UNIVERSIDAD DE LAS CIENCIAS INFORMÁTICAS

Facultad 5

TRABAJO DE DIPLOMA PARA OPTAR POR EL TÍTULO DE INGENIERO EN CIENCIAS INFORMÁTICAS

Título: Sistema de Vigilancia por Cámaras Cancerbero (SVCC).

Autora

Rosalina Puerto Sorio

Tutor

Ing. Alejandro Manuel Rubinos Carvajal

Co-Tutor

Ing. Ignais La Paz Trujillo

Junio de 2010

Declaración de autoría

Declaro ser autora del presente trabajo de diploma y autorizo al Centro de Informática Industrial de la Universidad de las Ciencias Informáticas para que haga el uso que estime pertinente con el mismo.

Para que así conste firmo la presente a los ___ días del mes de _____ del año _____.

_____
Firma de la Autora
Rosalina Puerto Sorio

_____
Firma del Tutor
Ing. Alejandro Manuel Rubinos Carvajal

_____
Firma del Co-Tutor
Ing. Ignais La Paz Trujillo

Datos de contacto

Tutor: Ing. Alejandro Manuel Rubinos Carvajal.
Edad: 25 años.
Título: Ingeniero en Ciencias Informáticas.
Categoría Docente: Profesor Instructor.
Correo: amrubinos@uci.cu
Tiene un año de graduado en la Universidad de las Ciencias Informáticas (UCI). Actualmente se desempeña como jefe de la línea Seguridad del proyecto SCADA y como arquitecto del Dominio de Supervisión y Control del Centro de Informática Industrial de la Facultad 5.

Co-Tutor: Ing. Ignais La Paz Trujillo.
Edad: 29 años.
Título: Ingeniero en Ciencias Informáticas.
Categoría Docente: Profesor Instructor.
Correo: ilapaz@uci.cu
Tiene tres años de graduado en la Universidad de las Ciencias Informáticas (UCI), dos años trabajando como arquitecto y uno como desarrollador.

Dedicatoria

A mis abuelitos,

Que desde el cielo me guían

Y son fuente de inspiración en mis pasos.

A mis padres,

Quienes han forjado mi camino

Y cada día me exhortan a ser un tilín mejor.

Agradecimientos

A mis padres, por su voto de confianza y por estar siempre; no los hay mejores.

A mi familia, por enseñarme a ser quien soy y por su preocupación constante.

A mi nene, por su amor infinito; qué bueno que te encontré.

A mi nueva familia habanera, por acogerme en su seno como una hija más.

A Meli, por ser la hermana que no tuve; gracias por existir.

A mi tutor, por la paciencia y el apoyo mostrados en todo momento, incluso estando lejos.

A Heidy, Pilar, Yurian, Gilberto, Ignais, Kike y Raudy, por sus acertadas recomendaciones.

A Ariel y Abascal, por las horas de esfuerzo.

A Pedro, por mostrarme los primeros pasos para el trabajo con QT.

A Lisi, Yusmary, Sardiñas, Yolier, Sosa, Argelio, Riolvi y Gustavo por tener siempre una respuesta para mí.

A Juan Carlos, por brindarme su computadora personal que tanto necesité en los últimos momentos.

A todos mis compañeros de aula y de proyecto, por ser como mi segunda familia.

A mis amigos de la FEU, por ser parte de la escuela de mi vida.

A Yaima, Leonardo, Michel, Alexander y Yunier, por ayudarme a limar errores.

A Osmany, por su apoyo incondicional en la etapa más crítica.

A Ide, por prestarme su webcam para el desarrollo de la aplicación.

A Lenia, por sus consejos sobre el vestuario.

A Iliana, por su magnífico regalo.

A Yadira y Ubalquis, por no dejarme caer.

A mi decana, al proyecto SCADA, a la Facultad 5, a la UCI, a la Revolución y a nuestro Fidel, por ser protagonistas de esta inmensa obra.

Y a todo el que de una forma u otra colaboró con la realización de este sueño.

…de todo corazón, GRACIAS...

Resumen

Debido al auge que ha tomado el crimen en el mundo y la necesidad de controlar los recursos, se piensa en el uso de sistemas informáticos que sirvan de apoyo en la detección de actos delictivos. Un ejemplo que forma parte de este grupo de sistemas es la vigilancia por cámaras. Una de las limitantes existentes actualmente es que la mayoría de estas aplicaciones son propietarias, lo cual limita el acceso a este tipo de tecnología. Para dar solución a dicha disyuntiva se presenta el siguiente trabajo de diploma, que tiene como objetivo principal desarrollar un sistema de vigilancia por cámaras que facilite el control y la monitorización de recursos, basado en las directivas del software libre.

Para dar cumplimiento al objetivo planteado se realizó un estudio de los sistemas de vigilancia por cámaras existentes, los principales fabricantes que se dedican al negocio de la video-vigilancia, así como las tecnologías usadas para ello. Durante el desarrollo de la aplicación se describe la modelación de la misma empleando diferentes diagramas que facilitan el entendimiento de su funcionamiento. De igual manera se hace alusión a todo lo relacionado con la implementación del software y con las pruebas realizadas para comprobar su nivel funcional.

Como resultado final de este proceso se puede decir que el Centro de Informática Industrial de la Universidad de las Ciencias Informáticas cuenta ya con la primera versión de un sistema de vigilancia por cámaras en software libre.

Palabras claves: implementación, modelación, monitorización, sistema de vigilancia por cámaras, software.

Summary

Due to the very peak that crime has reached all over the world and the necessity of controlling the resources, the use of computer systems for supporting criminal acts´ detection is widely taken into consideration. One of the examples forming part of this group of systems is surveillance cameras, which are increasingly used in the international scope. One of the currently existing limitations is that most of these applications are proprietary ones, which limits the access to this kind of technology. To solve this problem we have as our main objective to develop a surveillance camera system that facilitates controlling and monitoring resources, based on the free software existing directives.

To accomplish this objective not only a study of surveillance cameras systems was made but also the leading manufacturers engaged in the business of video surveillance and the technologies used to it as well. During the development of the application, the modeling of the different diagrams used for facilitating the understanding of its functioning is described. There is also reference to everything related to the software implementation and the tests applied to prove its efficiency and functional level.

As an outcome for the end of this process, it can be said that the Center for Industrial Computing, corresponding at the University of Informatics Sciences already has the first version of a surveillance camera system based on the free software.

Keywords: implementation, modeling, monitoring, surveillance camera system, software.

Índice

Índice de Figuras

Índice de Tablas

Introducción

El crimen y el desapego con las responsabilidades sociales se encuentran en constante crecimiento en la actualidad. Cada día son más los casos de robos, fraudes, secuestros, evasión de impuestos y otra gran cantidad de delitos los que se cometen en el continente.

Así como muchas otras naciones de América Latina y el Caribe, el crimen en México es una de las preocupaciones más urgentes que enfrenta este país. De manera general existe una tasa elevada de acciones delictivas y se torna difícil controlar aquellas actividades que en ocasiones dependen de la conciencia de las personas o que exista alguien asignado a tiempo completo para velar por su cumplimiento.

De igual manera que en México, las estadísticas de robos y secuestros en Venezuela son alarmantes, así como en Cuba, que a pesar de ser el país menos afectado por este fenómeno, también ha sido víctima en los últimos años de una escalada, aunque en menor por ciento, de violencia e indisciplinas sociales al igual que el resto de América.

Toda esta situación conlleva y obliga a los gobiernos, empresas, negocios e instituciones a velar por la salvaguarda de sus recursos, tanto humanos como materiales, mediante equipamiento y medidas para su aseguramiento. Resulta común el uso de tecnología moderna que facilite, agilice y lleve a cabo el proceso de monitoreo de zonas donde puedan detectarse acciones de índole delictiva en contra de personas o recursos materiales. Un ejemplo de ello es el empleo de sistemas de video-vigilancia que proporcionan mecanismos de captura, procesado, y almacenamiento de videos e imágenes.

Muchos de estos sistemas poseen herramientas adicionales de reconocimiento de patrones que conllevan a la identificación de acciones punibles. Es por ello que es muy frecuente su uso en negocios de primer nivel y en países del primer mundo, debido en particular al alto costo que pueden tener en cuanto a productos, equipamiento y soporte. En este aspecto toma principal fuerza el tema del pago de licencias en materia de software.

Países del tercer mundo tales como los de América Latina y el Caribe, así como pequeños negocios y

otras entidades, en muchas ocasiones no cuentan con el suficiente respaldo económico para poder implantar sistemas de vigilancia por cámaras; sin embargo son los más necesitados debido al alto grado de delincuencia existente y a la superación de los criminales que mejoran por día las técnicas que emplean para burlar la seguridad de los establecimientos, por lo que se va haciendo mucho más difícil para las autoridades y responsables del orden interceptar al protagonista del delito.

Cuba cuenta con la Universidad de las Ciencias Informáticas (UCI), una universidad productiva cuya misión radica en crear servicios informáticos y software a partir de la vinculación estudio-trabajo como modelo de formación. El Centro de Informática Industrial (CEDIN) de la Facultad 5 fue creado con este fin y actualmente cuenta con varios productos de gran importancia estratégica para el país. Dentro de los proyectos que conforman este centro se encuentra "Seguridad", el cual nace como un módulo del SCADA[1] Nacional "Guardián del Alba", sistema desarrollado para el control y la adquisición de datos. El proyecto Seguridad tiene como objetivo fundamental disminuir las vulnerabilidades que puedan tener sistemas de este tipo y proteger sus recursos mediante aplicaciones informáticas. Debido a ello, y teniendo en cuenta además que las relaciones de la UCI con países que aclaman soluciones más asequibles ante el afán delictivo como México y Venezuela han aumentado y se profundizan cada día más, nace la idea de vincular técnicas de vigilancia a nuestro quehacer informático. En este camino toma papel protagónico el hecho que factores de alto riesgo como la inseguridad obligan a requerir de un mecanismo de monitoreo constante con el fin de proteger los recursos del centro y de sus clientes, así como la posibilidad de lograr un fortalecimiento en las soluciones de software que oferta CEDIN al mercado, tanto fuera como dentro de Cuba.

Por tanto se puede definir la siguiente **situación problémica**: Necesidad de contar con una herramienta de monitoreo y control más asequible que facilite la protección de los recursos del Centro de Informática Industrial y sus clientes.

Ante esta situación se puede definir el siguiente **problema científico**: ¿Cómo lograr un sistema de vigilancia más asequible que facilite la protección de recursos para el Centro de Informática Industrial?

Para ello el **objeto de estudio** lo constituye: Los sistemas de vigilancia.

[1] Siglas en inglés: Supervisory Control And Data Adquisition (Control Supervisor y Adquisición de Datos).

Para dar respuesta al problema de la investigación se define que el **objetivo general** consista en: Desarrollar un sistema de vigilancia por cámaras bajo los principios del Software Libre para el Centro de Informática Industrial.

Derivándose como **campo de acción**: Los sistemas de vigilancia mediante el uso de cámaras para la monitorización y control de recursos.

Se establece como **idea a defender** que: El desarrollo de un sistema de vigilancia por cámaras basado en tecnologías libres para el Centro de Informática Industrial, permitirá la obtención de un producto más asequible que facilite la captura, procesado y almacenamiento de videos e imágenes.

Conforme a lo planteado como objetivo general se derivan las siguientes **tareas investigativas**:

- Estudio de los sistemas y técnicas de video-vigilancia de mayor uso a nivel mundial para establecer un diagnóstico de las tendencias actuales y definir las funcionalidades que debe tener la aplicación.
- Estudio de las herramientas y tecnologías libres existentes para el trabajo con cámaras para seleccionar las que se adapten a las necesidades actuales.
- Realización del análisis y diseño del sistema valorando posibles aplicaciones de patrones de diseño estudiados para lograr la obtención de un diseño flexible y reutilizable.
- Implementación del sistema y realización de pruebas al mismo, para obtener un producto que cumpla con las funcionalidades definidas.

Los **resultados** esperados al finalizar la investigación son:

- Obtención de la documentación relacionada con la primera versión del Sistema de Vigilancia por Cámaras (SVCC 1.0).
- Obtención de la primera versión funcional del producto SVCC.

La investigación está estructurada de la siguiente manera:

- **Capítulo 1:** Se hace referencia a la seguridad y a los sistemas de vigilancia existentes en el mundo describiendo sus características principales y las tecnologías usadas.

- **Capítulo 2:** Se explican las metodologías y herramientas seleccionadas para el desarrollo de la aplicación.

- **Capítulo 3:** Se describe la solución propuesta. Se explica el proceso de modelación del mismo y sus características generales mostrando algunos diagramas para un mejor entendimiento.

- **Capítulo 4:** Se explica cómo se lleva a cabo la implementación del software y se describen los casos de prueba realizadas al mismo.

Capítulo 1: Fundamentación teórica

1.1 Introducción al capítulo

En este capítulo se hará un análisis acerca de la importancia de la vigilancia como factor necesario para la seguridad mediante el uso de cámaras. Además, se pretende indagar acerca de las diferentes organizaciones que son vanguardia en este mercado, así como las aplicaciones existentes más conocidas tanto de software libre como de software propietario. De igual manera se hablará acerca de la tecnología implicada en los sistemas de vigilancia de manera general, explicando sus características y evolución a través de los años.

1.2 Seguridad mediante la video-vigilancia

El acceso a las nuevas tecnologías requiere una inversión importante en equipos informáticos. Esta inversión en muchos casos es difícil de recuperar a corto plazo, por lo que estos medios suponen una parte importante del activo de la empresa, de ahí la aconsejable necesidad de garantizar su seguridad.

En la computación, con el advenimiento de las redes de área local, los sistemas operativos empezaron a implementar diferentes niveles de seguridad, entre los cuales se encuentran: establecer fiabilidad de los datos guardados, otorgar confiabilidad del sistema frente a diferentes configuraciones de hardware, y establecer restricciones de acceso selectivas a diferentes recursos. Amén de la seguridad de los sistemas o de las políticas de seguridad implementadas por una oficina informática, las conductas humanas naturalmente atentan contra los sistemas implementados. Es por ello que la mayoría de las empresas requieren de un control exhaustivo de las existencias o la necesidad de tener bajo vigilancia una o varias zonas.

En cualquier negocio, la vigilancia es una herramienta que se necesita contra robos y delitos, tanto para su prevención, como para la identificación posterior de los culpables. El no poder detectar y controlar un amplio espectro de situaciones tales como intrusión, fuego, inundación y atraco, y el tener varias personas vigilando las zonas más importantes de una empresa, genera un gasto que en ocasiones puede ser muy elevado. Además, un error humano puede despistar un detalle de visión que una cámara deja grabado, y

si están estratégicamente instaladas se evita un importante gasto en personal y un mejor control de la situación.

De este ámbito se deriva la existencia de los sistemas de vigilancia mediante el uso de cámaras, los cuales se han convertido en un fuerte apoyo al tema de la seguridad integral, incluyendo entre sus virtudes ejercer una vigilancia preventiva mediante el registro visual de sucesos. Su incorporación y aplicación en el mercado va dirigida a asegurar un amplio espectro de ambientes y lugares tales como: centros comerciales, supermercados, aeropuertos, viviendas particulares, vías públicas, centros de eventos, transporte público, establecimientos educacionales y empresas de todo tipo.

1.2.1 En el mundo

Existen muchas compañías y organizaciones que se dedican al negocio de la video-vigilancia a nivel mundial. Algunas de ellas son:

- **Scati Labs:** Desarrolla, fabrica y comercializa sistemas de video-vigilancia que permiten la gestión de múltiples cámaras de seguridad para el control y supervisión de instalaciones locales y remotas (1). En la actualidad Scati Labs opera en los mercados español y latinoamericano.

- **Davantis**: Fabrica sistemas innovadores de análisis de video que analizan las imágenes provenientes de las cámaras de seguridad y generan alarmas cuando se produce una situación de riesgo (2). Davantis es el primer fabricante de sistemas de análisis de video en España.

- **SecureMaster S.A.**: Ofrece sistemas utilizados en grandes casinos y hoteles del mundo por la calidad, alta tecnología y soporte técnico (3).

- **Intplus S.L.**: Es una empresa que cuenta con una amplia experiencia en la comercialización de productos destinados a hacer más cómoda y segura la vida de sus clientes. Cuenta con un grupo de tiendas online que ayudan a la comercialización.

- **Securimport**: Es una empresa dedicada a la comercialización de productos relacionados con la seguridad, y la domótica[2] de los principales fabricantes del sector (4).

[2] Automatización de una vivienda mediante el uso de tecnología.

- **CirControl**: En los Sistemas de video-vigilancia dispone de una gama compuesta por todos aquellos elementos que pueden ser necesarios para dar solución a cualquier situación, tanto de equipos de gama convencional como de gama IP[3] (5).

- Y otras empresas como **Panasonic**, **Sony** y **Axis** que tienen como característica distintiva que los sistemas de vigilancia que desarrollan solamente funcionan para cámaras fabricadas por sus compañías, lo cual es una técnica para lograr comercializar sus productos en el mercado.

Además, existen tiendas virtuales que se dedican a la venta de productos de esta rama de la seguridad tales como:

- **areacctv.com**: Tienda on-line de material de circuito cerrado de televisión, video grabación digital, cámaras digitales, cámaras web, y videograbadores digitales (6).

- **rimaxonline.com**: Fabricante de productos de electrónica de consumo especializado en vigilancia infantil, video-vigilancia para el hogar, pequeños comercios y productos multimedia (7).

1.2.2 En Cuba

En Cuba el mundo de la video-vigilancia se ha ido introduciendo poco a poco. Ya algunas empresas hacen inversiones con entidades extranjeras para implantar sistemas de este tipo y actualmente los vemos instalados en algunas tiendas y centros de trabajo. El Ministerio del Interior (MININT) cuenta con una institución civil denominada DATYS que se dedica a la producción de software. Dicha institución se ha ido especializando paulatinamente en todo lo referente a tecnología y sistemas, ofreciendo soluciones informáticas a los organismos que lo soliciten.

Desde hace cuatro años DATYS ha venido trabajando en la realización de un sistema de video-vigilancia para el MININT denominado *XymaVision.* Actualmente no han finalizado la etapa de implementación de todos los módulos que completarían sus funcionalidades. No obstante a ello, *XymaVision* ya cuenta con una versión funcional que se encuentra desplegada por todo el país permitiendo el monitoreo de las cámaras digitales que han sido instaladas para el control del tránsito nacional.

[3] Siglas en inglés: Internet Protocol (Protocolo de Internet).

De igual manera, hace más de un año en la facultad 9 de la Universidad de las Ciencias Informáticas un equipo de trabajo compuesto por estudiantes y profesores desarrolla un sistema de vigilancia por cámaras basado en tecnología IP. Dicho sistema ya cumple con los requisitos funcionales para el cual fue concebido, aunque actualmente se sigue trabajando en la agregación de nuevas funcionalidades y potencialidades que garanticen mayor eficiencia y robustez.

Ambos proyectos presentan una problemática y es precisamente su total dependencia hacia el software propietario, ya que han sido desarrollados utilizando la plataforma .NET[4] y por ende no se aprecian posibilidades de aplicarse en entidades o centros como el Centro de Informática Industrial que tengan como premisa la utilización de tecnología libre en sus soluciones. Por lo anteriormente planteado se puede afirmar que no se conoce en Cuba la existencia de un sistema de video-vigilancia fabricado bajo los principios de software libre.

1.2.3 Sistemas desarrollados

Luego de analizar la existencia de empresas dedicadas a la fabricación de software de vigilancia por cámaras tanto en nuestro país como en el mundo, se deduce que con tantas personas e instituciones involucradas en este mercado, muchos han sido los productos que se han desarrollado para aplicarse en diferentes contextos. Ya fueron mencionados los existentes en Cuba por lo que a continuación se hace referencia a algunos de los desarrollados en todo el mundo:

- **CamUniversal**: Es una solución informática para todo tipo de cámara web, netcams[5], y en general para cualquier dispositivo de video que funcione en Windows. El objetivo de CamUniversal es actuar como sistema de vigilancia, conectando la cámara en cuestión, y avisando el programa en el momento que detecte algún cambio o movimiento en pantalla. Los avisos los puede configurar el usuario según le convenga, activando una alarma o realizando cualquier otra operación predefinida (8).

- **Campermanent:** Podría considerarse un auténtico sistema integral de vigilancia doméstica, que soporta cámaras web, netcams, o cualquier otro dispositivo de video que funcione bajo Windows

[4] Entorno lanzado por Microsoft diseñado para el desarrollo y ejecución de software en forma de servicios.
[5] Cámaras que funcionan por IP.

