Universidad de las Ciencias Informáticas

Facultad 1

Título:

Sistema para el control de flujo de personas en imágenes de video en tiempo real.

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

Autores:
Aynel Cruz Barrera.
Daniel Hernández Díaz.

Tutores:
MsC. Adrian Machado Cento.
Ing. Ramón Santana Fernández.

La Habana, 2013

Declaramos ser los autores del trabajo titulado “Sistema para el control de flujo de personas en imágenes de video en tiempo real” de la Universidad de las Ciencias Informáticas y autorizamos a la Universidad de las Ciencias Informáticas los derechos patrimoniales de la misma, con carácter exclusivo.

Para que así conste firmamos el presente a los 12 días del mes de junio del año 2013.

Aynel Cruz Barrera.

Daniel Hernández Díaz.

MsC. Adrian Machado Cento.

Ing. Ramón Santana Fernández.

Nunca consideres el estudio como una obligación, sino como una oportunidad para penetrar en el bello y maravilloso mundo del saber.

Albert Einstein.

Agradecer a mi mamá y mi hermana por estar siempre ahí cuando más lo necesite y el apoyo incondicional que me brindan, sin ellas esto no hubiera sido posible.

A mi familia por hacer suyos mis momentos, en especial a mi prima Yeney por sus cosas locas. A mi primo Asiel y mis tíos Lidy y Orlando, que siempre estuvieron al tanto de mí. A mi primo Henry (pucho) por ser mi amigo, entenderme y aconsejarme al igual que sus padres Rafi y Mary.

A Lázaro (El zurdo) mi amigo, por compartir momentos de amistad que siempre tendré presente.

A mis amigos de estos 5 años, mi pandilla de por vida que siempre llevaré conmigo y serán parte de mí: Daniel, José Miguel, Yam, Lieny, Herbert y Nilberto por alegrar siempre los momentos tensos. A mi amiga inolvidable por su dulzura y dedicación Yaricel (Yaricita),

A mi compañero de tesis Daniel por su confianza y hacer posible este sueño.

A mi grupo por compartir ese espacio conmigo, en especial a Adysmarys, Vivi, Adays, Sairenys, Liuba, Mabel.

A Soto, Yoevis, Royli.

A mis compañeros del proyecto que de una forma u otra aportaron su granito de arena en esta investigación.

A nuestros tutores Machado y Ramón por orientarnos y dedicarnos su tiempo.

A todas aquellas personas que aportaron a mi formación tanto personal como profesional a lo largo de mi vida.

Gracias a todos…

Aynel Cruz Barrera.

Agradecer a mi padres por creer en mí en todo momento, gracias por ayudarme a alcanzar el punto cumbre de esta etapa de mi vida.

A toda mi familia, los que están y los que no, gracias por todo el apoyo y amor que han sabido darme.

A Yoel, mi segundo padre y mi amigo, no alcanzarían palabras para agradecer todo lo que ha hecho por mí en estos 5 años, este sueño hecho realidad también es suyo.

A mi hermano Danilo, por estar ahí en cada momento que lo necesité.

A mi prima Dayamí, por toda su ayuda y apoyo a mí y a mi familia.

A mis amigos que me acompañaron durante estos 5 años, José Miguel, Omar, Yam, Aynel, Lieny, Sairenys, Adysmarys, en especial a José Miguel, sin su ayuda no hubiese sido posible llegar aquí, gracias a todos por ayudarme y estar ahí cuando los necesité.

A mi amigo y compañero de tesis Aynel, por confiar en mí y hacerme creer que si podíamos lograrlo.

A los nuevos amigos que aunque se incorporaron al grupo luego, no dejan de ser importantes, Herbert, Nilber, Yoevis, Royli, gracias por todo.

A Yanet por todos los momentos buenos y malos que pasamos juntos.

A los compañeros del proyecto Biometría, Adrián, Quiles, y todo los que de alguna manera contribuyeron a realizar este trabajo.

A nuestros tutores Machado y Ramón por ayudarnos y dedicarnos su tiempo siempre que lo necesitamos.

A todas las personas que de una forma u otra aportaron un granito de arena a mi formación como persona y como profesional.

Gracias a todos, de corazón.

Daniel Hernández Díaz.

A mi mamá y mi hermana, que son mi razón de ser.

Aynel Cruz Barrera.

A mi amigo inseparable, mi padre y mi eterna enamorada, mi madre.

A toda mi familia por tanto apoyo y dedicación.

Daniel Hernández Díaz.

Resumen

En la actualidad la mayoría de las empresas están enfrascadas en aumentar su seguridad y la de sus trabajadores. Para ello dichas empresas utilizan diferentes sistemas de seguridad integrados que contribuyen a conservar la confianza y el bienestar en sus instalaciones. Ejemplo de estos son los sistemas de videovigilancia, en particular, los encargados de llevar un control de flujo del personal con acceso a sus instalaciones.

En la Universidad de las Ciencias Informáticas (UCI), en el cual se encuentra el Centro de Identificación y Seguridad Digital (CISED), a partir de análisis realizados por la dirección de dicho centro, se ha detectado que el mecanismo de control de personas con acceso al CISED no garantiza un control de flujo confiable. Atendiendo a esto se decide desarrollar un sistema utilizando imágenes de video en tiempo real que permita mejorar la confiabilidad en el control de personas con acceso al CISED.

En el presente trabajo se hace un estudio de la detección y conteo de personas mediante imágenes de video en tiempo real. Como resultado de este análisis se desarrolla un sistema formado por un *plugin* detector de personas, un componente para el conteo de personas y una aplicación web de reportes.

El documento recoge los resultados de la investigación realizada, describiéndose las principales características de otras soluciones similares analizadas, la arquitectura y el diseño del sistema propuesto. Se describen las herramientas, tecnologías utilizadas y los artefactos generados en el proceso de desarrollo.

Palabras clave: Cámara, conteos de personas, fotograma, imagen digital, iSpy, video digital, videovigilancia.

Índice

Índice de Figuras

Índice de Tablas

Introducción

En la actualidad, es cada vez mayor la necesidad de presentar nuevas alternativas, ideas y experiencias innovadoras para mejorar la seguridad y bienestar de la sociedad. En este contexto, el futuro de la humanidad está sustentado, en gran medida, en el uso de las Tecnologías de la Información y las Comunicaciones (TIC). Las TIC pueden ser empleadas en beneficio de millones de personas a nivel global y con su aplicación se logran reducir muchos obstáculos tradicionales, especialmente los factores tiempo y distancia (1).

Existen mecanismos altamente sofisticados, entre los que se encuentran las tarjetas de seguridad, los scanner biométricos, las alarmas electrónicas y las cámaras de videovigilancia, aunque los demás no dejen de ser importantes, este último garantiza de una manera adecuada la protección de empresas, instituciones, bancos, aeropuertos, entre otros.

La videovigilancia es una rama de la informática que se encarga de la vigilancia a través de un sistema de cámaras móviles o fijas, para monitorear y controlar determinados lugares. Los sistemas de videovigilancia son utilizados en muchas áreas de la sociedad, como en la identificación de las matrículas de autos en movimiento, rastreo de personas en movimiento, detección de movimiento en áreas restringidas o de acceso prohibido, conteo de personas y control de flujos de las mismas, entre otros.

Los sistemas de conteo de personas varían en dependencia del fin para el que son utilizados. Muchos de estos sistemas brindan información específica sobre la cantidad de personas físicas que se encuentran en un lugar determinado, mientras que otros sistemas basan su funcionamiento en controlar el flujo de personas para obtener datos sobre la afluencia de público en un establecimiento.

El Centro de Identificación y Seguridad Digital (CISED) de la Universidad de las Ciencias Informáticas (UCI) está conformado por varios departamentos. En la práctica se ha demostrado que es muy engorroso llevar a cabo el control de todo el personal, pues aunque existe un sistema para el control de acceso a las instalaciones basado en la presentación de una credencial, este solo registra las personas que han ingresado, no haciéndolo así en el momento de la salida. Debido a esto el sistema no lleva un control de la cantidad de personas que salen del centro, tampoco así del número real de personas que permanecen trabajando en el CISED. La dirección de CISED, analizando lo anteriormente planteado, ha decidido

desarrollar un software que permita llevar un control de flujo de las personas utilizando imágenes de video en tiempo real.

