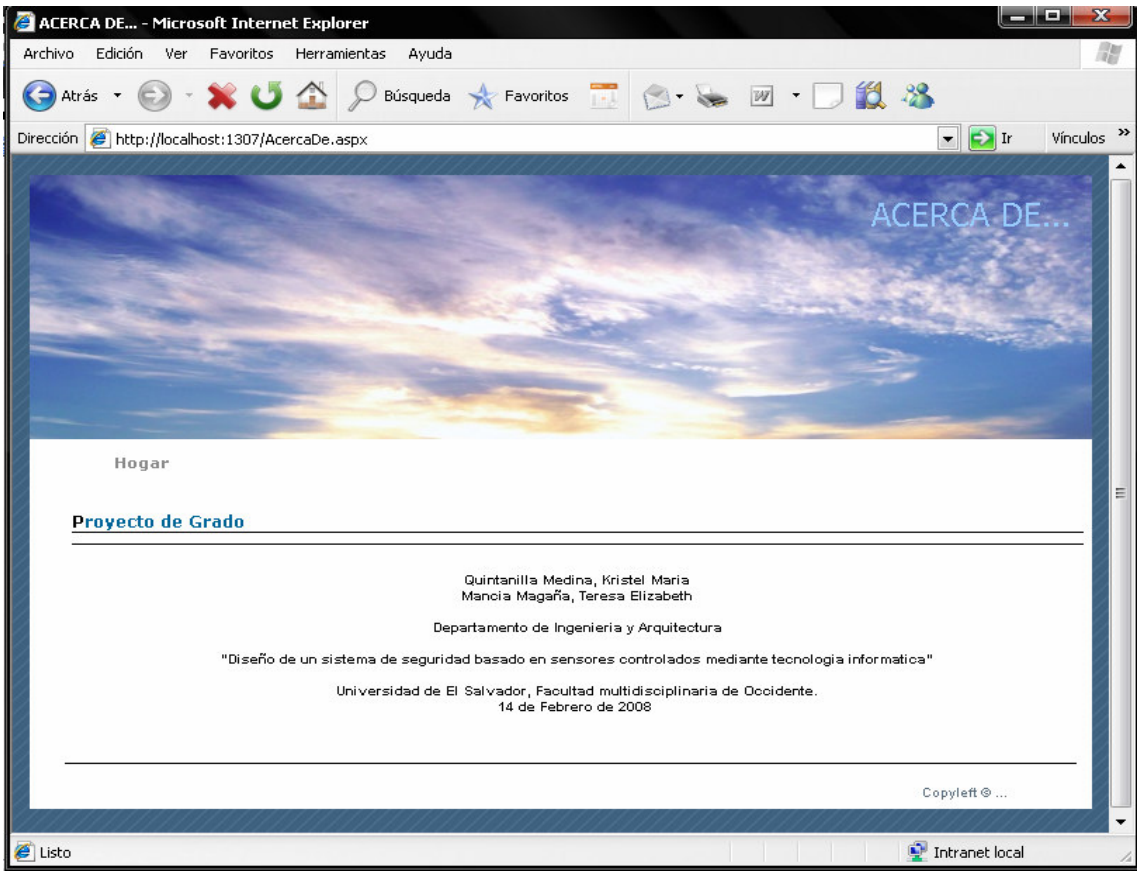


Al momento que su usuario pueda logiarse o entrar a la pagina de monitoreo, le mostrara el historia de sensores activados, desde que se instalo el sistema, con los datos como nombre del sensor, día y hora, así como el usuario que estaba en ese momento logiado o en turno.

Y en el listado superior mostrara el estado de sensores actualmente instalados en el sistema, y confirmara si ha sido activado o no.



Y para dar información de lo que es esta pagina, y quienes la construyeron, esta en la parte superior de la pagina la opción, acerca de, que muestra la siguiente información.