(9). El programa permite grabar continuamente e incorpora también detectores de movimiento integrado con variadas opciones muy potentes.

- **Eyeline Video Surveillance Software:** Es un sistema profesional de grabación y vigilancia de video en tiempo real[6], que se usa para capturar la emisión de video procedente de una o varias cámaras (10). Incluye monitorización de video en oficinas y edificios, tiene capacidad para grabar múltiples canales de video simultáneamente y para detectar automáticamente variaciones de movimiento en el área monitorizada.

Las aplicaciones mencionadas tienen como requisito fundamental que solo funcionan con el sistema operativo Windows, pues son aplicaciones de software propietario. A continuación se mencionan algunas de las aplicaciones de software libre existentes más utilizadas:

- **Devolution Security:** Soporta hasta 16 cámaras de vigilancia con detección de movimientos, streaming unicast y multicast[7]. Inicia el proceso de grabación al detectar movimientos, con una calidad de 25 fotogramas por segundo y formato MPEG-4[8]. Además, ofrece diferentes temas de escritorio y layouts (diseños) para las cámaras. Soporta multicast a través de una red local o unicast a una dirección IP (11).

- **Motion:** Es un programa que utiliza el V4L[9] (video para Linux) para recibir imágenes de una cámara, está grabando constantemente y, en el momento que algo entra en su campo de acción, detecta una diferencia entre frames (cuadros) y salva varios screenshots (captura de pantalla) mientras dure el movimiento, desde ese momento, comienza a grabar o a generar eventos previamente configurados. Su configuración es muy sencilla y no hace mucho uso de disco duro si el número de frames y la calidad de la imagen no son muy altos.

- **ZoneMinder:** Cuenta con varias funcionalidades que conjuntamente proporcionan una completa solución de video-vigilancia permitiendo capturar, analizar, grabar y monitorizar cualquier cámara

[6] Tipo de procesamiento en el cual una transacción es ejecutada y procesada sin espera alguna.
[7] Distribución de video ya sea desde un único emisor a un único receptor (unicast) o a varios de ellos (multicast).
[8] Es un formato contenedor especificado como parte del estándar internacional MPEG (en inglés: Moving Picture Experts Group) para almacenar formatos audiovisuales.
[9] Siglas en inglés: Video for Linux (video para Linux).

CCTV conectada a una máquina basada en Linux. Está diseñado para ejecutarse en distribuciones que soporten la interfaz de video para Linux y puede soportar múltiples cámaras sin pérdida aparente de rendimiento (12).

A continuación se muestra una comparación entre los sistemas de software libre antes mencionados.

	ZoneMinder	**Motion**	**Devolution Security**
Lenguaje de Programación	C++, JavaScript, PHP, Perl.	C.	C++.
Documentación	Abundante y variada.	No muy abundante.	Escasa.
Capacidad de análisis de imágenes	Media.	Baja.	Baja.
Visualización de captura de imagen	Utiliza una interfaz web que permite ver la imagen de las cámaras.	Es una aplicación de línea de comandos que no permite visualizar la imagen, todo es a través de la consola.	Permite la visualización de la imagen capturada por la cámara en pantalla.
Dispositivos que soporta	Todo tipo de cámaras que soporten el video para Linux.	Todo tipo de cámaras que soporten el video para Linux.	Hasta 16 cámaras que soporten el video para Linux.
Requisitos para su uso	Apache, PHP, MySQL.	Ninguno.	Ninguno.
Eventos	Mantiene un registro de eventos o de actividad sospechosa en las zonas vigiladas.	Notifica eventos en dependencia de cómo se haya configurado.	No tiene notificación de eventos.
Calidad	Media.	Baja.	Baja.

Tabla 1 Comparación entre aplicaciones de software libre.

Por las características ventajosas que presenta ZoneMinder respecto al resto de los sistemas comparados se decide realizar un estudio más profundo del mismo con el fin de validar su posible utilización en el proyecto ya que cumple con la característica de ser software libre.

ZoneMinder es un sistema que ha sido tema de debate en varios foros de la comunidad de software libre debido a las dificultades que presenta en cuanto a la consistencia de sus funcionalidades. Muchos son los comentarios de personas a las que no ha beneficiado la utilización de dicho software por problemas de

compatibilidad, instalación o configuración. Esto representa un problema ya que es de vital importancia contar con una herramienta de fácil instalación y manipulación, que no conlleve amplios conocimientos especializados para su uso y que posibilite la captura, procesado y almacenamiento de videos e imágenes de manera eficiente. ZoneMinder presenta además características que no cumplen con las expectativas que se tienen del sistema a desarrollar, tales como: visualización mediante una interfaz web y utilización de MySQL como gestor de base de datos.

Se decide que el sistema de vigilancia para CEDIN sea una aplicación de escritorio debido a que con ello se obtiene un software con mayor capacidad gráfica visual, mayor personalización y menor tiempo de respuesta, o sea, una aplicación más rápida. Si se tiene en cuenta que la esencia de la video-vigilancia es precisamente la visualización del flujo de video obtenido de la cámara en tiempo real, pues es de fácil entendimiento la necesidad de lograr rapidez en este proceso. Además, se desea utilizar PostgreSQL como gestor de base de datos con el objetivo de aprovechar sus bondades y de ser consecuentes con las políticas del Centro de Informática Industrial. Por los motivos antes mencionados se descarta la posibilidad de utilizar cualquiera de los sistemas desarrollados en software libre recién analizados.

1.3 Circuito Cerrado de Televisión

Debido a la diversidad existente de sistemas de vigilancia en el mercado actual y teniendo en cuenta la importancia que tiene para nuestra aplicación comprender sus potencialidades, se hace necesario salir a la búsqueda de un factor común. La mayoría de estos sistemas obedece al uso de una tecnología denominada Circuito Cerrado de Televisión.

El Circuito Cerrado de Televisión o su acrónimo CCTV, que viene del inglés: *Closed Circuit Television*, es una tecnología de video-vigilancia visual diseñada para supervisar una diversidad de ambientes y actividades.

La televisión comercial que conocemos está abierta al público ya que a través del aire, e incluso a través de cables (televisión por cable), se hace llegar a todo aquel que quiera observar la programación. En el caso del circuito cerrado, el video generado se conserva privado y únicamente son capaces de observarlo las personas asignadas para ello dentro de una organización. Mientras que en un sistema abierto, el propósito fundamental es la diversión y/o información, en un sistema cerrado el propósito fundamental es

la vigilancia.

El CCTV consiste en una combinación de múltiples cámaras, ya sean estacionarias o de rotación, conectadas a un conjunto correspondiente de monitores de circuito cerrado. Estos monitores son de similar aspecto al de un televisor común, pero carecen de los controles que permiten la sintonización de televisión a los espectadores al cambiar de canal. Básicamente, las cámaras utilizadas en un sistema de CCTV están conectadas a través del cableado mediante un codificador, que gestiona el flujo de información de lo que corresponde vigilar. Las imágenes del circuito cerrado de televisión permanecen dentro de la red de monitores y cámaras sin importar el tipo de conexión.

Los últimos avances tecnológicos han acercado el circuito cerrado de televisión a las computadoras y a la televisión en términos de complejidad. Fotografía e imagen digital han permitido a los proveedores de circuitos cerrados de televisión publicar sistemas que permiten más opciones de la cámara y mayor resolución de la imagen en los monitores. Además, la capacidad de hacer las cámaras más pequeñas permite sistemas de vigilancia menos evidentes y la capacidad de colocar cámaras en los lugares más pequeños.

1.3.1 Historia

El uso de esta tecnología comenzó como un elemento de la seguridad de la preparación militar. El primer uso documentado de circuito cerrado de televisión fue en el año 1942 por el ejército alemán. La instalación de cámaras remotas y monitores en blanco-negro era importante para la observación de los ensayos de misiles V2[10] en la preparación de los ataques militares de larga distancia.

Los alemanes no fueron los únicos en el uso del CCTV en la década de 1940, los Estados Unidos también utilizaron la tecnología en el Proyecto Manhattan. Este proyecto consistió en el desarrollo de un arma atómica en los desiertos del suroeste americano y el uso de este sistema permitió que científicos y militares observaran el éxito de las pruebas desde lejos.

Entre los años 1980 y 1990 se populariza como un instrumento de gobernabilidad local en Gran Bretaña, con la instalación de decenas de miles de sistemas de circuito cerrado para controlar el tráfico y ayudar a

[10] Vergeltungswaffe 2 (V2), conocido como “A4” en su fase de desarrollo.

combatir la creciente tasa de delincuencia. Además, se convirtió en una herramienta importante para las autoridades británicas y americanas de tránsito en lugares como Londres y Nueva York, con cámaras colocadas en taxis, autobuses y estaciones de tren, para evitar el vandalismo y garantizar el transporte oportuno de los clientes. En las ciudades de California, en la década de 1990, se instalaron cámaras en los semáforos, a fin de realizar el seguimiento de violadores de las normas de tránsito y enviar las multas a los propietarios de los automóviles.

1.3.2 Ventajas

Las ventajas del CCTV incluyen la capacidad de observar las situaciones de peligro a distancia, la posibilidad de proporcionar un ojo constante de las actividades rutinarias, y una gran herramienta de seguridad para el hogar y las empresas, en un esfuerzo por combatir la delincuencia.

1.3.3 Desventajas

Si bien el circuito cerrado de televisión brinda muchas ventajas en términos de seguridad, también ha provocado insatisfacción en la comunidad respecto a los derechos de privacidad. Un ejemplo potencial de la preocupación de las libertades civiles es la utilización de circuitos cerrados de televisión en la prevención de pérdidas en las grandes tiendas minoristas. Los expertos en prevención de pérdidas consideran que si pueden controlar a los consumidores en vestuarios y zonas de espera, los pueden enfrentar de manera adecuada y sancionar a los que quieran robar ropa y otros artículos. Sin embargo, la utilización de circuitos cerrados de televisión para observar al hombre y a la mujer vestirse y desvestirse se considera inoportuno y un paso innecesario de seguridad cuando las etiquetas electrónicas de seguridad son herramientas eficaces para capturar a los ladrones.

1.3.4 Aplicaciones

En la actualidad se mantiene el empleo del circuito cerrado por las fuerzas militares en todo el mundo para la realización de pruebas y simulacros, pero ya se ha expandido también a otros organismos gubernamentales y al sector privado.

El gobierno de las ciudades usa el circuito cerrado de televisión para observar la congestión del tráfico y

disuadir violaciones de exceso de velocidad, mientras que los fabricantes lo utilizan para observar los procesos de producción. Además, puede ser utilizado por los padres y los profesores para observar a los niños y estudiantes en diferentes salas, tanto para echar un vistazo al comportamiento de grupo como para asegurarse de que actúen adecuadamente. Tales actividades requieren de observación de rutina a fin de determinar el comportamiento común de los niños cuando los adultos no están en la sala.

También es utilizado el CCTV para monitorear el uso de determinadas vías de circulación en las ciudades, así como garantizar que las calles estén iluminadas y los sistemas de drenaje estén trabajando a su máxima eficiencia. Negocios como tiendas de regalo, bancos y estaciones de gasolina, han utilizado el CCTV durante años con el objetivo de prevenir posibles asaltos y detectar cualquier comportamiento sospechoso. Muchas veces la sensación de un sentimiento de seguridad es tan importante como prueba tangible de una mayor seguridad.

A continuación se listan algunos otros ejemplos de aplicaciones del circuito cerrado:

- Sondas médicas con micro cámaras introducidas en el cuerpo humano.
- Vigilancia en condiciones de absoluta oscuridad, utilizando luz infrarroja.
- Vigilancia de la población reclusa en los centros penitenciarios de máxima seguridad.
- Vigilancia de estacionamientos, incluyendo las placas del vehículo.
- Análisis facial para identificación de criminales en áreas públicas.

Lógicamente, en casi todos los casos el CCTV debe estar acompañado de la grabación de los eventos que se vigilan. Esto se realiza con el objetivo de obtener evidencia de los movimientos de mayor interés y de minimizar la presencia humana ante los monitores constantemente.

1.3.5 Evolución

La *Figura 1* muestra un sistema CCTV que emplea cable coaxial[11] para conectar los dispositivos

[11] Cable utilizado para transportar señales eléctricas de alta frecuencia

involucrados. Se establecen conexiones punto a punto desde las cámaras analógicas hasta el multiplexor (MUX) o concentrador de las señales de video, y además, se puede conectar un multiplexor entre la cámara y el VCR para grabar el video proveniente de cualquier cámara. Las grabadoras de video (VCR, Video Camera Recorder) almacenan las secuencias en cintas magnéticas.

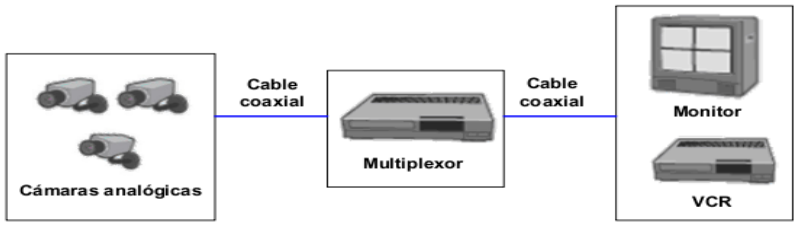

Figura 1 Esquema de la plataforma de video-vigilancia analógica.

1.3.5.1 CCTV usando DVR

En estos sistemas CCTV se usa un grabador de video digital (DVR, Digital Video Recorder) encargado de almacenar el video digitalizado tomado de las cámaras analógicas. (*Figura 2*) En esta variante el disco duro reemplaza las cintas magnéticas y se hace necesario comprimir el video para almacenar la máxima cantidad de imágenes en un día. En los primeros DVR el espacio del disco duro era limitado y representaba un inconveniente.

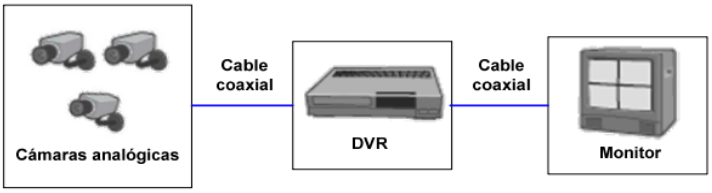

Figura 2 Sistema analógico que incluye un DVR.

En nuestros días el DVR puede incluir características IP y mediante una conexión a Internet el video digital se puede monitorizar remotamente.

1.3.5.2 Sistemas de video IP que utilizan servidores de video

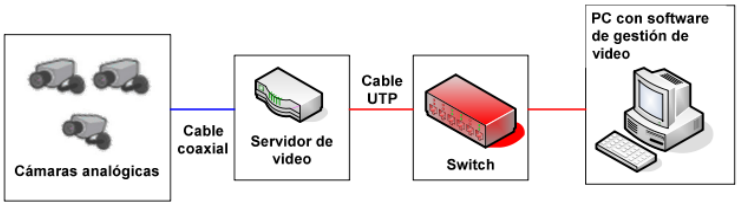

Figura 3 Sistema de video-vigilancia IP con servidor de video.

El video digital y las redes de datos dan origen a la video-vigilancia IP, la cual permite migrar a sistemas y medios de transmisión más flexibles. Un sistema de video-vigilancia IP puede emplear la infraestructura de una red de datos. El "servidor de video", como se ha definido en el mercado al encargado de digitalizar, comprimir y distribuir las secuencias de video en la red, es el nexo entre los sistemas de vigilancia analógicos y la video-vigilancia IP. (*Figura 3*)

1.3.5.3 Sistemas de video-vigilancia IP

Una cámara IP combina una cámara digital y un servidor de video en una unidad y posee como funciones principales: la digitalización, compresión y transmisión del video hacia una red de datos. El sistema de vigilancia a través del protocolo de comunicaciones IP rompe con las barreras de la video- vigilancia tradicional.

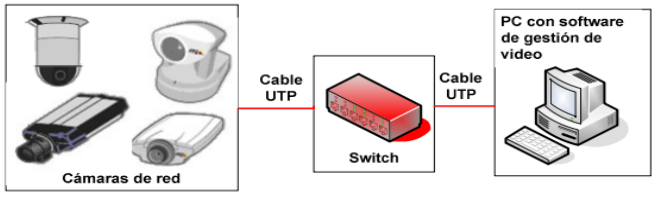

Figura 4 Sistema de vigilancia digital IP.

El desarrollo tecnológico va muy por delante de la adaptación comercial del sector de la seguridad. El mercado ha reaccionado rápidamente y con entusiasmo, mientras que la rama de la seguridad ha quedado un poco rezagada. No obstante, últimamente se ha descubierto el mercado potencial que la

vigilancia IP ofrece, aunque la falta de conocimiento sobre la tecnología es un aspecto importante a tener en cuenta ante la lenta adopción de este producto en la rama de la seguridad para la demanda existente en el mercado.

1.3.6 Comparación entre cámaras IP y analógicas

- **Cámara IP:** Lo que se conoce comúnmente como una cámara IP, es una cámara que digitaliza y procesa imágenes y señales analógicas, las comprime internamente y luego transmite el video en forma digital sobre una conexión de red a una computadora o a un dispositivo similar. Las cámaras IP están equipadas con un servidor web en su interior, esto facilita el acceso y control de las mismas a través de aplicaciones de software que permiten ver el video de manera local o remota.

- **Cámara Analógica:** Una cámara analógica de video-vigilancia contiene un sensor CCD[12] el cual digitaliza la imagen para procesarla. Después de esto puede transmitir el video, para lo cual necesita convertirlo de nuevo a su forma análoga y ser recibido por un dispositivo analógico, como un monitor o un videograbador. A diferencia de las cámaras IP no tienen servidores web dentro de la misma cámara o compresores de video, y no requieren mantenimiento. Esas funciones son implementadas en el dispositivo que graba o controla el video.

La principal diferencia entre ambas tecnologías de cámaras es la forma en la cual el video es transmitido, y en los últimos años el lugar donde el video se almacena y comprime.

	IP	**Analógica**
Calidad de video	Es excelente para capturar imágenes de alta definición. Presenta problemas para condiciones de poca luz, necesitando un apoyo con lámparas IR (infrarrojas). Está limitada en las herramientas para la compresión de video. La compresión del video introduce un retardo.	El sensor CCD que emplea la hace excelente para una variedad de condiciones de iluminación e imágenes en movimiento. La compresión del video se lleva a cabo en los DVR, por lo cual se permite verlo sin retardos.
Infraestructura	Utiliza la infraestructura de cableado de	El cable utilizado es coaxial. No se

[12] Sensor con diminutas células fotoeléctricas que registran la imagen.

de cableado	una red ya existente. Requiere un conocimiento en la configuración y establecimiento de redes.	requiere de conocimientos de configuración de redes.
Transmisión de video	El tráfico de video IP está sujeto a fallas potenciales, tales como: un ancho de banda limitada o red congestionada. Cada vez que se añade una cámara se debe de asegurar que el sistema soporta ese modelo en particular.	El tráfico de video analógico no está sujeto a riesgos de la red. Una cámara analógica se puede conectar a cualquier DVR, no hay incompatibilidad entre cámaras y DVRs.
Seguridad	Puede ser encriptado dificultando el ser interceptado. Por otro lado, la red misma esta propensa a virus y otros tipos de ataque.	Es menos seguro y puede ser interceptado y/o visto por cualquiera con acceso a la infraestructura del cableado.
Obsolescencia	Está en evolución constante. Hoy en día los modelos de cámaras son reemplazados rápidamente, buscando mayor calidad, eficiencia y rentabilidad.	Es una tecnología madura que posee una historia y un propósito bien definidos.
Costo	Puede costar tres veces más que su equivalente analógica. Sistemas grandes con este tipo de cámaras, requieren ser equipados con switch[13] y periféricos, lo cual los hace muy costosos.	Es significativamente menor en precio que su contraparte IP. Requiere de gran infraestructura de cableado aunque es menos costosa.

Tabla 2 Comparación entre cámaras IP y analógicas.

Resulta delicado concluir que una tecnología sea mejor que otra. Lo que se busca siempre es un compromiso entre la calidad de video y su almacenamiento contra el costo del sistema de video-vigilancia. Se puede también hacer una conexión híbrida entre componentes analógicos e IP para convivir en el mismo sistema de video-vigilancia.

1.4 Conclusiones del capítulo

El estudio de los diferentes sistemas de video-vigilancia que han sido mencionados en este capítulo ha permitido identificar las funcionalidades principales que deben poseer las aplicaciones de este tipo. Las mismas se engloban en dos grandes grupos que se especifican a continuación:

[13] Dispositivo digital de lógica de interconexión de redes de computadoras.

Gestión de monitores: Se refiere al trabajo con las cámaras que soporta el sistema.