Teniendo en cuenta lo antes analizado emerge el siguiente **problema de investigación:** El mecanismo de control de personas con acceso al CISED no garantiza un control de flujo confiable.
Para el desarrollo de este sistema se plantea como **objeto de estudio**, el control de flujo de personas mediante análisis de video en tiempo real. Para dar solución al problema existente se ha tomado como **objetivo general**, desarrollar un sistema mediante el análisis de video en tiempo real, para mejorar la confiabilidad en el control del personal con acceso al Centro de Identificación y Seguridad Digital, de la Universidad de las Ciencias informáticas.

Derivándose en los siguientes **objetivos específicos**:

1. Analizar las soluciones existentes para el desarrollo de un sistema de control de flujo de personas, mediante el análisis de video en tiempo real.
2. Identificar las tecnologías y herramientas relacionadas con la solución propuesta.
3. Realizar el análisis y diseño del sistema propuesto.
4. Implementar los componentes para el control de flujo de personas.

Para dar solución a los objetivos específicos, se utilizaron diferentes Métodos Científicos en la investigación. Estos métodos se clasifican en: Métodos Teóricos y Métodos Empíricos.

Métodos Teóricos: Permiten estudiar las características del objeto de investigación que no se observan directamente, además crean las condiciones para ir más allá de las características fenomenológicas y superficiales de la realidad. Se aplican en calidad de enfoque general como estrategia (2). Los utilizados fueron:

- **Analítico-Sintético:** Se utilizó en el estudio de la literatura especializada relacionada con el tema y la exploración de resultados de investigaciones afines, lo que permitió adoptar posiciones teóricas relacionadas con el objeto de investigación.
- **Histórico-Lógico:** Se utilizó para la revisión y análisis de los documentos y definir los principales antecedentes y fundamentos de la evolución de las aplicaciones, lo que facilitó la indagación de soluciones al problema planteado.

Métodos Empíricos: Estos métodos explican las características fenomenológicas del objeto, con ellos es posible representar un nivel de la investigación cuyo contenido procede de la experiencia y es sometido a cierta elaboración racional. Se aplican como procedimiento en el proceso de investigación (2). De los métodos empíricos existentes se utilizaron:

- **Entrevistas:** Se realizaron entrevistas a ejecutantes y directivos para obtener información relacionada con la aplicación, lo que contribuyó al perfeccionamiento de la propuesta realizada.
- **Consulta a expertos:** Se consultó a personas e instituciones capacitadas en el desarrollo de las herramientas a utilizar, ya que esta información constituyó un apoyo a la realización del producto.

Justificación de la Investigación.

La investigación proporcionará un sistema para el control del flujo de personal con acceso al CISED, de manera tal que contribuya a la seguridad del mismo. Además, facilitará el proceso de toma de decisiones que realiza la dirección del centro.

El siguiente trabajo estará estructurado en tres capítulos, de los cuales se realiza una breve descripción a continuación:

Capítulo I Marco teórico de la investigación.

Se hace referencia al marco conceptual, citando los principales conceptos relacionados con todo lo referente al proceso de detección de personas mediante las imágenes de video en tiempo real. Se analizan otros sistemas existentes, además del estado del arte, tecnologías y herramientas a utilizar en el desarrollo de la aplicación.

Capítulo II Características del Sistema.

Se hace referencia a las soluciones que se proponen para la solución del problema planteado en la situación problemática. Se explica toda la dinámica del proyecto en forma de historias de usuario y algunos modelos auxiliares además del plan de iteraciones para las entregas inmediatas.

Capítulo III Implementación y prueba.

Se da cumplimiento a los planes trazados a través de las fases: Iteraciones a primera liberación y Producción, se codifica la solución diseñada y finalmente se describen, diseñan, realizan y controlan los casos de pruebas aplicados al sistema. Se exponen los resultados obtenidos y se muestran las funcionalidades alcanzadas en el período de desarrollo.

Capítulo 1: Marco teórico de la investigación.

Introducción

En el presente capítulo se hace referencia al marco conceptual, citando los principales conceptos relacionados con todo lo referente al proceso de control de flujo y detección de personas. Se realizará un análisis detallado del estado del arte, de las herramientas y las tecnologías posibles a utilizar en el desarrollo de la aplicación. Asimismo se llega a una conclusión de que es más factible utilizar para dar cumplimiento a al objetivo general de la investigación y dar solución al problema planteado.

1.1. Conceptos asociados al dominio del problema.

Para lograr un mayor entendimiento del trabajo a continuación se reflejan conceptos asociados al dominio del problema.

1.1.1. Definiciones de Sistema Biométrico.

- Se denomina sistema biométrico a un sistema automatizado que realiza labores de biometría. Es decir, un sistema que fundamenta sus decisiones de reconocimiento mediante una característica personal que puede ser reconocida o verificada de manera automática (3).
- Un sistema Biométrico es un sistema automatizado de reconocimiento humano basado en las características físicas y comportamiento de las personas, como son la huella dactilar, el iris, el rostro, la palma de la mano, la voz, la forma de caminar, la silueta o aspecto de la persona, entre otros (4). Siendo esta última una de las más usadas en sistemas biométricos para el control del flujo de personas mediante el análisis de video en tiempo real.

1.1.2. Imagen Digital.

La imagen digital es un producto del desarrollo de la informática que tiene como antecesor a la fotografía, (que toma como punto de partida un objeto del mundo real) y a la pintura, (donde la imagen ha sido creada por un artista), en la imagen digital podemos ver incluidos los dos hechos.

De acuerdo con las valoraciones de Rafael C. González y Richard E. Woods**,** una imagen digital es una matriz bidimensional cuyo objetivo es recrear elementos imaginarios como el arte, o reales como el entorno mismo que rodea al hombre. Las imágenes digitales son muy utilizadas, hasta el punto de haber extinguido casi en su totalidad a la fotografía tradicional. Esta es aplicable en casi todos los campos, entre ellos: la fotografía, el arte, publicidad, comercio, medicina, investigación, educación y otras. Su alta aplicabilidad está dada principalmente por ser fácil de procesar, manejar, clonar y transportar (5).

1.1.3. Video Digital.

El video digital es un tipo de sistema de grabación de video que funciona usando una representación digital de la señal de vídeo. Los mismos pueden ser grabados en cintas magnéticas o directamente en discos duros y memorias flash.

Este consiste en mostrar una sucesión de imágenes digitales. Estas imágenes digitales se muestran a una frecuencia determinada, lo que hace posible saber la frecuencia de refresco, es decir, el número de bytes mostrados (o transferidos) por unidad de tiempo. De esta manera, la frecuencia necesaria para mostrar un video (en bytes por segundo) equivale al tamaño de la imagen multiplicado por el número de imágenes por segundo (6).

1.1.4. Videovigilancia.

Se denomina videovigilancia a un conjunto de cámaras o cámara, fijas o móviles, que responde a un sistema de supervisión. Ofrece un video en tiempo real o grabación del control visual de una zona determinada (7).

Este modo de vigilancia es de gran auge en la actualidad dada las grandes ventajas que brinda donde sea aplicado. Ofrece un control visual en tiempo real permitiendo conocer datos de afluencia, cantidad de personas en un local y su tiempo de estancia o empleándolo con otros fines, apoyando siempre la toma de decisiones de la dirección de una empresa o institución que la utilice.

1.1.5. Fotograma.

Se denomina fotograma o *frame* a cada una de las imágenes en una película fotográfica. Por extensión también se llama de ese modo a cada una de las imágenes individuales captadas por cámaras de video y registradas analógica o digitalmente en este caso. Según la real academia española se define como: “Cada una de las imágenes que se suceden en una película cinematográfica” (8).

Cuando una secuencia de fotogramas es visualizada de acuerdo a una determinada frecuencia de imágenes por segundo, se logra generar la sensación de movimiento en el espectador.

1.2. Estado del Arte.

La detección y conteo de personas mediante videos en tiempo real surge de manera gradual, aparece después de un período de investigación y desarrollo en el campo industrial, militar y académico. La aparición de esta tecnología llega acompañada por la madurez en otras, como los ordenadores y las cámaras digitales.

Analizando trabajos anteriores, se observa que uno de los principales requerimientos a la hora de la detección de personas es la localización de las zonas donde se produce el movimiento. Para la detección de dichas regiones, se suele utilizar lo que se conoce como técnicas de extracción del *background* (fondo). Las distintas técnicas existentes se diferencian entre sí solamente en el tipo de modelo de fondo que utilizan y en las técnicas basadas en dicho modelo de fondo para encontrar las regiones estáticas.

La **Figura 1** muestra la clasifican los diferentes métodos de segmentación entre regiones estáticas y zonas con movimiento basados en las técnicas de extracción del fondo. Para que la clasificación sea los más clara posible se han dividido las técnicas posibles en dos categorías, por un lado las aproximaciones que usan un modelo de fondo y por otro lado las aproximaciones que usan varios modelos de fondo.