ACERCA DE... - Microsoft Internet Explorer
Archivo Edición Ver Favoritos Herramientas Ayuda
Atrás
Búsqueda
Favoritos
Dirección
http://localhost:1307/AcercaDe.aspx
Ir
Vínculos
ACERCA DE...
Hogar
Proyecto de Grado
Quintanilla Medina, Kristel Maria
Mancia Magaña, Teresa Elizabeth
Departamento de Ingenieria y Arquitectura
"Diseño de un sistema de seguridad basado en sensores controlados mediante tecnologia informatica"
Universidad de El Salvador, Facultad multidisciplinaria de Occidente.
14 de Febrero de 2008
Copyleft © ...
Listo
Intranet local

PRUEBA PROTOTIPO E INSTALACIÓN

DESCRIPCIÓN DE LA PRUEBA

Como primer paso se procedió a escoger una de las áreas donde se pudieran hacer esta prueba del prototipo, por lo que se escogió el laboratorio de hardware del departamento de ingeniería, ya que se presta mucho a las necesidades y requerimientos para hacerlo de una forma aislada, sin que intervengan los alumnos, o haya mucha afluencia de personas.

PROCEDIMIENTOS O EJECUCIÓN

Ya tomada la decisión, y consultado con las personas encargadas del aula, se procedió a evaluar, los lugares o puntos adonde se debían instalar los sensores, y luego se procedió a hacer el cableado, con un punto final común, donde seria el lugar para interconectar los sensores al dispositivo de transmisión de datos, finalizando a la conexión con la PC en que esta instalado el software.

Los cables tienen instalados conectores RJ-11 machos, para poder conectarlos al dispositivo de transmisión de datos, por medio de los cuales se reciben las señales de apertura o cierre, de los circuitos. Y se envía el voltaje que los dispositivos necesitan para su funcionamiento, el cual oscila entre 5 y 12 voltios.

RESULTADOS

Al probar la interconexión de los dispositivos, se encontraron muchos problemas para recibir las señales emitidas por ellos, así como el recibimiento de voltaje en los dispositivos sensoriales, a causa de usar cable telefónico de filamentos, ya que es un producto poco resistente en la manipulación o instalación de esta clase de dispositivos.

Por lo tanto, se termino utilizando un cable especial, de marca Velden calibre 22, el cual es utilizado por las empresas de instalación de sistemas de seguridad en el país, el cual puede ser de 2 o 4 hilos.

En conclusión, la prueba final del circuito o sensores instalados fue satisfactoria, ya que se pudo tener comunicación entre los sensores y el sistema.

CONCLUSIONES

- Actualmente existen en el mercado una gran gama de sensores, los cuales pueden ofrecer más ventajas y tener una mejor tecnología, para implementar en exteriores, u adaptar otros dispositivos de vigilancia o seguridad.
- El sistema desarrollado en Lenguaje VB.net, puede ser explotado o ampliado en sus herramientas, para la mejora continua del prototipo y llegar a desarrollar un sistema mas completo.
- La investigación hecha para el uso o manipulación del puerto paralelo, puede seguir utilizándose para otros proyectos de estudio en el futuro, de la rama de la electrónica y de sistemas en esta institución.
- El uso de esta tecnología es muy rentable para los gastos de esta u otra institución, que deseen contar o invertir en un sistema de seguridad de este tipo.
- Debido al crecimiento de la infraestructura de la Institución, el apoyo tecnológico se convierte muy necesario cada día, para complementar y apoyar el trabajo de la unidad de custodios, y no para sustituir el recurso humano.
- Según las investigaciones hechas al estudiantado y el personal académico de la institución, la mayoría opina, que es necesario el uso de dispositivos de seguridad, para el resguardo del patrimonio estudiantil.
- Este Prototipo en sistema, es una alternativa innovadora, ya que no se ha desarrollado anteriormente, y se comprueba que por las cualidades y características que presenta es muy fácil de usar y amigable para los usuarios.
- El proyecto es económicamente ventajoso, ya que en todos los cálculos de la razón B/C, se obtuvo un resultado mayor de 1.0

RECOMENDACIONES

- Se necesita por parte de la FMO, la renovación de infraestructura de la institución, para que el sistema de seguridad respalde en forma optima, el patrimonio estudiantil.