- Añadir y eliminar cámaras.
- Configurar los datos de la cámara.
- Mostrar el video que captura la cámara según la zona donde está ubicada.

Gestión de usuarios: Se refiere al trabajo con las personas que de una manera u otra pueden interactuar con el sistema.

De esta manera, se ha obtenido una definición más cercana a las principales funcionalidades que debe poseer el sistema de vigilancia por cámaras.

Gracias al estudio del Circuito Cerrado de Televisión se logró conocer acerca de la existencia de los sistemas de video-vigilancia híbridos, los cuales pueden tener dos interpretaciones:

1. Permiten explotar las ventajas de cada una de las tecnologías.
2. La tecnología digital va desplazando lentamente a la analógica.

Teniendo en cuenta la comparación realizada entre las cámaras IP y analógicas se ha podido delimitar que el soporte del sistema a desarrollar sea para aquellas que no posean tecnología IP, pero que puedan ser conectadas a una computadora posibilitando así la obtención del video en formato digital.

Capítulo 2: Herramientas y tecnologías utilizadas

2.1 Introducción al capítulo

En este capítulo se hace referencia a las herramientas empleadas para el modelado del sistema, así como las tecnologías que se utilizan en el desarrollo y documentación de la aplicación.

2.2 Sistema operativo: GNU/Linux

Un sistema operativo puede ser contemplado como una colección organizada de extensiones software del hardware, consistente en rutinas de control que hacen funcionar al computador y proporcionan un entorno para la ejecución de programas. Estos programas utilizan las facilidades proporcionadas por el sistema operativo para obtener acceso a recursos del sistema informático como el procesador, archivos y dispositivos de entrada/salida (E/S). De esta forma, el sistema operativo constituye la base sobre la cual pueden escribirse los programas de aplicación, los cuales invocarían sus servicios por medio de llamadas al sistema. Por otro lado, los usuarios pueden interactuar directamente con él a través de órdenes concretas. En cualquier caso, actúan como interfaz entre los usuarios/aplicaciones y el hardware de un sistema informático. (13)

GNU/Linux es un sistema operativo que ha sido desarrollado bajo los principios del software libre. Software libre se refiere a la libertad de los usuarios para ejecutar, copiar, distribuir, estudiar, cambiar y mejorar el software. De modo más preciso, se refiere a cuatro libertades de los usuarios del software:

- La libertad de usar el programa, con cualquier propósito *(libertad 0).*
- La libertad de estudiar cómo funciona el programa, y adaptarlo a tus necesidades *(libertad 1)*. El acceso al código fuente es una condición previa para esto.
- La libertad de distribuir copias, con lo que puedes ayudar a tu vecino *(libertad 2).*

- La libertad de mejorar el programa y hacer públicas las mejoras a los demás, de modo que toda la comunidad se beneficie *(libertad 3)*. El acceso al código fuente es un requisito previo para esto. (14)

Se selecciona el sistema operativo GNU/Linux que por su característica de ser software libre, resulta un sistema de alta calidad tecnológica con menos errores que los sistemas comerciales, a un costo cero o muy bajo y con la disponibilidad del código fuente que permite aprender, modificar o ayudar al desarrollo del sistema.

Una distribución es una recopilación de programas y ficheros, organizados y preparados para su instalación. Existen muchas y variadas distribuciones de GNU/Linux creadas por diferentes empresas y organizaciones. En este caso se decide utilizar la distribución Ubuntu en su versión 9.04 debido a que es muy popular por el gran soporte que tiene en la comunidad, además está centrada en el usuario final y posee importantes características tales como: facilidad de uso, velocidad, seguridad, estabilidad y eficiencia.

2.3 Metodología de desarrollo: RUP

Una metodología de desarrollo es una colección de documentación formal referente a los procesos, las políticas y los procedimientos que intervienen en el desarrollo del software. La finalidad de una metodología de desarrollo es garantizar la eficacia (p. ej. cumplir los requisitos iniciales) y la eficiencia (p. ej. minimizar las pérdidas de tiempo) en el proceso de generación de software. (15)

Se selecciona RUP como metodología de desarrollo debido a sus características de ser: dirigido por casos de uso, iterativo e incremental y centrado en la arquitectura.

Dirigido por casos de uso[14]: Significa que el proceso de desarrollo sigue una trayectoria a través de flujos de trabajo generados por casos de uso.

Iterativo e incremental: Posibilita que se pueda dividir el trabajo en partes más pequeñas o en mini proyectos, permitiendo el equilibrio entre casos de uso y arquitectura durante cada mini proyecto. Cada

[14] Casos de uso: Describen la funcionalidad del sistema, en términos de su importancia para el usuario.

parte se puede ver como una iteración de la cual se obtiene un incremento, provocando un aumento del producto.

Centrado en la arquitectura: Facilita la visión del sistema completo. La arquitectura describe los elementos del modelo que son más importantes para su construcción, o sea, los cimientos del sistema que son necesarios como base para comprenderlo, desarrollarlo y producirlo económicamente.

El RUP divide el proceso de desarrollo en ciclos, obteniendo un producto final al culminar cada ciclo.

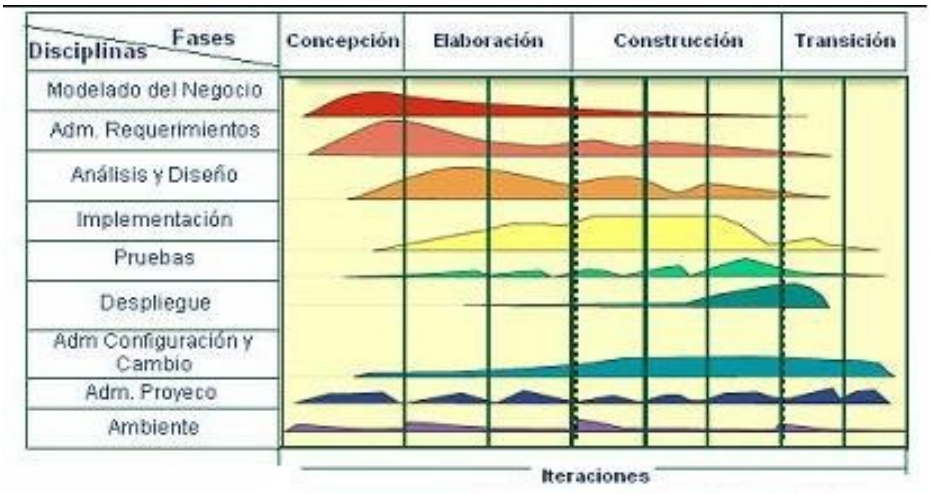

Figura 5 Metodología de desarrollo de software: RUP.

En cada ciclo se describe el negocio y se delimita el proyecto describiendo sus alcances con la identificación de los casos de uso. Además, se define la arquitectura del sistema cuyas decisiones se hacen sobre la base de la comprensión del mismo y los requisitos (funcionales y no funcionales) identificados de acuerdo al alcance definido. Por último, se obtiene un producto listo para su utilización y correctamente documentado.

2.4 Lenguaje de modelado: UML

El Lenguaje Unificado de Modelado (UML, por sus siglas en inglés, Unified Modeling Language) es el lenguaje de modelado de sistemas de software más conocido y utilizado en la actualidad. Es un lenguaje gráfico para visualizar, especificar, construir y documentar un sistema de software.

Debido a que la metodología de desarrollo seleccionada (RUP) utiliza UML como lenguaje para la modelación, se decide utilizar el mismo como lenguaje de modelado del sistema de vigilancia.

Es importante resaltar que UML es un "lenguaje" para especificar y no para describir métodos o procesos. Se utiliza para definir un sistema de software, para detallar los artefactos en el sistema y para documentar y construir. En otras palabras, es el lenguaje en el que está descrito el modelo. Se puede aplicar en una gran variedad de formas para dar soporte a una metodología de desarrollo de software (tal como el Proceso Unificado Racional), pero no especifica en sí mismo qué metodología o proceso usar. (16)

2.5 Lenguaje de programación: C++

Un lenguaje de programación es un conjunto de símbolos y reglas sintácticas y semánticas que definen su estructura y el significado de sus elementos y expresiones. Es utilizado para controlar el comportamiento físico y lógico de una máquina. (17)

C++ es un lenguaje de programación diseñado a mediados de los años 1980 con la intención de extender al exitoso lenguaje de programación C con mecanismos que permitiesen la manipulación de objetos. Posteriormente se añadieron facilidades de programación genérica, que se sumaron a los otros dos paradigmas que ya estaban admitidos (programación estructurada y la programación orientada a objetos). Por esto se suele decir que el C++ es un lenguaje multiparadigma. Una particularidad del C++ es la posibilidad de redefinir los operadores (sobrecarga de operadores), y de poder crear nuevos tipos que se comporten como tipos fundamentales. C++ permite trabajar tanto a alto como a bajo nivel (18).

Se decide utilizar C++ como lenguaje de programación para el desarrollo del sistema debido a:

- Es un lenguaje orientado a objetos.
- Es muy potente en lo que se refiere a creación de sistemas complejos debido a su robustez.
- Actualmente, puede compilar y ejecutar código de lenguaje C, ya que viene con bibliotecas incluidas para realizar esta labor.
- Es muy utilizado en el mundo por lo que existe abundante información sobre su uso.

- Existen muchos algoritmos cuyos pseudocódigos se encuentran ya desarrollados en C++, de manera que pueden tomarse y amoldarse a la solución deseada.

2.6 Bibliotecas Boost

Boost es un conjunto de bibliotecas de código abierto que permite extender las capacidades que oferta C++. Su licencia posibilita que sea empleada en cualquier tipo de proyecto, ya sea comercial o no. Está diseñada e implementada de una manera tal que puede utilizarse en un amplio espectro de aplicaciones y plataformas. Debido a ello se selecciona para ser utilizada en el desarrollo de la aplicación. Además, Boost brinda una funcionalidad que puede resultar muy útil en el desarrollo, y es la posibilidad de realizar el casteo[15] de diferentes tipos de datos primitivos (*string, int, float*).

2.7 Gestor de base de datos: PostgreSQL

Una base de datos es una colección de datos organizados y estructurados según un determinado modelo de información que refleja no sólo los datos en sí mismos, sino también las relaciones que existen entre ellos. Una base de datos se diseña con un propósito específico y debe ser organizada con una lógica coherente. Los datos podrán ser compartidos por distintos usuarios y aplicaciones, pero deben conservar su integridad y seguridad al margen de las interacciones de ambos. La definición y la descripción de los datos han de ser únicas para minimizar la redundancia y maximizar la independencia en su utilización. (19)

Un sistema de gestión de base de datos (SGBD); (en inglés: DataBase Management System, abreviado DBMS) es un tipo de software muy específico, dedicado a servir de interfaz entre la base de datos, el usuario y las aplicaciones que la utilizan. (20)

PostgreSQL es un gestor de base de datos orientado a objetos de software libre, liberado bajo la licencia BSD. Como muchos otros proyectos de código abierto (opensource), el desarrollo de PostgreSQL no es manejado por una sola compañía sino que es dirigido por una comunidad de desarrolladores y organizaciones comerciales las cuales trabajan en su desarrollo. Dicha comunidad es denominada el

[15] Permite forzar un dato, variable o expresión a convertirse o cambiarse a un nuevo tipo de dato.

Grupo de Desarrollo Global de PostgreSQL (PGDG), (en inglés: PostgreSQL Global Development Group). (21)

Se decide utilizar PostgreSQL como gestor de base de datos debido a que posee las siguientes características ventajosas:

- Instalación ilimitada: No hay costo asociado a la licencia del software. Esto trae consigo varias ventajas adicionales como:
    - No existe la posibilidad de ser auditado para verificar cumplimiento de licencia en ningún momento.
    - Flexibilidad para hacer investigación y desarrollo sin necesidad de incurrir en costos adicionales de licenciamiento.
- Mejor soporte que los proveedores comerciales: Existe una importante comunidad de profesionales y entusiastas de PostgreSQL de la cual se puede obtener beneficios y contribuir de la misma manera.
- Estabilidad y confiabilidad: En contraste con muchos sistemas de bases de datos comerciales, es extremadamente común que compañías reporten que PostgreSQL nunca ha presentado caídas en varios años de operación de alta actividad, lo cual demuestra un alto grado de estabilidad.
- Extensible: El código fuente está disponible para todos. Si se necesita extender o personalizar PostgreSQL de alguna manera, se puede hacer sin costos adicionales.
- Diseñado para ambientes de altos volúmenes de datos y usuarios: PostgreSQL utiliza una estrategia de almacenamiento de filas llamada MVCC[16] (*Control de Concurrencia Multi-versión*) para conseguir una mejor respuesta en ambientes de grandes volúmenes de datos y usuarios.
- Herramientas gráficas de administración de bases de datos: Existen varias herramientas gráficas de alta calidad para administrar las bases de datos (Ej. pgAdmin, pgAccess).

[16] Siglas en inglés: Multi-version Concurrency Control. Técnica que mejora las prestaciones de una BD en un entorno multiusuario.

2.8 Biblioteca Pqxx

Pqxx es una biblioteca que posibilita la comunicación de aplicaciones que utilizan C++ como lenguaje de programación con el motor de base de datos PostgreSQL.

Debido a la selección de C++ como lenguaje de programación y PostgreSQL como sistema gestor de base de datos para el desarrollo del sistema, se selecciona Pqxx como biblioteca de apoyo para facilitar la realización de consultas a la base de datos.

2.9 Herramienta CASE[17]: Visual Paradigm

Una herramienta CASE es un conjunto de métodos, utilidades y técnicas que facilitan la automatización del ciclo de vida del desarrollo de sistemas de información, completamente o en alguna de sus fases (22). Las herramientas CASE abarcan todos los pasos del desarrollo del software, y también aquellas actividades generales que se aplican a lo largo del proceso.

Visual Paradigm es una herramienta UML profesional que soporta el ciclo de vida completo del desarrollo de software: análisis y diseño orientados a objetos, construcción, pruebas y despliegue. El software de modelado UML ayuda a una más rápida construcción de aplicaciones de calidad, mejores y a un menor coste. Permite dibujar todos los tipos de diagramas de clases, código inverso, generar código desde diagramas y generar documentación. (23)

Se decide utilizar Visual Paradigm como herramienta CASE debido a:

- Ofrece un diseño centrado en casos de uso y enfocado al negocio.
- Posee capacidades de ingeniería directa e inversa.
- Puede integrarse a los principales entornos de desarrollo.
- Posee disponibilidad de múltiples versiones para cada necesidad.

[17] Siglas en inglés: Computer Aided Software Engineering (Ingeniería de Software Asistida por Ordenador).

2.10 Entorno de desarrollo: Eclipse

Un entorno de desarrollo integrado (en inglés Integrated Development Environment o IDE) es un programa compuesto por una serie de herramientas que utilizan los programadores para desarrollar código. Esta herramienta puede estar pensada para su utilización con un único lenguaje de programación o bien puede dar cabida a varios de éstos.

Los entornos de desarrollo proporcionan un marco de trabajo para la mayoría de los lenguajes de programación existentes en el mercado (por ejemplo C, C++, C#, Java, Python y Visual Basic entre otros). Además, es posible que un mismo entorno de desarrollo tenga la posibilidad de utilizar varios lenguajes de programación, como es el caso de Eclipse. (24)

Se decide utilizar Eclipse debido a que es un IDE gratuito, libre, potente, tiene gran soporte en la comunidad de software libre y permite instalar complementos o plugins[18] para lograr un entorno de desarrollo ampliado con posibilidades variadas de uso.

2.11 Framework para el desarrollo de interfaces gráficas de usuario: QT

Qt es una herramienta para desarrollar interfaces gráficas de usuario y también para el desarrollo de programas sin interfaz gráfica como herramientas de la consola y servidores.

Qt utiliza el lenguaje de programación C++ de forma nativa, pero permite usar también C, Python y Perl. Funciona en todas las principales plataformas, y tiene un amplio apoyo. El API[19] de Qt cuenta con métodos para acceder a bases de datos mediante SQL[20], así como uso de XML[21], gestión de hilos, soporte de red, una API multiplataforma unificada para la manipulación de archivos y una multitud de otros para el manejo de ficheros, además de estructuras de datos tradicionales. (25)

Se selecciona Qt para el desarrollo de la interfaz gráfica de la aplicación debido a que es de código abierto y posee características que la convierten en una herramienta de desarrollo muy potente, tales como:

[18] Aplicación que se relaciona con otra para aportarle una función nueva y generalmente muy específica.
[19] Siglas en inglés: Application Programming Interface (Interfaz de programación de aplicaciones).
[20] Lenguaje de consulta estructurado para manipular información en una base de datos.
[21] Siglas en inglés: eXtensible Markup Language (Lenguaje de marcado extensible). Diseñado especialmente para la web.

- Un completo juego de clases y métodos listos para usar incluso para cuestiones de programación no gráfica.
- Garantiza una buena solución en materia de interacción con el usuario mediante el uso de métodos virtuales y el mecanismo señal/contenedor.
- Un grupo de elementos gráficos predefinidos, llamados «widgets», que pueden ser fácilmente usados para crear los elementos visibles.

2.12 Biblioteca OpenCV

OpenCV es una biblioteca de visión artificial originalmente desarrollada por la compañía Intel y programada utilizando como lenguajes de programación C y C++. Uno de los objetivos para los que fue creada es proporcionar una infraestructura de visión por computadora que sea fácil de usar y que permita a sus usuarios desarrollar aplicaciones de manera rápida. Contiene más de 300 funciones que abarcan una gran gama de áreas en el proceso de visión.

Se decide utilizar OpenCV por ser una biblioteca libre que cuenta con una versión para Linux y que puede ser usada, sin restricciones, para propósitos comerciales y de investigación con las condiciones en ella expresadas. El hecho de ser diseñada para lograr eficiencia computacional con un fuerte enfoque en el desarrollo de aplicaciones de tiempo real resulta de vital importancia para considerarse su aplicación en la captura y visualización de cámaras digitales, así como en el tratamiento de las imágenes obtenidas del flujo de video.

2.13 Generador de documentación: Doxygen

El paquete Doxygen permite generar la documentación correspondiente a una aplicación. Puede generar archivos en HTML[22] y/o un manual de referencia a partir de un grupo de ficheros fuente documentados. Se decide utilizar Doxygen ya que contiene un sistema de documentación para C++ que es el lenguaje de programación que se utiliza en el sistema. Además, brinda la facilidad de que la documentación se extrae directamente de las fuentes, lo que hace mucho más fácil mantener la consistencia con el código fuente.

[22] Siglas en inglés: HyperText Markup Language (Lenguaje de Marcado de Hipertexto). Empleado para elaborar páginas web.

2.14 Conclusiones del capítulo

Enfocado en el estudio de las diferentes herramientas y tecnologías existentes y su desarrollo actual, se decide que es más recomendable utilizar las listadas en este capítulo por su peculiaridad de pertenecer a la categoría de software libre. Además, con el uso de las mismas se está siendo consecuente con las políticas trazadas por el Centro de Informática Industrial, y más específicamente el Proyecto Seguridad.

Capítulo 3: Descripción de la solución propuesta

3.1 Introducción al capítulo

En el presente capítulo se lleva a cabo el proceso de modelación del sistema y se ofrece una descripción de las características del mismo. Para ello se especifican los requisitos funcionales y no funcionales que debe tener la aplicación, se identifican los actores, se describen los principales casos de uso y se construyen varios diagramas que posibiliten un mejor entendimiento.

3.2 Propuesta de sistema

En la mitología griega el Can Cerbero era conocido como el perro de tres cabezas con una serpiente en la cola que resguardaba la puerta del inframundo y aseguraba que los muertos no salieran y que los vivos no pudiesen entrar. Debido a que realizaba su trabajo con mucho celo, con frecuencia se utiliza su nombre para referir la labor de todo guarda que cumple su función con mucha seriedad. Es por ello que se extrajo de la mitología para representar al proyecto Seguridad, y como el sistema a desarrollar forma parte de dicho proyecto, y por demás se ajusta en objetivos y contenido a la razón de ser de este animal mitológico, se decide bautizarlo con el nombre de Sistema de Vigilancia por Cámaras Cancerbero (SVCC).

La solución que se plantea es desarrollar un sistema que proporcione seguridad mediante el manejo de cámaras y la video-vigilancia. Dicha solución es un producto que, por sus características de desarrollarse con tecnologías libres, brinda una amplia gama de posibilidades para ser implantado en aquellos lugares que cuenten con el equipamiento de hardware necesario (computadora y cámaras) y no poseen el software que le permita trabajar con el mismo. La aplicación no sustituye completamente al personal de seguridad de una entidad pero si tiene la intención de disminuir la cantidad de personas necesitadas, las cuales se pueden apoyar en el sistema para realizar su trabajo. En cada punto de interés se situaría una cámara y el sistema obtendría los flujos de video capturados por esas cámaras para su visualización y análisis.

