UNIVERSIDAD DE LAS CIENCIAS INFORMÁTICAS

Facultad 6

Título: "Simulador de Cámaras IP para el sistema Video Vigilancia Suria."

Trabajo de Diploma para optar por el título de Ingeniero en Ciencias Informáticas.

Autores: Doina Yudith Ferro Fonseca

Yorbenys Pardo Rodríguez

Tutora: Ing. Olga María Rivera Correa

"Año 55 de la Revolución"

La Habana, junio de 2013

“No son las riquezas ni el esplendor, sino la tranquilidad y el trabajo, los que proporcionan la felicidad.”

Oscar Wilde

Declaración de Autoría

Declaramos ser autores de la presente tesis y reconocemos a la Universidad de las Ciencias Informáticas los derechos patrimoniales de la misma, con carácter exclusivo. Para que así conste se firma la presente a los ____ días del mes de _____ del año _____.

Doina Yudith Ferro Fonseca ___ Yorbenys Pardo Rodríguez

Nombre del Autor Nombre del Autor

Ing. Olga María Rivera Correa

Nombre del Tutor

Datos de Contacto

Tutor principal:

Ing. Olga María Rivera Correa

Graduada de la Universidad de las Ciencias Informáticas (UCI) en el año 2010. Pertenece al departamento Señales Digitales del Centro de Geoinformática y Señales Digitales (GEySED) de la Facultad 6. Labora en el proyecto Video Vigilancia (Suria), donde es la analista principal.
Correo electrónico: omrivera@uci.cu

Dedicatoria

A mi abuelo Modesto que siempre confió en mí y A mi madre a quien amo y le debo toda mi vida.

Doina

A mis padres que siempre han confiado en mí y me han dado todo su apoyo, a los cuales hoy les debo lo que soy, a mis hermanos a los cuales amo por siempre estar ahí cuando los necesito.

Yorbenys

Agradecimientos

A mi mamá por ser como una hermana y amiga, por su apoyo y confianza, por luchar siempre por darme lo mejor. A mi abuela Mariluz por su amor y su cariño.

A Ale por considerarme una hija y ayudarme incondicionalmente como un padre, por estar levantándome el ánimo y alegrándome siempre.

A mis hermanos Ale y Mary quienes forman una parte muy importante de mi, por quererme y respetarme.

A nuestra tutora Olga por su ayuda y preocupación, por estar siempre a disposición.

A mi compañero de tesis, por ser constante en el trabajo conmigo, por ayudarme y sobre todo por ser el mejor compañero de tesis que pude tener.

A todo el equipo de trabajo del proyecto Video Vigilancia con quienes tuve el placer de trabajar y que ayudaron de una forma u otra con la realización de esta tesis.

A los compañeros del tribunal que con sus críticas y consejos nos ayudaron a realizar esta tesis.

A Reidel por estar conmigo en los buenos y malos momentos, por quererme y apoyarme, por ser novio y amigo.

A todas aquellas personas que de una forma u otra han estado a mí lado apoyándome, fundamentalmente mis primas Ileana y Laura.

A los profesores Anay y Frank Alain por ser profesores y amigos. Por apoyarme en momentos difíciles e impulsarme a seguir adelante.

A los buenos amigos que no pudieron estar disfrutando de este momento y que siempre creyeron en mí, a Moya, Roberto, Yudelis, Yandy, Osnel, a todos ellos mi cariño y el esfuerzo de cada día.

A las chicas que fueron mis hermanas durante estos años y a quienes amo muchísimo a Tamara, Ori y Ceci. A las compañeras del 93106 a quienes a pesar de todo voy a extrañar muchísimo especialmente a Vivi y Yunet. A los amigos que quiero y que forman una parte muy importante de mi vida en esta escuela a Andris, José, Oscar y Darcy.

A los chicos del 111103 que siempre creyeron en mi Eric, Pedro, Daldis, José E. y Frank. A mis amigos de ahora y de la infancia El Chino, Jessica y Lili. A todos aquellos que no menciono pero que han formado parte de mi estancia en esta universidad y a quienes agradezco cada minuto compartido.

Doina

Agradecimientos

A mi madre, por ser madre y amiga al mismo tiempo, por darme todo su amor y apoyo incondicional, por estar presente en todos los momentos de mi vida, por luchar por mí y preocuparse por mi futuro.

A mi padre, por ser padre y amigo, por su amor y cariño, por su apoyo, por sus consejos, por su preocupación y por tratar de siempre darme lo mejor.

A mis hermanos por estar siempre presente cuando los he necesitado.

A todos mis familiares en general que de una manera u otra siempre han estado presentes en mi vida.

A mis amigos de la escuela, Adrián y Jorge con los cuales he compartido estos 5 años de carrera y se han convertido en más que amigos, son casi familiares para mí.

A mi amigo de prácticamente toda la vida, Daniel, por ser prácticamente un hermano para mí.

A mis amigos que hoy no están presentes en la universidad pero fueron excelentes compañeros, a Osnel, Yandy, Negrin, Papito, gracias por su amistad.

A mis compañeros de aula y de apartamento con los cuales he convivido todos estos años de universidad.

A mi compañera de tesis por ser muy responsable y preocupada, además de ser una excelente amiga.

A nuestra tutora Olga por su ayuda y preocupación, por estar siempre a disposición, prácticamente ha sido una compañera de tesis más, que ha trabajado a la par con nosotros.

A todo el equipo de trabajo del proyecto Video Vigilancia con quienes tuve el placer de trabajar y que ayudaron de una forma u otra con la realización de esta tesis.

A los compañeros del tribunal que con sus críticas y consejos nos ayudaron a realizar esta tesis.

A todos aquellos que no menciono pero que han formado parte de mi estancia en esta universidad y a quienes agradezco cada minuto compartido.

Gracias.

Yorbenys.

Resumen

En la facultad 6, el departamento de Señales Digitales cuenta con un proyecto dedicado al desarrollo de un sistema para la video vigilancia (Suria 1.0) utilizando cámaras IP[1] (Internet Protocol). Este producto tiene como objetivo el desarrollo de un Sistema Integral de Seguridad multiplataforma. El cual, para su despliegue, requiere de una infraestructura tecnológica que soporte un conjunto de cámaras IP de distintos modelos y que pueden gestionarse dentro de una red de datos, asegurando el almacenamiento y la trasmisión de los flujos de vídeos. Actualmente este sistema ofrece un conjunto de funcionalidades ya implementadas que no pueden ser probadas, pues no existe el entorno adecuado para realizar dichas pruebas, interrumpiéndose los procesos de liberación del producto y la implementación de nuevas funcionalidades.

Esta investigación ha sido realizada con el objetivo de desarrollar un Simulador de Cámaras IP para el sistema Video Vigilancia Suria de manera que se pueda probar la calidad del software en un ambiente lo más real posible, teniendo en cuenta los modelos de cámaras que puedan ser integrados y las funcionalidades que estas brindan. Este tipo de solución proporciona ahorro de recursos, pues no es necesaria la compra de cámaras para probar las funcionalidades existentes, ni se detendría la implementación de nuevas funcionalidades.

Palabras claves: cámaras IP, simulador, Suria, vigilancia.

[1] **IP: Internet Protocol o Protocolo de Internet.** es un protocolo de comunicación de datos digitales clasificado funcionalmente en la Capa de Red según el modelo internacional OSI.

Tabla de Contenido

Tabla de Contenido

Tabla de Contenido

Introducción

Introducción

El continuo desarrollo de la humanidad en el campo de la ciencia propició que el hombre sintiese la necesidad de corregir los errores encontrados en sus investigaciones, con el fin de crear herramientas para apoyar el proceso de la toma de decisiones. Para ello sus ideas y esfuerzos se enfocaron en la búsqueda y creación de sistemas matemáticos que les permitieran resolver, a través de ecuaciones y fórmulas, interrogantes en diversas disciplinas. Es entonces alrededor de 1777, cuando se da a conocer por primera vez el término de simulación para referirse a un método matemático que a partir de sucesivos intentos aproximaba el valor del número pi.(Wix, 2012) Posteriormente, a mediados de los años 40 se construyen los primeros computadores de propósito general proporcionándole a la simulación las bases para su rápida evolución.

El término de simulación por ordenador solía referirse a la construcción de modelos informáticos que describen la parte esencial del comportamiento de un sistema, así como diseñar y realizar experimentos con tales modelos con el fin de extraer conclusiones de sus resultados para apoyar la toma de decisiones. (Instituto Tecnológico de Acapulco, 2000) En la Segunda Guerra Mundial, se utilizaba simulación básica para resolver problemas de interés militar a través de ordenadores analógicos que usaban elementos electrónicos para resolver las operaciones matemáticas. En la década de los 60, la simulación comienza a enfocarse en resolver problemas de ámbito civil y luego de la revolución que se produjo en la informática a partir de los años 80, el uso de simuladores por ordenador se generaliza en prácticamente todos los ámbitos de la ciencia y la ingeniería(Ecured, 2011). Actualmente es una metodología de experimentación fundamental en campos tan diversos como la economía, la informática, la ingeniería y la física, con enormes aplicaciones industriales y comerciales.(Insua, 2008).

En Cuba, el avance científico-tecnológico de las últimas décadas y el uso de las tecnologías de la información y las comunicaciones (TIC), han impulsado el desarrollo de aplicaciones simuladoras, con el fin de preparar a personas en distintas esferas de la vida cotidiana, tales como la medicina o la infantería militar. Por las diversas ventajas que ofrecen los simuladores, su empleo es fundamental para el análisis, el entendimiento y la modificación de sistemas, pues son herramientas que aportan destreza, habilidad y formación continua al país. Su desarrollo disminuye las barreras tecnológicas y se evitan los altos costes que conllevan a la compra de equipos que pueden ser simulados.

que soporte un conjunto de cámaras IP de La Universidad de las Ciencias Informáticas es un centro de estudios universitarios, creado con los objetivos de informatizar el país resolviendo problemas sociales desde el punto de vista informático y ampliar la industria del software para contribuir al desarrollo económico del mismo. (Ecured, 2011) Específicamente en la facultad 6, el departamento de Señales Digitales cuenta con un proyecto dedicado al desarrollo de un sistema para la video vigilancia empleando cámaras IP. Este producto para su despliegue, requiere de una infraestructura tecnológica distintos modelos, asegurando la visualización y el almacenamiento de los flujos de vídeos generados en cada uno de los dispositivos de adquisición.(Redmine, 2006). Actualmente este sistema brinda un conjunto de funcionalidades ya implementadas que no pueden ser probadas pues no existe el entorno adecuado para realizar dichas pruebas, interrumpiéndose los procesos de liberación del producto y la implementación de nuevas funcionalidades.

La situación descrita anteriormente lleva a plantearse el siguiente **problema a resolver:** ¿Cómo lograr un entorno virtualizado de cámaras IP para el sistema Video Vigilancia Suria, garantizando probar las funcionalidades implementadas? Para realizar un estudio profundo de la problemática y desarrollar la investigación se define como **objeto de estudio** los procesos de visualización y control de video desde cámaras IP, enmarcándose como **campo de acción** el proceso de visualización y control de videos desde cámaras IP para el sistema Video Vigilancia Suria.

Para limitar el alcance de la investigación y darle solución al problema planteado se concreta como **objetivo general** desarrollar un Simulador de Cámaras IP para el sistema Video Vigilancia Suria. Como **idea a defender** se plantea que**:** “Con el desarrollo de un Simulador de Cámaras IP para el sistema Video Vigilancia Suria, se podrán probar todas las funcionalidades implementadas sin necesidad de dedicar un presupuesto para la compra de cámaras IP”.

Con el fin de dar cumplimiento a los objetivos anteriormente planteados se definieron las siguientes **tareas de investigación**:

- ✓ Caracterizar los procesos relacionados con la simulación de cámaras.
- ✓ Caracterizar las cámaras IP que actualmente soporta el sistema Video Vigilancia Suria.
- ✓ Caracterizar las herramientas y tecnologías seleccionadas para la implementación del Simulador de Cámaras IP para el sistema Video Vigilancia Suria.
- ✓ Realizar el diseño del Simulador de Cámaras IP para el sistema Video Vigilancia Suria.

- ✓ Implementar un Simulador de Cámaras IP para el sistema Video Vigilancia Suria.
- ✓ Realizar pruebas de caja negra al Simulador de Cámaras IP implementado.

Los **métodos científicos** de investigación que se utilizan con el fin de dar cumplimiento a los objetivos de este trabajo son:

Métodos Teóricos

- ✓ Analítico – Sintético: para realizar un estudio de la bibliografía referente a los diferentes conceptos asociados al tema permitiendo la extracción de los elementos más importantes que se relacionan con el objeto de estudio.
- ✓ Modelación: mediante este método se realizarán los modelos que especifican las características que va a desarrollar la aplicación y que permitirán una mejor comprensión en el posterior desarrollo del Simulador de Cámaras IP para el sistema Video Vigilancia Suria.
- ✓ Histórico – Lógico: para profundizar en el estudio de los simuladores, sus antecedentes y evolución a lo largo de la historia.

Métodos Empíricos

- ✓ Observación: para observar el funcionamiento del sistema Suria 1.0 y los modelos de cámaras que soporta, así como las operaciones que pueden ser realizadas sobre estas.
- ✓ Entrevista: se utiliza con el objetivo de entender el funcionamiento actual del sistema Suria 1.0 con el propósito de definir las principales funcionalidades que debe cubrir el simulador que se va a implementar. Para ello se toma como muestra a tres especialistas que trabajan en el desarrollo del sistema, dos desarrolladores y el líder del proyecto. Ver Anexo 1.

El presente trabajo está distribuido en los siguientes **Capítulos**:

Capítulo 1: Fundamentación Teórica: En este capítulo, se fundamentan términos técnicos de importancia para la investigación. Se realiza un estudio del arte sobre sistemas que existen en Cuba y el mundo con características similares a las requeridas para la investigación. También se incluye las herramientas y tecnologías a utilizar.

Introducción

Capítulo 2: Características del Sistema: En este capítulo se define el modelo del dominio del sistema que se va a desarrollar. Se describen las clases y flujos del diagrama modelo del dominio. También se identifican los requisitos funcionales y no funcionales. Se definen los casos de usos y actores del sistema.

Capítulo 3: Análisis y Diseño: En este capítulo se define la arquitectura base del Simulador de Cámaras IP mostrándose los artefactos generados en esta fase. También se muestran los resultados obtenidos en el desarrollo de los procesos de diseño. Así como los diagramas que fueron necesarios para obtener una mayor claridad a la hora de elaborar la solución que se propone tales como el diagrama de clases del diseño.

Capítulo 4: Implementación y Prueba: En este capítulo se aborda el tema referente a la construcción del Simulador de Cámaras IP para el sistema Video Vigilancia Suria, mostrando un conjunto de artefactos como los diagramas de despliegue y de componentes. Se realiza una descripción de las pruebas realizadas al sistema y los resultados alcanzados.

Capítulo 1: Fundamentación Teórica

Capítulo 1: Fundamentación Teórica

Introducción

En este capítulo se tiene como objetivo realizar una introducción al tema de la investigación, haciendo referencia a un conjunto de conceptos asociados al dominio del problema a resolver. Se realiza un estudio acerca de los simuladores de cámaras de red existentes en Cuba y el mundo. Además se mencionan las principales herramientas y tecnologías que serán utilizadas en el desarrollo de la aplicación.

1.1 Términos asociados al dominio de video vigilancia y los simuladores.

Simuladores o Simulación:

Según (Pierre, 2009)*"...son objetos de aprendizaje que mediante un programa de software, intentan modelar parte de una réplica de los fenómenos de la realidad y su propósito es que el usuario construya conocimiento a partir del trabajo exploratorio, la inferencia y el aprendizaje por descubrimiento."*

Según (Banks, 2007) *"…es el desarrollo de un modelo lógico matemático de un sistema, de tal forma que se tiene una imitación de la operación de un proceso de la vida real o de un sistema a través del tiempo. La simulación involucra la generación de una historia artificial de un sistema, la observación de esta historia mediante la manipulación experimental, ayuda a inferir las características operacionales de tal sistema."*

Teniendo en cuenta las definiciones anteriores y del objetivo de la presente investigación puede decirse que los simuladores son aplicaciones informáticas que permiten simular o modelar las características y propiedades de un sistema. Se plantea entonces la simulación como la técnica que permite representar sistemas utilizando modelos que imitan aspectos de la realidad de forma controlada en un ambiente artificial.

Video:

Según (Digitales, 2008-2009)*"...suele llamarse video a la captura, grabación, almacenamiento, y reconstrucción de una serie de imágenes y sonidos, las cuales representan escenas en movimiento."*

Capítulo 1: Fundamentación Teórica

Cámaras IP:

Según (Marrero, 2011) *"…es una cámara que se conecta directamente a la red empleando el protocolo IP. Se le asigna una dirección IP de forma tal que se pueda acceder a sus funciones desde la red, para ser empleadas por aplicaciones capaces de detectarla y emplearla."*

Según (Asesorias y computadores LTDA, 2005-2012)*"…Las cámaras IP son dispositivos autónomos que cuentan con un servidor web de video incorporado, lo que les permite transmitir su imagen a través de redes IP. Las cámaras IP permiten al usuario tener la cámara en una localización y ver el vídeo en tiempo real desde otro lugar a través de Internet."*

Sistema de vigilancia:

Según (Expósito, 2011)*"...se basan fundamentalmente en una infraestructura física que da como resultado instalaciones con un gran número de cámaras, ya sean IP o cualquier otra alternativa. Esta gran cantidad de cámaras es controlada en un centro de control por un personal calificado para la realización de este trabajo."*

Según (PCE, 2012)*"… todo tipo de aparatos para la detección inmediata y sistemática, la visualización o vigilancia de un proceso con ayuda técnica, sensores u otros sistemas de vigilancia, como por ejemplo una cámara."*

Teniendo en cuenta las definiciones mencionadas y las características de esta investigación se define que un sistema de vigilancia es la infraestructura compuesta por la instalación de un determinado número de cámaras que pueden ser de diferentes modelos y que son controladas por un personal capacitado en un centro de control con la ayuda de un software especializado que centraliza todas las funcionalidades de la infraestructura creada.