- La adecuada capacitación de los usuarios, servirá para aprovechar al máximo, los recursos del proyecto.

- El Mantenimiento y actualización del equipo, es vital para el funcionamiento óptimo del equipo.

- Toda red de distribución de sensores necesitara un backup en caso de corte de energía eléctrica.

- Este proyecto puede ampliar sus recursos y funciones, como adaptación cámaras de vigilancia u otras tecnologías de seguridad.

- El sistema de seguridad de la Institución, con el uso de tecnología de sensores es un gran respaldo para el que se encuentra en la actualidad, ya que es la mejor opción para cubrir áreas de vigilancia permanente o de alto riesgo.

BIBLIOGRAFÍA

Libros

- Kendall & kendall
  "Análisis y diseño de sistemas"
  sexta edición
  Pearson Prentice Hall

- Anthony j. Tarquín
  "Ingeniería Económica"
  Cuarta Edición
  Mc Graw Hill Interamericana S.A.

- Joseph Schmuller
  "Aprendiendo UML en 24 horas"
  Prentice Hall

Documentos electrónicos

- http:// www.uesocc.edu.sv
- http:// www.altesa.com.sv
- http:// www.GeSecurity.com

“ANEXOS”

ENCUESTA SOBRE LA SEGURIDAD EN LAS INSTALACIONES DE LA FMO

OBJETIVO: *Conocer la opinión de la comunidad universitaria, acerca de la seguridad institucional que brinda la FMO.*

Instrucciones: Marque con una X la casilla que según su criterio es la adecuada a lo que se le pregunta a continuación:

1. Departamento que pertenece:

_____

2. ¿Cómo califica el nivel de seguridad en los lugares que frecuenta en la FMO (aulas, departamento, canchas, etc.)?

Muy bueno ☐ Bueno ☐ Regular ☐ Mala ☐

3. ¿Cree que su departamento es un lugar seguro? ☐ S ☐ No

¿Por qué?_____

4. ¿En que zona de la FMO se siente mas seguro (aulas, bunker, edificio de usos múltiples, edificio de medicina, cafetería, etc.)?

_____

5. ¿Ha sufrido algún percance con su seguridad alguna vez? ☐ Si ☐ No Si su respuesta fue SI Especifique:

_____

6. ¿Considera usted que dentro de la FMO existan personas capaces de realizar actos de vandalismo? ☐ Si ☐ No

7. Según su criterio, ¿Los ataques a la seguridad de la FMO son hechos por personas ajenas a ella? ☐ Si ☐ No

8. ¿Cómo califica el trabajo realizado por los custodios (vigilantes), en sus años dentro de la comunidad universitaria?

Excelente ☐ Muy bueno ☐ Bueno ☐ Regular ☐ Mala ☐

9. ¿Ha tenido problemas con la unida de custodios alguna vez?

Si ☐ No ☐

Si su respuesta fue SI Especifique:

_____

10. Según su punto de vista, ¿Cuáles son las áreas con mayor seguridad?

_____

¿Por qué?_____

11. Según su opinión, ¿Cuáles son las áreas con menor seguridad?

_____

¿Por qué? _____

12. Según su criterio, ¿Cuáles son las áreas de la FMO que deben tener mayor seguridad?

_____

¿Por qué?_____

13. ¿Que sugerencias hace para que halla una mejor seguridad institucional?

_____

14. ¿Cree que al utilizar sensores y tecnología informática aumentaría la calidad de seguridad en la FMO? ☐ Si ☐ No

¿Por qué?_____

CUESTIONARIO PARA DEPARTAMENTOS

Departamento o unidad de:

_____

1. ¿Cómo contribuye la unidad de custodios a que su departamento sea un lugar seguro?

2. ¿La infraestructura que actualmente conforma su departamento es lo suficientemente segura? (Si no lo es, mencione porque).