Se les permite a los operadores visualizar los flujos de video obtenidos de cada una de las cámaras. En caso de ser un administrador del sistema, éste también puede adicionar, eliminar y modificar las cámaras en el mismo. El sistema brinda la posibilidad de gestionar todo el flujo de información de manera que esté

disponible y permita el análisis de cambios ocurridos en los datos. También posibilita la gestión de los usuarios registrados en la base de datos.

3.3 Especificación de los requisitos de software

Los requisitos se clasifican en funcionales y no funcionales. Los funcionales son capacidades o condiciones que el sistema debe cumplir, mientras que los no funcionales son propiedades o cualidades que hacen al producto atractivo, usable, rápido y confiable.

3.3.1 Requisitos funcionales del sistema

RF1 Gestionar monitor.

RF1.1 Adicionar monitor (*El sistema debe ser capaz de adicionar un nuevo monitor con sus datos).*

RF1.2 Editar monitor (*El sistema debe ser capaz de modificar los datos del monitor adicionado).*

RF1.3 Eliminar monitor (*El sistema debe ser capaz de eliminar un monitor adicionado).*

RF2 Almacenar imagen (*El sistema debe ser capaz de almacenar una imagen tomada de un monitor).*

RF3 Almacenar video (*El sistema debe ser capaz de almacenar el flujo de video de un monitor).*

RF4 Visualizar monitor (*El sistema debe ser capaz de visualizar un monitor añadido al sistema).*

RF5 Gestionar usuario.

RF5.1 Adicionar usuario (*El sistema debe ser capaz de adicionar un nuevo usuario).*

RF5.2 Editar usuario (*El sistema debe ser capaz de modificar los datos de un usuario adicionado).*

RF5.3 Eliminar usuario (*El sistema debe ser capaz de eliminar un usuario añadido).*

RF6 Autenticar (*El sistema debe permitir la autenticación de los usuarios existentes).*

RF7 Mostrar información de usuarios (*El sistema debe ser capaz de mostrar los datos de los usuarios existentes*).

RF8 Mostrar información de monitores (*El sistema debe ser capaz de mostrar los datos de los monitores existentes*).

3.3.2 Requisitos no funcionales del sistema

Rendimiento

RNF1 El sistema debe permitir la captura de todas las cámaras conectadas pero solo la visualización de una a la vez.

Hardware

RNF2 Para esta versión el sistema permite solamente el uso de cámaras digitales que no cuenten con tecnología IP y que sean compatibles con V4L.

RNF3 Los requerimientos de hardware son:

- 1 Gb de memoria RAM[23].
- Procesador Dual Core.
- 160 Gb de capacidad de disco duro.

Apariencia

RNF4 La interfaz debe ser de fácil comprensión en su funcionamiento permitiendo la utilización del sistema sin mucho adiestramiento.

3.4 Modelación del sistema

Definición de los actores

Los actores representan personas o sistemas externos que interactúan con el sistema. SVCC está diseñado de manera tal que pueda ser utilizado tanto por un operador con mínimos privilegios como por un administrador con todos los permisos de administración definidos.

[23] Siglas en inglés: Random Access Memory (memoria de acceso aleatorio).

Actor	Descripción
Administrador	Rol responsable de velar por el funcionamiento apropiado del sistema, configurarlo y realizar todas las demás funcionalidades que permite el mismo.
Operador	Rol que se beneficia de diversas funcionalidades del sistema pero no tiene acceso a las configuraciones del mismo.

Tabla 3 Actores del sistema.

Relación entre los actores del sistema

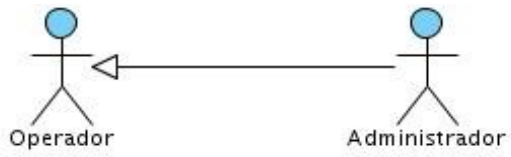

Figura 6 Relación entre actores del sistema.

Como se muestra en la *Figura 6*, la relación existente entre los actores del sistema es de herencia, o sea, el administrador se comporta como el rol hijo que hereda del rol padre que en este caso sería el operador. Esto implica que el administrador puede realizar todas las acciones que el operador hace además de las que desempeña por sí mismo en su rol administrativo, mientras que el operador solo puede realizar las que se corresponden con su rol.

3.4.1 Definición de los casos de uso

A continuación se muestra una tabla donde quedan definidos los casos de uso del sistema de vigilancia.

CU-1	**Gestionar monitor**
Actor	Administrador.
Descripción	Se adiciona, edita o elimina un monitor.
Referencia	RF1.1, RF1.2, RF1.3.
CU-2	**Almacenar imagen**
Actor	Operador.
Descripción	Se almacena una imagen tomada del flujo de video de un monitor.

Referencia	RF2.
CU-3	**Almacenar video**
Actor	Operador.
Descripción	Se almacena el flujo de video de un monitor.
Referencia	RF3.
CU-4	**Visualizar monitor**
Actor	Operador.
Descripción	Se visualiza el flujo de video de un monitor añadido al sistema.
Referencia	RF4.
CU-5	**Gestionar usuario**
Actor	Administrador.
Descripción	Se adiciona, edita o elimina un usuario.
Referencia	RF5.1, RF5.2, RF5.3.
CU-6	**Autenticar**
Actor	Operador.
Descripción	Autenticación en el sistema.
Referencia	RF6.
CU-7	**Mostrar datos de usuario**
Actor	Administrador.
Descripción	Se informa acerca de los usuarios existentes en el sistema.
Referencia	RF7.
CU-8	**Mostrar datos de monitor**
Actor	Operador.
Descripción	Se informa acerca de los monitores existentes en el sistema.
Referencia	RF8.

Tabla 4 Definición de los casos de uso.

3.4.2 Diagrama de casos de uso del sistema

Partiendo de la base de que cada requisito funcional se convierte en un caso de uso del sistema es que se definen los actores y casos de uso a partir de la captura de requisitos.

A continuación se modela la relación existente entre los actores y casos de uso definidos. En este diagrama se pueden observar las responsabilidades que tienen los actores sobre los casos de uso, así como la relación entre ellos. (*Figura 7*)

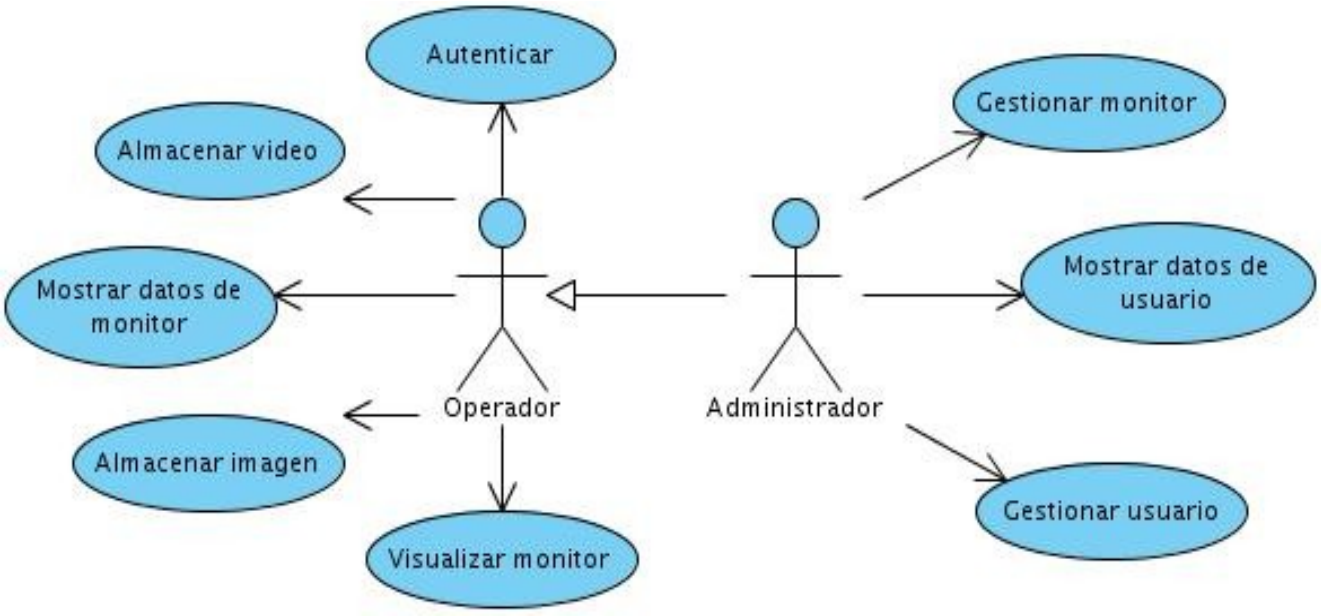

Figura 7 Diagrama de casos de uso del sistema.

3.4.3 Descripción de los casos de uso

Caso de uso	
CU-1	Gestionar monitor.
Propósito	Permitir el trabajo con los monitores que se manejarán en el sistema.
Actores: Administrador.	
Resumen: El caso de uso se inicia cuando el Administrador selecciona la opción de adicionar, eliminar o editar monitor.	
Referencias	RF1.1, RF1.2, RF1.3.
Flujo básico	
Acción del actor	**Respuesta del sistema**
Escenario Adicionar	
1. El Administrador selecciona la opción de	2. El sistema le muestra una forma para

adicionar un monitor.	introducir los datos del monitor (nombre, dirección).
3. El Administrador introduce los datos solicitados y acepta la entrada.	4. El sistema verifica que los datos estén correctos. (FA 1[24]) 5. El sistema almacena los datos del nuevo monitor.
Escenario Eliminar	
1. El Administrador selecciona la opción de eliminar un monitor.	2. El sistema muestra los monitores que se han añadido al sistema.
3. El Administrador selecciona el monitor que desee eliminar.	4. El sistema elimina el monitor.
Escenario Editar	
1. El Administrador selecciona la opción de editar un monitor.	2. El sistema muestra los monitores que se han añadido al sistema.
3. El Administrador selecciona el monitor que desea editar.	4. El sistema muestra una forma para introducir los nuevos datos (nombre, dirección).
5. El Administrador modifica los datos que desee y acepta.	6. El sistema verifica que los datos estén correctos (FA 2). 7. El sistema almacena los datos modificados según el identificador del monitor seleccionado.
Flujo alterno	
Acción del actor	**Respuesta del sistema**
Escenario Adicionar	
	1. (FA 1) Si todos los campos solicitados no han sido completados o no están en el formato requerido el sistema muestra una advertencia.
2. El Administrador rectifica lo que le ha sido señalado.	3. El sistema verifica los datos. 4. El sistema almacena los datos del nuevo monitor.
Escenario Editar	
	1. (FA 2) Si todos los campos solicitados no han sido completados o no están en el

[24] Se refiere al flujo alterno.

	formato requerido el sistema muestra una advertencia.
2. El Administrador rectifica lo que le ha sido señalado.	3. El sistema verifica los datos. 4. El sistema almacena los datos modificados según el identificador del monitor seleccionado.

Tabla 5 Descripción CU Gestionar monitor.

Caso de uso	
CU-2	Almacenar imagen.
Propósito	Permitir el almacenamiento de imágenes a partir del flujo de video del monitor.
Actores: Operador.	
Resumen: El caso de uso se inicia cuando el Operador elige la opción de almacenar imagen.	
Referencias	RF2.
Precondiciones	Debe existir algún monitor añadido visualizándose.
Flujo básico	
Acción del actor	**Respuesta del sistema**
1. El Operador elige la opción de configurar dirección de almacenamiento de imagen.	2. El sistema solicita al Operador que seleccione una dirección para el almacenamiento.
3. El Operador selecciona la dirección donde serán almacenadas las imágenes y acepta.	4. El sistema habilita las opciones de almacenar imagen y configurar intervalo de almacenamiento (el tiempo entre la captura de una imagen y otra es 5 segundos por defecto). (FA1)
5. El Operador selecciona la opción de almacenar imagen.	6. El sistema comienza el almacenamiento de imágenes según la configuración hecha. 7. El sistema habilita la opción de detener almacenamiento de imagen y deshabilita la opción de almacenar imagen ya que se está llevando a cabo dicho procedimiento.
8. El Operador selecciona la opción de detener almacenamiento.	9. El sistema detiene el almacenamiento de imagen, deshabilita la opción de detener

	almacenamiento, y habilita la opción de almacenar imagen.
Flujos alternos	
Acción del actor	**Respuesta del sistema**
1. (FA 1) El Operador selecciona la opción de configurar intervalo de almacenamiento.	2. El sistema le muestra una forma para introducir el tiempo entre la captura de una imagen y otra.
3. El operador introduce el dato solicitado y acepta.	4. El sistema verifica el dato introducido. (FA2) y continúa con el paso 5 del flujo normal.
Acción del actor	**Respuesta del sistema**
	1. (FA2) Si el dato introducido no es válido el sistema muestra una advertencia.
2. El Operador rectifica lo que le ha sido señalado y acepta.	5. El sistema verifica los datos y continúa con el paso 5 del flujo normal.

Tabla 6 Descripción CU Almacenar imagen.

Caso de uso	
CU-3	Almacenar video.
Propósito	Permitir el almacenamiento del flujo de video de un monitor.
Actores: Operador.	
Resumen: El caso de uso se inicia cuando el Operador elige la opción de almacenar video.	
Referencias	RF3.
Precondiciones	Debe existir algún monitor añadido visualizándose.
Flujo básico	
Acción del actor	**Respuesta del sistema**
1. El Operador elige la opción de configurar dirección de almacenamiento de video.	2. El sistema solicita al Operador que seleccione una dirección para el almacenamiento.
3. El Operador selecciona la dirección donde será almacenado el video.	4. El sistema habilita la opción de almacenar video.
5. El Operador selecciona la opción de almacenar video.	6. El sistema comienza el almacenamiento del video según la configuración hecha. 7. El sistema habilita la opción de detener

	almacenamiento de video y deshabilita la opción de almacenar video ya que se está llevando a cabo dicho procedimiento.
8. El Operador selecciona la opción de detener almacenamiento de video.	9. El sistema detiene el almacenamiento del video, deshabilita la opción de detener almacenamiento, y habilita la opción de almacenar video.

Tabla 7 Descripción CU Almacenar video.

Caso de uso	
CU-4	Visualizar monitor.
Propósito	Permitir la visualización del flujo de video de la cámara añadida al sistema.
Actores: Operador.	
Resumen: El caso de uso se inicia cuando el Operador elige la opción de visualizar monitor.	
Referencias	RF4.
Precondiciones	Debe existir al menos un monitor en la base de datos.
Flujo básico	
Acción del actor	**Respuesta del sistema**
1. El Operador selecciona la opción de visualizar monitor.	2. El sistema lista todos los monitores existentes y solicita la selección de uno de ellos para su visualización.
3. El Operador selecciona un monitor.	4. El sistema verifica si el monitor seleccionado puede ser visualizado. (FA1) 5. El sistema visualiza el monitor seleccionado.
Flujos alternos	
Acción del actor	**Respuesta del sistema**
	1. (FA1) Si el monitor seleccionado no puede ser visualizado por error en la captura del video de la dirección especificada o porque la dirección especificada es incorrecta (atributo dirección que se especifica al añadir un

	monitor) el sistema muestra una advertencia.
2. El Operador rectifica lo señalado y continúa con el paso 1 del flujo normal.	

Tabla 8 Descripción CU Visualizar monitor.

Caso de uso	
CU-5	Gestionar usuario.
Propósito	Garantizar la seguridad en el sistema mediante el uso de usuarios con privilegios distintos, protegidos por contraseña.
Actores: Administrador.	
Resumen: El caso de uso se inicia cuando el Administrador selecciona la opción de adicionar, eliminar o editar usuario.	
Referencias	RF5.1, RF5.2, RF5.3.
Flujo básico	
Acción del actor	**Respuesta del sistema**
Escenario Adicionar	
1. El Administrador selecciona la opción de adicionar un usuario.	2. El sistema le muestra una forma para introducir los datos del usuario (nombre de usuario, contraseña y si es administrador o no).
3. El Administrador introduce los datos solicitados y acepta la entrada.	4. El sistema verifica que la contraseña tiene más de 5 caracteres y que los datos introducidos son correctos. (FA 1) 5. El sistema almacena el nuevo usuario.
Escenario Eliminar	
1. El Administrador selecciona la opción de eliminar un usuario.	2. El sistema lista los usuarios que existen.
3. El Administrador selecciona un usuario y acepta.	4. El sistema elimina el usuario seleccionado.
Escenario Editar	
1. El Administrador selecciona la opción de editar un usuario.	2. El sistema muestra los usuarios existentes.
3. El Administrador selecciona un usuario.	4. El sistema muestra una forma para introducir los nuevos datos del usuario

	(nombre de usuario, contraseña y rol).
5. El Administrador modifica los datos y acepta.	6. El sistema verifica que los datos son válidos. (FA 2) 7. El sistema almacena los datos del usuario modificado.
Flujo alterno	
Acción del actor	**Respuesta del sistema**
Escenario Adicionar	
	1. (FA 1) Si la contraseña no se escribió correctamente con al menos 6 caracteres o alguno de los datos fueron introducidos incorrectamente, el sistema muestra una advertencia.
2. El Administrador rectifica los datos y acepta.	3. El sistema verifica los datos. 4. El sistema almacena el nuevo usuario.
Escenario Editar	
	1. (FA 2) Si los datos no son válidos el sistema muestra una advertencia.
2. El Administrador rectifica los datos y acepta.	3. El sistema verifica los datos. 4. El sistema almacena los datos del usuario modificado.

Tabla 9 Descripción CU Gestionar usuario.

Caso de uso	
CU-6	Autenticar.
Propósito	Garantizar que la persona que esté operando el sistema tenga los derechos requeridos.
Actores: Operador.	
Resumen: El caso de uso se inicia cuando se ejecuta la aplicación.	
Referencias	RF6.
Precondiciones	Debe existir al menos un usuario en la base de datos.
Flujo básico	
Acción del actor	**Respuesta del sistema**
	1. El sistema muestra una forma para entrar

	los datos (usuario y contraseña).
2. El Operador introduce los datos y acepta.	3. El sistema verifica los datos introducidos por el Operador. (FA 1) 4. El sistema permite la entrada al sistema según los permisos que tenga el tipo de Operador que entró (Operador, Administrador).
Flujo alterno	
Acción del actor	**Respuesta del sistema**
	1. (FA 1) Si el usuario y la contraseña no coinciden con los datos de un usuario válido, el sistema muestra una advertencia.
2. El Operador introduce los datos nuevamente.	3. El sistema verifica datos. 4. El sistema permite la entrada al sistema según los permisos que tenga el tipo de Operador que entró (Operador, Administrador).

Tabla 10 Descripción CU Autenticar.

Caso de uso	
CU-7	Mostrar datos de monitor.
Propósito	Conocer los monitores que existen en el sistema para verificar si los datos son los deseados.
Actores: Operador.	
Resumen: El caso de uso se inicia cuando el Operador selecciona la opción de información de monitores.	
Referencias	RF7.
Flujo básico	
Acción del actor	**Respuesta del sistema**
1. El Operador selecciona la opción de información de monitores.	2. El sistema le muestra una forma con los datos (id, nombre y dirección) de todos los monitores existentes.

Tabla 11 Descripción CU Mostrar datos de usuario.

Caso de uso	
CU-8	Mostrar datos de usuario.
Propósito	Conocer los usuarios que existen en el sistema para verificar si los datos son los deseados.
Actores: Administrador.	
Resumen: El caso de uso se inicia cuando el Administrador selecciona la opción de información de usuarios.	
Referencias	RF8.
Flujo básico	
Acción del actor	**Respuesta del sistema**
1. El Administrador selecciona la opción de información de usuarios.	2. El sistema le muestra una forma con los datos (nombre de usuario y rol) de todos los usuarios existentes.

Tabla 12 Descripción CU Mostrar datos de usuario.