1.1.1 ¿Qué es un Simulador de Cámaras IP?

Atendiendo a las definiciones anteriores se plantea entonces que un Simulador de Cámaras IP es una aplicación informática que permite modelar y representar el comportamiento y las propiedades de una cámara IP. Permitiendo acceder a sus funcionalidades dependiendo del modelo de la cámara que se desee simular, creando una representación de la realidad a la que están sujetas las mismas y que

conduzcan a entender, comprobar, modificar y analizar el funcionamiento de los sistemas a los que se integran.

1.2 Descripción de la situación problémica.

La Universidad de las Ciencias Informáticas cuenta con un conjunto de centros productivos dedicados al desarrollo de software. Los mismos están distribuidos en las distintas facultades que posee la universidad. Entre ellos se encuentra GEySED, centro compuesto por los departamentos Geoinformática y Señales Digitales perteneciente a la facultad 6. Específicamente en el departamento de Señales Digitales existe un proyecto que se centra en el desarrollo de un sistema para la video vigilancia empleando cámaras IP (Suria 1.0) de forma que permita fortalecer la seguridad en cualquier institución o entidad donde se utilice.

Este sistema incluye una solución de software que permite la visualización de múltiples cámaras en distintos modos de trabajo y monitores. Dichas cámaras pueden ser de distintos modelos por lo que las funcionalidades que se realicen sobre ellas varían en dependencia de sus capacidades. Así como también posibilita la grabación bajo demanda o basada en calendarios de los flujos de video generados. Para su despliegue, la aplicación requiere de una infraestructura tecnológica capaz de soportar un conjunto de cámaras IP, las cuales pueden gestionarse dentro de una red de datos.(Redmine, 2006).

Suria 1.0 cuenta actualmente con un conjunto de funcionalidades implementadas que no han sido probadas, pues no existe el entorno adecuado para realizar dichas pruebas. Frenándose así, el proceso de desarrollo para versiones posteriores de este producto. Ante nuevas solicitudes de cambios, relacionadas con los modelos de cámaras que soporta el sistema y con las funcionalidades que estas brindan se vuelve difícil probar cualquier modificación que se realice. Esto trae consigo que se interrumpan los procesos de liberación del producto por calidad; deteniéndose la certificación y comercialización del mismo.

1.3 Descripción del objeto de estudio.

Para la realización de esta investigación es necesario profundizar en el estudio de las cámaras IP. Centrándose principalmente en los procesos de visualización y control de video desde cámaras IP. Logrando una mejor compresión del funcionamiento de estos equipos, como visualizan los videos y como acceden a sus funcionalidades. Identificándose características que puedan ser incluidas en el posterior desarrollo de la aplicación.

1.3.1 Descripción General de las Cámaras IP.

Las cámaras de red o cámaras IP, son dispositivos autónomos que cuentan con un servidor web de video incorporado, lo que les permite transmitir a través de redes IP**.** Los componentes principales que integran este tipo de cámaras de red incluyen un objetivo, un sensor de imagen, uno o más procesadores y memoria. Los procesadores se utilizan para la compresión, el análisis de vídeos y para realizar funciones de red. La memoria se utiliza para fines de almacenamiento del programa informático (*firmware*) de la cámara de red y para la grabación local de secuencias de vídeo (Axis, 2011). Las cámaras digitalizan la señal de vídeo con un codificador especializado que se conecta directamente a las redes IP y permite a los usuarios visualizar, almacenar y analizar el video capturado a través de un navegador estándar. Asimismo, es necesaria la existencia de un software instalado en el ordenador que permita a los operadores manipular sus funciones.

1.3.2 Visualización y control de videos desde cámaras IP.

Las cámaras IP pueden ser conectadas fácilmente a las redes y permiten actualizaciones en tiempo real del video de alta calidad. Poseen una interfaz de programación de aplicaciones (API) que permite a los usuarios la visualización simultánea de las cámaras, el control, la administración y por supuesto la reproducción de los videos que se hayan almacenado mediante grabación programada, o como consecuencia de alarmas.(Camino, 2009) Además, poseen un SDK[2] que les permite integrarse en cualquier aplicación, de forma fácil y flexible.

Los SDK están compuestos en su mayoría por bibliotecas que permiten la realización de distintas operaciones sobre las cámaras tales como la visualización de videos en vivo o grabaciones, configuración y movimiento de cámaras, así como la activación de video servidores.(Video Insight, 2012) Frecuentemente incluyen algunos códigos de ejemplos y notas técnicas de soporte u otra documentación para facilitar el trabajo con las cámaras. Estos SDK son específicos para cada fabricante y en algunas ocasiones para cada cámara IP. Para llevar a cabo sus funciones los SDK trabajan mediante el envío de comandos a través de direcciones URL, estos comandos son recibidos por el servidor que los procesa y ejecuta las operaciones solicitadas.

[2]**SDK: Kit de desarrollo de software.** conjunto de herramientas de desarrollo de software que le permite al programador crear aplicaciones para un sistema concreto

Capítulo 1: Fundamentación Teórica

1.4 Características del sistema Video Vigilancia Suria.

El producto Suria 1.0 se concibe sobre la base del desarrollo de un sistema de video vigilancia soportado sobre tecnología IP. Su base está formada por estaciones de monitoreo simultáneo y en tiempo real de los diferentes flujos de videos generados por cada una de las cámaras de seguridad. Cuenta además con servidores de almacenamiento y recuperación de los flujos de videos entre cada uno de los nodos de la red. Todo el funcionamiento es coordinado por un manejador de eventos y reglas que facilita la adaptabilidad del sistema en diferentes entornos y modos de funcionamiento. Integra varias aplicaciones, con el objetivo de facilitar la realización de múltiples funciones enfocadas a mejorar la vigilancia en las instituciones.

Módulo Gestor: Es el módulo fundamental de la aplicación, y todos los demás son agentes que se encargan de realizar tareas específicas. Todo el tráfico de información pasa por el Gestor facilitando el control y supervisión, permitiendo además la flexibilidad del sistema. Es el único que interactúa directamente con la Base de datos.

Módulo Visor: Es el módulo con el que interactúan los usuarios que permite visualizar los flujos de videos capturados por las cámaras. Pueden existir varias instancias de esta estación corriendo en los dominios físicos del sistema o ninguna. Tiene la capacidad de reflejar todo el aspecto organizativo con que se manejan las cámaras internamente, además de poder visualizarlas de manera independiente o colectiva (a manera de vistas). También permite manipularlas en la medida de las capacidades de cada una.

Algunas funcionalidades que ofrece:

- ✓ Visualizar las cámaras insertadas en el sistema organizadas según su disposición física.
- ✓ Insertar, editar y eliminar cualquier modelo o tipo de cámara del sistema.
- ✓ Insertar, editar y eliminar zonas de cámaras en el sistema.
- ✓ Visualizar videos obtenidos desde las cámaras y una vez visualizado un video podrá:
    1. Refrescar la visualización del video mostrado.
    2. Adicionar visualizadores.
    3. Ver modo “Pantalla completa” el video visualizado.
    4. Ver en modo “Visualizador Independiente” el video.
    5. Ajustar visualizador.
    6. Mostrar información de la cámara visualizada.

Módulo Grabador: Es el módulo encargado de almacenar los flujos de video obtenidos de las cámaras, bajo petición de un cliente determinado o por configuración, que puede ser por horarios determinados o por la ocurrencia

de algún evento. Pueden existir varias instancias de esta estación corriendo en los dominios físicos del sistema o ninguna.

Módulo Recuperador: Es el módulo encargado de recuperar los videos almacenados en los servidores, hacer búsquedas en estos, servir los resultados al usuario y darle la posibilidad de la reproducción de los mismos a diferentes niveles de velocidad.

Modelos de Cámaras que soporta Suria:

El producto Suria está concebido para que soporte un gran número de cámaras que varían en dependencia de su modelo o fabricante. Algunos de los modelos que actualmente soporta este sistema se encuentran en la siguiente tabla, aunque las más probadas son las Planet y las Axis:

Cámaras	Operaciones
Axis 211 network camera.	Hasta 20 conexiones simultáneas. Control de nivel del color, brillo y contraste. Realizar zoom. Rotar la imagen del video. Configurar cámara. Transmitir video. Envío de mensajes por SMTP. Capacidades de pantalla (hora, fecha, máscara de privacidad, texto o imagen.) Trasmisión en formatos M-JPEG y MPEG-4.
Axis 2460 Network DVR.	Control de nivel de color, brillo y contraste. Capacidades de pantalla (hora, fecha, máscara de privacidad, texto o imagen.) Compresión de imagen. Visualizar video. Transmitir video. Configurar cámara. Hasta 20 conexiones simultáneas. Trasmisión en formato JPEG-video.

Axis 214 PTZ.	Visualizar video. Transmitir video. Configurar cámara. Función de zoom 18x. Capacidades de pantalla (hora, fecha, máscara de privacidad, texto o imagen.). Hasta 20 conexiones simultáneas. Rotación de imagen 90 grados. Control de brillo y contraste. Trasmisión en formatos Motion JPEG y MPEG-4.
Panasonic BB_HCM381 network camera.	Visualizar video. Transmitir video. 30 accesos simultáneos. Control de brillo y contraste. Función de zoom42x.
Planet ICA-HCM100 IP camera.	Visualizar video. Transmitir video. Control de brillo y contraste. Trasmisión en formatos H.264 / MPEG-4 y M-JPEG.
Planet ICA-750 IP camera.	Visualizar video. Transmitir video. Control de brillo y contraste. Trasmisión en formatos H.264 / MPEG-4 y M-JPEG.
VIVOTEK 7131.	Visualizar video. Transmitir video. Trasmisión en formatos MPEG-4. 10 accesos simultáneos. Control de brillo y contraste.

Tabla 1: Modelos de Cámaras IP que soporta Suria.

1.5 Análisis de soluciones existentes.

Durante años la simulación ha jugado un papel significativo en programas de formación de importantes sectores de la economía. Los primeros simuladores se originaron mucho antes de la década de los 60 con el objetivo de reducir el nivel de error humano en las investigaciones. Actualmente la simulación está siendo utilizada con éxito para el desarrollo de una amplia gama de ciencias tales como la informática, la ingeniería, la industria, la física, entre otras.

1.5.1 Simuladores de cámaras existentes en Cuba.

En Cuba, la simulación también se ha destacado a través de los años en disímiles esferas de la economía, contando con herramientas propias para confeccionar simuladores, desarrollados en el Centro de Cibernética Aplicada a la Medicina (CECAM). Sin embargo, donde más resalta su utilización es en la industria militar, siendo el Centro de Investigación y Desarrollo de Simuladores (SIMPRO) de las Fuerzas Armadas Revolucionarias (FAR) uno de los primeros centros que se destacó por crear dispositivos de realidad virtual, los cuales dirige en su mayoría a la preparación de futuros oficiales en campos como el tiro de tanques y el vuelo en aeronaves. SIMPRO se ha destacado también por la construcción de simuladores destinados al aprendizaje de conducción de vehículos que se utilizan en escuelas de automovilismo. (Cáceres, 2011)

A pesar de la creciente evolución del desarrollo de software que experimenta el país, actualmente no existe un uso generalizado de los sistemas de vigilancias, ni un modelo de cámaras IP estándar que utilicen los mismos. La capacitación del personal de vigilancia para trabajar en estos entornos, se realiza con cámaras reales utilizando los softwares que por defecto traen las cámaras instaladas lo que limita la integración de otros modelos de cámaras.

1.5.2 Simuladores de cámaras existentes en el mundo.

A nivel internacional han sido desarrollados diferentes simuladores atendiendo a las necesidades específicas de los diferentes niveles económicos y a la toma de decisiones a nivel institucional.(Paniagua, 2004) Sin embargo en el ámbito de la video vigilancia solo se identificaron algunos simuladores destinados a simular cámaras IP físicamente, y cuyo propósito no va más allá de impresionar a los intrusos y consisten en instalaciones de cámaras falsas. Entre los simuladores que existen en el ámbito mundial se identificaron los siguientes:

Capítulo 1: Fundamentación Teórica

Simuladores Elro

Estos simuladores tienen el propósito de impresionar a los intrusos y no son más que una instalación de cámaras falsas, junto con las reales. Poseen un sensor de movimiento y una luz de activación que los hace parecer una verdadera cámara de seguridad. Estas falsificaciones se pueden mover, rotar e incluso hacer zoom y sonidos, son fáciles de mantener, y funcionan con baterías. Son ubicadas en zonas menos asequibles junto a las cámaras reales que se ubican en los lugares vulnerables. Por su bajo costo en el mercado, son muy usados en las instalaciones, para aparentar tener una mayor cantidad de cámaras, o en muchos casos para simular tener un sistema de seguridad lo cual desanima a cualquier intruso. (Elro, 2013)

Axis Network Camera Emulator

AxisCameraEmulator es un emulador de cámara de vídeo IP simple, registrado por SourceForge.net el 11 de febrero de 2008. Emula parcialmente algunas de las funcionalidades de la cámara de red Axis 223M, como la visualización y trasmisión de video, así como el uso de archivos JPEG. Además es capaz de soportar un pequeño subconjunto de los parámetros Axis VAPIX (específicamente los fotogramas por segundo y la resolución de los flujos de videos). Axis Camera Emulator es utilizado para probar el software de circuito cerrado de televisión. Se puede descargar gratis desde la página oficial de SourceForge.net y obtener una versión totalmente funcional. (Sourceforge.net, 2012)

Por no existir ningún otro simulador que incluyera características que fueran de interés a la aplicación que se desea desarrollar, se realizó un análisis de las API que brindan alguna de las cámaras IP usadas por Suria.

VAPIX

VAPIX es la interfaz de programación de aplicaciones (API) propia de Axis. Integra dentro de sí varias aplicaciones API para el acceso a funciones incorporadas al dispositivo que posibilitan la configuración, visualización y la trasmisión los flujos de videos, dentro de estas aplicaciones se encuentra RTSP[3] API (Axis Communications, 2012)

[3]**RTSP: (Real Time Streaming Protocol)** es un protocolo de control de flujos de medios proporcionados por un servidor de medios.

Capítulo 1: Fundamentación Teórica

RTSP API: Es una interfaz de programación de aplicaciones que proporciona comandos para el control de flujos de videos tales como la resolución, la compresión, la tasa de bits de vídeo y audio, así como los parámetros que controlan la apariencia de la imagen, incluyendo, por ejemplo, el texto y la configuración de la imagen de superposición. (Axis Communications, 2008) Algunos de estos comandos son:

- ✓ El comando PLAY que posibilita la reproducción del flujo de video.
- ✓ El comando PAUSE que posibilita la pausa en la trasmisión del flujo de video.

Además, RTSP API proporciona los comandos de control de flujo de video tales como:

- ✓ El comando RESOLUTION que posibilita cambiar la resolución del flujo de video que se está visualizando.
- ✓ El comando COMPRESSION que permite ajustar el nivel de compresión de la imagen con valores pertenecientes al intervalo de 1 a 100.
- ✓ El comando AUDIO que especifica si el audio estará disponible en la trasmisión tomando valores de 0 o 1 (0 = sin audio, 1 = audio).
- ✓ El comando VIDEO BIT RATE: Permite ajustar la tasa de bits de vídeo.
- ✓ El comando SETUP se usa para configurar el método de entrega de datos.
- ✓ La solicitud de TEARDOWN se utiliza para cerrar el suministro de datos desde el producto Axis.
- ✓ El comando SET_PARAMETER se utiliza para cambiar los parámetros de sesión.

RTSP API puede crear un túnel a través de HTTP, funcionalidad que posibilita el envió de comandos RTSP en la conexión POST y las respuestas sobre la conexión de GET. Tanto las solicitudes GET como las POST son aceptadas tanto en el puerto HTTP (por defecto 80) como en el puerto RTSP servidor (por defecto 554). Además RTSP API proporciona los parámetros para solicitar secuencias de videos con propiedades específicas y para establecer la apariencia de la imagen. Los parámetros se introducen en la URL RTSP que utiliza la siguiente sintaxis:

rtsp ://<servername>/axis-media/media.amp[?<parameter>=<value> [&<parameter>=<value>...]]

Algunos de los parámetros que se introducen en esta sentencia son:

- ✓ colorlevel: Ajusta el nivel de color o escala de grises. (0 escala de grises, 100 a todo color).
- ✓ color: Activar / desactivar el color. (0negroyblanco, 1 color).
- ✓ rotation: Girar la imagen en sentido horario. (0, 90, 180, 270).
- ✓ text: Mostrar /ocultar el texto. (0ocultar, 1 mostrar).