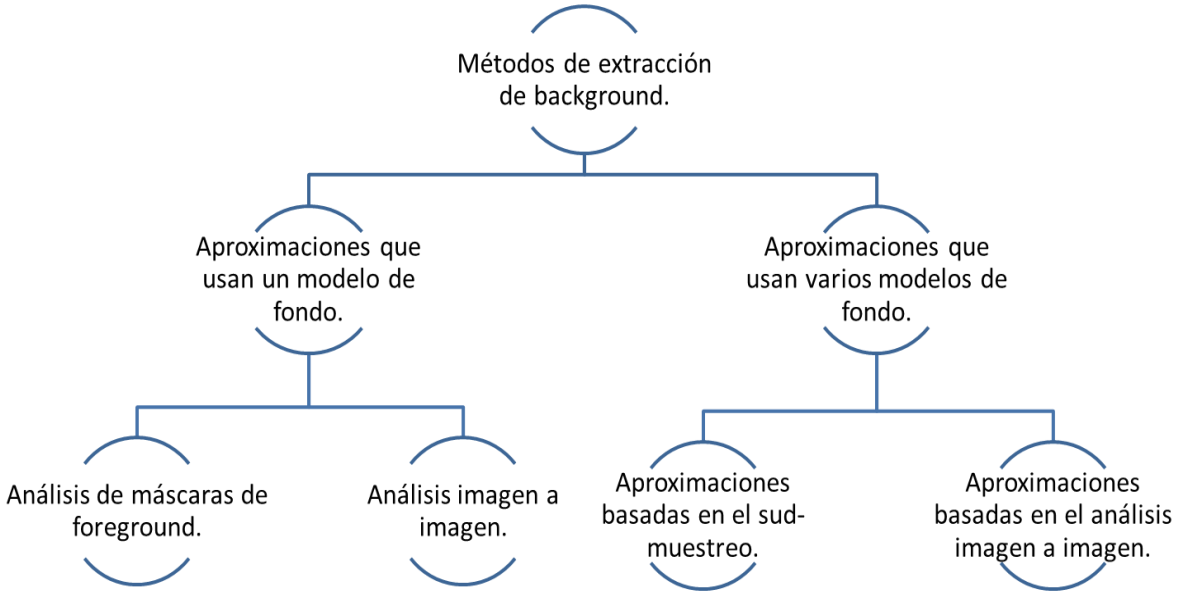

Figura 1. Clasificación de métodos de extracción del fondo.

Una vez realizada la clasificación general anterior, cabe destacar que los algoritmos más importantes y utilizados para la detección de personas en la actualidad según (9) son, el modelo de mezcla Gaussianas (por sus siglas en inglés *Mixture of Gaussians*: MOG) y el modelo Bayesiano (por sus siglas en inglés *Foreground Detection based on background modeling and Bayes classification*: FGD).

Los métodos mencionados deben ser capaces de detectar objetos en movimiento en tiempo real. Además todos los algoritmos deben solventar problemas como:

- **Ruido:** Este ruido es ocasionado por el sensor de la cámara o al medio de transmisión de la señal, se manifiesta en píxeles aislados que toman un valor diferente al de sus vecinos. Eliminar el ruido procedente de la cámara de video, puede provocar la detección de zonas del frente incorrectas.
- **Sombras y reflejos:** Es uno de los principales problemas ya que no pertenecen ni al fondo ni al frente y en la mayoría de los casos generan interferencias que hacen que el algoritmo no funcione de forma adecuada, por ello hay que eliminarlos.
- **Cambios de iluminación:** Son variaciones de la iluminación, estas variaciones pueden tener lugar tanto si la escena ha sido capturada en el exterior como si ha sido tomada en el interior (distintas fuentes de iluminación). Es un factor que influye en gran medida en la complejidad del algoritmo, teniendo en cuenta que es más sencillo modificar la iluminación que cambiar un algoritmo complejo. Se debe dedicar el tiempo necesario para conseguir una buena iluminación de forma que el algoritmo no se complique más de lo necesario.
- **Actualización del fondo de escena:** Su importancia radica en: primeramente se tiene que la inicialización del fondo de la secuencia, generalmente no coincide con el fondo, ya que puede haber algún objeto o persona en movimiento, por lo que el algoritmo debe ser capaz de no clasificarlo como fondo, el segundo punto por lo que la actualización de fondo es importante es debido a que pueden haber cambios importantes en la secuencia.
- **Determinar los parámetros de funcionamiento de los algoritmos:** Es una de las tareas más complejas ya que dependiendo de los parámetros utilizados los algoritmos mostrarán unos resultados u otros, que en la mayoría de los casos difieren en gran medida.

1.2.1. *Foreground Detection based on background modeling and Bayes classification (* FGD).

El FGD propone un marco bayesiano para incorporar características espectrales, espaciales y temporales en el modelado de fondo, debido a esto se encuentra dentro de los métodos de análisis de máscaras de *foreground* (frente) mencionados anteriormente. Deriva una nueva fórmula de la regla de decisión de Bayes para la clasificación de fondo y frente. El fondo es representado usando estadísticas de las principales características asociadas con objetos de fondos estacionarios y no estacionarios. Se propone un método nuevo para aprender y actualizar las características del fondo a cambios graduales y repentinos. Este método consiste en cuatro pasos fundamentales (9):

1. **Detección de cambio:** los píxeles inalterados son filtrados y separados en puntos estáticos y dinámicos acorde a los cambios del *interframe* (cambios entre un fotograma y otro).

2. **Clasificación de cambio:** dichos puntos obtenidos del paso uno son clasificados como fondo o frente usando el modelo bayesiano y las estadísticas de las características principales.
3. **Segmentación de objeto del frente:** los objetos detectados en el frente son segmentados mediante la combinación de los resultados de clasificación, tanto los puntos estáticos como los dinámicos.
4. **Mantenimiento o actualización del fondo:** se mantienen actualizados los fondos.

En la **Figura 2** se muestra un diagrama que representa los diferentes pasos mencionados anteriormente que dan cumplimiento al algoritmo. Los bloques en gris corresponden al paso cuatro y los bloques blancos de izquierda a derecha a los tres primeros pasos.