3. ¿Cuál es el equipo o material de riesgo, que necesita mayor vigilancia dentro de su departamento?

4. Puede citar algunos casos en que haya sido violentada la seguridad de su departamento.

5. Puede estimar el monto de perdidas que ha tenido su departamento en los últimos 5 años, o especifique el monto por año.

6. Considera usted que el personal de su departamento estaría de acuerdo con la instalación de un sistema seguridad basado en sensores. Si o no ¿porque?

7. Si en su departamento se instalara este sistema, ¿Cuáles serian los puntos que a su criterio, son más vulnerables y necesitan mayor vigilancia? ¿Porque?

ENTREVISTA ELABORADA HACIA EL ENCARGADO
DE LA UNIDAD DE CUSTODIOS:
LIC. GERBERT SALVADOR RIVAS FLORES.

OBJETIVO: *Conocer el estado de la Unidad de Custodios, y la situación de la seguridad institucional, actualmente en la FMO***.**

2) Con cuántos miembros cuenta la Unidad de Custodios actualmente.
3) Cuáles son los turnos que desempeñan, y con cuantos custodios cuenta cada turno.
4) Que procedimientos llevan acabo los empleados de la Unidad de Custodios.
5) En que zonas está distribuido geográficamente la vigilancia.
6) Existen herramientas o instrumentos que faciliten a los vigilantes desempeñar sus labores.
7) Cumple actualmente, las necesidades y expectativas de Seguridad en la FMO.
8) Cuáles son los desafíos que presenta la FMO en el tema de Seguridad.
9) Cuáles son los puntos más vulnerables en la institución, según su punto de vista.
10) Cuáles son las zonas que requieren mayor seguridad.
11) Qué técnicas o metodologías han empleado para violentar la seguridad institucional.
12) Mencione algunos casos específicos, que han dañado más a la institución.
13) Con que frecuencia se dan estos actos.
14) Dentro de las indagaciones hechas a estos actos delictivos realizados dentro de la Universidad, han determinado si estas personas son ajenas o pertenecientes a la institución.
15) Cree que se ha mejorado la seguridad institucional dentro de los 5 años anteriores.
16) A su punto de vista, que formas o técnicas se pueden emplear para el resguardo de la institución y el equipo valioso que posee.
17) Previamente ya se les han dado alternativas o proyectos, para mejorar o dar solución a dicha problemática. ¿Cuáles? ¿Por qué no se han implementado?

18) Cree que el incorporar el RRHH y el recurso tecnológico (nuestro proyecto), beneficiaría a la institución. ¿Por qué?
19) Sugerencias y/o Recomendaciones al proyecto de tesis que se desarrollará.

ENTREVISTA ELABORADA HACIA LOS CUSTODIOS:

OBJETIVO: *Conocer el estado de los que laboran en la Unidad de Custodios, y la situación de la seguridad institucional, actualmente en la FMO***.**

1. ¿Qué necesidades tiene en el desempeño de sus funciones?
2. ¿Cuáles son las herramientas que se le facilitan para el desempeño de sus labores?
3. A su criterio, cuales son los puntos más vulnerables o críticos que requieren de mayor vigilancia.
4. ¿Cuáles son los desafíos que tienen como Unidad de Custodios?
5. ¿Qué sugiere cambiar del sistema de trabajo que desarrolla actualmente?
6. Según su experiencia, en que turno es donde se encuentra el mayor numero de casos delictivos ocurridos en estos 2 últimos años.
7. Mencione algunos casos específicos, que han dañado más a la institución.
8. Con que frecuencia se dan estos actos.
9. Sugerencias y/o Recomendaciones, para mejorar el sistema de vigilancia que tiene la FMO actualmente.
10. Cree usted que les beneficiaria la implementación de tecnología sensorial, para el resguardo de la institución. ¿Por que?

DIAGRAMA ELECTRONICO DEL DISPOSITIVO CONTROLADOR DEL PUERTO PARALELO Y LOS SENSORES