Capítulo 1: Fundamentación Teórica

VIVOTEK

VIVOTEK es un sistema de vigilancia de red, que permite controlar de forma local o remota la seguridad de propiedades en cualquier lugar y en cualquier momento. Posee una avanzada colección de desarrollo de software VIVOTEK SDK, que le proporciona una gestión centralizada de software, con grabación fiable, fácil administración del sistema, y una gran escalabilidad para diversas aplicaciones de vigilancia IP. Su avanzada tecnología de *codec* le permite visualizar, controlar y administrar fácilmente todas las cámaras de red usando cualquier navegador estándar o un software de administración de vigilancia desde cualquier PC en la red.(VIVOTEK., 2011)

VIVOTEK realiza el envío de comandos a través de la conexión POST y las respuestas sobre la conexión de GET a través de HTTP (por defecto 80) y el puerto RTSP (por defecto 554). Los parámetros que permiten solicitar funciones específicas y establecer la apariencia de la imagen se introducen en la URL que utiliza la siguiente sintaxis:

http://<servername>/cgi-bin/<subdir>[/<subdir>..]/<cgi>.<ext>[?<parameter>=<value>[&...]]

Algunos de los parámetros que se introducen en esta sentencia son:

- ✓ brightness: Ajuste el brillo de la imagen. (intervalo de -5 a 5).
- ✓ color: Activar / desactivar el color. (0negroyblanco, 1 color).
- ✓ saturation: Ajuste de la saturación de la imagen. (intervalo de -5 a 5).
- ✓ contrast: Ajuste el contraste de la imagen. (intervalo de -5 a 5).
- ✓ sharpness: Ajuste la nitidez de la imagen. (intervalo de -3 a 3).
- ✓ text: Mostrar /ocultar el texto. (0ocultar, 1 mostrar).

1.5.3 Conclusiones sobre las Soluciones Existentes.

Luego de una investigación detallada sobre los simuladores de cámaras a nivel nacional, se puede concluir que actualmente en Cuba a pesar de existir empresas dedicadas al desarrollo de software y a la producción de simuladores, no existe una aplicación que simule las funcionalidades de una cámara IP por la insipiente inserción que tiene el país en este campo. A nivel internacional tampoco fueron encontradas aplicaciones simuladoras que sirvieran de base para el desarrollo de la investigación. Pese a que se identificaron simuladores de cámaras IP como los Elro, por la particularidad de que consisten en instalaciones de equipos falsos, estos tipos de simuladores no son los idóneos para el desarrollo de esta

investigación. Por otra parte el análisis del Axis Camera Emulator permitió identificar funcionalidades propias de la cámara Axis 223M tales como: la visualización y trasmisión de los flujos de videos.

La búsqueda y análisis de los softwares que emplean las cámaras IP, centrándose principalmente en aquellos modelos que son soportados por el sistema Video Vigilancia Suria, ofreció características que pueden incorporase a la aplicación que se desea desarrollar tales como: la utilización de comandos para invocar las distintas funcionalidades sobre el flujo de video obtenido de las cámaras IP y la utilización de URL para el envío de los comandos. Además, permitió identificar un conjunto de operaciones comunes que realizan las cámaras IP, características de las cuales puede nutrirse el simulador que se desea desarrollar, tales como:

- ✓ Rotación de cámara, horizontal y verticalmente.
- ✓ Transmisión de video.
- ✓ Ajuste de los colores en el video.
- ✓ Ajuste del brillo y contraste de la imagen en el video que se trasmite.
- ✓ Superposición de texto sobre el video que se trasmite.
- ✓ Funcionalidad de zoom en el video que se trasmite**.**

1.6 Herramientas y tecnologías para el desarrollo del software.

Teniendo en cuenta que la aplicación es de interés exclusivo del proyecto Video Vigilancia Suria se empleará para su desarrollo las herramientas y las tecnologías que se definen en este proyecto, con el objetivo de lograr una fácil integración con sus módulos de ser necesario.

1.6.1 Metodologías de Desarrollo.

Las metodologías de desarrollo de software son el marco de trabajo que colecciona un conjunto de pasos y procedimientos que se deben seguir para organizar, controlar y planear el proceso de desarrollo de un software.(Pressman, 2010) Hoy en día existen significativas metodologías que brindan soluciones a los desarrolladores. Su uso posee una enorme importancia a la hora de gestionar de forma eficiente un proyecto.

Proceso Unificado de Desarrollo de Software (RUP).

El Proceso Unificado de Desarrollo (RUP) es una metodología de desarrollo de software que está basado en componentes e interfaces bien definidas, y junto con el Lenguaje Unificado de Modelado (UML),

constituye la metodología estándar más utilizada para el análisis, implementación y documentación de sistemas orientados a objetos. (Ecured, 2012)

Es un proceso que puede especializarse para una gran variedad de sistemas de software, en diferentes áreas de aplicación, diferentes tipos de organizaciones, diferentes niveles de aptitud y diferentes tamaños de proyecto. Su objetivo es asegurar la producción de software de alta calidad, que satisfaga los requerimientos de los usuarios finales respetando el cronograma y el presupuesto. Es guiado por casos de uso lo cual guía el proceso de desarrollo ya que los modelos que se obtienen, como resultado de los diferentes flujos de trabajo, representan la realización de los casos de uso. (Ecured, 2012)

Principales características:

- ✓ Forma disciplinada de asignar tareas y responsabilidades entre los miembros del equipo de desarrollo, lo que facilita el trabajo, determinado quién hace determinadas actividades, cuándo lo hace y cómo lo hace.
- ✓ Desarrollo iterativo lo cual posibilita que en cada flujo de trabajo se obtenga un producto con un determinado nivel, y que posibilite según los cambios que se produzcan seguir trabajando sobre esa nueva versión terminada.
- ✓ Administración de requisitos, lo cual facilita una mejor comprensión y control de los requerimientos de software.
- ✓ Modelado visual del software utilizando UML garantizando la construcción de la arquitectura del software posibilitando organizar el desarrollo de la aplicación, plantear la reutilización del software, hacerlo evolucionar y reducir los riesgos.
- ✓ Verificación de la calidad del software por medio de las pruebas de caja negra las cuales se concentran en los requisitos funcionales del sistema y comprueban si funcionan adecuadamente.

1.6.2 Lenguaje de Modelado Unificado (UML).

EL Lenguaje de Modelado Unificado (UML) es un lenguaje gráfico para visualizar, especificar, construir y documentar un sistema de software. UML ofrece un estándar para describir un modelo del sistema, incluyendo aspectos conceptuales tales como procesos de negocios y funciones del sistema, y aspectos concretos como expresiones de lenguajes de programación, esquemas de bases de datos y componentes de software reutilizables. Además es un método de modelado, lo cual aporta las siguientes ventajas:(Ecured, 2012)

- ✓ Permite especificar, visualizar y documentar los artefactos que son generados a lo largo del ciclo de desarrollo de la aplicación, fundamentalmente en las fases de análisis, diseño e implementación.
- ✓ Se puede aplicar en una gran variedad de formas para dar soporte a una metodología de desarrollo de software, lo cual posibilita que se pueda usar como lenguaje de modelado en la metodología RUP.

1.6.3 Herramientas CASE[4].

Las Herramientas CASE son un conjunto de programas y ayudas que dan asistencia a los analistas, ingenieros de software y desarrolladores, durante todos los pasos del ciclo de vida de desarrollo de un software. Las herramientas CASE también permiten a los analistas tener más tiempo para el análisis y diseño y minimizar el tiempo para codificar y probar.(INEI, 1999)

Visual Paradigm 8.0.

Con el fin de automatizar los aspectos claves del proceso de desarrollo del Simulador de Cámaras IP para el sistema Video Vigilancia Suria, se hace necesaria la utilización de las herramientas CASE. Para el desarrollo de esta investigación se utilizará la herramienta Visual Paradigm 8.0 por sus características de:

- ✓ Disponibilidad en múltiples plataformas, lo cual facilita que la realización de diagramas de modelado no dependan del sistema operativo que se tenga instalado en el equipo de trabajo.
- ✓ Uso de un lenguaje estándar que es común a todo el equipo de desarrollo facilitando la comunicación, permitiendo la realización de diagramas y modelos durante el proceso de desarrollo.
- ✓ Licencia: gratuita y comercial, fácil de instalar y actualizar.

1.6.4 Lenguaje de Programación y Biblioteca.

Un lenguaje de programación es un conjunto de sintaxis y reglas semánticas que definen los programas del computador. Es una técnica estándar de comunicación para entregarle instrucciones al computador. Un lenguaje le da la capacidad al programador de especificarle al computador, qué tipo de datos actúan y que acciones tomar bajo una variada gama de circunstancias, utilizando un lenguaje relativamente próximo al lenguaje humano. (Veranes., 2012)

[4] **CASE**: Ingeniería de Software Asistida por Computación

Capítulo 1: Fundamentación Teórica

C++.

C++ es una versión ampliada del lenguaje C incluyendo además algunas mejoras considerables como el soporte de la programación orientada a objetos. Se utilizará este lenguaje por las siguientes características:

- ✓ Se puede realizar una aplicación que se puede comercializar sin la necesidad de pagar una licencia, lo cual posibilita la realización de la aplicación en su totalidad sin tener que pagar por la utilización de este lenguaje.
- ✓ Presenta la facilidad de que es portable, propiciando que el código del simulador creado en este lenguaje se puede compilar en cualquier sistema operativo.
- ✓ Proporciona facilidades para utilizar código o bibliotecas existentes además de ser uno de los lenguajes más rápidos en cuanto a ejecución, ventaja que puede ser aprovechada para el procesamiento de los flujos de video.

Biblioteca GStreamer 10.3.

Las bibliotecas son conjuntos de subprogramas que contienen datos y códigos que brindan servicios a otras aplicaciones y ayudan a los programadores en el desarrollo de software. Estas no necesitan ser modificadas y el código que contienen se añade al programa principal cuando se genera.

GStreamer es un framework (marco de trabajo) multimedia de software libre multiplataforma escrito en el lenguaje de programación C, usando la biblioteca GObject que posibilita a los programadores la oportunidad de crear aplicaciones de video. Utilizando GStreamer se logra garantizar funcionalidades básicas y útiles del sistema, como el procesamiento del video que incluye el ajuste de los colores del mismo, realizar función de zoom así como rotar vertical u horizontalmente el video y que además posibilite la trasmisión del mismo en directo.

Biblioteca VLC 2.04.

VLC es un reproductor y también un *framework (marco de trabajo)* multimedia del proyecto VideoLAN. Es además un software libre distribuido bajo la licencia GPL que soporta formatos de audio (MP3, QDM2/QDMC, RealAudio, Speex, Screamtracker 3/S3M, TTA, Vorbis y WMA.) y video (H.263, H.264/MPEG-4 AVC, MJPEG, MPEG-1, MPEG-2, MPEG-4 Part 2 y WMV) , además de DVD, VCD y varios protocolos de streaming. Tiene la capacidad de transmitir datos streaming a través de redes y convertir archivos multimedia en formatos distintos al original.(Ecured, 2012)

La utilización de esta biblioteca garantiza la correcta trasmisión de los flujos de video en el simulador independientemente del sistema operativo en que este corriendo, puesto que posee versiones para GNU/Linux, Microsoft Windows, Mac OS X, entre otros.

1.6.5 Entorno de Desarrollo Integrado.

Un Entorno de Desarrollo Integrado (IDE) es un programa que está compuesto por un conjunto de herramientas para un programador. Facilita un marco de trabajo amigable para una gran cantidad de lenguajes de programación tales como C++, Java, C#, logrando utilizarse en el mismo uno o varios lenguajes de programación.(Veranes., 2012)

Qt Creator 2.6.

En esta investigación se utilizará el IDE Qt Creator 2.6 como plataforma de desarrollo del Simulador de Cámara IP pues proporciona amplios beneficios para los desarrolladores debido a que existe abundante documentación sobre cómo trabajar en su entorno que ayuda a los nuevos usuarios de Qt a aprender y comenzara desarrollar rápidamente. Además era necesaria la utilización de un IDE que aprovechara toda la potencialidad del lenguaje C++ y que ofreciera características tales como:

- ✓ Las bibliotecas Qt: clases escritas en C++ que facilitan el desarrollo.
  - QtDesigner: para diseñar formularios visualmente.
  - QtAssistant: acceso rápido a la documentación.
  - QtLinguist: traducción rápida de programas.
  - Qmake: simplifica el proceso de construcción de proyectos en las diferentes plataformas soportadas.
- ✓ Posee un avanzado editor de código C++, lo cual facilita la programación de la aplicación ofreciendo completado de código.
- ✓ Posee una GUI integrada y diseñador de formularios.

1.7 XML[5]

XML (Lenguaje de Marcado Extensible) es un lenguaje de marcado sencillo similar al HTML[6]. Su objetivo es facilitar la representación, almacenamiento y trasmisión de información varia por parte de aplicaciones informáticas, computadoras y medios de comunicación digital en general. Para el desarrollo de la

[5] **XML**: Extensible Markup Language o Lenguaje de Marcado Extensible.
[6] **HTML**: Hipertext Markup Language o Lenguaje de Marcado de Hipertexto.

investigación es necesaria la utilización de XML con el propósito de garantizar el almacenamiento de la información de los distintos perfiles de cámaras IP que soporta el sistema. Además este lenguaje proporciona la ventaja de que si se desea usar un documento creado en XML, es sencillo entender su estructura y procesarla. (Ecured, 2013)

Conclusiones

En este capítulo se precisaron una serie de conceptos relacionados con la problemática planteada, estudiándose el desarrollo y la evolución de los simuladores, tanto a nivel nacional como internacional, analizándose detalladamente sus ventajas, y características generales.

- ✓ Se caracterizaron las cámaras IP que soporta actualmente el sistema Suria, centrándose principalmente en los procesos de visualización y control de video que estas utilizan, propiciando una descripción amplia del objeto de estudio, garantizando un mejor entendimiento sobre el funcionamiento de estos dispositivos y permitiendo identificar funcionalidades a incorporaren el simulador que se desea desarrollar.
- ✓ Se analizaron soluciones existentes tanto a nivel nacional como internacional, determinándose que no existe un simulador de este tipo, por lo cual fue necesario centrar el estudio en los software que incluyen las cámaras IP obteniendo de estos características de funcionamiento que sirven de guía para el desarrollo de la investigación.
- ✓ Se caracterizaron las herramientas y tecnologías a utilizar en el desarrollo de la aplicación enunciándose de cada una de ellas las principales características y las facilidades que ofrecen al equipo de desarrollo, favoreciendo posibles integraciones entre el simulador y los módulos del proyecto Video Vigilancia y garantizando futuras actualizaciones del sistema.

Capítulo 2: Características del Sistema

Introducción

En este capítulo se definen temas relacionados con el análisis de la aplicación que se desea desarrollar, atendiendo a la información que se maneja. Se realiza un estudio de la situación problémica con el objetivo de facilitar la comprensión de los principales conceptos que se utilizan para la descripción de la solución propuesta, utilizándose para ello un modelo de dominio. Se describe el flujo del diagrama del dominio y la descripción de las clases que lo componen. Se identifican los requisitos funcionales y no funcionales agrupándose en casos de usos y presentando una descripción de cada uno de ellos.

2.1 Modelo de dominio.

Al no existir un negocio real, no es posible precisar la estructura de los procesos de negocio que intervienen en el dominio del problema, por ello se realizó un modelo de dominio. El modelo de dominio permite de manera visual mostrar al usuario los principales conceptos asociados a la situación problemática planteada. Es una representación de las clases conceptuales del mundo real, no de componentes software y emplea un glosario de términos para lograr una mejor concepción de los conceptos asociados (Loja, 2008)

2.1.1 Descripción del flujo del diagrama del modelo de dominio.

El sistema de Video Vigilancia Suria integra varias aplicaciones o módulos (Gestor, Visor, Grabador y Recuperador) que pueden tener varias instancia corriendo simultáneamente en los dominios físicos del sistema. Entre estos, el módulo Visor es el encargado de visualizar los flujos de video provenientes de las cámaras IP soportadas por el sistema. Estas cámaras poseen un conjunto de operaciones que varían según su tipo o modelo; las mismas, pueden ser accedidas y manipuladas desde la interfaz del Visor por medio de plugins según las funcionalidades que posean.

2.1.2 Diagrama de Modelo de dominio.

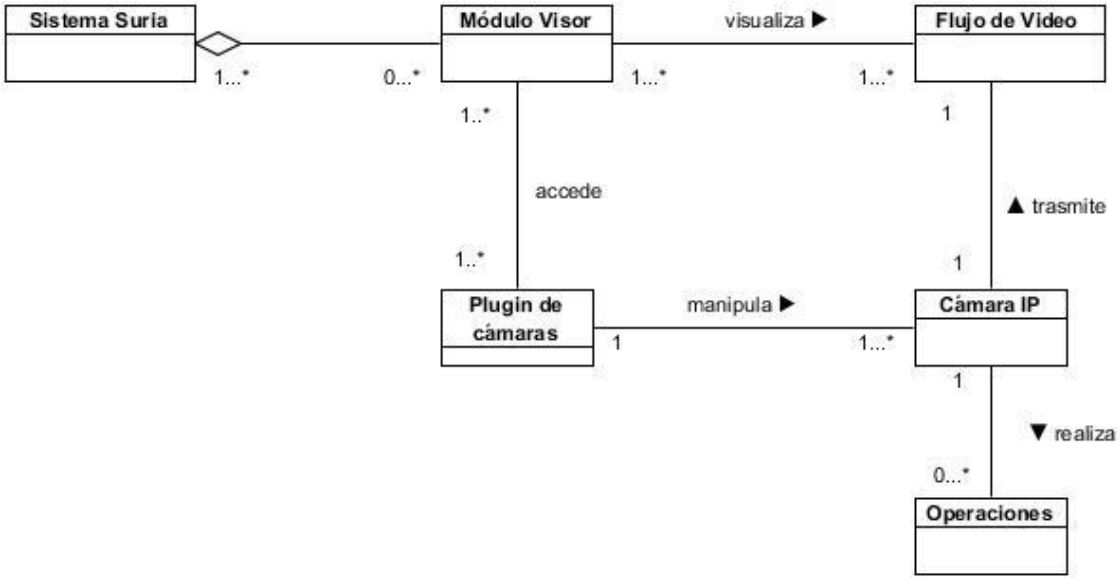

Fig. 1: Modelo de dominio Simulador de Cámaras IP para el sistema Video Vigilancia Suria.