3.4.4 Diagrama de clases persistentes

La persistencia es la capacidad de un objeto de mantener su valor en el espacio y en el tiempo. Las clases persistentes se definen para conocer la información real representada en las tablas de la base de datos. (26)

Con el objetivo de mostrar los atributos y tipos de dato que presentan cada una de las clases persistentes para una mejor comprensión, se realiza una descripción de las mismas. (*Tabla 13*)

3.4.4.1 Descripción de las tablas

Nombre:	**Users**	
Descripción:	La tabla almacena los datos relacionados con los usuarios del sistema.	
Atributo	**Tipo**	**Descripción**
username	string	Se refiere al nombre de usuario para manipular el sistema. Es lo que identifica a los usuarios, o sea, dos usuarios no pueden tener el mismo nombre de usuario.
password	string	Se refiere a la contraseña que posee el usuario para entrar al sistema, la cual debe tener al menos 6 caracteres. La contraseña se almacena cifrada en la base de datos para lograr un mayor nivel de seguridad en el sistema, o sea, ni siquiera el administrador puede conocer la contraseña de ningún

		usuario.
isadmin	boolean	Se refiere al rol del usuario en el sistema. Si es *true* (verdadero) significa que es administrador, si es *false* (falso) significa que es un operador.
Nombre:	**Monitors**	
Descripción:	La tabla almacena los datos relacionados con los monitores del sistema.	
Atributo	**Tipo**	**Descripción**
id	int	Se refiere a un número entero que se comporta como el identificador de un monitor. Dos monitores no pueden tener el mismo id.
name	string	Se refiere al nombre con el cual se conoce al monitor para hacer más intuitivo el trabajo con el mismo.
path	string	Se refiere a la dirección de donde se obtiene el flujo de video proveniente de la cámara conectada a la computadora. (/dev/video/path)

Tabla 13 Descripción de las tablas.

El diagrama de clases persistentes se realizó para modelar la estructura lógica de la base de datos. (*Figura 8*)

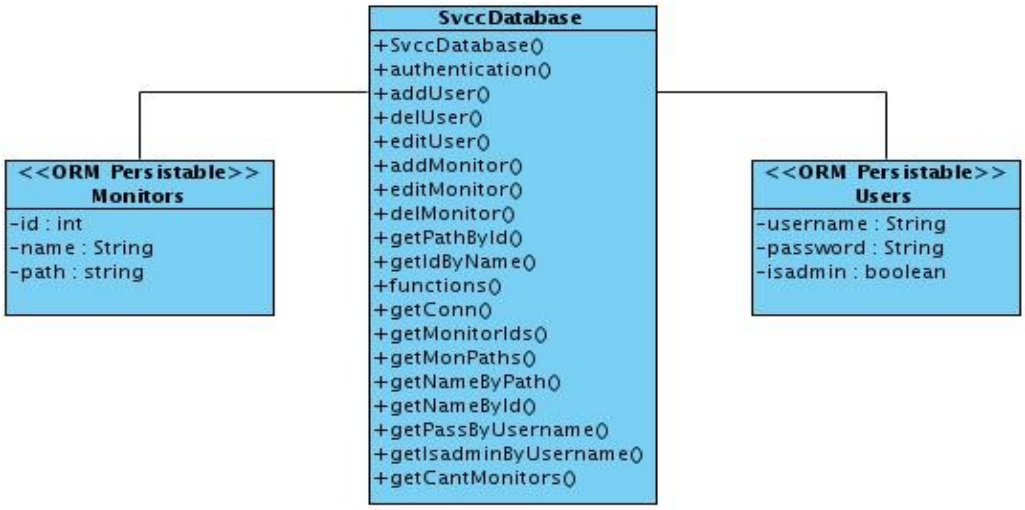

Figura 8 Diagrama de clases persistentes.

En este caso la clase SvccDatabase se encarga de realizar las consultas a la base de datos, y de esta manera obtener, modificar o eliminar información de las tablas *Users* y *Monitors*.

3.4.5 Diagrama de clases

Los diagramas de clases muestran las diferentes entidades que componen un sistema y cómo se

relacionan unas con otras. A continuación se muestra el diagrama que recoge la relación entre las clases elaboradas para la solución propuesta:

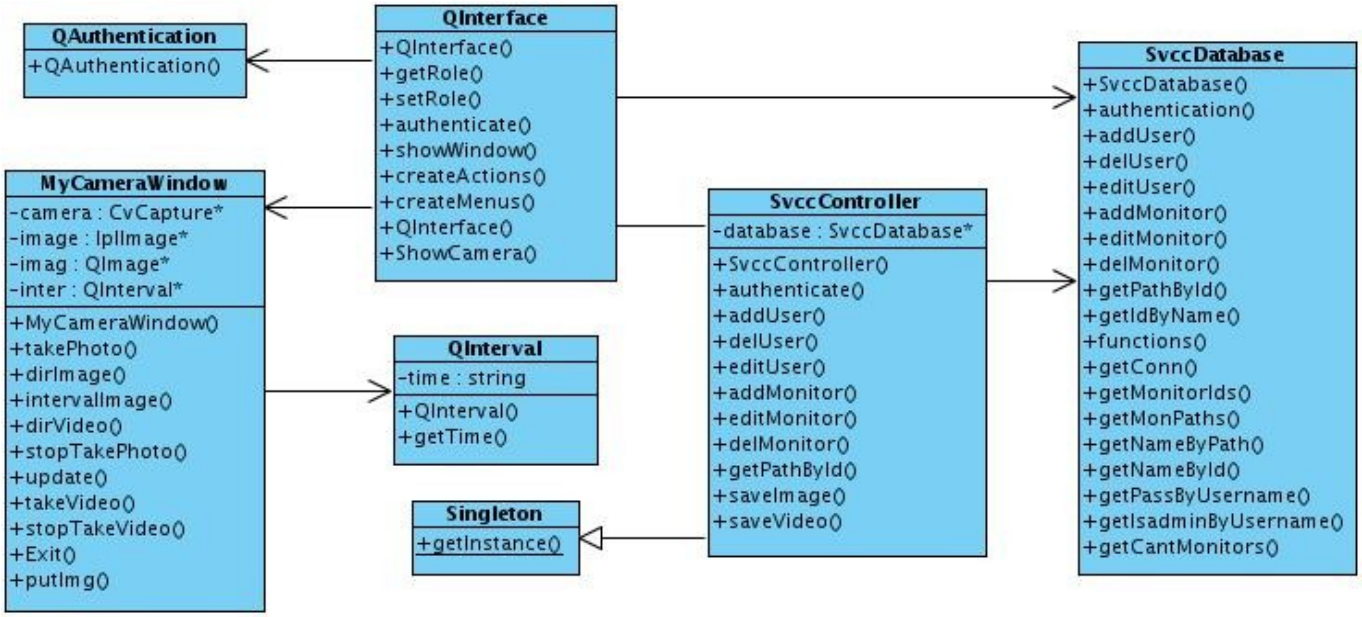

Figura 9 Diagrama de clases.

3.4.5.1 Descripción de las principales clases

En el *anexo 1* de este documento se puede ver una descripción de las principales clases del sistema, donde se detallan cada una de las funcionalidades y atributos que las componen. (Ver Anexo 1)

3.4.6 Diagramas de interacción

El diagrama de secuencia es un tipo de diagrama de interacción ya que muestra el intercambio de mensajes en un determinado espacio de tiempo resaltando el orden y momento en que se envían los mensajes a los objetos. A continuación se muestran los diagramas de secuencia de los principales casos de uso de la aplicación:

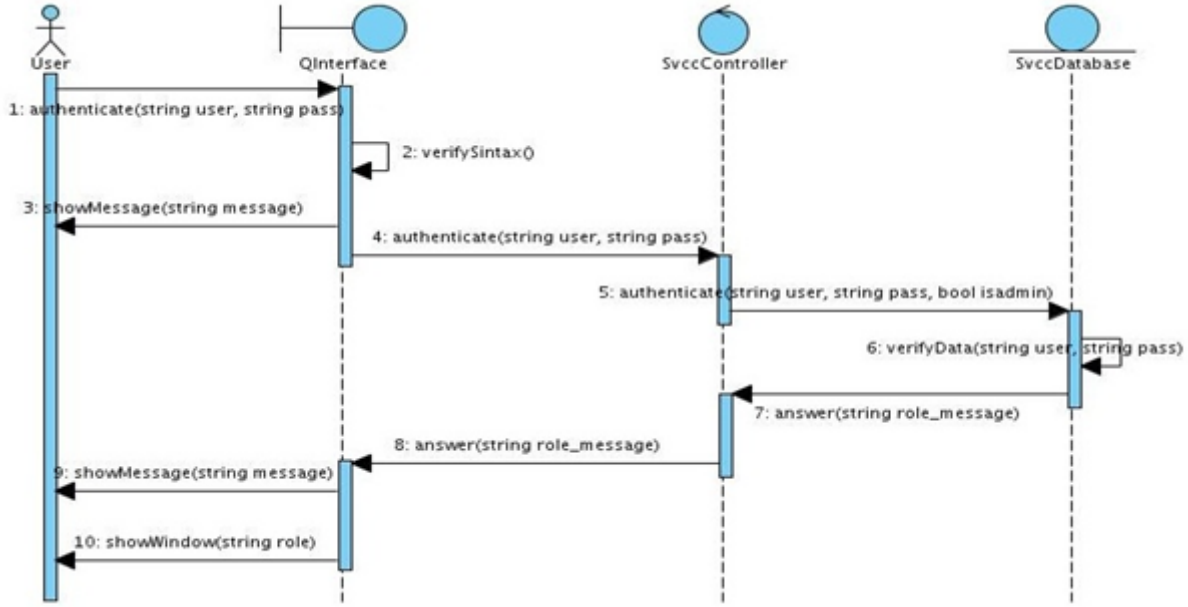

Figura 10 Diagrama de secuencia CU Autenticar.

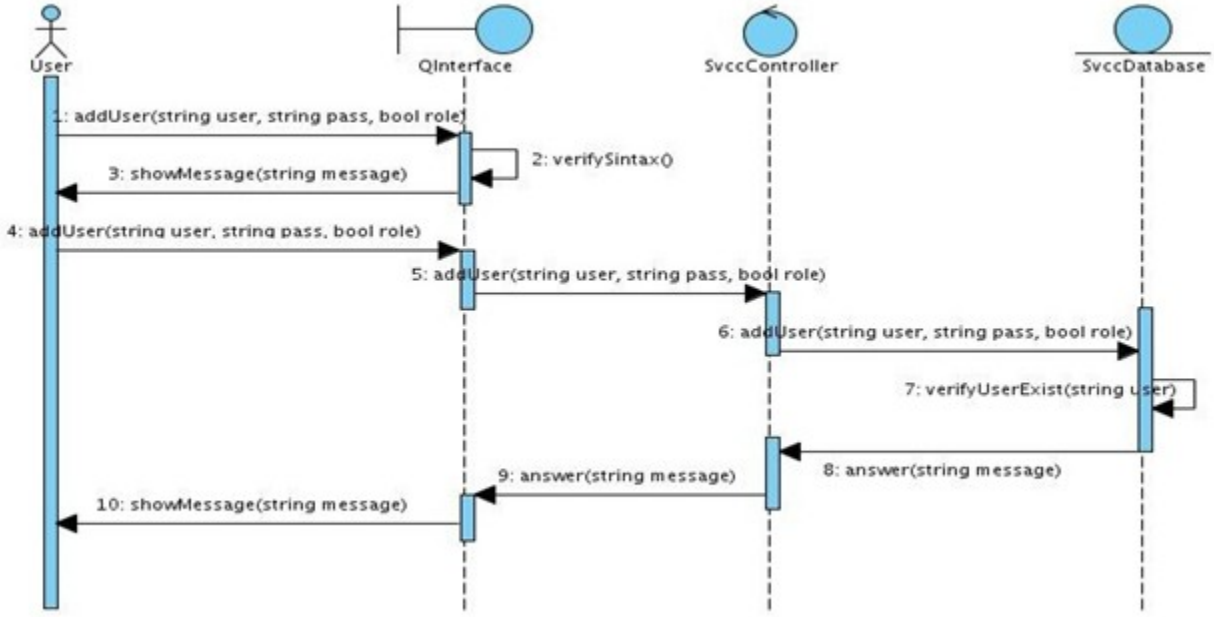

Figura 11 Diagrama de secuencia CU Adicionar usuario.

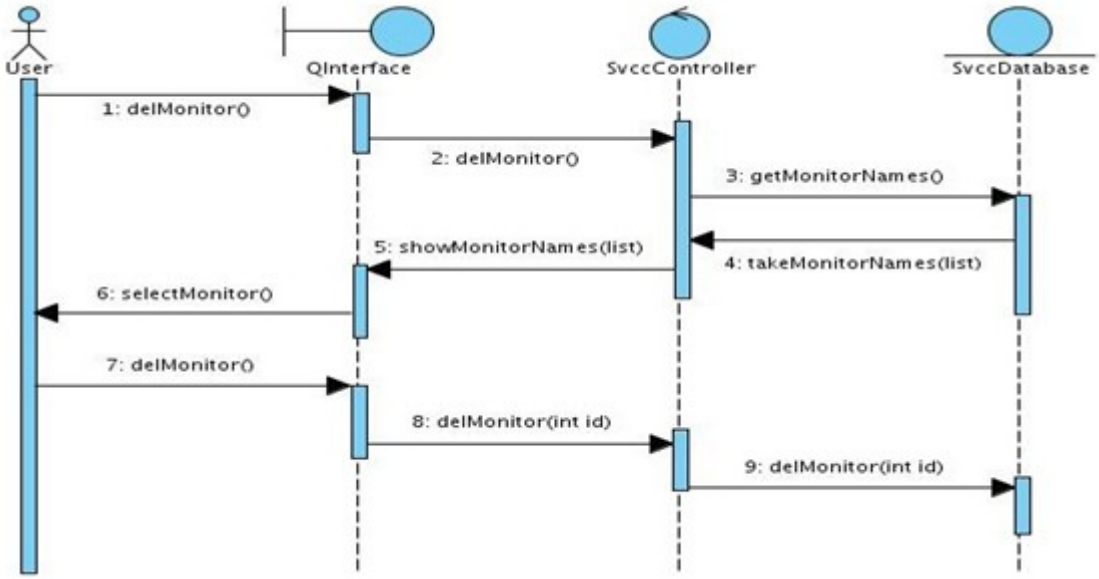

Figura 12 Diagrama de secuencia CU Eliminar monitor.

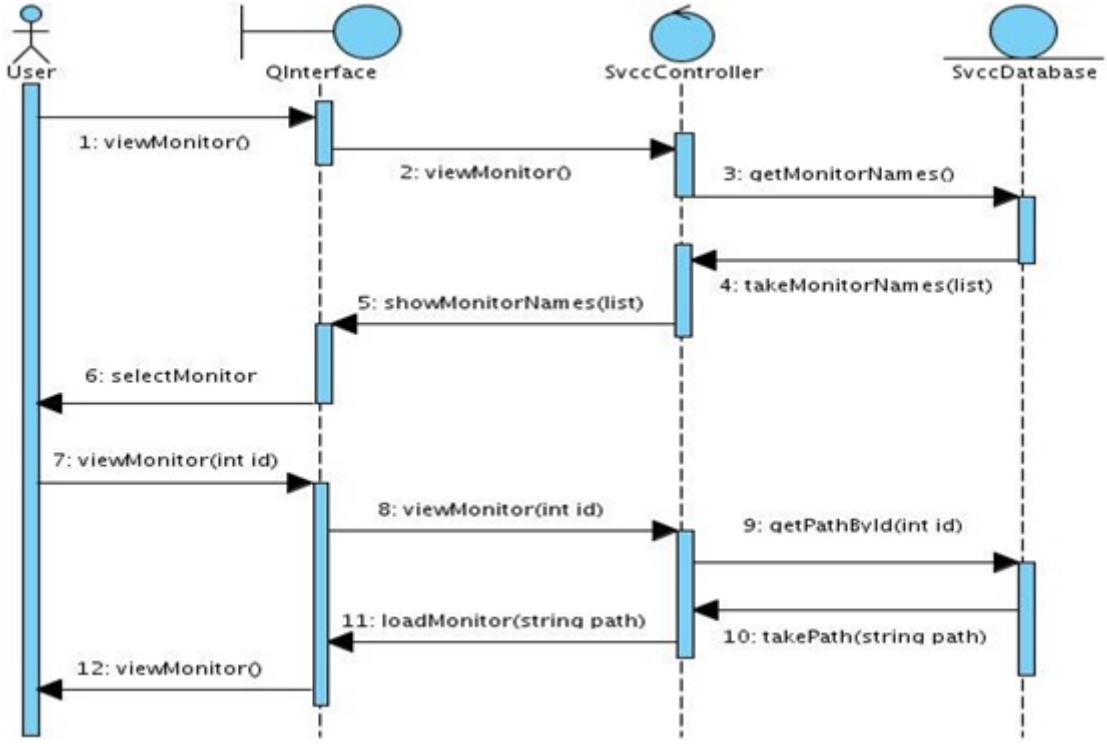

Figura 13 Diagrama de secuencia CU Visualizar monitor.

3.4.7 Patrones

Un patrón es una solución a un problema en un contexto determinado, aplicando el conocimiento específico acumulado por la experiencia en un dominio. Existen patrones arquitectónicos y de diseño. A continuación se explica cómo fueron utilizados en el sistema desarrollado.

3.4.7.1 Patrón arquitectónico

El sistema utiliza el patrón Modelo-Vista-Controlador (MVC) el cual afronta el hecho de las modificaciones en la interfaz de usuario que surgen como consecuencia de los cambios frecuentes en los soportes de hardware y de la necesidad de añadir nuevas funcionalidades. Separa el modelado del dominio, la presentación y las acciones basadas en datos ingresados por el usuario en tres clases diferentes: Modelo, Vista y Controlador. (*Figura 14*)

El Modelo es inmutable a los cambios en la interfaz de usuario y se encarga de toda la gestión interna del negocio. Administra el comportamiento y los datos del dominio de aplicación, responde a requerimientos de información sobre su estado (usualmente formulados desde la vista) y responde a instrucciones de cambiar el estado (habitualmente desde el controlador). La Vista se encarga de presentar los datos y de interactuar con el usuario. Maneja la visualización de la información. El Controlador tiene la función de controlar el flujo entre la vista y el modelo (los datos).

Tanto la vista como el controlador dependen del modelo, el cual no depende de las otras clases. Esta separación permite construir y probar el modelo, independientemente de la representación visual.

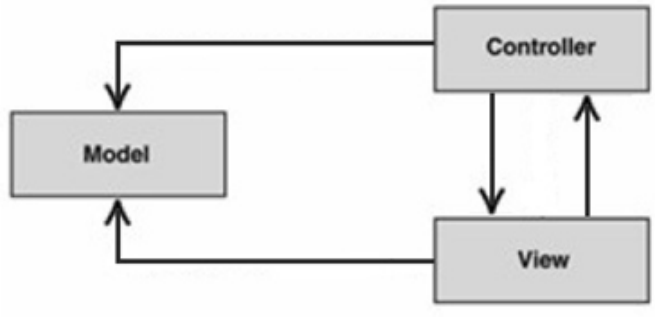

Figura 14 Patrón Modelo-Vista-Controlador.

En el sistema se comporta de la siguiente manera:

- SvccDatabase representa la clase **Model** del MVC.
- SvccController representa la clase **Controller** del MVC.
- QInterface representa la clase **View** del MVC.

Entre las ventajas del estilo MVC se encuentran las siguientes:

Soporte de múltiples vistas: Dado que la vista se halla separada del modelo y no hay dependencia directa del modelo con respecto a la vista, la interfaz de usuario puede mostrar múltiples vistas de los mismos datos simultáneamente.

Adaptación al cambio: Los requisitos de interfaz de usuario tienden a cambiar con rapidez. Además, los usuarios pueden requerir soporte para nuevos dispositivos como teléfonos celulares o PDAs[25]. Dado que el modelo no depende de las vistas, agregar nuevas opciones de presentación no afecta al modelo.

3.4.7.2 Patrones de diseño

Los patrones de diseño se clasifican en dependencia de su funcionalidad.

- Patrones de creación: Muestran la guía de cómo crear objetos cuando sus creaciones requieren tomar decisiones. Estas decisiones normalmente serán resueltas dinámicamente decidiendo qué clases instanciar o sobre qué objetos delegará responsabilidades.
- Patrones estructurales: Describen la forma en que diferentes tipos de objetos pueden ser organizados para trabajar unos con otros. Tratan de facilitar el diseño identificando formas simples de establecer relaciones entre entidades.
- Patrones de comportamiento: Se utilizan para organizar, manejar y combinar comportamientos. Identifican patrones comunes de comunicación entre objetos y tratan de incrementar la flexibilidad de esta comunicación.

[25] Siglas en inglés: *Personal Digital Assistant* (Asistente Digital Personal). Mini-computador diseñado como agenda electrónica.

Dentro de los patrones de creación se tiene en cuenta el siguiente:

- **Singleton o Instancia única:** Garantiza que una clase sólo tenga una instancia, y proporciona un punto de acceso global a ella.