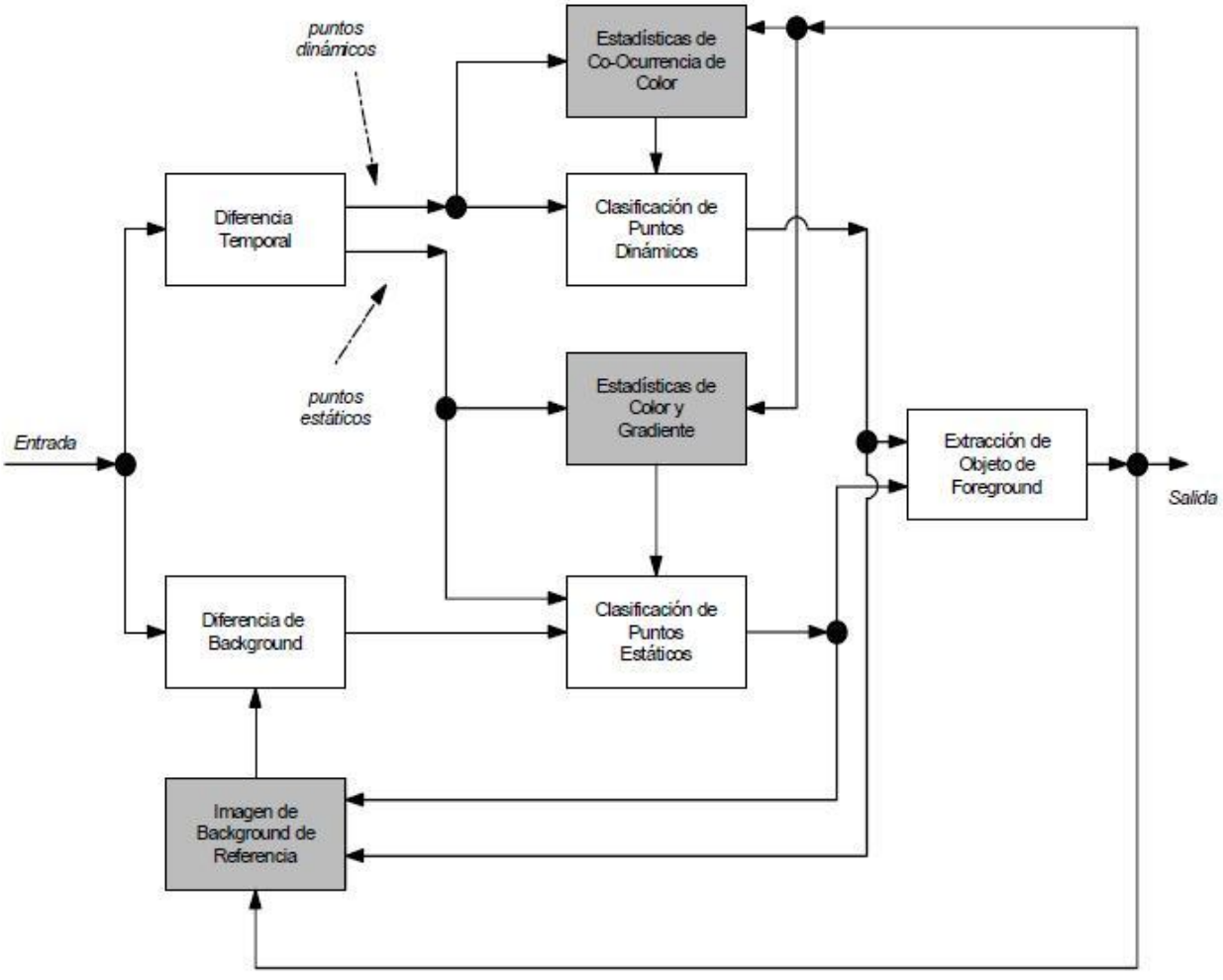

Figura 2. Diagrama de flujo del comportamiento de FGD **(9)**.

1.2.2. *Mixture of Gaussians (* MOG).

El MOG está basado en el método de mezcla de Gaussianas para la caracterización de los píxeles del fondo por lo que se encuentra dentro de los métodos de análisis imágenes a imágenes, clasificación mencionada anteriormente. Se caracteriza por tener en cuenta a la hora de modelar el fondo los posibles cambios de iluminación en la imagen, secuencias multimodales, objetos moviéndose lentamente, y el ruido introducido por la cámara.

Cada cierto tiempo los parámetros de las gaussianas son actualizados. Las gaussianas son evaluadas usando una simple heurística para hacer hipótesis de cuáles son más probables que sean parte del "proceso de fondo". Los valores del píxel que no concuerdan con los píxeles del "fondo" son agrupados usando componentes relacionados. Finalmente, los componentes relacionados son rastreados fotograma a fotograma usando un rastreador de hipótesis múltiple. En la siguiente figura se muestran los pasos principales que desarrolla este algoritmo en el procesado del fotograma.

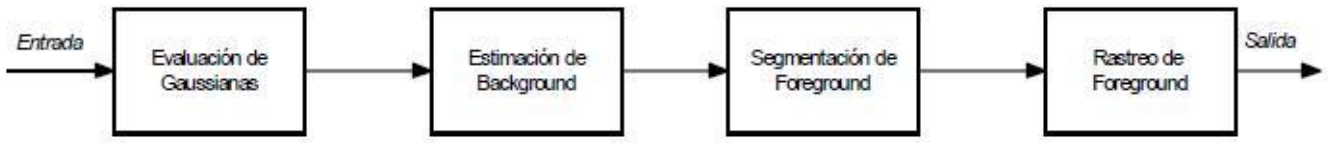

Figura 3. Muestra un diagrama del flujo de procesos del algoritmo MOG **(9)**.

Luego de un profundo estudio se decidió hacer uso del algoritmo MOG, siendo este el que más se adapta a las necesidades del sistema, ya que el FGD luego de haber detectado un objeto en movimiento en la escena y este detenerse pasa a formar parte del fondo, no recomendada su aplicación en esta investigación. El análisis de la selección incluyó un estudio realizado por Luis Rodríguez López en (9) ofreciendo datos estadísticos y resultados convincentes de la aplicación de dichos métodos.

1.2.3. Análisis de otras soluciones similares.

Después de realizar un análisis sobre otras soluciones existentes, se encontraron los trabajos que se muestran a continuación, los cuales han servido de guía para el desarrollo de esta investigación.

1.2.3.1. SENSOURCE.

SENSOURCE es un sistema para el análisis y detección de tráfico de clientes, desarrollado por la empresa **SenSource Inc.** destinada al desarrollo de videovigilancia y conteo de personas. Este sistema cuenta con un contador de personas, el cual incorpora tecnología moderna para la detección, colección de datos y reportes. El mismo incluye tecnología de fácil expansión y actualización. Esta aplicación hace uso de una tecnología avanzada llamada PC-THI60, utilizada en la proyección de imágenes termales para

contar el tráfico de personas desde un punto de vista superior en una puerta de entrada y salida. Esta tecnología permite rastrear continuamente las personas que entran o salen al mismo tiempo a través del procesamiento de las imágenes termales correspondientes a las personas que se mueven debajo del lente de la cámara (10).

1.2.3.2. Análisis de video: conteo de personas.

Su funcionalidad está basada en técnicas de análisis de imagen para el conteo de personas, estimación de afluencias y control visual de operaciones de terminales de venta, este es desarrollado por la empresa **Visual Tools S.A** la cual desarrolla, fabrica y comercializa productos y soluciones de video digital para el monitoreo, supervisión y gestión de cualquier tipo de instalación, independientemente del número de equipos y del tipo de vigilancia. Este sistema cuenta con funcionalidades basadas en las necesidades de cada cliente, posee equipos PeCo altamente capacitados que realizan conteo de personas basados en el análisis de video. Son equipos dotados de sensores inteligentes que analizan las imágenes de las cámaras de CCTV (por sus siglas en inglés *Closed Circuit Television).*

Dentro de las características de este producto se destacan:

1. Funcionamiento con cámaras estándar sin necesidad de un software ni de hardware adicional.
2. Alto porcentaje de precisión registrando con una fiabilidad del 95% todos los visitantes de una zona determinada, diferenciando entradas y salidas (bidireccional) y el paso de varias personas simultáneamente.
3. Datos de conteo para el control de operaciones, esto permite saber cuántos visitantes tiene un establecimiento dado, las horas y días de mayor afluencia.

Todo lo anterior posibilita tener un mayor conocimiento de un negocio específico y de esta manera tomar decisiones que aumenten su rentabilidad (11).

A nivel nacional, el desarrollo en este campo se encuentra concentrado en instituciones específicas y universidades como CENATAV[1], DATYS[2] y la UCI.

[1] Centro de Aplicaciones de Tecnologías de Avanzada, http://www.cenatav.co.cu

[2] DATYS, empresa cubana de software, http://www.datys.cu/

[3] SXP es un híbrido cubano de metodologías ágiles que tiene como base las metodologías SCRUM y XP.

1.2.3.3. Sistema Integral de Seguridad (Xyma Safe Suite).

Es un conjunto de soluciones tecnológicas de video monitoreo, videovigilancia y seguridad física orientadas a apoyar a las instituciones públicas y privadas en los esfuerzos para garantizar la seguridad de los ciudadanos y las organizaciones, está desarrollado por la empresa cubana DATYS.

Se basa en la Integración de tecnologías de telecomunicaciones, cómputo, aplicaciones de seguridad, visualización, almacenamiento y correlación de la información. El sistema está preparado para adaptarse a cambios tecnológicos continuos, con lo que se evita la obsolescencia de la tecnología, además de aplicar las mejores prácticas en este tipo de soluciones (12).

Integra aplicaciones desarrolladas por sus diferentes departamentos como:

- Circuito cerrado de televisión.
- Control de acceso.
- Control de citas y visitantes.
- Detección de intrusos.
- Detección de incendios.
- Aplicaciones de telefonía.

1.2.3.4. Desarrollo de un video sensor para el conteo de personas.

En el Proyecto de Videovigilancia SURIA, se desarrolló un trabajo de Diploma titulado “Desarrollo de un video sensor para el conteo de personas” por Lisandra Michelena Rodríguez. Este trabajo tenía como objetivos ampliar prestaciones al sistema de vigilancia SURIA agregando un componente video sensor con la capacidad de realizar conteos de personas a través de cámaras IP (13). Actualmente este componente forma parte del centro GEYSED, donde se encuentra en un proceso de integración a una plataforma de servicio. Este componente no se ajusta a las necesidades de esta investigación ya que fue creado específicamente para satisfacer las necesidades de SURIA. Este es enfocado hacia otro tipo de escenario de vigilancia, donde requiere una vista desde un ángulo superior y un enfoque directamente hacia abajo, detectando solo la parte superior de la personas (cabeza y hombros). No cuenta con una base de datos que le permita almacenar la información que más tarde el usuario podría necesitar, solo muestra datos estadísticos del momento en que está activo el sistema.