2.1.3 Descripción de las clases.

Sistema Suria: sistema para la video vigilancia compuesto por los módulos: Gestor, Grabador, Visor y Recuperador. Soportado por tecnología IP. Concebido con el objetivo de facilitar la realización de múltiples funciones sobre cámaras IP, enfocadas a mejorar y fortalecer la vigilancia en las instituciones donde sea desplegado.

Módulo Visor: módulo con el que interactúan los usuarios, que permite la visualización de los flujos de videos generados por las cámaras. Tiene la capacidad de reflejar todo el aspecto organizativo con que se manejan las cámaras internamente, además de poder visualizarlas de manera independiente o colectiva (a manera de vistas). También permite manipularlas en medida de las capacidades de cada una.

Flujos de Video: representan la secuencia de imágenes que transmiten las cámaras IP. Se pueden visualizar desde el Módulo Visor.

Plugin de cámaras: Son clases que permiten al sistema Video Vigilancia Suria integrar distintos modelos de cámaras, permitiéndole al sistema manipularlas en la medida de sus capacidades.

Cámaras IP: videocámaras diseñadas para enviar los flujos de video obtenidos de un área en específico a través de la red y que poseen funcionalidades que varían en dependencia del modelo.

Operaciones: representa todas las operaciones que son posibles realizar con una cámara IP en dependencia de su tipo o modelo, ejemplo ajustar brillo, ajustar contraste, transmitir video, entre otras.

2.2 Especificación de los requisitos de software.

La especificación de requisitos software no es más que la descripción completa del comportamiento del sistema que se desea desarrollar. La misma incluye los requisitos funcionales que describen todas las características requeridas y los requisitos no funcionales o complementarios que son aquellos que imponen restricciones en el diseño o la implementación.

2.2.1 Requisitos Funcionales.

Los requisitos funcionales son declaraciones de servicios que el sistema debe proporcionar, define la manera en que éste debe reaccionar a determinadas entradas y cómo se debe comportar en situaciones particulares (Pressman, 2010). Estos requisitos son las características fundamentales del sistema y expresan la capacidad de acción del mismo.

Nota: Se usa el prefijo RF en su nomenclatura.

RF1Transmitir video.

Descripción: El sistema debe permitir al administrador comenzar a trasmitir un flujo de video desde una cámara en la red simulando la trasmisión que realizan las cámaras IP.

Entrada: Dirección URL del video que se desea trasmitir (Cadena de caracteres).

Salida: Trasmisión de flujo de video.

RF2 Detener trasmisión.

Descripción: El sistema debe permitirle al administrador detener la trasmisión de video desde una cámara en el momento que lo desee.

Entrada: Nombre de la cámara que desea detener (Cadena de caracteres).

Salida: Fin de la trasmisión del flujo de video desde el simulador.

RF3 Simular rotación de cámara de forma horizontal, en el video que se trasmite.

Descripción: El sistema debe permitir simular el movimiento de rotación de las cámaras de forma horizontal en el flujo de video que se trasmite.

Entrada: Comando de rotación horizontal (Cadena de caracteres).
Salida: Transmisión de flujo de video procesado.

RF4 Simular rotación de cámara de forma vertical, en el video que se trasmite.
Descripción: El sistema debe permitir simular el movimiento de rotación de las cámaras de forma vertical en el flujo de video que se trasmite.
Entrada: Comando de rotación vertical (Cadena de caracteres).
Salida: Transmisión de flujo de video procesado.

RF5 Simular ajuste de contraste en el video que se trasmite.
Descripción: El sistema debe permitirle al administrador según sus preferencias, ajustar el contraste en el video que se trasmite.
Entrada: Comando para ajustar contraste (Cadena de caracteres).
Salida: Transmisión de flujo de video procesado.

RF6 Simular ajuste del brillo en el video que se trasmite.
Descripción: El sistema debe permitirle al administrador según sus preferencias, ajustar el brillo en el video que se trasmite.
Entrada: Comando para ajustar brillo (Cadena de caracteres).
Salida: Transmisión de flujo de video procesado.

RF7 Simular ajuste de color en el video que se trasmite.
Descripción: El sistema debe permitirle al administrador según sus preferencias, ajustar el color en el video que se trasmite.
Entrada: Comando para ajustar color (Cadena de caracteres).
Salida: Transmisión de flujo de video procesado.

RF8 Simular ajuste de saturación en el video que se trasmite.
Descripción: El sistema debe permitirle al administrador según sus preferencias, ajustar la saturación en el video que se trasmite.
Entrada: Comando para ajustar saturación (Cadena de caracteres).
Salida: Transmisión de flujo de video procesado.

RF9 Simular zoom en el video que se trasmite.

Descripción: El sistema debe permitirle al administrador según sus preferencias, realizar la funcionalidad de zoom sobre el video que se trasmite.

Entrada: Comando para realizar zoom (Cadena de caracteres).

Salida: Transmisión de flujo de video procesado.

RF10 Mostrar última operación realizada en el video que se trasmite.

Descripción: El sistema debe permitir mostrar la última operación realizada en el video que se trasmite.

Entrada: Ultima operación realizada (Cadena de caracteres).

Salida: Transmisión de flujo de video procesado.

RF11 Agregar nuevo perfil de cámara IP al sistema.

Descripción: El sistema debe permitir al administrador agregar nuevos perfiles de cámara IP.

Entrada: Protocolo de trasmisión (RTSP o HTTP), puerto de trasmisión, resolución de la cámara (ancho y alto), fotogramas por segundo y formato de trasmisión (MPEG2, H264-TS, H264-MP4) y la URL de trasmisión de esa cámara nueva a agregar. (Todos los parámetros son Cadenas de caracteres)

Salida: Archivo XML modificado según los parámetros insertados.

RF12 Editar perfil de cámara IP.

Descripción: El sistema debe permitir al administrador modificar un perfil de cámara IP que el simulador tenga almacenado en su archivo XML.

Entrada: Protocolo de trasmisión (RTSP o HTTP), puerto de trasmisión, resolución de la cámara (ancho y alto), fotogramas por segundo y formato de trasmisión (MPEG2, H264-TS, H264-MP4) y la URL de trasmisión de esa cámara que se desea modificar. (Todos los parámetros son Cadenas de caracteres)

Salida: Archivo XML actualizado según los parámetros modificados.

RF13 Eliminar perfil de cámara IP.

Descripción: El sistema debe permitir al administrador eliminar un perfil de cámara IP que el simulador tenga almacenado en su archivo XML.

Entrada: Nombre de la cámara que se desea eliminar. (Cadenas de caracteres)

Salida: Se elimina la cámara del archivo XML.

2.2.2 Requisitos No Funcionales.

Los requisitos no funcionales son las restricciones de los servicios o funciones ofrecidas por el sistema, restringen el espacio de posibles soluciones (Pressman, 2010). Estos requisitos son los que se centran fundamentalmente en las características del sistema y no en los rasgos particulares del mismo.

Nota: Se usa el prefijo RnF en su nomenclatura.

Usabilidad

RnF 1. Tipo de usuario Final: El sistema debe poder ser usado por los especialistas del equipo de desarrollo del proyecto Video Vigilancia.

RnF 2. Tipo de Aplicación Informática: El simulador de cámaras IP para el proyecto Video Vigilancia será una aplicación de escritorio.

RnF 3. Finalidad: Proveer al proyecto de Video Vigilancia de un simulador de cámaras IP garantizando un entorno de pruebas favorable, que permita comprobar todas las funcionalidades referentes a la manipulación de cámaras IP sin necesidad de adquirirlas previamente.

RnF 4. Ambiente: Para el funcionamiento del sistema se debe requerir como mínimo del siguiente equipamiento (Estas prestaciones permiten simular 5 flujos en trasmisión simultánea:

Estación de Procesamiento

Hardware: 512 Mb de memoria RAM, pero es recomendado 1 GB de memoria RAM; además de un CPU de dos núcleos a 2.4 GHz

Software: Sistema Operativo: Windows o Linux.

Biblioteca GStreaming 10.3.

Biblioteca VLC 2.04.

Confiabilidad

RnF 5. Si se interrumpe la energía o la red en la estación de procesamiento donde está corriendo el Simulador de Cámaras IP, el sistema Suria, no debe obtener el flujo de video solicitado, parecido a lo que ocurre cuando se interrumpe la conexión con las cámaras. Una vez reanudada la energía en la estación debe ejecutarse manualmente el sistema para que inicie sus funcionalidades.

RnF 6. El sistema debe garantizar un tratamiento adecuado cuando ocurra una excepción, mostrando mensajes al administrador del sistema informando las causas del error.

Eficiencia

RnF 7. El tiempo de respuesta por transacción: promedio 2 segundos, máximo 4segundos (probado para 20 conexiones simultáneas).

RnF 8. Capacidad: el sistema debe permitir más de 20 conexiones simultáneas al simulador.

Restricciones del diseño

RnF 9. El lenguaje de programación a utilizar en la implementación del componente será C++, con el framework (marco de trabajo) de desarrollo Qt Creator v2.6.

RnF 10. Para el correcto funcionamiento del sistema las bibliotecas que se deben utilizar serán GStreamer10.3 y VLC 2.04.

RnF 11. El sistema se implementará siguiendo una arquitectura en capas específicamente en 3 capas.

Interfaz

RnF 12. Interfaces de usuario: Las interfaces gráficas implementadas por el sistema deben concebirse con un ambiente sencillo para el usuario.

2.3 Descripción del sistema.

Luego de identificados los requisitos funcionales del simulador de cámaras IP para el sistema Suria, se agruparon en casos de usos (CU), los cuales se representan el diagrama de casos de uso del sistema. Cada uno de los casos de usos identificados engloba un conjunto de acciones, las cuales son inicializadas por los actores del sistema.

2.3.1 Definición de los actores.

Un actor del sistema no es más que un conjunto de roles que los usuarios desempeñan cuando interactúan con los casos de uso. Los actores identificados para el simulador de cámaras IP son el administrador y el sistema Suria.

Actor	Descripción
Administrador	Persona encargada cargar y crear los perfiles de cámaras IP, así como ejecutar funciones sobre el flujo de video que se trasmite.
Suria	Es que se comunica con el simulador de cámaras IP a través del envío de peticiones HTTP para ordenar el procesamiento sobre el flujo de video que se trasmite.

Tabla 2: Actores del sistema.

2.3.2 Listado de los Casos de Uso.

De acuerdo a los RF definidos anteriormente, se agruparon en los siguientes casos de usos:

CU Trasmitir video.

RF1 Transmitir video.

CU Detener trasmisión de video.

RF2 Detener trasmisión de video

CU Realizar procesamiento.

RF3 Simular rotación de cámara de forma horizontal, en el video que se trasmite.

RF4 Simular rotación de cámara de forma vertical, en el video que se trasmite.

RF5 Simular ajuste de contraste en el video que se trasmite.

RF6 Simular ajuste del brillo en el video que se trasmite.

RF7 Simular ajuste de color en el video que se trasmite.

RF8 Simular ajuste de saturación en el video que se trasmite.

RF9 Mostrar última operación realizada en el video que se trasmite.

RF10 Simular zoom en el video que se trasmite.

CU Gestionar perfil de cámara IP.

RF11 Agregar nuevo perfil de cámara IP al sistema.

RF12 Editar perfil de cámara IP.

RF13 Eliminar perfil de cámara IP.

2.3.3 Diagrama de Casos de Uso.

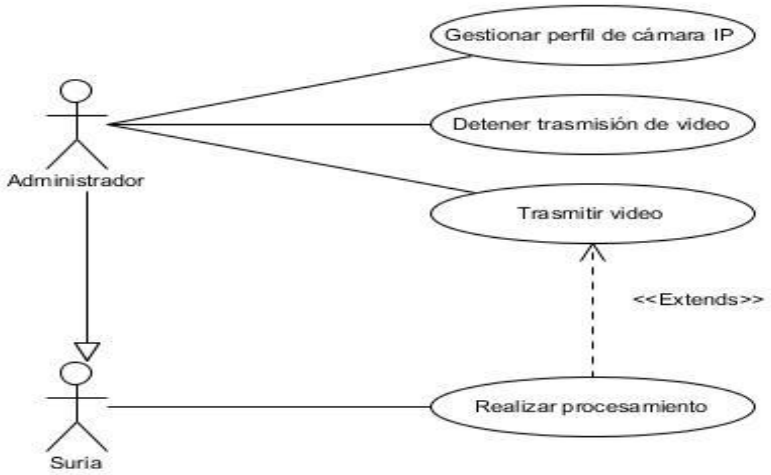

Fig. 2: Diagrama de CU Simulador de Cámaras IP para el sistema Video Vigilancia Suria.

2.3.4 Especificación de los Casos de Uso del Sistema.

En el presente epígrafe se presenta la descripción textual de los CU Realizar procesamiento y Trasmitir video por ser considerados arquitectónicamente significativos. Las descripciones de los CU Gestionar perfil de cámara IP y Detener trasmisión de video pueden encontrarse en el Anexo 2 del presente trabajo.

CU	Realizar procesamiento.
Objetivo	El objetivo de este CU es permitir la realización de procesamiento sobre el video que está trasmitiendo desde el simulador.
Actores	Suria: (Inicia) Procesamiento del flujo de video que se trasmite.
Resumen	El caso de uso inicia cuando el sistema Video Vigilancia Suria se comunica con el simulador a través del envió de comandos en direcciones URLs para solicitar la visualización del mismo o cuando el Administrador inicia la trasmisión del video desde la interfaz del simulador. Una vez iniciado permite realizar acciones relacionadas con el simulador y el video que se visualiza, tales como ajustes de brillo, contraste, color y saturación.
Complejidad	Media.
Prioridad	Alta.
Precondiciones	Simulación de trasmisión activa para la cámara que se desea procesar.

Postcondiciones	Trasmisión de flujo de video procesado según operación solicitada. (Brillo, contraste, color, saturación, rotación horizontal o vertical y zoom).	
Flujo de eventos		
Flujo básico: Realizar procesamiento.		
	Actor	**Sistema**
1.	Realiza una de las siguientes acciones por medio de peticiones HTTP: ✓ Rotación horizontal. ✓ Rotación vertical. Ver Sección 1:"Rotación vertical". ✓ Ajustar contraste. Ver Sección 2:" Ajustar contraste". ✓ Ajustar brillo. Ver Sección 3:" Ajustar brillo". ✓ Ajustar color. Ver Sección 4:" Ajustar color". ✓ Ajustar saturación. Ver Sección 5:" Ajustar saturación". ✓ Hacer zoom. Ver Sección 6:" Hacer zoom". **Observación:** Las URLs para las peticiones HTTP tienen la siguiente sintaxis. http://ip:puerto/URLdecomandos?variable=valor	
2.	Invoca la operación de Rotación horizontal.	
3.		Procesa el video modificando las propiedades del plugin videocrop de la biblioteca GStreamming teniendo como entrada la dirección en que se va a mover la cámara (izquierda o derecha). En este caso por defecto

		se mueve 10 px.
4.		Guarda en una variable el nombre de la operación realizada y ejecuta una función para mostrar en la trasmisión el valor de la variable. Termina el caso de uso.
Flujos alternos		
2a. La trasmisión del video se encuentra en los límites horizontales.		
	Actor	**Sistema**
2a.		Mantiene la trasmisión en el límite del video. Termina el caso de uso. **Observación**: Las cámaras IP normalmente cuando llegan a los límites, no visualizan los efectos de la rotación.
Sección 1: "Rotación vertical**"**		
Flujo básico: Rotación vertical.		
	Actor	**Sistema**
2.	Invoca la operación de Rotación vertical.	
3.		Procesa el video modificando las propiedades del plugin videocrop de la biblioteca GStreamming teniendo como entrada la dirección en que se va a mover la cámara (arriba o abajo). En este caso por defecto se mueve 10 px. Regresa al paso 4 del flujo básico: Realizar procesamiento.
Flujos alternos		
3a. La trasmisión del video se encuentra en los límites verticales.		
	Actor	**Sistema**
3a.		Mantiene la trasmisión en el límite del video. Termina el caso de uso.