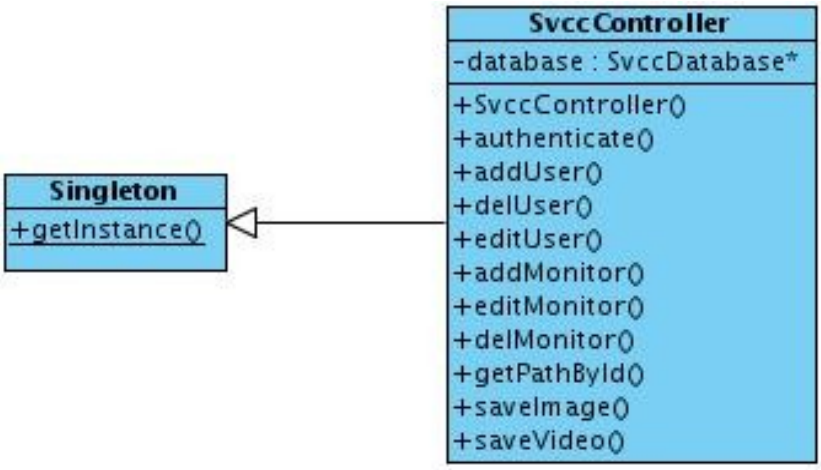

Figura 15 Uso del patrón Singleton.

Dentro de los patrones estructurales se tienen en cuenta el siguiente:

- **Facade o Fachada:** Proporciona una interfaz unificada para un conjunto de interfaces de un subsistema. Define una interfaz de alto nivel que hace que el subsistema sea más fácil de usar.

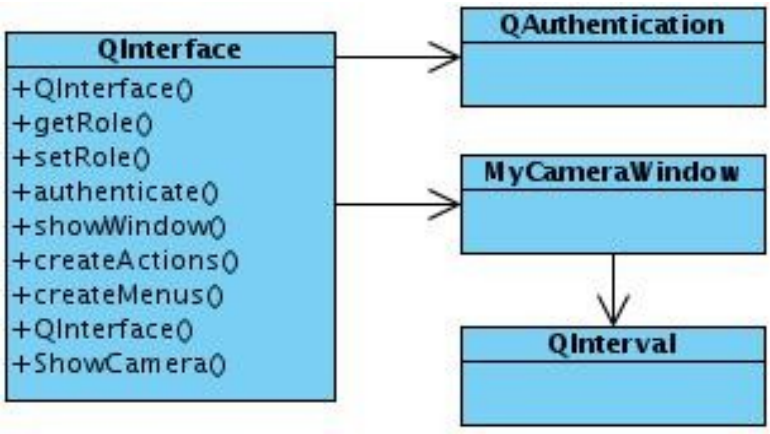

Figura 16 Uso del patrón Facade.

3.5 Conclusiones del capítulo

En este capítulo se aborda acerca de las características del Sistema de Vigilancia por Cámaras Cancerbero. Uno de los aspectos fundamentales tratados fue la captura de requisitos tanto funcionales como no funcionales. Gracias a ello se lleva a cabo el análisis y diseño del sistema, aspecto de vital importancia en todo desarrollo, ya que sirve de base para la implementación y posterior realización de pruebas que permitan verificar el correcto funcionamiento del software. De esto último se estará abordando en el próximo capítulo.

Capítulo 4: Implementación y pruebas

4.1 Introducción al capítulo

En este capítulo se explicarán los estándares de codificación y documentación utilizados para el desarrollo del software. También se explicará cómo se distribuye físicamente el hardware utilizado y cómo se llevaron a cabo las diferentes pruebas de sistema realizadas al software.

4.2 Estándar de codificación

4.2.1 Nombres

- Los nombres de las clases son sustantivos singulares.
- Los nombres de clases y objetos deben reflejar qué hacen y no cómo lo hacen.
- Escoger nombres lo suficientemente largos que expresen correctamente el sentido de lo que se quiere, pero evitando manejar nombres que dificulten la labor de implementación.
- Evitar nombres que permitan una interpretación subjetiva (evitar ambigüedad y asegurar abstracción).
- Evitar redundancia no repitiendo nombres de clases en sus elementos.
- Dado que los nombres generalmente son el producto de concatenar varias palabras, se debe emplear la primera palabra en minúscula, mayúscula para denotar la letra de inicio de cada una de las palabras restantes por las que esté formado y minúscula para las letras intermedias en el caso de los nombres de métodos y funciones. Para el caso de los nombres de variables y atributos debe aplicarse la misma convención.
- Las variables booleanas deben contener la palabra *is* en su nombre.
- Los nombres de constantes deben contener solo letras mayúsculas.
- Minimizar el uso de abreviaturas. En caso de ser requeridas, se debe ser consistente en su uso y cada abreviatura debe significar solo una cosa.
- Los nombres de los métodos son frases que incluyen verbos.
- Los nombres de los atributos y parámetros son frases con sustantivos.

4.2.2 Codificación

- Se establece un tamaño de indentación[26] estándar de tres espacios, sin tabulaciones.
- Alinear secciones del código.
- Alinear verticalmente llaves de apertura y cierre.
- Usar espacios antes y después de los operadores que el lenguaje de programación permita.
- Emplear líneas en blanco para organizar el código, permitiendo crear párrafos de código para una mejor lectura.
- Evitar colocar más de una sentencia por línea.
- Emplear constantes en sustitución de números o cadenas de caracteres literales.
- Minimizar el alcance de las variables para evitar confusión y facilitar el mantenimiento.
- Emplear cada variable y rutina solo para un propósito.
- Evitar el uso de variables públicas, sustituirlas por variables privadas y métodos que provean el valor de tal variable, para mantener el encapsulamiento.
- Emplear las letras i, j, k, l, m, p, q, r para contadores en ciclos.
- Mantener la modularidad del código bajo el criterio de la lógica que encierra, no exagerar la modularidad.
- Emplear correctamente los tipos de ciclos: si es al menos una vez usar *do-while*, si es ninguna o más veces usar *while-do*, y si se conoce el número exacto de ciclos usar *for*.
- Inicializar todas las variables.
- Emplear líneas en blanco para separar pasos lógicos (Ej.: declaraciones, lazos).

4.3 Estándar de documentación

Los formatos brindados por la herramienta Doxygen empleada para la documentación del sistema se explican a continuación:

- **Estilo de bloques de documentación.**

Se adopta el estilo de bloques de documentación JavaDoc, el cual consiste en un bloque de comentario de estilo C, los asteriscos que se encuentran en la línea de la mitad son opcionales. Ejemplo:

[26] Espacio o sangría que se pone a la derecha de cada línea de código.

```
/**
 *  Texto
 */
```

- **Descripción breve con el comando \brief o @brief.**

Para hacer una descripción breve se adopta el uso del comando \brief o @brief en el bloque de comentarios ya descrito.

La acción de este comando termina al final de un párrafo, de tal manera que la descripción detallada sigue después de una línea vacía, tal como lo muestra el ejemplo:

```
/**
 * @brief descripción breve.
 * Continuación de la descripción breve.
 *
 * La descripción detallada comienza aquí, nótese
 * que se debe dejar una línea en blanco para lograr
 * tener las dos descripciones (breve y detallada)
 */
```

Nota: Si no se utiliza el comando @brief Doxygen tomará la descripción hecha en el bloque como una descripción detallada y no habrá descripción breve.

- **Descripción de argumentos y métodos.**

A la hora de hacer una descripción de los argumentos de métodos y funciones ésta se hará en línea, es decir, luego de la declaración de cada uno de los argumentos se da una breve descripción de cada uno de ellos, en el siguiente ejemplo se muestra el formato a utilizar:

```
int multFunction (     int c , /**<Primer operando de la operación */
                       int d , /**<Segundo operando de la operación */
                       int e /**<Tercer operando de la operación */ );
```

- **Documentación de tipos de datos.**

Para describir la función y los elementos que componen los tipos de datos definidos se pueden utilizar los formatos especificados en los apartados anteriores. A continuación se muestra un ejemplo:

```
/**
 * @brief Enumerado de los tipos de datos admitidos
 *
 * Descripción más detallada de la función de este tipo de dato.
 */
enum DataTypes{ INTEGER, /**<Puede ser un valor de tipo entero */
                DOUBLE, /**<Puede ser un valor de tipo double */
                STRING /**<Puede ser un valor de tipo string */ };
```

Observamos que se genera una descripción breve seguida de una descripción más detallada para la función, luego se explica brevemente el valor de retorno de la misma y la descripción de los parámetros usando el formato de documentación en línea.

- **Comandos @author y @date.**

Es importante que se especifique el nombre del autor y la fecha de creación de cualquier estructura en un código, para ello se utilizan los comandos @author y @date para el nombre del autor y la fecha respectivamente, en el siguiente ejemplo se puede ver la acción de estos comandos:

```
/**@brief Descripción breve
 *
 * Aquí comienza la descripción detallada
 * @author Rosalina Puerto rpuerto@estudiantes.uci.cu
 * @date 13-04-2010
 */
```

- **Comando @see.**

Existe otro comando útil que permite hacer referencias a otras clases o métodos cuando esto sea necesario, es importante usar este comando en la documentación a la hora de implantar los métodos o funciones propias de alguna clase, este comando es @see y su uso se muestra en el siguiente ejemplo:

```
/**
* Constructor de la clase
* @see Test::Test()
*/
Test1();
```

Esto generará en la documentación para el constructor de la clase de prueba Test1 una referencia hacia la documentación existente para el constructor de la clase de prueba Test.

Nota: Si se quiere hacer una referencia desde cualquier estructura hacia la documentación completa de una clase ya documentada, solo se debe utilizar el comando @see seguido del nombre de la clase de esta forma:

```
/**
* Constructor de la clase
* @see Clase
*/
Test1();
```

4.4 Implementación

4.4.1 Diagrama de componentes

La aplicación está integrada por los componentes que a continuación se detallan:

- Validation: Es una biblioteca desarrollada por el proyecto Seguridad para verificar y validar la correcta sintaxis de los diferentes datos introducidos. La misma es utilizada en el software con el

objetivo de chequear algunos de los datos de usuarios y monitores que el usuario teclea.

- Encrypt: Es una biblioteca desarrollada por el proyecto Seguridad para el cifrado de cadenas de caracteres. La misma es utilizada en el software con el objetivo de almacenar la contraseña de usuario de manera cifrada en la base de datos, y así brindar mayor seguridad al sistema.
- Pqxx: Es una biblioteca que facilita el trabajo con la base de datos. En esencia es utilizada en el software para obtener el resultado de las consultas realizadas.
- Boost/lexical_cast: Es un componente que se utiliza para realizar conversiones entre los tipos de datos que se manejan en la aplicación.
- SvccDatabase: Es la clase desarrollada con el objetivo de responsabilizarse con todas las operaciones relacionadas con la base de datos, ya sea para seleccionar, insertar, eliminar o modificar los datos almacenados en la misma. (Ver Anexo 1)
- SvccController: Es la clase desarrollada con el objetivo de controlar el flujo de peticiones entre la interfaz de usuario y el acceso a los datos almacenados. (Ver Anexo 1)
- OpenCV: Es la biblioteca que posibilita el trabajo con las cámaras incorporadas al sistema. Es utilizada ya que brinda estructuras y funciones para el manejo de archivos de video e imágenes tales como:
    - cvCapture: Estructura de captura de video.
    - cvCreateCameraCapture: Permite inicializar la captura de video de una cámara conectada.
    - cvSaveImage: Permite almacenar una imagen en un directorio especificado.
    - cvQueryFrame: Permite obtener imágenes de una cámara y descomprimirlas en tiempo real.
- QInterface: Es la clase desarrollada con el objetivo de permitir la interacción con el usuario que manipula el sistema y mostrar los resultados de sus peticiones. (Ver Anexo 1)
- Qt (QtGui y QtCore): Los módulos QtGui y QtCore pertenecen al framework Qt, utilizado para el desarrollo de la interfaz gráfica de la aplicación. Ambos módulos brindan una amplia gama de clases que facilitan, tanto el diseño de la interfaz con la cual el usuario debe interactuar, como la

configuración de sus funcionalidades, teniendo en cuenta los requisitos y características propias del sistema.

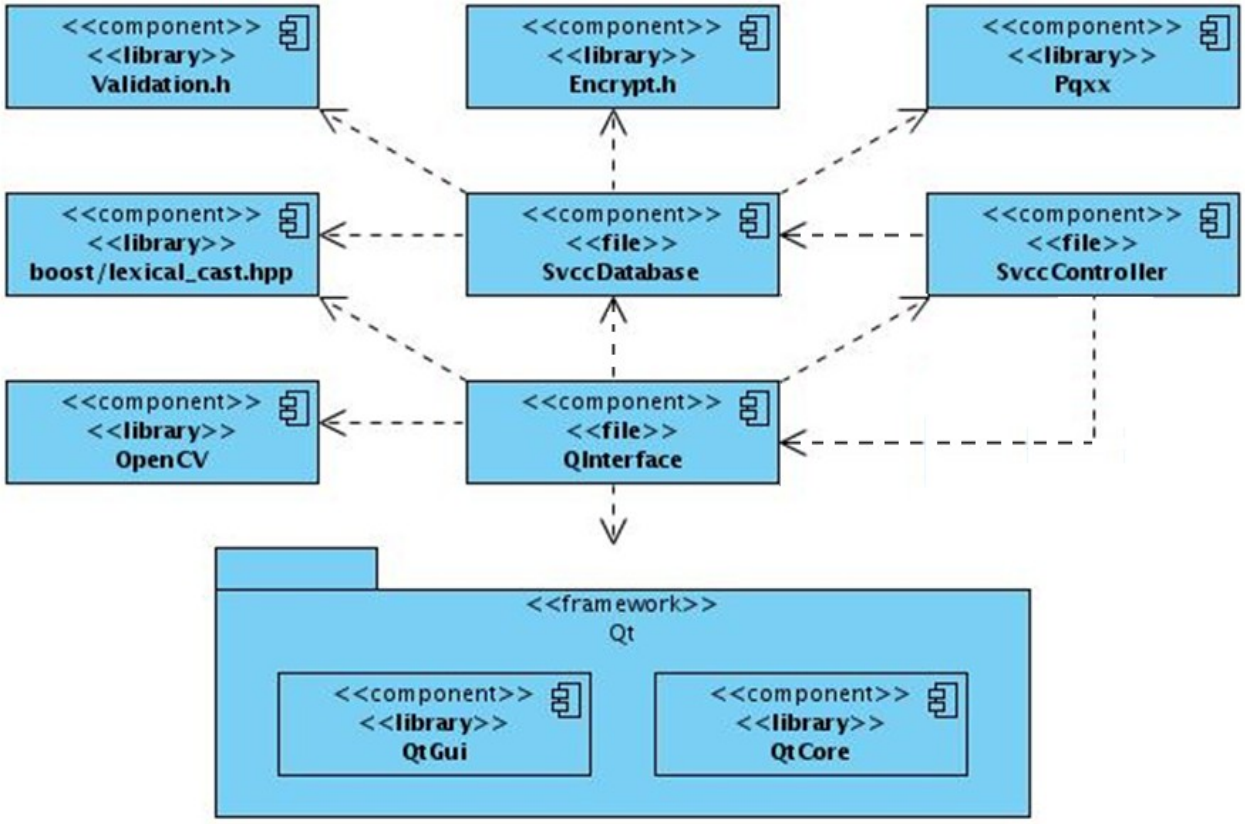

Figura 17 Diagrama de componentes.

4.4.2 Diagrama de despliegue

El diagrama de despliegue muestra las relaciones físicas de los distintos nodos que componen el sistema y el reparto de los componentes sobre dichos nodos. En la figura 18 se muestra una PC[27] cliente que tiene instalado el sistema SVCC y por lo tanto necesita conectarse al servidor que almacena todos los datos de los monitores y los usuarios que gestiona el sistema. Además se puede observar que las diferentes cámaras se conectan mediante cables USB a la PC cliente que es quien realiza el trabajo con los flujos de video obtenidos de la misma.

[27] Siglas en inglés: Personal Computer. Se refiere a una computadora personal.

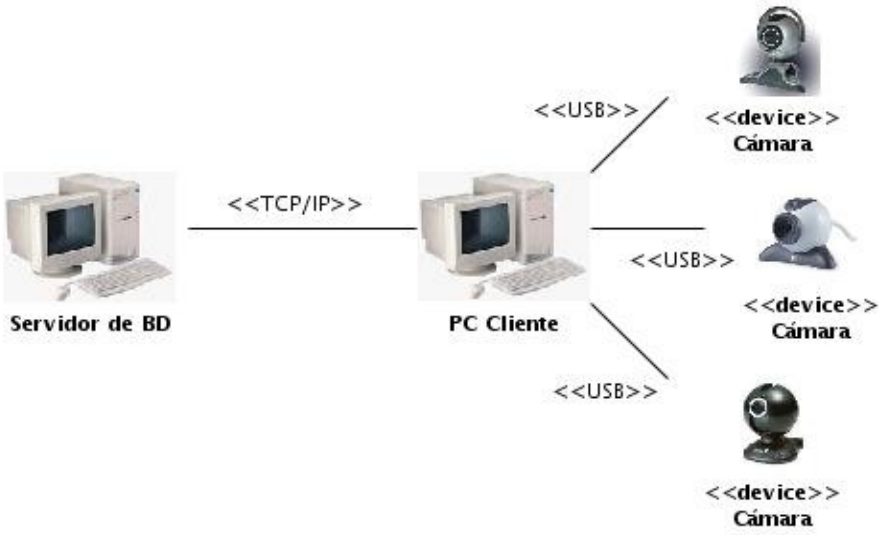

Figura 18 Diagrama de despliegue.

4.5 Pruebas

4.5.1 Descripción de los casos de prueba

La realización de pruebas es una actividad en la cual un sistema o componente es ejecutado bajo ciertas condiciones o requerimientos específicos, cuyos resultados son observados, registrados y evaluados.

Uno de los niveles de pruebas que existen son las pruebas de sistema, las cuales tienen como objetivo fundamental verificar el software ejerciendo como un todo. A continuación se detallan los casos de prueba elaborados por cada caso de uso para comprobar el correcto funcionamiento del sistema bajo las diferentes condiciones a las que puede someterse.

Nombre del caso de uso:	Almacenar imagen.	
Entrada	**Resultados esperados**	**Resultados obtenidos**
Caso de Prueba #1		
Al configurar el intervalo de almacenamiento se deja el campo vacío.	Acción fallida. El sistema muestra el mensaje “Debe introducir el dato solicitado”.	Prueba exitosa. Se obtienen los resultados esperados.
Caso de Prueba #2		
Se configura la dirección donde será almacenada la	El sistema muestra una ventana permitiendo seleccionar la dirección	Prueba exitosa. Se obtienen los resultados

imagen.	deseada.	esperados.
Caso de Prueba #3		
Se selecciona almacenar imagen.	El sistema almacena imágenes en la dirección especificada, con el intervalo especificado entre imagen e imagen, y las almacena con el siguiente nombre: “Svcc-año-mes-día-hora-minutos-segundos”.	Prueba exitosa. Se obtienen los resultados esperados.
Caso de Prueba #4		
Se selecciona detener almacenamiento de imagen.	El sistema detiene el almacenamiento, por lo que las imágenes pueden ser verificadas.	Prueba exitosa. Se obtienen los resultados esperados.

Tabla 14 Casos de prueba para CU Almacenar imagen.

Nombre del caso de uso:	Almacenar video.	
Entrada	**Resultados esperados**	**Resultados obtenidos**
Caso de Prueba #1		
Se configura la dirección donde será almacenado el video.	El sistema muestra una ventana permitiendo seleccionar la dirección deseada.	Prueba exitosa. Se obtienen los resultados esperados.
Caso de Prueba #2		
Se selecciona almacenar video.	El sistema almacena un video con lo que se está visualizando en pantalla en la dirección especificada.	Prueba exitosa. Se obtienen los resultados esperados.
Caso de Prueba #3		
Se selecciona detener almacenamiento de video.	El sistema detiene el almacenamiento, por lo que el video puede ser verificado.	Prueba exitosa. Se obtienen los resultados esperados.

Tabla 15 Casos de prueba para CU Almacenar video.

Nombre del caso de uso:	Visualizar monitor.	
Entrada	**Resultados esperados**	**Resultados obtenidos**
Caso de Prueba #1		
Visualizar un monitor que no está conectado.	Acción fallida. El sistema muestra el mensaje “Error en la captura del monitor de la dirección especificada”.	Prueba exitosa. Se obtienen los resultados esperados.
Caso de Prueba #2		
Visualizar un monitor que	Acción fallida. El sistema muestra el	Prueba exitosa.

está conectado pero la dirección especificada no se corresponde con la real.	mensaje “Error en la captura del monitor de la dirección especificada”.	Se obtienen los resultados esperados.
Caso de Prueba #3		
Visualizar un monitor con los datos correctos.	Se muestra una ventana con el monitor visualizándose.	Prueba exitosa. Se obtienen los resultados esperados.