Luego de un estudio realizado a los sistemas existentes, se ha llegado a la conclusión que no cumplen con lo que la dirección del CISED desea, ya que **SENSOURCE** como el **Análisis de Video: Conteo de personas**, cuentan con una buena precisión de la información pero a su vez, tienen como desventajas que son propietarios, poseen un límite de comercialización, y son altamente costosos, ya que para su

desarrollo emplean tecnología de última generación como PC-THI60 y equipos PeCo. El **Sistema Integral de Seguridad** posee dentro de su conjunto de soluciones varias prestaciones pero no cuenta con una que permita el conteo de las personas. Por otro lado la solución desarrollada por Lisandra Michelena Rodríguez en (13) es un componente muy ligero creado específicamente para el Proyecto de Videovigilancia SURIA, satisfaciendo sus necesidades específicas. Además este pequeño componente no brinda la información necesaria para que el usuario pueda tomar sus propias decisiones, por estas razones se hace necesario buscar otra solución, la cual se propone en este trabajo, que se ajusta a las necesidades específicas del centro.

1.3. Metodología, tecnologías y herramientas a utilizar.

1.3.1. Selección de la metodología para el desarrollo de Software.

La selección de la metodología adecuada para el desarrollo de software es un factor determinante en el éxito de un proyecto. Aunque no existe una metodología mejor que otra, algunas se ajustan mejor a las características y necesidades específicas de los proyectos de desarrollo.

Dentro de las metodologías de desarrollo existen dos grandes grupos, las metodologías tradicionales y las metodologías ágiles. Las primeras enfatizan en el uso exhaustivo de documentación durante todo el ciclo de vida del proyecto y es recomendada para los proyectos con grandes equipos de desarrollo. Un ejemplo de esta es *Microsoft Solution Framework* (MSF) y *Rational Unified Process* (RUP) entre otras, siendo esta última la líder dentro del grupo. Por otra parte, las ágiles dan mayor importancia a la capacidad de respuesta a los cambios, se enfatiza en la satisfacción del cliente y promueve el trabajo en equipo. Dentro de este grupo se encuentra: *Feature Driven Development* (FDD), SCRUM y SXP[3] entre otras, pero una de las más usadas es *eXtreme Programming* (XP).

1.3.1.1. Proceso Unificado de Desarrollo de Software (RUP)

RUP tiene tres características esenciales: Está **dirigido por casos de uso**, donde los casos de uso definen lo que el usuario desea a partir de la captura de requisitos y la modelación del negocio, **centrado en la arquitectura**, característica que está relacionada con la toma de decisiones que indican cómo tiene que ser construido el sistema, la definición de la arquitectura debe tomar en consideración elementos de calidad del sistema, rendimiento, reutilización y capacidad de evolución por lo que debe ser flexible durante todo el proceso de desarrollo, **iterativo e incremental,** donde cada fase se divide en iteraciones, dividiendo el producto en pequeños proyectos para el desarrollo e incremento del mismo. Durante todo el

[3] SXP es un híbrido cubano de metodologías ágiles que tiene como base las metodologías SCRUM y XP.

proceso de desarrollo de software se producen versiones incrementales (que se acercan al producto terminado) del producto en desarrollo (14).

Ventajas:

- Evaluación en cada fase que permite cambios de objetivos.
- Funciona bien en proyectos de innovación.
- Es sencillo ya que sigue los pasos intuitivos necesarios a la hora de desarrollar el software.
- Seguimiento detallado en cada una de las fases.

Desventajas:

- La evaluación de riesgos es compleja.
- Excesiva flexibilidad para algunos proyectos.
- El cliente deberá ser capaz de describir y entender a un gran nivel de detalle, acordando así un alcance del proyecto con él.
- Está creado para proyectos de gran envergadura lo cual hace difícil su uso para pocas personas.

El ciclo de vida de esta metodología, posee 4 fases en su ciclo de vida, dichas fases poseen actividades las cuales son agrupadas en grupos lógicos, definiéndose 9 flujos de trabajo principales. Los 6 primeros son conocidos como flujos de ingeniería y los tres últimos como apoyo o soporte (Ver Anexo 1).

1.3.1.2. Programación Extrema (*eXtreme Programming*, XP).

Esta metodología está centrada en potenciar las relaciones interpersonales como clave para el éxito en desarrollo de software, promoviendo el trabajo en equipo, preocupándose por el aprendizaje de los desarrolladores, y propiciando un buen clima de trabajo. XP se basa en realimentación continua entre el cliente y el equipo de desarrollo, comunicación fluida entre todos los participantes, simplicidad en las soluciones implementadas y coraje para enfrentar los cambios. XP se define como especialmente adecuada para proyectos con requisitos imprecisos y muy cambiantes, donde existe un alto riesgo técnico. Los principios y prácticas son de sentido común pero llevadas al extremo, de ahí proviene su nombre (15). A continuación detallamos las características esenciales de XP.

Ventajas:

- Es más apropiado para la ejecución de proyectos a corto plazo por equipos de desarrollo pequeños (de 2 a 10 desarrolladores), así como en entornos volátiles y en proyectos de alto riesgo.
- Permite una mejor adaptabilidad a los cambios, que se traduce en una reducción de costos.
- Planificación a corto plazo y más transparente para los clientes, ya que conocen las fechas de entrega de funcionalidades vitales para su negocio.

- Permite tener retroalimentación continua de los usuarios a través de las entregas frecuentes.

Desventajas:

- A veces cuesta delimitar el alcance del proyecto con el cliente.

Como en otra cualquier actividad humana se necesitan valores, prácticas y variables para desarrollar el trabajo y conseguir los planteamientos iniciales, en XP se proponen los siguientes:

Valores:

- Comunicación
- Sencillez
- Retroalimentación
- Valentía

Prácticas:

El mayor beneficio de las prácticas se consigue con su aplicación conjunta y equilibrada puesto que se apoyan unas en otras. Esto se ilustra en el Anexo 2, donde una línea entre dos prácticas significa que las dos prácticas se refuerzan entre sí. El mérito de XP es integrarlas de una forma efectiva y complementarlas con otras ideas desde la perspectiva del negocio, los valores humanos y el trabajo en equipo.

La metodología XP fue seleccionada para el desarrollo de este trabajo teniendo en cuenta los siguientes elementos:

1.3.1.2.1. Fundamentación de XP como metodología a utilizar.

- Permite un grupo pequeño de desarrolladores.
- Necesidad de resultados tangibles a corto plazo.
- Los problemas de programación se resuelven más rápido, posibilita la transferencia de conocimientos de programación entre los miembros del equipo.
- Imposibilidad para un grupo de desarrollo pequeño, de asumir una metodología robusta debido a la cantidad excesiva de roles y documentación generada en el ciclo de vida del proyecto.
- La realimentación continua entre el cliente y el equipo de desarrollo.
- La producción de versiones del sistema de manera rápida.
- Diseñar la solución más simple que pueda funcionar y ser implementada en un momento determinado del proyecto.

1.3.2. Selección de la plataforma de trabajo.

.Net framework 4.0**,** es una moderna plataforma que facilita la creación de todo tipo de aplicaciones: de escritorio, de servidor, para XBOX o móviles. Se integra a la perfección en los sistemas operativos Windows. Está compuesto, por una parte, por el motor en tiempo de ejecución CLR[4], el cual administra la memoria, la ejecución de subprocesos y de código, realiza la comprobación de la seguridad del código y la compilación. Por otra parte, incluye la librería de clases base (BCL) orientada a objetos, cuyo uso reduce los tiempos de aprendizaje de las nuevas características de .NET Framework. .NET Framework 4.0, ha aumentado la compatibilidad con Surface 2.0 SDK y con las nuevas características de Windows 7 (16) . Además, ha incluido diversas mejoras relacionadas con:

- *Common Language Runtime* y la librería de clases base.
- Depuración de código.
- ASP.NET.

1.3.3. Lenguaje de programación.

1.3.3.1. C#

C# fue creado por el danés Anders Hejlsberg que diseñó también los lenguajes Turbo Pascal y Delphi. El C**#** (pronunciado en inglés “C *sharp*” o en español “C sostenido”) es un lenguaje de programación orientado a objetos. Con su surgimiento se quiso mejorar con respecto a los dos lenguajes anteriores de los que deriva, el C y el C++.