Sección 2: "Ajustar contraste**"**		
Flujo básico: Ajustar contraste		
	Actor	**Sistema**
2.	Invoca la operación de Ajustar contraste.	
3.		Procesa el video modificando las propiedades del plugin videobalance de la biblioteca GStreamming teniendo como entrada el número de contraste de 0 a 2. Regresa al paso 4 del flujo básico: Realizar procesamiento.
Flujos alternos		
3a. Alcanza uno de los valores límites de contraste.		
	Actor	**Sistema**
3a.		Trasmite el video en el límite de contraste que se encuentra. Termina el caso de uso.
Sección 3: "Ajustar brillo**"**		
Flujo básico: Ajustar brillo		
	Actor	**Sistema**
2.	Invoca la operación de Ajustar brillo.	
3.		Procesa el video modificando las propiedades del plugin videobalance de la biblioteca GStreamming teniendo como entrada el número de brillo de -1 a 1. Regresa al paso 4 del flujo básico: Realizar procesamiento.
Flujos alternos		
3a. Alcanza uno de los valores límites de brillo.		
	Actor	**Sistema**
3a.		Trasmite el video en el límite de brillo que se encuentra. Termina el caso de uso.
Sección 4: "Ajustar color**"**		
Flujo básico: Ajustar color		

	Actor	**Sistema**
2.	Invoca la operación de Ajustar color.	
3.		Procesa el video modificando las propiedades del plugin videobalance de la biblioteca GStreamming teniendo como entrada el número de color de -1 a 1. Regresa al paso 4 del flujo básico: Realizar procesamiento.
Flujos alternos		
3a. Alcanza uno de los valores límites de color.		
	Actor	**Sistema**
3a.		Trasmite el video en el límite de color que se encuentra. Termina el caso de uso.
Sección 5: "Ajustar saturación **"**		
Flujo básico: Ajustar saturación		
	Actor	**Sistema**
2.	Invoca la operación de Ajustar saturación.	
3.		Procesa el video modificando las propiedades del plugin videobalance de la biblioteca GStreamming teniendo como entrada el número de color de 0 a 2. Regresa al paso 4 del flujo básico: Realizar procesamiento.
Flujos alternos		
3a. Alcanza uno de los valores límites de saturación.		
	Actor	**Sistema**
3a.		Trasmite el video en el límite de saturación que se encuentra. Termina el caso de uso.
Sección 6: "Hacer zoom**"**		
Flujo básico: Hacer zoom		
	Actor	**Sistema**
2.	Invoca la operación de Hacer zoom.	

3.		Procesa el video modificando las propiedades del plugin videobox de la biblioteca GStreamming teniendo como entrada el número de zoom de 0 a 100 en cualquier dirección. Regresa al paso 4 del flujo básico: Realizar procesamiento.
Flujos alternos		
3a. Alcanza uno de los valores límites de zoom.		
	Actor	**Sistema**
3a.		Trasmite el video en el límite de zoom que se encuentra. Termina el caso de uso.

Relaciones	**CU Incluidos**	No procede.
	CU Extendidos	No procede.
RnF	RnF 5, RnF 6, RnF 7, RnF 8, RnF 9, RnF 10, RnF 11.	
Asuntos pendientes	No procede.	

Tabla 3: Descripción del CU Realizar Procesamiento.

CU	Trasmitir video.
Objetivo	El objetivo de este CU es permitir el inicio de la trasmisión del video desde el simulador.
Actores	Administrador.
Resumen	El caso de uso inicia cuando el Administrador selecciona el perfil de cámara IP que desea simular e inicia la trasmisión. Una vez iniciado comienza la trasmisión del flujo de video por los protocolos RTSP o HTTP en dependencia de la cámara IP que se esté simulando, una vez finalizado cesa la trasmisión de video.
Complejidad	Baja.
Prioridad	Alta.
Precondiciones	Perfil de cámara IP creado.

<table>
<tr><td colspan="2"></td><td colspan="2">Perfil seleccionado.</td></tr>
<tr><td colspan="2">Postcondiciones</td><td colspan="2">Trasmisión de flujo de video por el puerto UDP y dirección IP local de la PC donde se encuentra corriendo el simulador.</td></tr>
<tr><td colspan="4">Flujo de eventos</td></tr>
<tr><td colspan="4">Flujo básico: Trasmitir video.</td></tr>
<tr><td></td><td colspan="2">Actor</td><td>Sistema</td></tr>
<tr><td>1.</td><td colspan="2">Selecciona el perfil de la cámara IP que desea simular y selecciona la opción iniciar trasmisión (A).</td><td></td></tr>
<tr><td>2.</td><td colspan="2"></td><td>Recibe la dirección del video a trasmitir.</td></tr>
<tr><td>3.</td><td colspan="2"></td><td>Configura los plugin videobox, videocrop, videobalance, para realizar la trasmisión permitiendo procesar la misma en tiempo de ejecución.</td></tr>
<tr><td>4.</td><td colspan="2"></td><td>Comienza a trasmitir por el protocolo UDP.</td></tr>
<tr><td>5.</td><td colspan="2"></td><td>Captura la trasmisión por UDP con VLC.</td></tr>
<tr><td>6.</td><td colspan="2"></td><td>Si el protocolo de la cámara simulada es RTSP trasmite por ese protocolo. Termina el caso de uso.
Observación: Una vez que termina la reproducción del video, el sistema inicia nuevamente la reproducción desde el inicio.</td></tr>
<tr><td colspan="4">Flujos alternos</td></tr>
<tr><td colspan="4">5a. La cámara trasmite por el protocolo HTTP.</td></tr>
<tr><td></td><td colspan="2">Actor</td><td>Sistema</td></tr>
<tr><td>5a.</td><td colspan="2"></td><td>Trasmite por el protocolo HTTP. Termina el caso de uso.
Observación: Una vez que termina la reproducción del video, el sistema inicia nuevamente la reproducción desde el inicio.</td></tr>
<tr><td colspan="2" rowspan="2">Relaciones</td><td>CU Incluidos</td><td>No procede.</td></tr>
<tr><td>CU Extendidos</td><td>Realizar procesamiento. Ver CU Realizar</td></tr>
</table>

		procesamiento.
RnF	RnF 5, RnF 8, RnF 9, RnF 10, RnF 11, RnF 12.	
Asuntos pendientes	No procede.	

Tabla 4: Descripción del CU Trasmitir video.

Conclusiones

En este capítulo se especificaron los temas relacionados con el análisis de la aplicación que se desea desarrollar, a partir de la información recopilada con anterioridad.

- ✓ Se realizó una modelación de la situación problémica obteniéndose el modelo de dominio correspondiente con su respectivo diagrama, además de una descripción del flujo de dominio facilitando con esto una mejor comprensión sobre el funcionamiento del sistema así como de las clases que interactúan dentro del mismo.
- ✓ Se especificaron los requisitos del software funcionales y no funcionales, para un total de 14 requisitos funcionales que fueron agrupados en 4 casos de uso de ellos 2 críticos y el resto secundarios, obteniéndose con esto una mejor organización de las funcionalidades del sistema, permitiéndole al desarrollador identificar sus funcionalidades básicas.
- ✓ Se identificaron los actores que interactúan con cada uno de los casos de usos, obteniéndose finalmente el diagrama de casos de uso del sistema, lo cual facilitó la realización de las especificaciones pertinentes, que permitirán al desarrollador conocer de forma detallada el flujo de cada una de las funcionalidades.

Capítulo 3: Análisis y Diseño.

Introducción

En este capítulo se definen los procesos relacionados con el análisis y diseño de la aplicación que se desea construir. Se caracteriza la arquitectura N-capas, concretándose la arquitectura del Simulador de Cámaras IP. Se definen los patrones de diseño a utilizar, presentándose el modelo de clases del diseño en el que se muestran las relaciones que existen entre las mismas evidenciándose la utilización de los patrones de diseños los cuales garantizan el buen estructurado de las clases.

3.1 Descripción de la Arquitectura.

La arquitectura de software, está relacionada con el diseño y la implementación de estructuras de software de alto nivel. Es el resultado de ensamblar un cierto número de elementos arquitectónicos de forma adecuada para satisfacer la mayor funcionalidad y requerimientos de desempeño de un sistema, así como requisitos no funcionales, como la confiabilidad, escalabilidad, portabilidad, y disponibilidad.(Rodríguez, 2011)

3.1.1 Arquitectura en capas.

El patrón arquitectónico en capas pertenece a la familia del estilo en llamada y retorno, donde cada capa proporciona servicios a la capa superior y se sirve de las prestaciones que le brinda la inferior. Al dividir un sistema en N-capas, cada capa puede tratarse de forma independiente, sin tener que conocer los detalles de las demás. La división de un sistema en capas facilita el diseño modular, permite la construcción de sistemas débilmente acoplados, lo que significa que si se minimiza las dependencias entre capas, resulta más fácil sustituir la implementación de una capa sin afectar al resto del sistema (González, 2011),

3.1.2 Arquitectura 3 capas.

Para el desarrollo del Simulador de Cámaras IP que se propone se definió una arquitectura en tres capas como una variante del patrón N-capas. Las mismas son denominadas como capa de presentación, capa de negocio y capa de acceso a datos. La capa de presentación es la que ve el usuario, también se la denomina "capa de usuario", comunica y captura la información. Por otra parte la capa de negocio es donde residen los programas que se ejecutan, se reciben las peticiones HTTP del usuario y se envían las respuestas tras el proceso. En esta investigación esta capa es la que se encarga del procesamiento y la trasmisión de los flujos de video. La capa de acceso a datos es la encargada de almacenar los datos del

sistema. Su función es almacenar y devolver datos a la capa de negocio, en este caso la información almacenada sobre las cámaras en los ficheros XML.

Fig. 3: Representación de la Arquitectura 3 capas.

Se escogió este tipo de arquitectura para el desarrollo de la aplicación, pues con la misma se reducen las dependencias existentes entre las clases, de forma tal que las capas más bajas no son conscientes de ningún detalle o interfaz de las superiores. Posibilitando con esto mejorar el soporte del sistema y poseer una mejor organización durante el desarrollo del software; además cada nivel posee funcionalidades diferentes lo que permite el diseño de una arquitectura escalable que puede ser ampliada en caso de ser necesario.

3.2 Patrones de diseño.

Los patrones de diseño ayudan a estructurar las clases y los objetos para guiar todos los procesos de creación de software. Componen soluciones concretas a problemas que se presentan durante el diseño de una aplicación. Entre los patrones de diseño más utilizados se encuentran los patrones GRASP y los patrones GoF.(Rodríguez, 2011)

Los patrones GRASP (*General Responsibility Assignment Software Patterns* o Patrones Generales de Software para Asignar Responsabilidades) describen los principios fundamentales de diseño de objetos para la asignación de responsabilidades y aplica el razonamiento para el diseño de una forma sistemática, racional y explicable.(Alcolea, 2012)

Los patrones GoF (*Gang of Four* o Grupo de Cuatro) son soluciones técnicas basadas en Programación Orientada a Objetos (POO). Se clasifican en patrones estructurales, creacionales y de comportamiento.(Expósito, 2011)

3.2.1 Patrones GRASP empleados.

- ✓ **Experto:** Es el principio básico de asignación de responsabilidades. Este patrón garantizará que cada clase cumpla con sus responsabilidades, según la información que contenga y las funcionalidades que se deseen implementar. (Larman, 2000) En la aplicación cada clase mantiene este principio, por ejemplo la clase GstStreaming, es la experta en conocer todo lo relacionado al video en trasmisión (*streaming*), controlando las acciones relacionadas al procesamiento del flujo de video que se trasmite.

- ✓ **Alta Cohesión:** Permite simplificar el mantenimiento y brindar un alto grado de funcionalidad. Se utilizó Alta Cohesión para asignar responsabilidades a clases que estén altamente relacionadas con la trasmisión de video garantizando la contribución entre estas para realizar tareas de elevada complejidad en vez de utilizar una sola clase para ello.

- ✓ **Controlador:** Permite asignar la responsabilidad de controlar el flujo de eventos del sistema, a clases específicas. Se utilizó para decidir qué clase es la responsable de recibir o manejar un evento del sistema, por ejemplo la clase CameraManagement es la controladora pues es la encargada de recibir los datos que son enviados desde la interfaz y enviarlos a las distintas clases según el método llamado.

- ✓ **Bajo Acoplamiento:** El acoplamiento es una medida de la fuerza con que una clase está conectada a otras clases. Permite la reutilización aumentando la productividad, el diseño de las clases más independientes y reduce el impacto de los cambios. Se utilizó para asignar responsabilidades a otras clases reduciendo las dependencias al mínimo, lo más que se pueda facilitando la reutilización de código.

- ✓ **Creador:** Cada clase instancia y crea las clases que le son necesarias para cumplir sus funcionalidades. Se utilizó para decidir qué clase es la responsable de la construcción de objetos es decir, la que le puede asignar los datos al constructor por ejemplo en las clases CameraManagement, CameraControl y ReadProfile.

3.2.2 Patrones GoF empleados.

Fachada: Proporciona una única interfaz a un conjunto de clases de un subsistema. Se utilizó para simplificar el acceso al conjunto de clases proporcionando una única clase a utilizar para comunicarse con las demás en este caso la clase CameraManagement.

Capítulo 3: Análisis y Diseño.

3.3 Modelo del diseño.

Un modelo de diseño es un modelo de objetos físicos que describen la realización física de los casos usos centrándose en cómo los requisitos funcionales y no funcionales, junto con otras restricciones relacionadas con el entorno de implementación, tienen impacto en el sistema. El diseño representa el centro de atención al final de la fase de elaboración y es el comienzo de las iteraciones de construcción, este representa un plano del modelo de implementación garantizando de esta manera una arquitectura sólida y estable. El modelo de diseño perdurara durante todo el ciclo de vida del software y constituye la entrada fundamental utilizada para el correcto desarrollo de la implementación.(Figueras., 2011)

3.3.1 Diagrama de secuencia del diseño.

Un diagrama de secuencia es una forma de diagrama de interacción, que muestra los objetos como líneas de vida a lo largo de la página y con sus interacciones en el tiempo, representadas como mensajes dibujados como flechas desde la línea de vida origen hasta la línea de vida destino. Los diagramas de secuencia no están pensados para mostrar lógicas de procedimientos complejos.(Moser, 2011.)

A continuación se muestran los diagramas de secuencia para los CU Realizar procesamiento y Trasmitir video. Los diagrama de secuencias para los CU Gestionar perfil de cámara IP y Detener trasmisión de video pueden encontrarse en el Anexo 3 del presente trabajo.

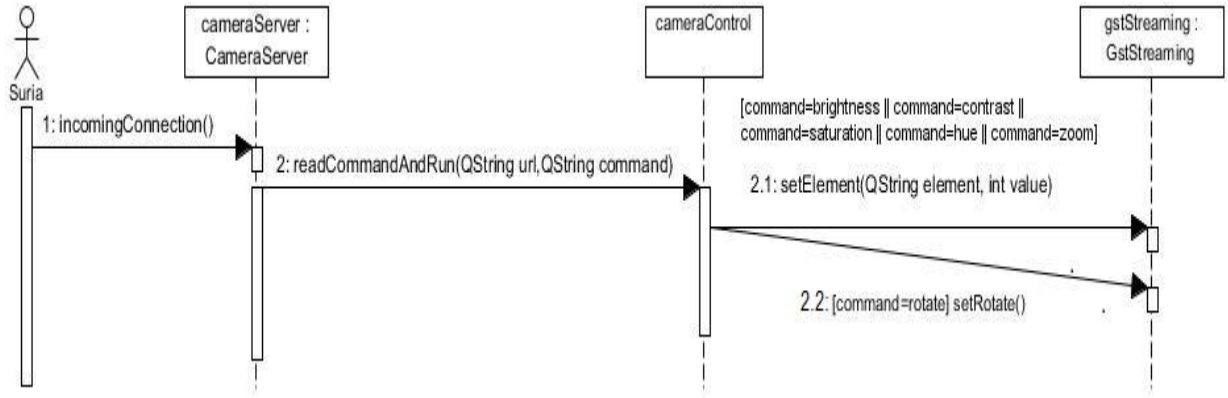

Fig. 4: Diagrama de Secuencia CU Realizar procesamiento.

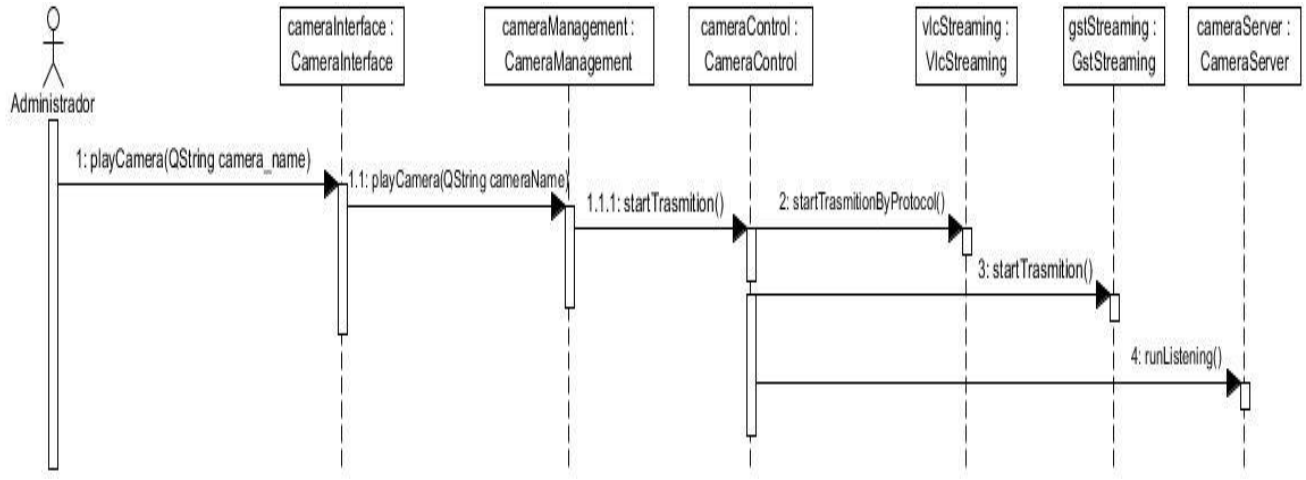

Fig. 5: Diagrama de Secuencia CU Trasmitir video.