Tabla 16 Casos de prueba para CU Visualizar monitor.

Nombre del caso de uso:	Autenticar.	
Entrada	**Resultados esperados**	**Resultados obtenidos**
Caso de Prueba #1		
Usuario: Correcto Contraseña: Correcto	Autenticación válida. El sistema muestra la ventana correspondiente según el rol del usuario autenticado.	Prueba exitosa. Se obtienen los resultados esperados.
Caso de Prueba #2		
Usuario: Incorrecto Contraseña: Correcto	Autenticación fallida. El sistema muestra el mensaje de advertencia “Error de autenticación”.	Prueba exitosa. Se obtienen los resultados esperados.
Caso de Prueba #3		
Usuario: Correcto Contraseña: Incorrecto	Autenticación fallida. El sistema muestra el mensaje de advertencia “Error de autenticación”.	Prueba exitosa. Se obtienen los resultados esperados.
Caso de Prueba #4		
Usuario: Incorrecto Contraseña: Incorrecto	Autenticación fallida. El sistema muestra el mensaje de advertencia “Error de autenticación”.	Prueba exitosa. Se obtienen los resultados esperados.
Caso de Prueba #5		
Usuario: Sin datos Contraseña: Con datos	Autenticación fallida. El sistema muestra el mensaje de advertencia “Debe llenar todos los campos”.	Prueba exitosa. Se obtienen los resultados esperados.
Caso de Prueba #6		
Usuario: Con datos Contraseña: Sin datos	Autenticación fallida. El sistema muestra el mensaje de advertencia “Debe llenar todos los campos”.	Prueba exitosa. Se obtienen los resultados esperados.
Caso de Prueba #7		
Usuario: Sin datos	Autenticación fallida. El sistema	Prueba exitosa.

Contraseña: Sin datos	muestra el mensaje de advertencia “Debe llenar todos los campos”.	Se obtienen los resultados esperados.

Tabla 17 Casos de prueba para CU Autenticar.

Nombre del caso de uso:	Adicionar monitor.	
Entrada	**Resultados esperados**	**Resultados obtenidos**
Caso de Prueba #1		
Nombre: Sin datos Dirección: Sin datos	Acción fallida. El sistema muestra el mensaje de advertencia “Debe llenar todos los campos”.	Prueba exitosa. Se obtienen los resultados esperados.
Caso de Prueba #2		
Nombre: Sin datos Dirección: Con datos	Acción fallida. El sistema muestra el mensaje de advertencia “Debe llenar todos los campos”.	Prueba exitosa. Se obtienen los resultados esperados.
Caso de Prueba #3		
Nombre: Con datos Dirección: Sin datos	Acción fallida. El sistema muestra el mensaje de advertencia “Debe llenar todos los campos”.	Prueba exitosa. Se obtienen los resultados esperados.
Caso de Prueba #4		
Nombre: Ya existente Dirección: Nueva	Acción fallida. El sistema muestra el mensaje de advertencia “Ya existe un monitor con ese nombre”.	Prueba exitosa. Se obtienen los resultados esperados.
Caso de Prueba #5		
Nombre: Nuevo Dirección: Ya existente	Acción fallida. El sistema muestra el mensaje de advertencia “Ya existe un monitor con esa dirección”.	Prueba exitosa. Se obtienen los resultados esperados.
Caso de Prueba #6		
Nombre: Ya existente Dirección: Ya existente	Acción fallida. El sistema muestra el mensaje de advertencia “El monitor ya existe”.	Prueba fallida. Se muestra el mensaje “Ya existe un monitor con ese nombre”.
Caso de Prueba #7		
Nombre: Nuevo Dirección: Nueva	Acción válida. El sistema adiciona el monitor a la base de datos. El sistema muestra el mensaje “El monitor ha sido insertado exitosamente”	Prueba exitosa. Se obtienen los resultados esperados.

Tabla 18 Casos de prueba para CU Adicionar monitor.

Nombre del caso de uso:	Editar monitor	
Entrada	**Resultados esperados**	**Resultados obtenidos**
Caso de Prueba #1		
No existen monitores en la base de datos.	Acción fallida. El sistema muestra el mensaje de advertencia “No existen monitores”.	Prueba fallida. Se muestra el mensaje “Debe seleccionar un monitor”.
Caso de Prueba #2		
Nombre: Sin datos Dirección: Sin datos	Acción fallida. El sistema muestra el mensaje de advertencia “Debe llenar todos los campos”.	Prueba exitosa. Se obtienen los resultados esperados.
Caso de Prueba #3		
Nombre: Con datos Dirección: Sin datos	Acción fallida. El sistema muestra el mensaje de advertencia “Debe llenar todos los campos”.	Prueba exitosa. Se obtienen los resultados esperados.
Caso de Prueba #4		
Nombre: Sin datos Dirección: Con datos	Acción fallida. El sistema muestra el mensaje de advertencia “Debe llenar todos los campos”.	Prueba exitosa. Se obtienen los resultados esperados.
Caso de Prueba #5		
Nombre: Igual Dirección: Nueva	Acción válida. El sistema almacena los nuevos datos del monitor en la base de datos y muestra el mensaje “El monitor ha sido editado exitosamente”.	Prueba fallida. Se muestra el mensaje “Error de sintaxis, vuelva a introducir los datos”.
Caso de Prueba #6		
Nombre: Igual Dirección: Igual	Acción fallida. El sistema muestra el mensaje de advertencia “Al menos un campo debe ser modificado. Acción denegada”.	Prueba exitosa. Se obtienen los resultados esperados.
Caso de Prueba #7		
Nombre: Nuevo Dirección: Igual	Acción válida. El sistema almacena los nuevos datos del monitor en la base de datos y muestra el mensaje “El monitor ha sido editado exitosamente”.	Prueba exitosa. Se obtienen los resultados esperados.
Caso de Prueba #8		
Nombre: Existente Dirección: Nueva	Acción fallida. El sistema muestra el mensaje de advertencia “Ya existe un monitor con ese nombre”.	Prueba exitosa. Se obtienen los resultados esperados.
Caso de Prueba #9		

Nombre: Nuevo Dirección: Existente	Acción fallida. El sistema muestra el mensaje de advertencia “Ya existe un monitor con esa dirección”.	Prueba exitosa. Se obtienen los resultados esperados.

Tabla 19 Casos de prueba para CU Editar monitor.

Nombre del caso de uso:	Eliminar monitor.	
Entrada	**Resultados esperados**	**Resultados obtenidos**
Caso de Prueba #1		
No se selecciona ningún monitor.	Acción fallida. El sistema muestra el mensaje de advertencia “Debe seleccionar un monitor”.	Prueba exitosa. Se obtienen los resultados esperados.
Caso de Prueba #2		
Se selecciona un monitor.	Acción válida. El sistema elimina el monitor de la base de datos y muestra el mensaje “El monitor ha sido eliminado exitosamente”.	Prueba exitosa. Se obtienen los resultados esperados.
Caso de Prueba #3		
No existen monitores en la base de datos.	Acción fallida. El sistema muestra el mensaje de advertencia “No existen monitores”.	Prueba fallida. Se muestra el mensaje “Debe seleccionar un monitor”.

Tabla 20 Casos de prueba para CU Eliminar monitor.

Nombre del caso de uso:	Adicionar usuario.	
Entrada	**Resultados esperados**	**Resultados obtenidos**
Caso de Prueba #1		
Todos los campos solicitados sin datos.	Acción fallida. El sistema muestra el mensaje de advertencia “Debe llenar todos los campos”.	Prueba exitosa. Se obtienen los resultados esperados.
Caso de Prueba #2		
Al menos uno de los campos solicitados sin datos.	Acción fallida. El sistema muestra el mensaje de advertencia “Debe llenar todos los campos”.	Prueba exitosa. Se obtienen los resultados esperados.
Caso de Prueba #3		
Usuario: Existente	Acción fallida. El sistema muestra el mensaje de advertencia “El usuario ya existe”.	Prueba exitosa. Se obtienen los resultados esperados.
Caso de Prueba #4		

Usuario: Nuevo Contraseña: Correcta (>5) Rep. contraseña: Incorrecta Es Admin.: Con datos	Acción fallida. El sistema muestra el mensaje de advertencia “Las contraseñas deben coincidir”.	Prueba exitosa. Se obtienen los resultados esperados.
Caso de Prueba #5		
Usuario: Nuevo Contraseña: Correcta (<5) Rep. contraseña: Correcta Es Admin.: Con datos	Acción fallida. El sistema muestra el mensaje de advertencia “La contraseña debe tener al menos 6 caracteres”.	Prueba exitosa. Se obtienen los resultados esperados.
Caso de Prueba #6		
Usuario: Nuevo Contraseña: Correcta (>5) Rep. contraseña: Correcta Es Admin.: Con datos	Acción válida. El sistema adiciona el usuario a la base de datos y muestra el mensaje “El usuario ha sido insertado exitosamente”.	Prueba exitosa. Se obtienen los resultados esperados.

Tabla 21 Casos de prueba para CU Adicionar usuario.

Nombre del caso de uso:	Editar usuario.	
Entrada	**Resultados esperados**	**Resultados obtenidos**
Caso de Prueba #1		
Todos los campos solicitados sin datos.	Acción fallida. El sistema muestra el mensaje de advertencia “Debe llenar todos los campos”.	Prueba exitosa. Se obtienen los resultados esperados.
Caso de Prueba #2		
Al menos uno de los campos solicitados sin datos.	Acción fallida. El sistema muestra el mensaje de advertencia “Debe llenar todos los campos”.	Prueba exitosa. Se obtienen los resultados esperados.
Caso de Prueba #3		
Usuario: Nuevo Contraseña: Correcta (>5) Rep. contraseña: Incorrecta Es Admin.: Con datos	Acción fallida. El sistema muestra el mensaje de advertencia “Las contraseñas deben coincidir”.	Prueba exitosa. Se obtienen los resultados esperados.
Caso de Prueba #4		
Usuario: Nuevo Contraseña: Correcta (<5) Rep. contraseña: Correcta Es Admin.: Con datos	Acción fallida. El sistema muestra el mensaje de advertencia “La contraseña debe tener al menos 6 caracteres”.	Prueba exitosa. Se obtienen los resultados esperados.

Caso de Prueba #5		
Usuario: Igual Contraseña: Igual Rep. contraseña: Correcta Es Admin.: Igual	Acción fallida. El sistema muestra el mensaje de advertencia “Al menos un campo debe ser modificado. Acción denegada”.	Prueba fallida. Se muestra el mensaje “El usuario ha sido editado exitosamente”.
Caso de Prueba #6		
Usuario: Igual Contraseña: Igual Rep. contraseña: Correcta Es Admin.: Nuevo	Acción válida. El sistema almacena los nuevos datos del usuario en la base de datos y muestra el mensaje “El usuario ha sido editado exitosamente. Vuelva a ejecutar SVCC para ver los cambios”.	Prueba fallida. Se muestra el mensaje “El usuario ha sido editado exitosamente”.
Caso de Prueba #7		
Usuario: Existente Contraseña: Nueva Rep. contraseña: Correcta Es Admin.: Con datos	Acción fallida. El sistema muestra el mensaje de advertencia “El usuario ya existe”.	Prueba exitosa. Se obtienen los resultados esperados.
Caso de Prueba #8		
Usuario: Nuevo Contraseña: igual Rep. contraseña: Correcta Es Admin.: Igual	Acción válida. El sistema almacena los nuevos datos del usuario en la base de datos y muestra el mensaje “El usuario ha sido editado exitosamente”.	Prueba exitosa. Se obtienen los resultados esperados.
Caso de Prueba #9		
Usuario: Igual Contraseña: Nuevo Rep. contraseña: Correcta Es Admin.: Igual	Acción válida. El sistema almacena los nuevos datos del usuario en la base de datos y muestra el mensaje “El usuario ha sido editado exitosamente”.	Prueba exitosa. Se obtienen los resultados esperados.
Caso de Prueba #10		
Usuario: Nuevo Contraseña: Nuevo Rep. contraseña: Correcta Es Admin.: Nuevo	Acción válida. El sistema almacena los nuevos datos del usuario en la base de datos y muestra el mensaje “El usuario ha sido editado exitosamente”.	Prueba exitosa. Se obtienen los resultados esperados.

Tabla 22 Casos de prueba para CU Editar usuario.

Nombre del caso de uso:	Eliminar usuario.	
Entrada	**Resultados esperados**	**Resultados obtenidos**
Caso de Prueba #1		
No se selecciona ningún usuario.	Acción fallida. El sistema muestra el mensaje de advertencia “Debe	Prueba exitosa. Se obtienen los resultados

	seleccionar un usuario".	esperados.
Caso de Prueba #2		
Se selecciona un usuario.	Acción válida. El sistema elimina el usuario de la base de datos y muestra el mensaje "El usuario ha sido eliminado exitosamente".	Prueba exitosa. Se obtienen los resultados esperados.

Tabla 23 Casos de prueba para CU Eliminar usuario.

Nombre del caso de uso:	Mostrar datos de monitor.	
Entrada	**Resultados esperados**	**Resultados obtenidos**
Caso de Prueba #1		
Adicionar monitor	Se muestra una tabla con los datos actualizados de los monitores existentes en la base de datos (id, nombre y dirección).	Prueba exitosa. Se obtienen los resultados esperados.
Caso de Prueba #2		
Eliminar monitor	Se muestra una tabla con los datos actualizados de los monitores existentes en la base de datos (id, nombre y dirección).	Prueba exitosa. Se obtienen los resultados esperados.
Caso de Prueba #3		
Editar monitor	Se muestra una tabla con los datos actualizados de los monitores existentes en la base de datos (id, nombre y dirección).	Prueba exitosa. Se obtienen los resultados esperados.

Tabla 24 Casos de prueba para CU Mostrar datos de monitor.

Nombre del caso de uso:	Mostrar datos de usuario.	
Entrada	**Resultados esperados**	**Resultados obtenidos**
Caso de Prueba #1		
Adicionar usuario	Se muestra una tabla con los datos actualizados de los usuarios (usuario y rol).	Prueba exitosa. Se obtienen los resultados esperados.
Caso de Prueba #2		
Eliminar usuario	Se muestra una tabla con los datos actualizados de los usuarios existentes (usuario y rol).	Prueba exitosa. Se obtienen los resultados esperados.

Caso de Prueba #3		
Editar usuario	Se muestra una tabla con los datos actualizados de los usuarios existentes (usuario y rol).	Prueba exitosa. Se obtienen los resultados esperados.

Tabla 25 Casos de prueba para CU Mostrar datos de usuario.

4.6 Conclusiones del capítulo

Los estilos de codificación y documentación explicados en este capítulo son aplicados en el Centro de Informática Industrial. Esto justifica el empleo de los mismos en el sistema de vigilancia por cámaras desarrollado, lo cual evidencia la intención de lograr una aplicación que pueda ser comprendida por usuarios comunes sin necesidad de tener un amplio conocimiento respecto al tema.

Además, en este capítulo se exponen los resultados obtenidos a partir de la aplicación de los diferentes casos de prueba efectuados a las funcionalidades del sistema, de los cuales se deriva la solidez del mismo, ya que fueron tratados los errores fundamentales identificados. Como conclusión de estas pruebas se puede decir que de un total de 60 casos, 54 resultaron satisfactorios y 6 demostraron vulnerabilidades del software. Todos los errores detectados fueron solucionados, lo cual demuestra que el proceso de validar la primera versión funcional del SVCC concluye de manera exitosa.

Conclusiones generales

Con la culminación del presente trabajo de diploma se logró desarrollar la primera versión de un sistema de video-vigilancia basado en tecnologías libres en su totalidad. Además, se obtiene un producto asequible y de fácil manipulación. La utilización de dicho sistema proveerá al Centro de Informática Industrial (CEDIN) de una herramienta que facilite la protección de sus recursos tanto humanos como materiales.

De manera general se cumplió con el objetivo trazado al inicio de la investigación y se alcanzaron los resultados esperados.

Recomendaciones

A continuación se recomiendan posibles aspectos a tener en cuenta para continuar con la mejora y perfeccionamiento del Sistema de Vigilancia por Cámaras (SVCC 1.0):

- Incorporar al sistema la posibilidad de visualizar las cámaras de manera simultánea.
- Añadir al sistema la funcionalidad de detectar movimiento para de esta manera disminuir la necesidad de la presencia humana de manera continua frente al monitor.
- Incorporar al sistema la posibilidad de configurar atributos de las cámaras tales como brillo, contraste, saturación y color.
- Estudiar la posibilidad de incluirle al sistema el reconocimiento de cámaras con tecnología IP y la posibilidad de monitorearlas a través de la red.

Si se logra la incorporación en el sistema desarrollado de algunos de los aspectos mencionados, sería de gran impacto ya que se lograría un sistema de video-vigilancia mucho más potente y se podría obtener la segunda versión del software.

Referencias bibliográficas

1. Sistema de videovigilancia. [En línea] 16 de Febrero de 2010. http://www.seguridadydefensa.com/noticias/sistema-de-videovigilancia-20796.html.

2. Davantis. [En línea] 18 de Febrero de 2010. http://www.davantis.com/la-empresa/conozcanos.php.

3. SecureMaster SA. [En línea] 16 de Febrero de 2010. http://www.sistemadevigilanciadigital.com/laempresa.php.

4. Secureimport. [En línea] 16 de Febrero de 2010. http://www.securimport.com/textos.php/id/21?osCsid=1a39468a7a1eee4f72030b682367fa82.

5. CirControl, identificación y seguridad. [En línea] 19 de Febrero de 2010. http://www.cirlatina.com.ar/circontrol/pdf/presentacion.pdf.

6. Videocámaras. [En línea] 15 de Febrero de 2010. http://www.hotfrog.es/Productos/Videocamaras/BARCELONA.

7. Rimaxonline.com. [En línea] 16 de Febrero de 2010. http://www.hotfrog.es/Empresas/Rimaxonline-com.

8. Camuniversal. [En línea] 16 de Febrero de 2010. http://camuniversal.programas-gratis.net/.

9. Campermanent. [En línea] 16 de Febrero de 2010. http://campermanent.programas-gratis.net/.

10. Eyeline Video Surveillance Software. [En línea] 16 de Febrero de 2010. http://eyeline-video-surveillance-software.programas-gratis.net/.

11. Domótica en Linux. [En línea] 23 de Febrero de 2010. http://www.ubuntu-es.org/index.php?q=node/14035.

12. ZoneMinder. [En línea] 18 de Febrero de 2010. http://www.softwarelibre.gob.ve/wiki-sl/index.php/ZoneMinder.

13. Ruiz Muzquiz, Pablo. Sistemas operativos. 2003.

14. ¿Qué es el software libre? [En línea] 24 de Febrero de 2010. http://hispalinux.es/SoftwareLibre.

15. Metodologías de desarrollo. [En línea] 19 de Febrero de 2010. http://www.marblestation.com/?p=644.

16. Reyes Bermúdez, Enrique. Sistemas de autenticación para el módulo Seguridad del proyecto Guardián del Alba. UCI : Trabajo de Diploma, 2009.

17. Historia del computador y evolución, arquitectura, lenguajes de programación y algoritmos. [En línea] 18 de Febrero de 2010. http://nectacellcomputer.jimdo.com/historia-del-computador-arquitectura-lenguajes-de-programaci%C3%B3n-y-algoritmos/.

18. Lenguaje de Programación. [En línea] 23 de Febrero de 2010. http://www.scribd.com/doc/13719562/Lenguaje-de-Programacion.

19. Microsoft. Base de Dato. [En línea] 22 de Febrero de 2010. http://gloriaisabelparra.spaces.live.com/blog/cns!409FD0578C87C97D!153.entry.

20. Sistema de gestión de base de datos. [En línea] 22 de Febrero de 2010. http://io.uvmnet.edu/revistadyn/app/articulo/ArticuloDyn.aspx?id=867.

21. PostgreSQL . [En línea] 23 de Febrero de 2010. http://www.teracat.com/soporte_programacion/.

22. Ingeniería de software I. [En línea] 24 de Febrero de 2010. http://www.scribd.com/doc/3062020/Capitulo-I-HERRAMIENTAS-CASE.

23. Visual Paradigm for UML. [En línea] 17 de Febrero de 2010. http://www.freedownloadmanager.org/es/downloads/Paradigma_Visual_para_UML_%28M%C3%8D%29_14720_p/.