C# es un lenguaje de programación de uso generalmente sencillo. Visual C# ofrece a los desarrolladores herramientas eficaces centradas en código, compatibilidad de lenguajes para crear aplicaciones Web completas y conectadas en .NET Framework. El lenguaje C# también incluye soporte para la programación orientada a componentes.

C# 4.0 es la versión del potente lenguaje de programación incluido en el .NET Framework 4.0, abarca una notable variedad de características: tipiado dinámico, excepciones, genéricos, colecciones, LINQ (por sus siglas en inglés, *Language Integrated Query*), eventos, delegados, métodos anónimos, patrones, etc.

Tiene como principal característica que se inserta como una opción para programación dinámica. Este lenguaje sigue siendo esencialmente de tipo estático, aunque se ha agregado la capacidad de interactuar de manera más eficaz con objetos dinámicos (17). Su utilización sirvió para el desarrollo de los

[4] Common Language Runtime o CLR (Lenguaje común en tiempo de ejecución) es el componente de máquina virtual de la plataforma .Net de Microsoft.

componentes Plugin detector de personas, Control de flujo y en conjunto con ASP.NET para la Aplicación de reportes.

1.3.3.2. ASP.NET

Para el desarrollo web se utilizó ASP.NET**,** que es un *framework* desarrollado y comercializado por Microsoft. Es usado para construir sitios web dinámicos, aplicaciones web y servicios web XML. Apareció en enero de 2002 con la versión 1.0 del .NET Framework, y es la tecnología sucesora de *Active Server Pages* (ASP). ASP.NET está construido sobre el *Common Language Runtime*, permitiendo a los programadores escribir código ASP.NET usando cualquier lenguaje admitido por el .NET Framework. Corre código compilado sobre el entorno NGWS[5] en el servidor. Distinto a sus predecesores interpretados, ASP.NET usa amarres tempranos ("*early binding*"), así como compilación justo a tiempo ("*just-in-time compilation*"), optimización nativa y servicios de caché sin configuración adicional. Esto significa eficiencia dramáticamente superior antes de escribir la primera línea de código (18).

1.3.4. jQuery

jQuery es un *framework* de JavaScript *open source*, que funciona en múltiples navegadores. Su objetivo principal es hacer la programación "scripting" mucho más fácil y rápida del lado del cliente. Con jQuery se pueden producir páginas dinámicas así como animaciones parecidas a Flash en relativamente corto tiempo. Este se encuentra bajo la licencia MIT[6] (19).

jQuery permitió agregar efectos y funcionalidades a la aplicación web, como la validación de formularios y calendarios, mejorando así la interfaz de la web.

- Es flexible y rápido para el desarrollo web.
- Se encuentra bajo la licencia MIT y es open source.
- Cuenta con una excelente comunidad de soporte.

1.3.5. Librería de visión OpenCV.

OpenCV es una librería libre de visión artificial originalmente desarrollada por Intel en el año 1999 y publicada bajo licencia BSD[7], permite que sea usada libremente para propósitos comerciales y de

[5] New Generation Windows Services (Nueva Generación de Servicios Windows).

[6] Instituto Tecnológico de Massachusetts (MIT, *Massachusetts Institute of Technology*).

[7] **BSD (Berkeley Software Distribution):** es una licencia de software libre permisiva como la licencia de OpenSSL o la MIT License.

investigación con las condiciones en ella expresadas. La librería es multiplataforma y puede ser usada en Mac OS X, Windows y Linux. Está orientada al tratamiento de imágenes en tiempo real, lo que es posible si se encuentran en el sistema las Primitivas de Rendimiento Integradas de Intel (IPP).

OpenCV presenta un conjunto de funciones implementadas en lenguaje C, que permitió el procesamiento de imágenes. OpenCV posee una estructura modular, está compuesto por diferentes módulos, dentro de los cuales se destacan los siguientes (20):

- Core: en este módulo se encuentran las estructuras básicas de OpenCV como Mat (estructura que representa matrices) y funciones básicas que utilizan el resto de los módulos.
- *Imgproc*: este módulo se encarga del procesamiento de imágenes mediante algunas de las operaciones: filtrado, transformaciones geométricas, conversiones de color, histogramas, entre otros.
- Video: módulo para el análisis de videos que incluye estimación de movimiento, sustracción de fondo y seguimiento de objetos.
- *Objdetect*: módulo de detección de objetos e instancias de los tipos predefinidos como por ejemplo personas, autos, rostros y ojos entre otros.

1.3.6. Liberia EmguCV.

EmguCV es una plataforma de encapsulamiento que contiene a la librería OpenCV de procesamiento de imágenes (21). También es conocida como un envoltorio de la librería OpenCV que permitió que sus funciones, pudieran ser llamadas desde C#.

En el Anexo 3 se muestra un esquema de la descripción de la arquitectura de EmguCV con dos capas de envoltura.

1.3.7. Entorno de desarrollo integrado (IDE, por sus siglas en inglés).

Microsoft Visual Studio 2010 Ultimate proporciona un entorno integrado de herramientas y la infraestructura de servidor que simplifica el proceso de solicitud de desarrollo. Ofrece resultados empresariales usando procesos productivos, predecibles y personalizables, aumenta la transparencia y el seguimiento durante el ciclo de vida con análisis detallados. Permite la creación de nuevas soluciones o mejoras a las aplicaciones existentes con la creación de un prototipo de gran alcance, arquitectura y herramientas de desarrollo que le permiten llevar su visión a la vida, dirigidas a un número creciente de plataformas y tecnologías, incluyendo las nubes y la computación paralela. Permite comprender el incremento de la productividad del equipo mediante la utilización de características avanzadas de

colaboración y el uso integrado de las pruebas y herramientas de depuración para encontrar y corregir errores de forma rápida y sencilla (22).

1.3.8. Herramienta CASE[8].

Como herramienta CASE se utilizó el Visual Paradigm que propicia un conjunto de ayudas para el desarrollo de programas informáticos, desde la planificación, pasando por el análisis y el diseño, hasta la generación del código fuente de los programas y la documentación. Soporta el ciclo de vida completo del desarrollo de software: análisis y diseño orientados a objetos, construcción, pruebas y despliegue. También proporciona abundantes tutoriales de UML, demostraciones interactivas y proyectos. Presenta licencia gratuita y comercial. Es fácil de instalar, actualizar y compatible entre ediciones (23). Su utilización permitió dibujar todos los tipos de diagramas de clases, de componentes y de despliegue así como generar código desde diagramas. Entre sus principales características se destacan:

- Disponibilidad en múltiples plataformas.
- Entorno de creación de diagramas para UML.
- Capacidades de ingeniería directa e inversa.
- Diseño centrado en casos de uso, equivalente a las historias de usuario en XP y enfocado al negocio que genera un software de mayor calidad.
- Uso de un lenguaje estándar común a todo el equipo de desarrollo que facilita la comunicación.

1.3.9. Selección del gestor de base de datos.

1.3.9.1. MySQL.

MySQL es un sistema multiplataforma de manejo, creación y administración de base de datos *open source* (*Database Management System*, DBMS) para bases de datos relacionales. Cuenta con un completo sistema multihilo, que ofrece un soporte completo para diferentes formas de manera eficiente y veloz, permitiendo acceder a todos los campos que resguardan los datos de trabajo (24).

Características:

- Cuenta con la capacidad de realizar tareas multiprocesador, debido a que posee la opción de trabajo multihilo.
- Puede ingresar una enorme cantidad de datos por columna de trabajo.
- Cuenta con API's disponibles para los principales lenguajes de programación que existen.
- Aplicación con una portabilidad sobresaliente.

[8] CASE del inglés Computer Aided Software Engineering, que significa Ingeniería de Software Asistida por Computadora.

- Capacidad de soportar hasta 32 índices de tablas diferentes.
- Un nivel de seguridad que permite gestionar varios usuarios mediante contraseñas individuales.
- Sistema de sub-consultas un poco arcaico en relación a opciones más modernas, lo que implica que el desarrollador tenga que buscar opciones más complicadas para solventar esta situación.
- Todavía se espera la inclusión de diversas funciones de Oracle.

1.3.9.2. PostgreSQL

PostgreSQL es un servidor de base de datos relacional orientado a objetos de software libre. Como muchos otros proyectos, es de código abierto, por lo que el desarrollo de PostgreSQL no es manejado por una sola compañía sino que es dirigido por una comunidad de desarrolladores y organizaciones comerciales, las cuales trabajan en su desarrollo. Algunas de sus principales características son la alta concurrencia. Mediante un sistema denominado MVCC (por sus siglas en inglés Acceso concurrente multiversión) PostgreSQL permite que mientras un proceso escribe en una tabla, otros accedan a la misma tabla sin necesidad de bloqueos. Cada usuario obtiene una visión consistente de lo último que se actualizó (25).