3.3.2 Clase del diseño.

Una clase del diseño es una abstracción sin costuras de una clase o construcción similar en la implementación del sistema. Lo cual significa que el lenguaje utilizado para especificar una clase del diseño es lo mismo que el lenguaje de programación, esto conlleva a que las operaciones, atributos, tipos y demás son especificados utilizando la sintaxis del lenguaje de programación elegido.(Rodríguez, 2011)

3.3.3 Diagrama de clases del diseño.

Un diagrama de clases muestra un conjunto de clases, interfaces y colaboraciones, así como sus relaciones. Los diagramas de clases se utilizan para modelar la vista de diseño estática de un sistema. Esto incluye modelar el vocabulario del sistema, modelar las colaboraciones o modelar esquemas. Los diagramas de clases son importantes no sólo para visualizar, especificar y documentar modelos estructurales; sino también para construir sistemas ejecutables, aplicando ingeniería directa e inversa. En la siguiente ilustración, se puede observar el diagrama de clases del diseño de manera conceptual. Las ilustraciones con los atributos y métodos de las clases se encuentran en el Anexo 4 del presente trabajo.

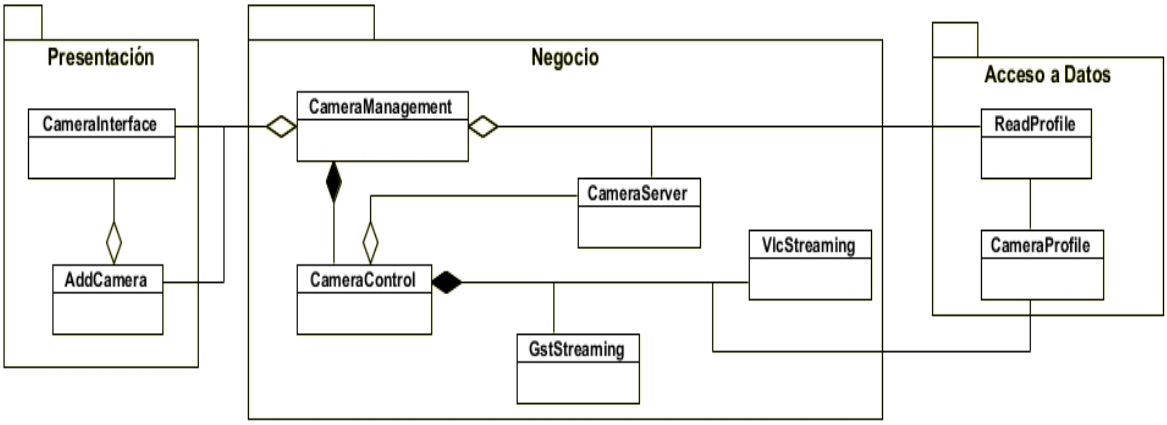

Fig. 6: Diagrama de clases del diseño.

La clase CameraInterface es la clase a través de la cual el administrador puede acceder a las funcionalidades del sistema. Posee un objeto de tipo AddCamera que le permite acceder a la interfaz para agregar un nuevo perfil. Ambas clases poseen un objeto de tipo CameraManagement que es la clase controladora; esta clase recibe los eventos desde las interfaces y los envía a las clases CameraControl o ReadProfile según corresponda. La clase CameraControl permite controlar los diferentes perfiles de cámaras que se estén simulando y la clase ReadProfile posibilita acceder a los perfiles de cámaras IP almacenados en el archivo XML.

La clase CameraControl es la clase encargada de manipular los flujos de video para su procesamiento y trasmisión a través de un objeto de tipo GstStreaming, esta clase permite realizar funciones, como ajuste de brillo, contraste, función de zoom y de rotación. Además recibe las peticiones HTTP que le son enviados desde la clase CameraServer y realiza la trasmisión del video procesado a través de los protocolos RTSP y HTTP (en dependencia de la cámara que se esté simulando) a través de un objeto de tipo VlcStreaming. Posee además un objeto de tipo CameraProfile que le facilita el mapeo de los perfiles pues es la clase encargada de almacenar las características de cada cámara IP.

Conclusiones

En este capítulo se definieron temas relacionados con el análisis y el diseño del Simulador de Cámaras IP para el sistema Video Vigilancia Suria.

- ✓ Se estableció la línea base de la arquitectura que tendrá el simulador que se quiere desarrollar empleándose el patrón arquitectónico en tres capas como una especificación de la arquitectura N-

capas, facilitando la reutilización de las capas, la estandarización y la contención de cambios, proporcionando con esto que se logre garantizar el mejor desempeño, robustez, portabilidad, flexibilidad y escalabilidad de la aplicación.

- ✓ Se identificaron los patrones de diseño GRASP y GoF utilizados en el desarrollo de la aplicación, permitiendo facilitar la asignación de responsabilidades logrando un diseño de software que sirva de apoyo a la implementación del sistema.
- ✓ Se realizaron los diagramas de secuencias del diseño, mostrándose en cada uno de ellos las relaciones entre las clases por medio de funciones, garantizando una mejor comprensión del flujo de vida en cada caso de uso. Además se realizó el diagrama de clases de diseño, obteniendo con esto una visión estática del sistema, facilitando la comunicación entre los programadores y el descubrimiento de fallas de la aplicación en el diseño

Capítulo 4: Implementación y Prueba.

Introducción

En este capítulo se presentan todos los elementos relacionados al flujo de trabajo de implementación, describiéndose detalladamente las técnicas utilizadas en el desarrollo de la aplicación. Se realizan los diagramas de despliegue y de componentes de implementación resultantes de esta etapa de construcción. Se describen las pruebas de caja negra aplicadas a la aplicación, con el objetivo de validar su correcto funcionamiento.

4.1 Modelo de Implementación.

El modelo de implementación está compuesto por un conjunto de subsistemas y componentes que establecen la composición física de la implementación del sistema, tiene como objetivos implementar las clases de diseño como componentes, asignar los componentes a los nodos, probar los componentes individualmente e integrar los componentes en un sistema ejecutable.(Ramos, 2011)

4.1.1 Implementación.

En la etapa de implementación existen una serie de artefactos que tienen como elementos de entrada para su confección, los desarrollados en la fase anterior de análisis y diseño. Su objetivo es ir conformando el proyecto como un sistema completo y lograr un acabado correspondiente a los requisitos definidos.

4.1.2 Diagrama de Despliegue.

El Diagrama de Despliegue es un tipo de diagrama del Lenguaje Unificado de Modelado que muestra las relaciones físicas de los distintos nodos que componen un sistema y el reparto de los componentes sobre dichos nodos. En la siguiente figura se representa el diagrama de despliegue correspondiente a la aplicación:

Fig. 7: Diagrama de Despliegue.

Capítulo 4: Implementación y Prueba.

4.1.3 Diagrama de Componentes de Implementación.

Los diagramas de componentes de implementación permiten describir los elementos físicos del sistema y sus relaciones. Los componentes representan todos los tipos de elementos de software que entran en la fabricación de aplicaciones informáticas. Pueden ser simples archivos, paquetes y bibliotecas cargadas dinámicamente.(Ramos, 2011) A continuación se muestra en la siguiente figura el diagrama de componente de implementación:

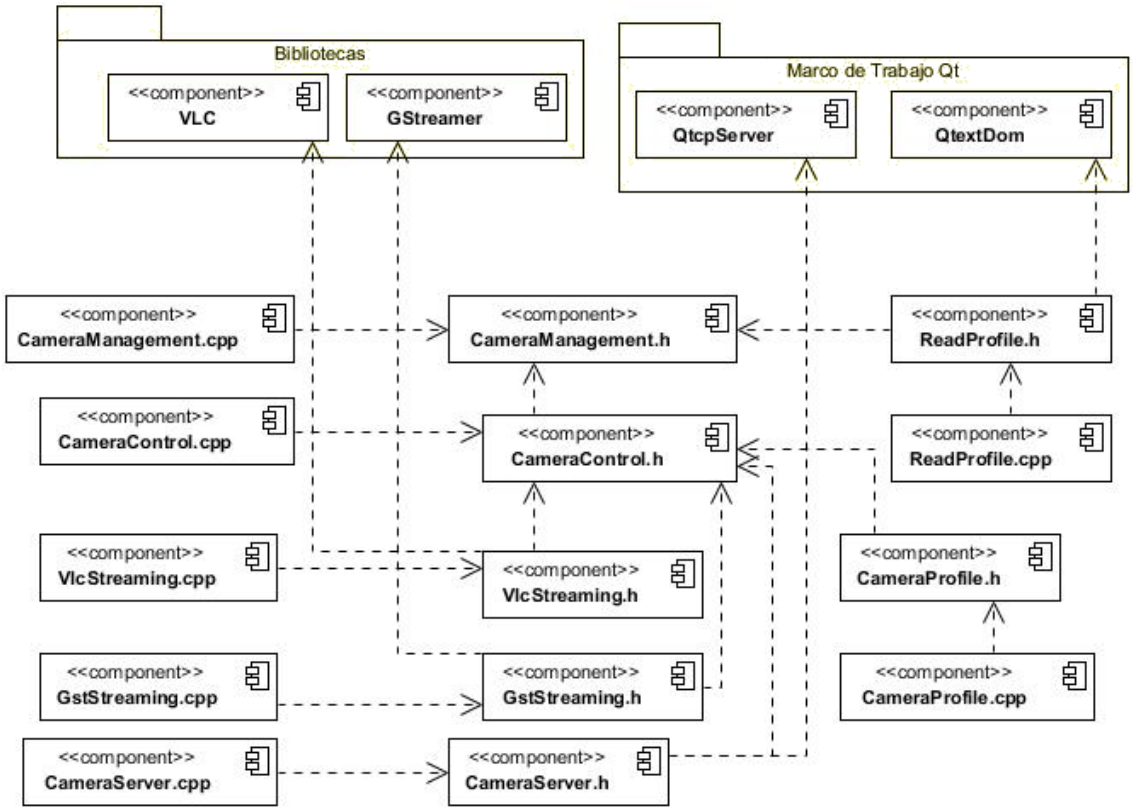

Fig. 8: Diagrama de Componente de Implementación del Simulador de Cámaras IP para el sistema Video Vigilancia Suria.

4.2 Prueba del Software.

Las pruebas de software no son más que técnicas de validación para verificar el resultado de la implementación probando cada construcción. El objetivo de estas pruebas es descubrir errores en la aplicación que antes no se habían descubierto.(Ramos, 2011)

Capítulo 4: Implementación y Prueba.

4.3 Métodos de Pruebas.

Existen dos métodos para la realización de las pruebas de software, las pruebas de caja blanca y las pruebas de caja negra siendo estas últimas las que se le realizarán a la aplicación, ya que el principal objetivo de estas pruebas es verificar las entradas y salidas de los datos.

4.3.1 Pruebas de Caja Negra.

Para la realización de pruebas a la aplicación se seleccionó la prueba de caja negra. Estas pruebas también denominadas pruebas de comportamiento, se concentran en los requisitos funcionales del software. Esto permite a quien la aplique derivar conjuntos de condiciones de entrada que ejercitan por completo todos los requisitos funcionales del programa. En esta prueba se utilizó la técnica de partición equivalente. Esta técnica divide el dominio de entrada en clases de datos a partir de las cuales pueden derivarse casos de prueba.

La partición equivalente se esfuerza por definir un caso de prueba que descubra varias clases de errores, reduciendo así el número de casos de prueba que deben desarrollarse. Se definen un conjunto de estados válidos y no válidos para las condiciones de entrada y se verifica que los datos de salida sean los esperados.(Pressman, 2010) A continuación se muestran los casos de prueba para los CU Realizar procesamiento y Trasmitir video. Los casos de prueba para los CU Administrar perfil de cámara IP y Detener trasmisión de video pueden encontrarse en el Anexo 5 del presente trabajo.

Caso de Prueba #1 “CU Realizar Procesamiento”.

Sección 1 "Rotación horizontal"

Escenario	Descripción	Respuesta del sistema	Flujo central
EC 1.1 Rotar cámara correctamente.	El sistema recibe las peticiones HTTP que indican la dirección en que se va a mover la cámara (izquierda o derecha). En este caso por defecto se mueve 10 px.	El sistema rota de forma horizontal 10pxen la dirección especifica que puede ser izquierda o derecha.	Simulador de Cámaras IP/ clic en la cámara que desea simular/ botón rotación horizontal.

EC 1.2 Rotar cámara en los límites horizontales.	El sistema recibe las peticiones HTTP que indican la dirección en que se va a mover la cámara pero la trasmisión del video se encuentra en cualquiera de los límites horizontales.	El sistema mantiene la trasmisión en el límite del video.	Simulador de Cámaras IP/ clic en la cámara que desea simular/ botón rotación horizontal.

Tabla 5: Diseño de caso de prueba Sección "Rotación horizontal".

Sección2 "Rotación vertical"

Escenario	Descripción	Respuesta del sistema	Flujo central
EC 2.1 Rotar cámara correctamente.	El sistema recibe las peticiones HTTP que indican la dirección en que se va a mover la cámara (arriba o abajo). En este caso por defecto se mueve 10 px.	El sistema rota de forma vertical 10pxen la dirección especifica que puede ser arriba o abajo.	Simulador de Cámaras IP/ clic en la cámara que desea simular/ botón rotación vertical.
EC 2.2 Rotar cámara en los límites horizontales.	El sistema recibe las peticiones HTTP que indican la dirección en que se va a mover la cámara pero la trasmisión del video se encuentra en cualquiera de los límites verticales.	El sistema mantiene la trasmisión en el límite del video.	Simulador de Cámaras IP/ clic en la cámara que desea simular/ botón rotación vertical.

Tabla 6: Diseño de caso de prueba Sección "Rotación vertical".

Sección3 “Ajustar contraste”.

Escenario	Descripción	Respuesta del sistema	Flujo central
EC 3.1 Ajustar contraste	El sistema recibe las peticiones HTTP que indican	El sistema modifica el contraste del video que se	Simulador de Cámaras IP/ clic en

correctamente.	el número para el ajuste de contraste.(valores de 0 a 2)	trasmite.	la cámara que desea simular/ botón contraste.
EC 3.2 Ajustar contraste en los límites.	El sistema recibe las peticiones HTTP que indican el número para el ajuste de contraste pero la trasmisión del video se encuentra en su valor límite de contraste.	El sistema trasmite el video en el límite de contraste que se encuentra.	Simulador de Cámaras IP/ clic en la cámara que desea simular/ botón contraste.

Tabla 7: Diseño de caso de prueba Sección "Ajustar contraste".

Sección4 “Ajustar brillo”.

Escenario	Descripción	Respuesta del sistema	Flujo central
EC 4.1 Ajustar brillo correctamente.	El sistema recibe las peticiones HTTP que indican el número para el ajuste de brillo.(valores de -1 a 1)	El sistema modifica el brillo del video que se trasmite.	Simulador de Cámaras IP/ clic en la cámara que desea simular/ botón brillo.
EC 4.2 Ajustar brillo en los límites.	El sistema recibe las peticiones HTTP que indican el número para el ajuste de brillo pero la trasmisión del video se encuentra en su valor límite de brillo.	El sistema trasmite el video en el límite de brillo que se encuentra.	Simulador de Cámaras IP/ clic en la cámara que desea simular/ botón brillo.

Tabla 8: Diseño de caso de prueba Sección "Ajustar brillo".

Sección 5 “Ajustar color”.

Escenario	Descripción	Respuesta del sistema	Flujo central
EC 5.1 Ajustar color correctamente.	El sistema recibe las peticiones HTTP que indican el número para el ajuste de	El sistema modifica el color del video que se trasmite.	Simulador de Cámaras IP/ clic en la cámara que

	color.(valores de -1 a 1)		desea simular/ botón color.
EC 5.2 Ajustar coloren los límites.	El sistema recibe las peticiones HTTP que indican el número para el ajuste de color pero la trasmisión del video se encuentra en su valor límite de color.	El sistema trasmite el video en el límite de color que se encuentra.	Simulador de Cámaras IP/ clic en la cámara que desea simular/ botón color.

Tabla 9: Diseño de caso de prueba Sección "Ajustar color".

Sección 6 “Ajustar saturación”.

Escenario	Descripción	Respuesta del sistema	Flujo central
EC 6.1 Ajustar saturación correctamente.	El sistema recibe las peticiones HTTP que indican el número para el ajuste de saturación.(valores de 0 a 2)	El sistema modifica la saturación del video que se trasmite.	Simulador de Cámaras IP/ clic en la cámara que desea simular/ botón saturación.
EC 6.2 Ajustar saturación en los límites.	El sistema recibe las peticiones HTTP que indican el número para el ajuste de saturación pero la trasmisión del video se encuentra en su valor límite de saturación.	El sistema trasmite el video en el límite de saturación que se encuentra.	Simulador de Cámaras IP/ clic en la cámara que desea simular/ botón saturación.

Tabla 10: Diseño de caso de prueba Sección "Ajustar saturación".

Sección 7 “Hacer Zoom”.