24. Entornos de Desarrollo Integrado. [En línea] 17 de Febrero de 2010. http://petra.euitio.uniovi.es/~i1667065/HD/documentos/Entornos%20de%20Desarrollo%20Integrado.pdf.

25. Biblioteca Qt. [En línea] 26 de Febrero de 2010. http://es.wikipedia.org/wiki/Qt_%28biblioteca%29.

26. Valiente Mesa, Rosayda y Castillo Pérez, Idelsis. LIMS de Calidad del Centro de Ingeniería Genética y Biotecnología: Desarrollo de la Base de Datos del Módulo Sección de Mejoramiento de la Calidad. UCI : Trabajo de Diploma, 2008.

Bibliografía

1. Biblioteca OpenCV online [En línea] 29 de Abril de 2010. http://homepages.inf.ed.ac.uk/cgi/rbf/CVONLINE/entries.pl?TAG572.

2. Bilbao sustituye las cámaras de vigilancia análogicas por otras con tecnología IP. [En línea] 18 de Febrero de 2010. http://www.idg.es/computerworld.

3. Boost, c++ libraries. [En línea] 15 de Abril de 2010. http://www.boost.org/.

4. Bradski, Gary y Kaehler, Adrian. Learning OpenCV. [ed.] Mike Loukides. Primera edición. s.l. : O'Reilly Media, Inc., 2008.

5. Cámaras para sistemas de video-vigilancia (2da parte). [En línea] 18 de Febrero de 2010. http://www.aulati.net/?p=405.

6. CCTV. [En línea] 11 de Marzo de 2010. http://www.seguridadindustrialycomercial.com/cctv.php.

7. Conferencia 2: Arquitectura y Patrones de diseño. UCI : Ingeniería de Software II, 2010.

8. García, Pedro Luis. Cámaras de Seguridad con Motion. [En línea] 24 de Febrero de 2010. http://www.linuca.org/body.phtml?nIdNoticia=189.

9. Hayet, J. B. y Martínez, H. C. Introducción al uso de OpenCV. Centro de Investigación en Matemáticas, 2010.

10. Intplus. [En línea] 16 de Febrero de 2010. http://www.hotfrog.es/Empresas/Intplus.

11. QT (biblioteca). [En línea] 21 de Febrero de 2010. http://es.wikipedia.org/wiki/Qt_%28biblioteca%29.

12. Pressman, Roger S. Ingeniería del Software, un enfoque práctico. Sexta edición. s. l. : McGraw-Gill Interamericana. p. 980.

13. Rumbaugh, James, Booch, Grady and Jacobson, Ivar. El proceso unificado de desarrollo de software. s.l. : Félix Varela, 2004. Vol. I.

14. Sistemas de seguridad y vigilancia mediante cámaras. [En línea] 23 de Febrero de 2010. http://www.camarasip.cl/sistema_de_seguridad_mediante_camaras.html.

15. Sistemas de vigilancia por internet. [En línea] 23 de Febrero de 2010. http://www.sistemasdevigilancia.net.

16. Software de vigilancia. [En línea] 11 de Marzo de 2010. http://www.antirrobo.net/vigilancia/software-de-vigilancia.html.

17. Tipos de cámara. [En línea] 11 de Marzo de 2010. http://www.videovigilancia.com/tiposcamaras.htm.

18. Tipos de cámara de seguridad. [En línea] 11 de Marzo de 2010. http://www.sistemas-seguridad.com/blogseguridad/Tipos-de-Camaras-de-Seguridad.

19. Ventajas de PostgreSql. [En línea] 20 de Marzo de 2010. http://soporte.tiendalinux.com/portal/Portfolio/postgresql_ventajas_html.

20. Video-vigilancia. [En línea] 23 de Febrero de 2010. http://www.hotfrog.es/Productos/Videovigilancia.

21. ¿Video-Vigilancia con Software libre?, la Solución es ZoneMinder. [En línea] 19 de febrero de 2010 http://www.alcancelibre.org.

22. ZoneMinder. [En línea] 19 de Febrero de 2010. http://www.zoneminder.com.

Anexos

Anexo 1: Descripción de las clases utilizadas

Nombre: QInterface	
Tipo de clase: Interfaz	
Atributo	**Tipo**
Nombre:	QInterface()
Descripción:	Constructor por defecto de la clase.
Nombre:	~QInterface()
Descripción:	Destructor de la clase.
Nombre:	authenticate()
Descripción:	Permite la autenticación en el sistema mediante un llamado a la clase SvccController.
Nombre:	showWindow()
Descripción:	Permite mostrar la ventana correspondiente al usuario que está operando según su rol.
Nombre:	createActions()
Descripción:	Permite crear las acciones que hará cada una de las opciones del menú creado.
Nombre:	createMenus()
Descripción:	Permite la creación del menú de trabajo que observa el usuario en la interfaz.
Nombre:	showCamera (index, camera)
Descripción:	Permite visualizar un monitor en pantalla.

Tabla 26 Descripción de la clase QInterface.

Nombre: SvccController	
Tipo de clase: Controladora	
Atributo	**Tipo**
database	SvccDatabase*
Nombre:	SvccController()
Descripción:	Constructor por defecto de la clase.
Nombre:	~SvccController()
Descripción:	Destructor de la clase.
Nombre:	authenticate()

Descripción:	Envía a la clase SvccDatabase los datos de autenticación obtenidos de la clase QInterface.
Nombre:	addUser(string userName, string password, bool isAdmin)
Descripción:	Permite añadir un usuario al sistema. Esta operación la delega en la clase SvccDatabase.
Nombre:	delUser(string userName)
Descripción:	Permite eliminar un usuario del sistema. Esta operación la delega en la clase SvccDatabase.
Nombre:	editUser(string userName, string n_user, string n_pass, bool n_isAdmin)
Descripción:	Permite modificar los datos de un usuario del sistema. Esta operación la delega en la clase SvccDatabase.
Nombre:	addmonitor(int id, string name, string path)
Descripción:	Permite añadir un monitor al sistema. Esta operación la delega en la clase SvccDatabase.
Nombre:	delMonitor(int id)
Descripción:	Permite eliminar un monitor del sistema. Esta operación la delega en la clase SvccDatabase.
Nombre:	editMonitor(int id, string n_name, string n_path)
Descripción:	Permite modificar los atributos de un monitor del sistema. Esta operación la delega en la clase SvccDatabase.
Nombre:	getPathById(int id)
Descripción:	Permite devolver la dirección de un monitor del sistema cuyo id es pasado por parámetro. Esta operación la delega en la clase SvccDatabase.
Nombre:	saveImage(string path, int time)
Descripción:	Permite almacenar las imágenes de manera local teniendo en cuenta la configuración pasada por parámetro.
Nombre:	saveVideo(string path)
Descripción:	Permite almacenar el video de manera local teniendo en cuenta la configuración pasada por parámetro.

Tabla 27 Descripción de la clase SvccController.

Nombre: SvccDatabase	
Tipo de clase: Acceso a datos	
Atributo	**Tipo**
validate	Validation*
encrypt	Encrypt*
Nombre:	SvccDatabase()

Descripción:	Constructor por defecto de la clase.
Nombre:	~SvccDatabase()
Descripción:	Destructor de la clase.
Nombre:	authentication()
Descripción:	Permite verificar los datos de autenticación en la base de datos.
Nombre:	addUser(string userName, string password, bool isAdmin)
Descripción:	Permite añadir un usuario a la base de datos con los datos pasados por parámetro.
Nombre:	delUser(string userName)
Descripción:	Permite eliminar el usuario de la base de datos cuyo nombre de usuario (userName) es pasado por parámetro.
Nombre:	editUser(string userName, string n_user, string n_pass, bool n_isAdmin)
Descripción:	Permite modificar los atributos userName, password e isAdmin pasados por parámetro del usuario almacenado en la base de datos cuyo nombre de usuario es userName.
Nombre:	getUsernames()
Descripción:	Permite obtener todos los usuarios existentes en la base de datos.
Nombre:	getRole()
Descripción:	Permite obtener un listado de los roles de los usuarios existentes en la base de datos.
Nombre:	getIdNewMonitor()
Descripción:	Permite conocer cuál es el id que va a tener el nuevo monitor que se va a añadir a la base de datos (los números id son enteros positivos que van creciendo consecutivamente a medida que se añaden los monitores).
Nombre:	getMonitorIds()
Descripción:	Permite obtener los identificadores de todos los monitores existentes en la base de datos.
Nombre:	getMonPaths()
Descripción:	Permite obtener las direcciones de los monitores existentes en la base de datos.
Nombre:	addmonitor(int id, string name, string path)
Descripción:	Permite añadir un monitor a la base de datos con los datos dados.
Nombre:	delMonitor(int id)
Descripción:	Permite eliminar el monitor de la base de datos cuyo identificador (id) es pasado por parámetro.
Nombre:	editMonitor(int id, string n_name, string n_path)
Descripción:	Permite modificar en la base de datos los atributos name y path

	pasados por parámetro del monitor cuyo id es pasado por parámetro.
Nombre:	getMonitorNames()
Descripción:	Permite obtener un listado de los nombres de cada monitor de la base de datos.
Nombre:	getNameByPath(int path)
Descripción:	Permite obtener el nombre del monitor cuya dirección se conoce.
Nombre:	getIdByName(string name)
Descripción:	Permite obtener de la base de datos el id del monitor cuyo nombre es pasado por parámetro.
Nombre:	getPathById(int id)
Descripción:	Permite obtener de la base de datos la dirección del monitor cuyo id es pasado por parámetro.
Nombre:	getNameById(int id)
Descripción:	Permite obtener el nombre del monitor cuyo id se conoce.
Nombre:	getPassByUsername(string userName)
Descripción:	Permite obtener la contraseña de un usuario cuyo nombre de usuario se conoce.
Nombre:	getIsadminByUsername(string userName)
Descripción:	Permite obtener el rol de un usuario cuyo nombre de usuario se conoce.
Nombre:	getCantMonitors()
Descripción:	Permite obtener la cantidad de monitores existentes en la base de datos.
Nombre:	functions()
Descripción:	Permite ejecutar consultas a la base de datos.
Nombre:	getConn()
Descripción:	Devuelve una conexión nueva por cada vez que es instanciado.

Tabla 28 Descripción de la clase SvccDatabase.

Anexo 2: Imágenes de la aplicación

Figura 19 Ventana de inicio al cargar SVCC.

Figura 20 Ventana de autenticación.

Figura 21 Ventana de información de monitores.

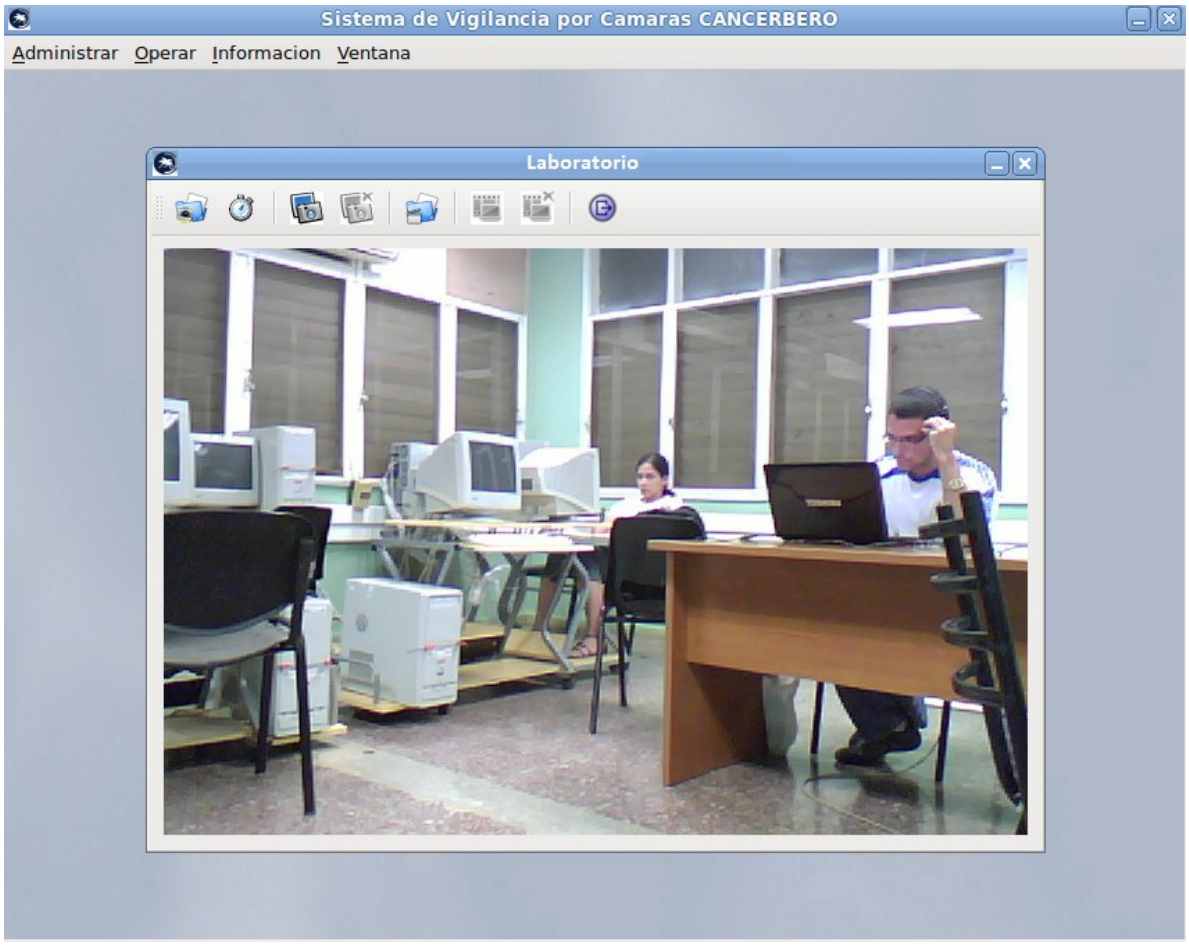

Figura 22 Ventana de visualización del monitor Laboratorio.

Figura 23 Ventana de información de usuarios.

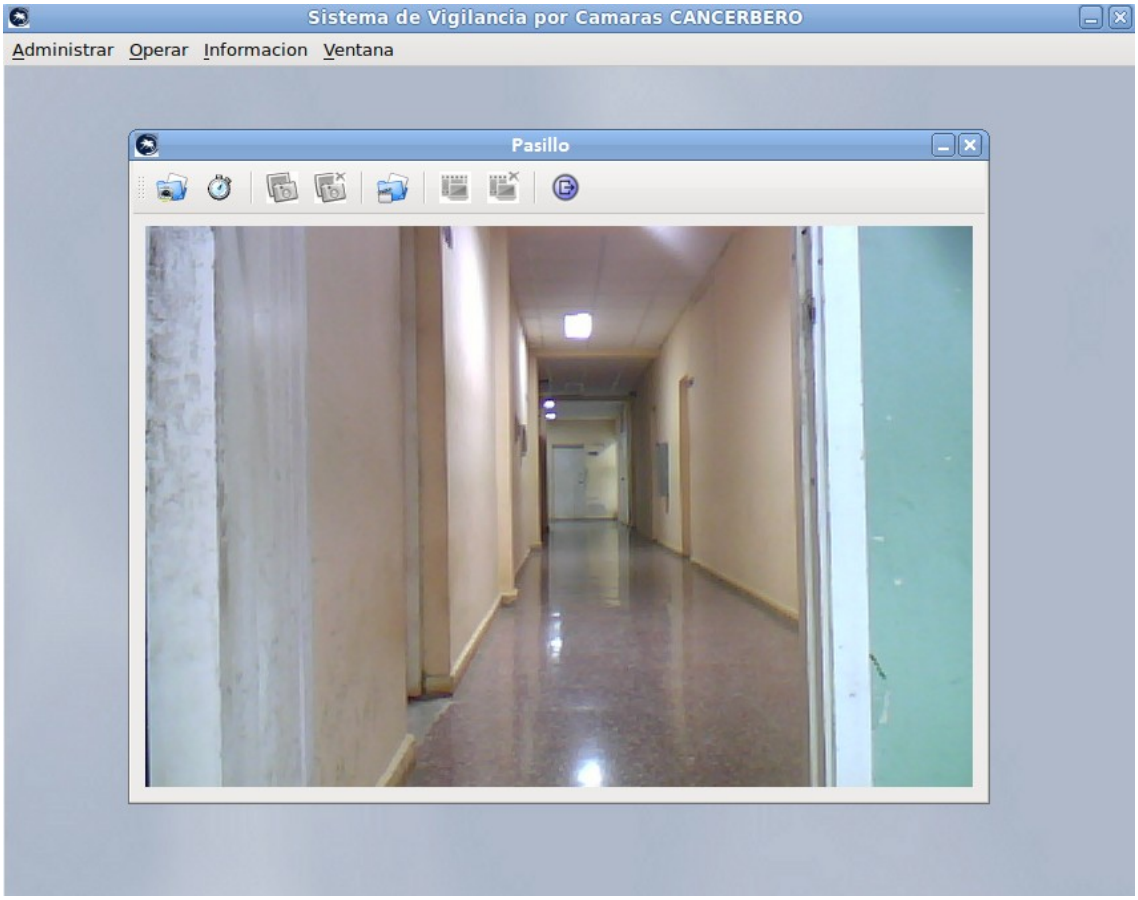

Figura 24 Ventana de visualización del monitor Pasillo.

Figura 25 Ventana de adicionar usuario.

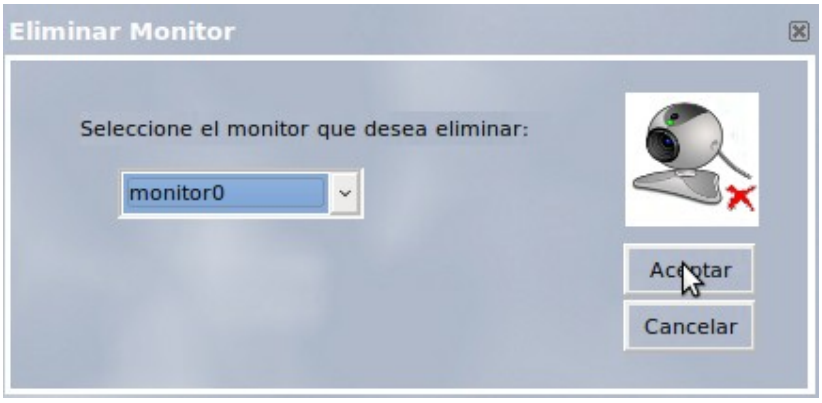

Figura 26 Ventana de eliminar monitor.

Glosario de términos

A

- Aplicación: Terminología técnica para referirse a un programa de software.
- Arquitectura: Indica la estructura, funcionamiento e interacción entre las partes del software.
- Autenticación: Verificación de la identidad de una persona o de un proceso para acceder a un recurso o poder realizar determinada actividad.

C

- CEDIN: Siglas que identifican el Centro de Informática Industrial de la facultad 5 de la Universidad de las Ciencias Informáticas.
- Cliente: Un sistema o proceso que solicita a otro sistema o proceso que le preste un servicio.
- Contraseña: Información confidencial constituida por una cadena de caracteres, usada para la autenticación de un usuario o para el acceso a un recurso.
- Control de acceso: Mecanismo que en función de la identificación ya autenticada permite acceder a datos o recursos.

F

- Framework: Marco de trabajo que proporciona una estructura de soporte mediante la cual un proyecto de software puede ser organizado y desarrollado.

H

- Hardware: Componente físico de una computadora o de una red, en contraposición con los programas o elementos lógicos que la hacen funcionar.

I

- IDE: Siglas en inglés que identifican un Entorno de Desarrollo Integrado (*Integrated Development Environment)*. Es un entorno de programación que ha sido empaquetado como un programa de aplicación, es decir, consiste en un editor de código, un compilador, un depurador y un constructor de interfaz gráfica.

- Identificación del usuario: Procedimiento de reconocimiento de la identidad de un usuario.

P

- Programa: Conjunto de instrucciones que una vez ejecutadas realizan una o varias tareas en una computadora.

R

- Recurso: Cualquier parte componente de un sistema de información.

S

- SCADA: Siglas que identifican a un sistema desarrollado para el control y la supervisión de datos (Supervisory Control And Data Adquisition).

- Software: Conjunto de programas, documentos, procesamientos y rutinas asociadas con la operación de un sistema de computadoras, es decir, la parte intangible o lógica de una computadora.

U

- Usuario: Sujeto o proceso autorizado para acceder a datos o recursos.

V

- Video-vigilancia: Mecanismo de supervisión mediante el uso de cámaras.