Ventajas:

- Es compatible en los principales sistemas operativos: Linux, Unix, BSDs, Mac OS, Beos, Windows, etc.
- Soporte nativo para los lenguajes más populares: PHP, C, C++, Perl, Python, etc.
- Gran escalabilidad, se ajusta al número de CPUs y a la cantidad de memoria que posee el sistema de forma óptima, permitiéndole soportar una mayor cantidad de peticiones simultáneas de manera correcta (en algunos casos, se dice que ha llegado a soportar el triple de carga de lo que soporta MySQL).
- Soporta distintos tipos de datos, además del soporte para los tipos base, también soporta datos de tipo fecha, monetarios, elementos gráficos, datos sobre redes (MAC, IP), cadenas de bits y permite la creación de tipos propios de datos.
- Permite la gestión de diferentes usuarios y los permisos asignados a cada uno de ellos.
- Puede comprobar la integridad referencial y almacenar procedimientos en la propia base de datos, equiparándolo con los gestores de base de datos de alto nivel, tales como Oracle.

Se decide utilizar PostgreSQL en la aplicación que se desea desarrollar, pues sus ventajas y prestaciones son amplias y ajustables al objetivo propuesto. Es un sistema potente y multiplataforma, lo que permite el

constante desarrollo y perfeccionamiento de sus funcionalidades, está publicado bajo licencia BSD que pertenece al grupo de licencias de software libre.

1.3.10. RabbitMQ

RabbitMQ 3.0.2 es un software robusto de negociación de mensajes de código abierto que entra dentro de la categoría de middleware de mensajería. Implementa el estándar *Advanced Message Queuing Protocol* (AMQP). El servidor RabbitMQ está escrito en *Erlang* y utiliza el *framework Open Telecom Platform* (OTP) para construir sus capacidades de ejecución distribuida y conmutación ante errores. El código fuente está liberado bajo la licencia Mozilla Public License (26). Es fácil de usar y multiplataforma, soportando una gran cantidad de plataformas para desarrolladores. Este software sirvió de ayuda con respecto al manejo de las colas, almacenando los datos en un sistema seguro y consistente, esto permitiendo que luego se puedan consumir los datos definidos en esta.

1.3.11. iSpy

iSpy es un software que se encarga de manipular cámaras. La versión utilizada, iSpy 4.8.4 puede ser configurada para trabajar con distintas fuentes de entrada, ya sea una cámara web convencional, cámara IP, tarjeta de captura de escritorio, cámara, micrófono o video. La configuración de una cámara para iSpy es sencilla, a su vez permite la incorporación de nuevos *plugin* a su estructura para la manipulación de las cámaras (27).

iSpy se puede ejecutar en varios ordenadores simultáneamente y puede utilizar el servidor web integrado o el sitio web del desarrollador para ver los archivos multimedia capturados, fuentes de cámara web en vivo y controlar de forma remota iSpy. El programa es un software gratuito y de código abierto, que cuenta con las siguientes características:

- Conecta y supervisa tantas cámaras y micrófonos como se desee.
- Conecta varios ordenadores en un grupo y los gestiona a través de la web.
- Detecta, resalta, da seguimiento y graba el movimiento.
- Personaliza las áreas de detección de movimiento en las cámaras.
- Detecta y graba el sonido
- Envía un correo electrónico o SMS de alerta al detectar movimiento o sonido.
- Recibe alertas de movimiento por correo electrónico con imágenes adjuntas seleccionadas del marco desde las cámaras web.
- Permite el control y acceso de forma remota.
- Permite la protección por contraseña.

- Registra el tiempo transcurrido desde cualquier cámara.
- Permite la inclusión de nuevos *plugin*.

1.3.12. Telerik OpenAccess

Telerik OpenAccess es una herramienta de mapeo objeto-relacional (ORM) que apoya el desarrollo de aplicaciones orientadas a datos y proporciona persistencia transparente que se ajusta plenamente a .NET. Proporciona una capa de acceso a datos inteligente, que brinda servicios de persistencia para construir efectivamente aplicaciones orientadas a objetos bien diseñadas.

Con su uso, los arquitectos y desarrolladores tendrán la capacidad de concentrarse en la lógica de los problemas que están resolviendo, en lugar de resolver los problemas con los motores de datos utilizados para almacenar y recuperar los datos. Permite a los desarrolladores trabajar con los datos en forma de objetos y gráficos de objetos específicos de dominio, sin tener que preocuparse de las tablas de base de datos subyacente y columnas donde se almacenan los datos reales. Esto aumenta el nivel de abstracción en el que los desarrolladores pueden trabajar cuando se trata de los datos y reduce el código que se requiere para crear y mantener aplicaciones orientadas a datos (28).

Beneficios de su utilización:

- Las aplicaciones pueden trabajar con los datos persistentes utilizando las prácticas de desarrollo orientadas a objetos, incluyendo tipos con herencia, miembros complejos y relaciones.
- Las solicitudes son liberadas de las capas de acceso a datos no modificables y dependencias en un almacén de datos en particular o el esquema de almacenamiento.
- Las asignaciones entre el modelo de datos y el esquema de almacenamiento específicos pueden variar realizando mínimos cambios en el código de la aplicación de usuario.
- Los desarrolladores pueden trabajar con datos de la persistencia del modelo de objetos que se pueden asignar a varios almacenes de datos.
- Varios modelos de la persistencia de datos se pueden asignar a un solo esquema de almacenamiento. Asimismo un modelo de dominio único podría ser asignado a varios almacenes de datos o esquemas.
- Proporciona la validación de sintaxis en tiempo de compilación de consultas en un modelo de persistencia de datos mediante el apoyo de *Language-Integrated Query* (LINQ).

1.4. Conclusiones Parciales.

En el estudio realizado de este capítulo se demostró la necesidad de desarrollar un sistema para la detección y conteo de personas a través de una cámara. Para esto:

- Mediante el estudio de los conceptos asociados a la solución se pudo obtener un mejor dimensionamiento y entendimiento del problema.
- El estudio de las diferentes técnicas de extracción del fondo permitió obtener conocimiento referente a los métodos existentes, para seleccionar el adecuado a las características del problema planteado.
- El estudio de los sistemas existentes en el ámbito nacional e internacional demostró que no existe una solución que resuelva la situación presente en el CISED, por lo cual se propone la implementación del **Sistema para el control de flujo de personas en imágenes de video en tiempo real.**
- Además se seleccionaron la metodología, tecnologías y herramientas, potenciando así la satisfacción de los requisitos establecidos y posibilitando la obtención de un producto con la calidad requerida.

Capítulo II: Características del Sistema.

Introducción

Este capítulo está dedicado a las características del sistema. En él se estará tratando la propuesta de solución para responder a la situación problémica, además quedarán definidas las principales funcionalidades y los requisitos no funcionales, las funcionalidades se describen mediante las historias de usuario. Se realiza el plan de entrega, en el cual se indican las historias de usuario que se crearán para cada versión de la aplicación y las fechas en las que se publicarán estas versiones. Se realiza el plan de iteraciones donde se muestran las HU que se realizarán en cada iteración según su prioridad en el negocio.

2.1. Propuesta de solución.

El presente trabajo propone el desarrollo de tres componentes que integrados, le permitirán al CISED conocer la cantidad de personas que se encuentran físicamente en el centro en un determinado momento del día, conocer los datos estadísticos de un día, así como el nivel de afluencia de personas hacia el centro y viceversa.

El componente **Plugin detector de personas**, permitirá la conexión con el iSpy, donde obtendrá los fotogramas que este brinda, luego se procederá a la segmentación del fondo del fotograma y a la eliminación del ruido mediante la aplicación de los diferentes filtros (tamaño del blob, área del rectángulo englobante, aspecto del blob y *Smooth_gaussian*). Posteriormente se detectan las personas que puedan existir en el fotograma, obteniendo su posición y almacenándola en la cola del RabbitMQ.

El componente **Control de flujo** de tipo *notifyIcon*, se encargará de extraer los datos de la cola del RabbitMQ. Luego de extraídos los datos de la cola correspondientes a la persona (id, imagen recortada del rectángulo englobante, posición actual de la persona, ancho y alto del fotograma), se contará con una lista donde se almacenarán los datos de la personas extraídas. En cada iteración se verificará si la persona que está en el frente de la cola ya está en dicha lista comprobando su id (único para cada persona en la escena), en caso que ya esté en la lista, solamente se le actualiza su posición, guardando en memoria su posición anterior antes de actualizarla, en caso contrario se añade a la lista.