Escenario	Descripción	Respuesta del sistema	Flujo central
EC 7.1 Hacer Zoom correctamente.	El sistema recibe las peticiones HTTP que indican el número de pixeles para el	El sistema hace zoom en el video que se trasmite.	Simulador de Cámaras IP/ clic en la cámara que desea

	zoom.(valores de 0 a 100)		simular/ botón zoom.
EC 7.2 Hacer Zoom en los límites.	El sistema recibe las peticiones HTTP que indican el número de pixeles para el zoom pero la trasmisión del video se encuentra en su valor límite.	El sistema trasmite el video en el límite de zoom que se encuentra.	Simulador de Cámaras IP/ clic en la cámara que desea simular/ botón zoom.

Tabla 11: Diseño de caso de prueba Sección "Hacer zoom".

Caso de Prueba #2 "CU Trasmitir video".

Sección "Trasmitir video".

Escenario	Descripción	Respuesta del sistema	Flujo central
Trasmisión del flujo de video correctamente	El Administrador selecciona el perfil de cámara IP que desea simular e inicia la trasmisión del flujo de video.)	El sistema comienza la trasmisión del flujo de video correctamente por el protocolo RTSP o HTTP según la cámara simulada.	Simulador de Cámaras IP/ clic en la cámara que desea simular/ botón Iniciar Trasmisión.

Tabla 12: Diseño de caso de prueba Sección "Trasmitir video".

4.3.2 Resultados de las Pruebas

Para la ejecución de esta técnica se realizó un diseño de caso de prueba (CP) para cada CU obteniéndose un total de 4CP. Como resultado de la primera iteración se obtuvieron 3 no conformidades relacionadas con el procesamiento y la trasmisión de los flujos de video, de ellas 2 altas y 1 baja. Las mismas se muestran en la siguiente tabla:

No.	No conformidad	CP	Clasif.
1.	No se podían simular más de una cámara en trasmisión de forma simultánea.	CP # 2	Alta
2.	Problemas en la trasmisión de diferentes formatos de video en los protocolos RTSP y HTTP.	CP # 2	Alta
3.	Los botones de la aplicación estaban algunos en inglés y otros en español.	-	Baja

Tabla 13: Resultado de la primera iteración de pruebas.

Como resultado de la segunda iteración se obtuvieron 2 no conformidades relacionadas con la estética de la aplicación y la trasmisión de video de ellas 1 alta y 1 baja. Estas no conformidades se muestran en la siguiente tabla:

No.	No conformidad	CP	Clasif.
1.	Cuando se procesaba el video de un perfil de cámara el sistema automáticamente procesaba los videos de todas las cámaras en trasmisión.	CP # 1	Alta
2.	Cuando finalizaba el video en trasmisión, se detenía la misma.	CP # 2	Baja

Tabla 14: Resultado de la segunda iteración de pruebas.

En la tercera iteración no se encontraron no conformidades, lo que significa que la aplicación desarrollada cumple con la calidad requerida para el proyecto Video Vigilancia, proporcionándoles un entorno de pruebas que les da la posibilidad trabajar en un ambiente lo más real posible aun cuando no existan cámaras IP físicamente.

Conclusiones

En este capítulo se definieron temas relacionados con los procesos de implementación y prueba del Simulador de Cámaras IP para el sistema Video Vigilancia Suria.

- ✓ Se realizaron los diagramas de despliegue y de componentes de implementación, los cuales proporcionan una vista de la implementación del sistema garantizando una descripción detallada de su estructura.
- ✓ Se realizaron pruebas de caja negra, específicamente se aplicó la técnica de partición equivalente, la cual posibilitó comprobar el correcto funcionamiento de la aplicación en cuanto a los requisitos funcionales de software definidos anteriormente, probándose que la aplicación desarrollada cumple con la calidad requerida para ser utilizada por los miembros del proyecto Video Vigilancia proporcionándoles un entorno de pruebas adecuado para cuando no existan cámaras IP.

Conclusiones Generales.

Conclusiones Generales.

Con el presente trabajo de diploma se logró dar cumplimiento a las tareas de la investigación propuestas inicialmente, logrando obtener como resultado un Simulador de Cámaras IP para el sistema Video Vigilancia Suria. Lo cual permitió llegar a las siguientes conclusiones:

- ✓ El análisis de otros sistemas simuladores demostró que no existe a nivel nacional e internacional una aplicación con características similares a las de esta investigación, sin embargo el estudio de los softwares que emplean las cámaras IP permitió identificar funcionalidades que posteriormente fueron incluidas en el simulador, lográndose con esto obtener un producto cuyo funcionamiento es lo más similar posible a una cámara IP.
- ✓ Las herramientas y tecnologías utilizadas en el desarrollo de la aplicación son las definidas por el proyecto Video Vigilancia, favoreciendo con esto posibles integraciones entre el simulador y los módulos de Suria 1.0, garantizando futuras actualizaciones del sistema.
- ✓ Al usar una arquitectura tres capas se facilitó la reutilización de las capas reduciendo las dependencias existentes entre las clases, garantizándose un mejor soporte del sistema y poseer una mejor organización durante el desarrollo del software.
- ✓ Los artefactos generados durante el proceso de desarrollo de software que han sido abordados en la presente investigación proveen de documentación sobre el sistema para futuras actualizaciones y desarrollo de nuevas funcionalidades.

Recomendaciones.

Recomendaciones.

Durante el desarrollo de la presente investigación surgieron ideas que pueden permitir la realización de un Simulador de Cámaras IP más completo, por lo que se recomienda:

- ✓ Incorporar en el sistema otras funcionalidades que se pueden realizar con las cámaras IP que no hayan sido incluidas en la presente investigación como el control del audio, el desenfoque y el auto iris en el video que se trasmite.
- ✓ Implementar la funcionalidad de simular la autenticación de las cámaras IP para lograr una mejor seguridad en el sistema.

Referencias Bibliográficas.

Referencias Bibliográficas.

Alcolea, Walfrido Lora. 2012.*"Desarrollo de un servidor de grabación multiplataforma para el sistema de video vigilancia Suria Visión.".* La Habana, Cuba : UCI, 2012.

Asesorias y computadores LTDA. 2005-2012. Asesoría Informática. *Asesoría Informática.* [Online] 2005-2012. [Cited: diciembre 11, 2012.] http://www.aseinformatica.com/camarasip.php.

Axis Communications. 2012. "Axis". *"Axis".* [Online] 2012. http://www.axis.com/techsup/cam_servers/dev/cam_http_api_index.php.

Axis. 2011. La informatica. [Online] 2011. [Cited: enero 26, 2013.] http://www.lainformatica.com/_camaras.shtml.

Banks, Jerry. 2007. Dirección Nacional de Servicios Académicos Virtuales. *Dirección Nacional de Servicios Académicos Virtuales.* [Online] 2007. [Cited: diciembre 11, 2012.] http://www.virtual.unal.edu.co/cursos/sedes/manizales/4060010/lecciones/Capitulo1/simulacion.htm.

By Kim Hamilton, Russell Miles. 2006.*Learning UML 2.0 .* s.l. : O'Reilly, 2006. 978-0-59-600982-3.

Cáceres, Patricia. 2011. Juventud Rebelde. *Juventud Rebelde.* [Online] agosto 4, 2011. [Cited: noviembre 6, 2012.] http://www.juventudrebelde.cu/cuba/2011-08-04/victoria-en-terreno-virtual/.

Camino, Jorge Relanzón. 2009.*"Diseño y planificación de una red inteligente de videovigilancia".* Madrid, España : Universidad Carlos III de Madrid, 2009.

Claramunt., Javier Lambert. 2012.*"Implementación de un prototipo funcional para automatizar el proceso de pruebas de Caja Blanca del departamento Señales Digitales del Centro GEySED.".* La Habana, Cuba : UCI, 2012.

Definicion abc. 2007-2012. definicionabc. *definicionabc.* [Online] 2007-2012. [Cited: diciembre 11, 2012.] http://www.definicionabc.com/tecnologia/video.php.

Digitales, Señales. 2008-2009.*Conferencia: "Características del video. Conceptos".* La Habana, Cuba : UCI, 2008-2009.

Ecured. 2012. [Online] mayo 2012. [Cited: diciembre 17, 2012.] http://www.ecured.cu/index.php/UML.

Ecured. 2012.*Ecured.* [Online] septiembre 2012. [Cited: abril 1, 2013.] http://www.ecured.cu/index.php/VLC.

Ecured. 2013. [Online] 2013. [Cited: mayo 26, 2013.] http://www.ecured.cu/index.php/XML.

Ecured. 2012. Ecured. *Ecured.* [Online] 2012. [Cited: diciembre 2, 2012.] http://www.ecured.cu/index.php/RUP.

Referencias Bibliográficas.

Elro. 2013. Elro. [Online] 2013. [Cited: enero 8, 2013.] http://www.elro.eu/es/.

Expósito, Yuliet Hernández. 2011.*"Desarrollo de un video sensor para la detección de objetos abandonados.".* La Habana, Cuba : UCI, 2011. 5.

Figueras., Alejandro Alvarez. 2011.*"Sistema para el control autónomo de espacio en disco en servidores de media".* La Habana, Cuba : UCI, 2011.

Hector Daniel Pagán Arias, Annierys Martínez González. 2011.*"Propuesta de arquitectura para Juegos en Línea sobre plataforma web.".* La Habana, Cuba : UCI, 2011.

INEI. 1999.*"Herramientas Case: COLECCION CULTURA INFORMATICA".* s.l. : Talleres de la Oficina de Impresiones de la Oficina Técnica de Difusión del Instituto Nacional de Estadística e Informática (INEI), 1999.

Instituto Tecnológico de Acapulco. 2000.*"Simulación".* Mexico : s.n., 2000.

Insua, David Ríos. 2008.*"Simulación: métodos y aplicaciones".* España : RA-MA EDITORIAL, 2008. p. 387 p. ISBN.

Jacobson, I. 1998.*"El lenguaje Unificado de Modelado, Manual de referencia.".* s.l. : Addison Wesley, 1998.

Larman, Craig. 2000.*"UML y Patrones".* México : Prentice Hall, 2000.

Loja. 2008. Loja. [Online] Enero 2008. [Cited: Enero 15, 2012.] http://www.utpl.edu.ec/eva/descargas/material/175/G18401.8.pdf.

Marrero, Danieyis Santiago. 2011.*"Módulo Web de Monitorización y Administración del Sistema de Video Vigilancia".* La Habana : s.n., 2011.

Moser, Christian. 2011.. WPF Tutorial | Model-View-ViewModel Pattern. [Online] 2011. http://www.wpftutorial.net/MVVM.html. .

Paniagua, Soraya. 2004.*"Aprender haciendo, formación basada en simuladores".* 2004.

PCE. 2012 . pce-iberica. *pce-iberica.* [Online] mayo 25, 2012 . [Cited: diciembre 11, 2012.] http://www.pce-iberica.es/instrumentos-de-medida/medidores/sistemas-vigilancia.htm.

Pierre. 2009. "Simuladores". *"Simuladores".* 2009.

Pressman, Roger S. 2010.*"Software Engineering".* 7ma. New York : Higher Education, 2010.

PRmob. 2012. PRmob. *PRmob.* [Online] 2012. [Cited: febrero 21, 2013.] http://es.prmob.net/c%C3%A1mara-ip/circuito-cerrado-de-televisi%C3%B3n/c%C3%A1mara-663190.html.

Ramos, Yurisdelvys Almarales. 2011.*"Cliente de grabación y reporte para el sistema de video vigilancia Suria Web.".* La Habana, Cuba : UCI, 2011.

Referencias Bibliográficas.

Redmine. 2006. Redmine. *Redmine.* [Online] 2006. [Cited: noviembre 1, 2012.] http://gespro.geysed.prod.uci.cu/projects/video-vigilancia?jump=welcome.

Rodríguez, Lisandra Michelena. 2011.*"Desarrollo de un video sensor para el conteo de personas.".* La Habana, Cuba : UCI, 2011.

Sourceforge.net. 2012. [Online] 2012. [Cited: mayo 31, 2013.] http://sourceforge.net/projects/ipcameraemu/.

Veranes., Alejandro Miguel Rodríguez. 2012.*"Componente para localizar regiones de subtítulos en videos".* La Habana, Cuba : UCI, 2012.

Video Insight. 2012. [Online] 2012. [Cited: abril 2, 2013.] http://www.video-insight.com/Products/SDK.aspx.

VIVOTEK. 2011. VIVOTEK. *VIVOTEK.* [Online] 2011. http://www.vivotek.com/products/model.php?soft=vast..

Wix. 2012. Wix.com. *Wix.com.* [Online] 2012. [Cited: enero 25, 2013.] http://simsystemsecurity.wix.com/tesis_2011#!__historia-de-la-sim-copy1.

Bibliografía.

Bibliografía.

Asesorias y computadores LTDA. 2005-2012. Asesoría Informática. *Asesoría Informática.* [Online] 2005-2012. [Cited: diciembre 11, 2012.] http://www.aseinformatica.com/camarasip.php.

Axis. 2008.*"VAPIX RTSP API".* 2008.

Banks, Jerry. 2007. Dirección Nacional de Servicios Académicos Virtuales. *Dirección Nacional de Servicios Académicos Virtuales.* [Online] 2007. [Cited: diciembre 11, 2012.] http://www.virtual.unal.edu.co/cursos/sedes/manizales/4060010/lecciones/Capitulo1/simulacion.htm.

By Kim Hamilton, Russell Miles. 2006.*Learning UML 2.0 .* s.l. : O'Reilly, 2006. 978-0-59-600982-3.

Cáceres, Patricia. 2011. Juventud Rebelde. *Juventud Rebelde.* [Online] agosto 4, 2011. [Cited: noviembre 6, 2012.] http://www.juventudrebelde.cu/cuba/2011-08-04/victoria-en-terreno-virtual/.

Ecured. 2012. [Online] mayo 2012. [Cited: diciembre 17, 2012.] http://www.ecured.cu/index.php/UML.

Ecured. 2012.*Ecured.* [Online] septiembre 2012. [Cited: abril 1, 2013.] http://www.ecured.cu/index.php/VLC.

Ecured. 2013. [Online] 2013. [Cited: mayo 26, 2013.] http://www.ecured.cu/index.php/XML.

Ecured .2011. Ecured. *Ecured.* [Online] marzo 21, 2011. [Cited: noviembre 12, 2012.] http://www.ecured.cu/index.php/UCI.

Ecured. 2012. Ecured. *Ecured.* [Online] 2012. [Cited: diciembre 2, 2012.] http://www.ecured.cu/index.php/RUP.

Elro. 2013. Elro. [Online] 2013. [Cited: enero 8, 2013.] http://www.elro.eu/es/.

Instituto Tecnológico de Acapulco. 2000.*"Simulación".* Mexico : s.n., 2000.

Insua, David Ríos. 2008.*"Simulación: métodos y aplicaciones".* España : RA-MA EDITORIAL, 2008. p. 387 p. ISBN.

Jacobson, I. 1998.*"El lenguaje Unificado de Modelado, Manual de referencia.".* s.l. : Addison Wesley, 1998.

Larman, Craig. 2000.*"UML y Patrones".* México : Prentice Hall, 2000.

Loja. 2008. Loja. [Online] Enero 2008. [Cited: Enero 15, 2012.] http://www.utpl.edu.ec/eva/descargas/material/175/G18401.8.pdf.

Moser, Christian. 2011.. WPF Tutorial | Model-View-ViewModel Pattern. [Online] 2011. http://www.wpftutorial.net/MVVM.html. .

Paniagua, Soraya. 2004.*"Aprender haciendo, formación basada en simuladores".* 2004.

Pierre. 2009. "Simuladores". *"Simuladores".* 2009.

Bibliografía.

Pressman, Roger S. 2010.*"Software Engineering".* 7ma. New York : Higher Education, 2010.

Sourceforge.net. 2012. [Online] 2012. [Cited: mayo 31, 2013.] http://sourceforge.net/projects/ipcameraemu/.

Video Insight. 2012. [Online] 2012. [Cited: abril 2, 2013.] http://www.video-insight.com/Products/SDK.aspx.

VIVOTEK. 2011. VIVOTEK. *VIVOTEK.* [Online] 2011. http://www.vivotek.com/products/model.php?soft=vast..

Wix. 2012. Wix.com. *Wix.com.* [Online] 2012. [Cited: enero 25, 2013.] http://simsystemsecurity.wix.com/tesis_2011#!__historia-de-la-sim-copy1.

Anexos.

Anexos.

I. Entrevista realizada.

Fecha: diciembre 2012

Entrevistados: Rayner Pupo Gómez, Olga María Rivera Correa y Reynier Pupo Gómez.

Preguntas:

1. ¿Considera usted importante el desarrollo de una aplicación simuladora de cámaras IP? ¿Qué beneficios traería para el proyecto Video Vigilancia una aplicación de este tipo?
2. ¿Qué requisitos considera que sean importantes a desplegar por la aplicación?

Resultados generales de la entrevista por preguntas:

1. Una aplicación como el simulador provee al proyecto Video Vigilancia de un entorno de pruebas que permita a los desarrolladores experimentar y comprobar las funcionalidades que incorpora Suria. A grandes rasgos algunos de los beneficios que traería para el proyecto Video Vigilancia un Simulador de Cámaras IP son:

✓ Se puede probar la calidad del software Suria en un ambiente lo más real posible, a partir de los modelos de cámaras que puedan ser integrados y las funcionalidades que estas brindan.

✓ Ahorro de recursos para el centro GEySED pues con una aplicación de este tipo no es necesario la compra de cámaras para probar las funcionalidades que existen ya implementadas ni las que se desarrollen posteriormente.

2. Algunos de los requisitos coincidentes mencionados por los entrevistados son:

✓ Simular rotación de cámara.

✓ Transmitir video.

✓ Simular ajuste de los colores del video.

✓ Simular zoom en el video que se trasmite**.**

II. Descripción de los CU Gestionar perfil de cámara IP y Detener trasmisión de video

CU	Gestionar perfil de cámara IP.
Objetivo	El objetivo de este CU es posibilitarle al Administrador el adicionar o modificar un perfil de cámara IP en el simulador.
Actores	Administrador.

Anexos.

Resumen	El caso de uso inicia cuando el Administrador selecciona la opción de adicionar, modificar o eliminar un perfil de cámara IP en el simulador. Una vez iniciado el usuario puede introducir las características de la cámara IP que desea adicionar o modificar o seleccionar la cámara que desea eliminar, una vez finalizado actualiza el archivo XML.	
Complejidad	Baja.	
Prioridad	Media.	
Precondiciones	Para Modificar o Eliminar: Perfil de cámara IP creado. Perfil seleccionado.	
Postcondiciones	Lista de perfiles actualizada.	
Flujo de eventos		
Flujo básico: Trasmitir video.		
	Actor	**Sistema**
1.	Selecciona en la interfaz de la aplicación una de las siguientes opciones: A: Insertar nuevo perfil de cámara IP. B: Modificar perfil de cámara IP, Ver Sección 1:"Modificar perfil de cámara IP". C: Eliminar perfil de cámara IP, Ver Sección 2:"Eliminar perfil de cámara IP".	
2.	Selecciona la opción Insertar Perfil (A).	
3.		Muestra una interfaz con los campos necesarios para agregar una nueva cámara IP: A: Nombre. B: URL Trasmisión. C: URL Rotación. D: URL Brillo. E: URL Saturación.

		F: URL Color. G: URL Zoom. H: Formato. I: Protocolo. J: Puerto. K: Ancho. L: Alto. M: Fps (fotogramas por segundo). N: Puerto del Servidor.
4.	Introduce los campos necesarios para el nuevo perfil de cámara IP a agregar.	
5.	Selecciona la opción Insertar Perfil (A).	
6.		Valida que los datos de entrada sean correctos y guarda el perfil en el archivo XML.
7.		Muestra un mensaje de confirmación al usuario indicando que el perfil fue adicionado correctamente. Termina el caso de uso.
Flujos alternos		
5a. Selecciona la opción Cancelar.		
	Actor	**Sistema**
5a		Cancela la operación y regresa a la interfaz principal. Termina el caso de uso.
Flujos alternos		
6a. Existen campos vacíos, los puertos de escucha y trasmisión son iguales o ya existe un perfil de cámara con el nombre.		
	Actor	**Sistema**
6a		El sistema muestra un mensaje

		informando al usuario que no se pudo agregar el perfil al sistema y el error ocurrido. Termina el caso de uso.
Sección1: “Modificar perfil de cámara IP”		
Flujo básico: “Modificar perfil de cámara IP”		
	Actor	**Sistema**
2.	Selecciona la opción Modificar Perfil (B).	
3.		Muestra una lista de los perfiles que puede modificar. **Observación:** Los perfiles en trasmisión no pueden ser modificados.
4.	Selecciona el perfil que desea modificar.	
5.		Muestra una interfaz para modificar el perfil seleccionado, cargando los datos almacenados de dicho perfil permitiendo su modificación. Tales como: A: Nombre. B: URL Trasmisión. C: URL Rotación. D: URL Brillo. E: URL Saturación. F: URL Color. G: URL Zoom. H: Formato. I: Protocolo. J: Puerto. K: Ancho. L: Alto. M: Fps (fotogramas por segundo). N: Puerto del Servidor.

6.	Modifica los campos deseados.	
7.	Selecciona la opción Modificar Perfil.	
8		Comprueba la entrada correcta de todos los campos y modifica los datos correspondientes a ese perfil almacenados en el archivo XML.
9.		Muestra un mensaje de confirmación, informando al usuario que se modificó el perfil correctamente. Termina el caso de uso.
Flujos alternos		
7a. Selecciona la opción Cancelar.		
	Actor	**Sistema**
7a		Cancela la operación y regresa a la interfaz principal. Termina el caso de uso.
Flujos alternos		
8a. Existen campos vacíos, los puertos de escucha y trasmisión son iguales o ya existe un perfil de cámara con el nombre.		
	Actor	**Sistema**
8a		El sistema muestra un mensaje informando al usuario que no se pudo modificar el perfil y el error ocurrido. Termina el caso de uso.
Sección2: "Eliminar perfil de cámara IP"		
Flujo básico: "Eliminar perfil de cámara IP"		
	Actor	**Sistema**
2.	Selecciona la opción Eliminar Perfil (C).	
3.		Muestra una lista de los perfiles que puede eliminar.

		Observación: Los perfiles en trasmisión no pueden ser eliminados.
4.	Selecciona el perfil que desea eliminar y Selecciona la opción Eliminar Perfil.	
5.		Busca el perfil seleccionado en el archivo XML por el nombre de la cámara y lo elimina.
6.		Muestra un mensaje de confirmación, informando al usuario que se eliminó el perfil correctamente. Termina el caso de uso.
Flujos alternos		
4a. Selecciona la opción Cancelar.		
Actor		**Sistema**
		Cancela la operación y regresa a la interfaz principal. Termina el caso de uso.
Relaciones	**CU Incluidos**	No procede.
	CU Extendidos	No procede.
Requisitos no funcionales	RnF 5, RnF 6, RnF 7, RnF 8, RnF 9, RnF 10, RnF 12.	
Asuntos pendientes	No procede.	

Tabla 1: Descripción del CU Gestionar perfil de cámara IP.

CU	Detener trasmisión de video.
Objetivo	El objetivo de este caso de uso es permitir al Administrador detener la simulación de un determinado perfil de cámara IP que se esté simulando.
Actores	Administrador.
Resumen	El caso de uso inicia cuando el Administrador selecciona la opción de detener trasmisión de video en el simulador. Una vez iniciado el usuario puede

	seleccionar la cámara que desea dejar de simular, una vez finalizado se detiene la trasmisión del video.		
Complejidad	Baja.		
Prioridad	Media.		
Precondiciones	Perfil de cámara IP creado. Perfil seleccionado.		
Postcondiciones	Trasmisión finalizada.		
Flujo de eventos			
Flujo básico: Trasmitir video.			
	Actor		**Sistema**
1.	Selecciona el perfil de cámara que desea detener.		
2.	Selecciona la opción de Detener Trasmisión (A).		
3.			Busca la cámara seleccionada en los perfiles de cámara que se están simulando y detiene la simulación de esta cámara. Termina el caso de uso.
Relaciones	**CU Incluidos**	No procede.	
	CU Extendidos	No procede.	
Requisitos no funcionales	RnF 5, RnF 6, RnF 7, RnF 8, RnF 9, RnF 10, RnF 12.		
Asuntos pendientes	No procede.		

Anexos.

III. Diagrama de secuencia de los CU Gestionar perfil de cámara IP y Detener trasmisión de video

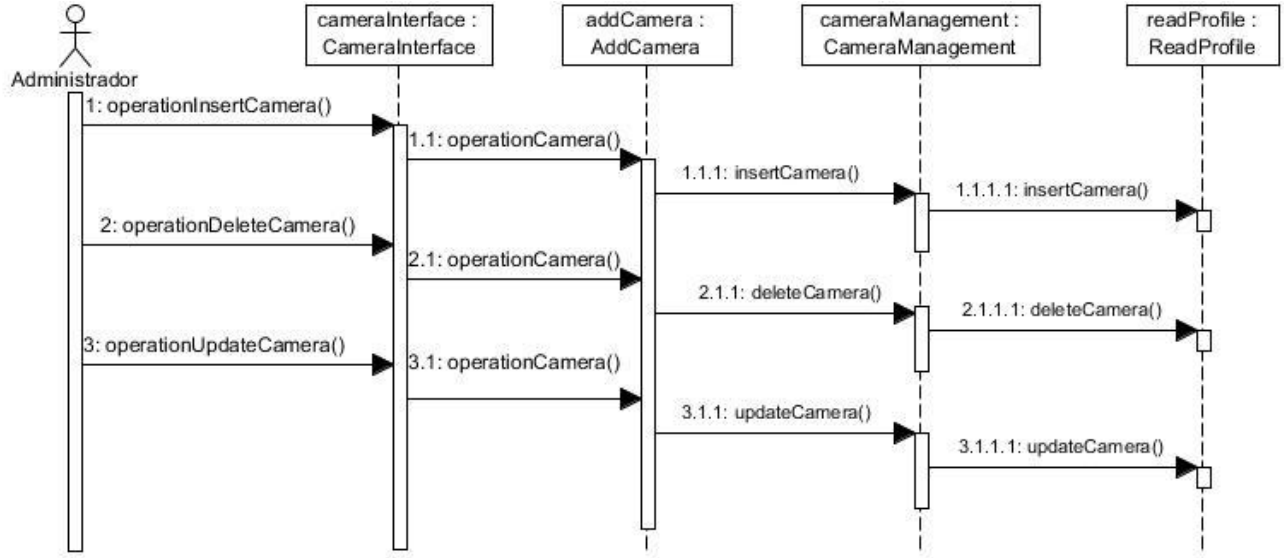

Fig. 9: Diagrama de Secuencia del CU Gestionar perfil de cámara IP.

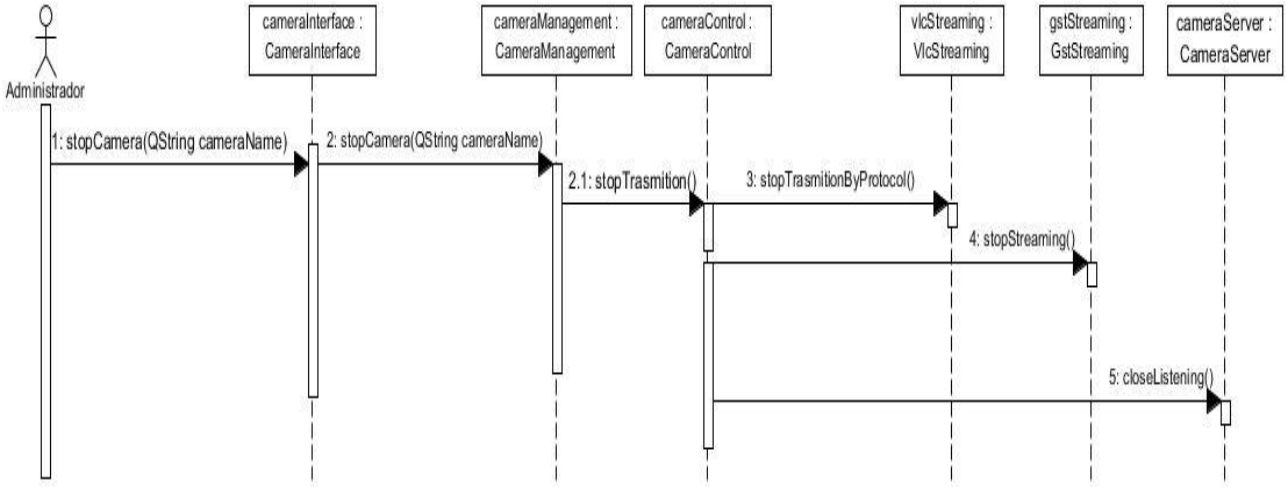

Fig. 10: Diagrama de Secuencia del CU Detener trasmisión de video.

IV. Clases del sistema.

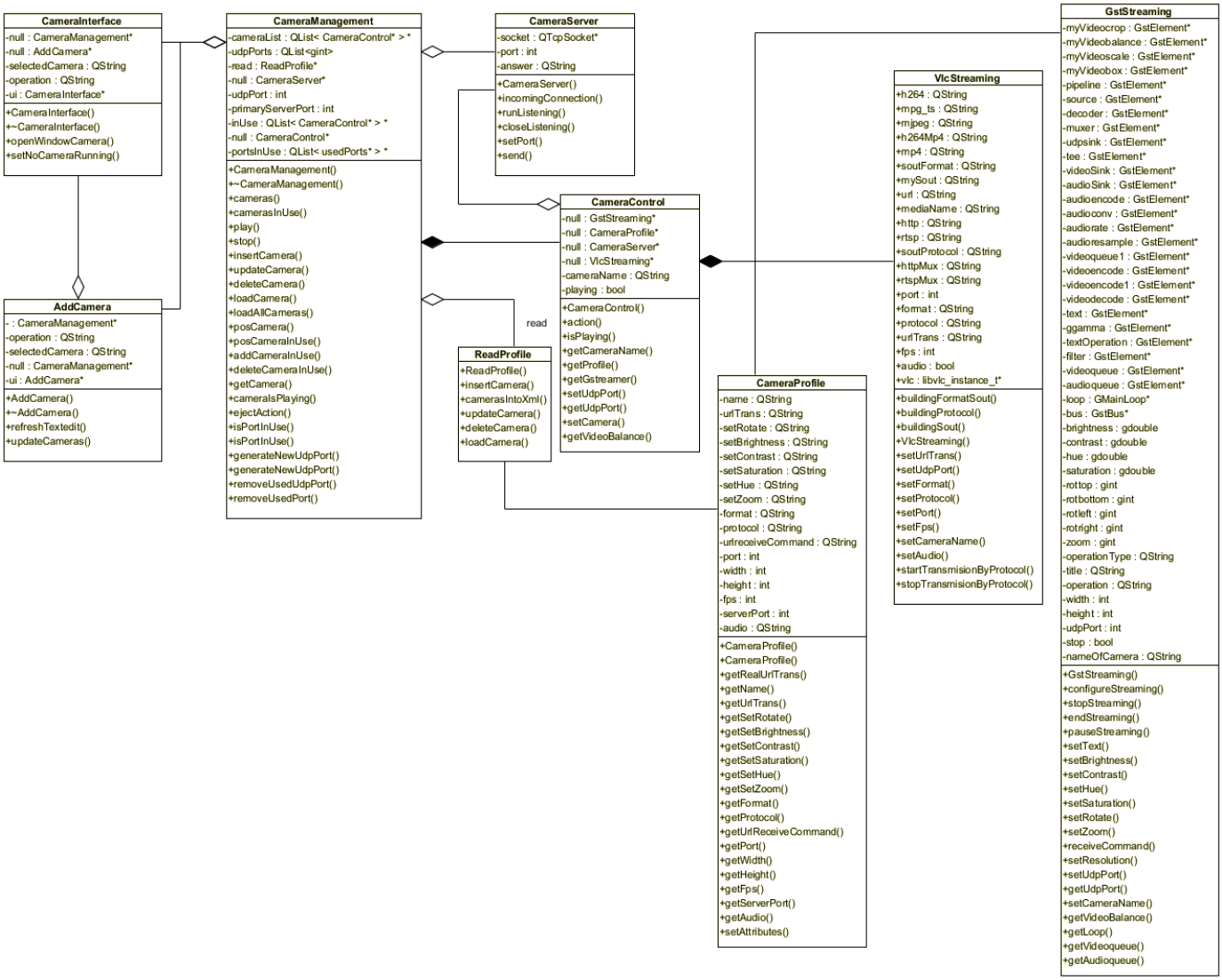

Fig. 11: Diagrama de Clases del diseño.

Glosario de Términos.

Glosario de Términos.

API: Interfaz de Programación de Aplicaciones.

Biblioteca: Colección de clases y funciones.

CASE: Ingeniería de Software Asistida por Computación.

CCTV: Circuito cerrado de televisión**.** Tecnología de video vigilancia visual diseñada para supervisar una diversidad de ambientes y actividades.

Codec: Es la abreviatura de codificador-decodificador. Describe una especificación desarrollada en software, hardware o una combinación de ambos.

Componente: Es una unidad de composición de aplicaciones de software, que posee un conjunto de interfaces y un conjunto de requisitos.

Framework (marco de trabajo): define, en términos generales, un conjunto estandarizado de conceptos, prácticas y criterios para enfocar un tipo de problemática particular, que sirve como referencia para enfrentar y resolver nuevos problemas de índole similar.

IP: Internet Protocol, es un protocolo de comunicación de datos digitales clasificado funcionalmente en la Capa de Red según el modelo internacional OSI.

JPEG: Joint Photographic Experts Group. Es la norma de compresión de imágenes más utilizada a nivel mundial.

LAN: red de área local.

Píxel: Menor unidad que compone una imagen en un medio electrónico.

PTZ: "Pan-Tilt-Zoom". Las cámaras "Pan-Tilt-Zoom" pueden rotar alrededor de dos ejes, uno horizontal y otro vertical, así como acercarse o alejarse.

Rate: Cantidad de imágenes por segundos en un video.

Sensor: Algoritmos que apoyan a los guardias de seguridad durante la vigilancia.

Umbral: es la cantidad mínima de señal que ha de estar presente para ser registrada por un sistema.

Video: Número consecutivo de imágenes y sonido que representan escenas en movimiento.

WAN: red de área amplia.

XML: Es una versión de SGML, diseñado especialmente para los documentos de la web. Permite que los diseñadores creen sus propias etiquetas, permitiendo la definición, transmisión, validación e interpretación de datos entre aplicaciones y entre organizaciones.