Posteriormente se definirá el sentido del movimiento de la persona y se aplicará una función matemática basada en la contención o no de un punto(X, Y) dentro de un rectángulo, dicha función será aplicada para poder contar la persona. El rectángulo será definido en la pantalla según el área donde se decida realizar

el conteo, para esto se utiliza el ancho y el alto del fotograma. La funcionalidad implementada se basa en si la posición actual de la persona está contenida dentro del rectángulo y la posición anterior no está contenida, en este caso se puede decir que la persona está entrando al área de conteo. En caso contrario si la posición actual de la persona no está contenida en el rectángulo y la anterior si está contenida, se puede afirmar que la persona está saliendo del área de conteo, en otro caso, la acción que ejecute la persona no es de interés. En el Anexo 4 se muestra un fotograma donde se aplica el proceso anteriormente descrito.

Luego de esto se almacenará en la base de datos la información correspondientes a cada persona en el instante que se ejecutó la acción de entrada o salida, dígase id de la persona, fecha y hora exacta que realizó la acción, imagen recortada del rectángulo englobante, tipo de acción que realizó. El tipo de acción se almacenará en una variable de tipo booleana, tomando valor *true* en caso de la entrada y valor *false* en caso contrario.

Finalmente la **Aplicación de reportes** se encargará de extraer de la base de datos la información correspondiente a la solicitud que ejecute el usuario como: consultar los datos estadísticos en tiempo real, consultar datos estadísticos de un día determinado y comparar datos estadísticos entre dos días.

En la **Figura 4** se muestra lo expresado anteriormente:

Figura 4. Propuesta de solución.

2.2. Modelo de dominio.

Dada la propuesta anterior se propone la definición de un modelo de dominio o modelo conceptual el cual mostrará los principales conceptos a utilizar en el desarrollo de este sistema. Se decide emplear el modelo conceptual ya que se cuenta con pocos datos, los cuales no brindan gran cantidad de información para poder crear un modelo de negocio.

El modelo de dominio como se muestra en la **Figura 5,** contempla la representación de conceptos u objetivos que son importantes dentro de un problema. Estos conceptos representan y simbolizan objetos del mundo real que se encuentran dentro del negocio y ofrecen a su vez un entendimiento del problema en cuestión.

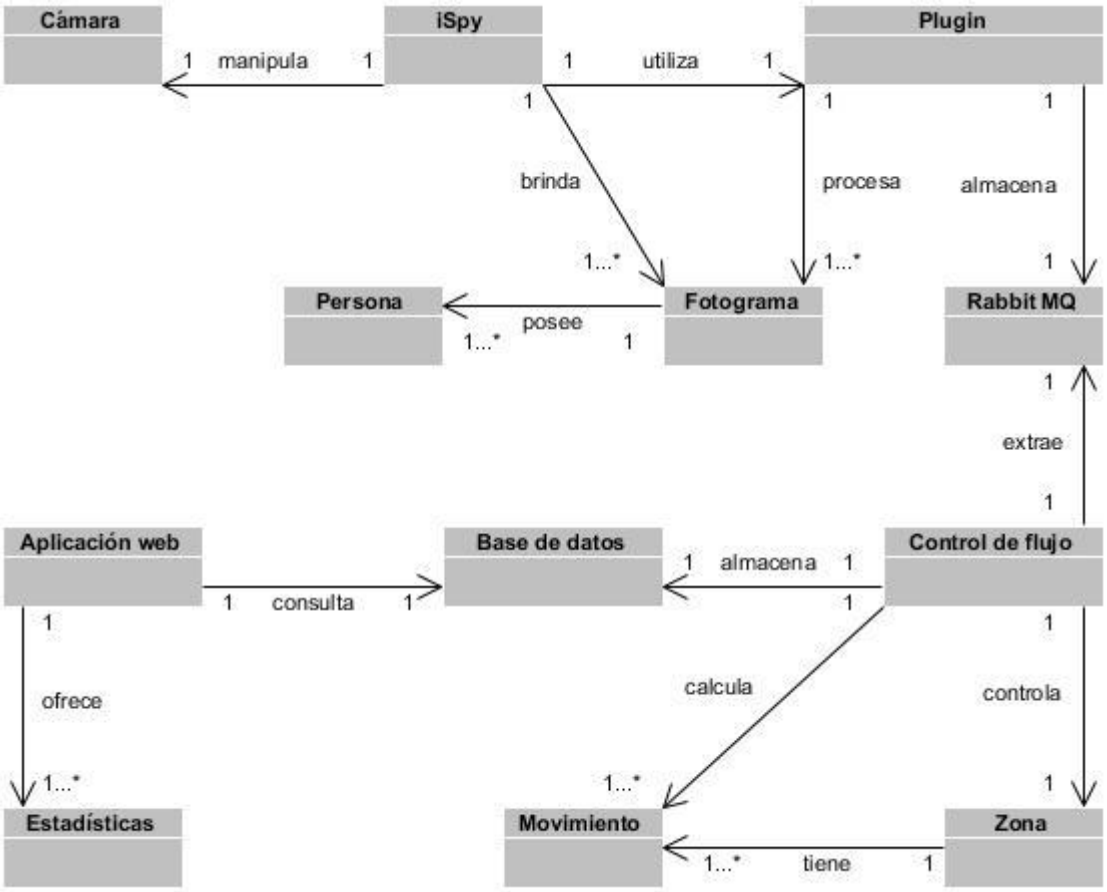

Figura 5. Modelo de dominio.

2.2.1. Glosarios de términos del modelo de dominio.

- Cámara: Dispositivo de captura del video.
- iSpy: Manipula cámara.
- Plugin: Componente que almacena muchas personas en el RabbitMQ.
- RabbitMQ: Componente que controla una cola.
- Control de flujo: Determina la dirección de movimiento de las personas en la escena.
- Fotograma: Una imagen determinada de la escena.
- Persona: Objeto localizado en el Fotograma.
- Zona: Área definida para ser controlada.
- Movimiento: Acción detectada en la escena.
- Base de datos: Almacena información.

- Aplicación web: Extrae la información de la Base de datos.
- Estadísticas: Resultados del Control de flujo.

2.3. Principales funcionalidades.

Como principales funcionalidades para guiar el desarrollo de este sistema y satisfacer las necesidades del cliente se han detectado las siguientes, las cuales se agrupan por componentes:

El componente **Plugin detector de personas**:

- Obtener los fotogramas del iSpy.
- Segmentar el fondo del fotograma.
- Eliminar ruido en fotograma.
- Detectar persona.
- Almacenar información a la cola del RabbitMQ.

El componente **Control de flujo**:

- Extraer información de la cola del RabbitMQ.
- Calcular dirección del movimiento.
- Contar persona.
- Almacenar información en la base de datos.

El componente **Aplicación de reportes**:

- Extraer información de la base de datos.
- Consultar datos estadísticos en tiempo real.
- Consultar datos estadísticos de un día determinado.
- Comparar datos estadísticos entre dos días.

2.4. Requerimientos no funcionales.

Los requerimientos no funcionales son propiedades o cualidades que el producto debe tener. Las propiedades o cualidades que hacen este producto más confiable y robusto se definen a continuación:

Apariencia o Interfaz Externa

- La interfaz debe contar con un diseño sencillo y de fácil uso, que contenga sólo los gráficos necesarios, para acelerar la velocidad de respuesta hacia el usuario.

Rendimiento

- Los tiempos de respuestas de la aplicación deben ser menores a cuatro segundos.

Disponibilidad

- El componente funcionará las 24 horas, los siete días de la semana.

Software

- El sistema debe funcionar sobre el sistema operativo Windows XP o superior.
- Debe desarrollarse sobre la plataforma .NET en C# 4.0 y ASP.NET 4.0 como lenguajes de programación.
- Utiliza la librería EmguCV 2.3.2 para el procesamiento de las imágenes.
- Debe haber un servidor iSpy 4.8 que proporcione los videos.
- Se debe tener PostgreSQL 9.1 para el manejo de la base de datos.
- Se debe contar con el manejador de colas de mensajes RabbitMQ 3.0 para el envío de información.

Hardware

En el **iSpy:**

- Se debe contar con el sistema operativo Windows XP o superior.
- Un microprocesador CPU 1.0 GHz o superior.

En el **RabbitMQ:**

- Un microprocesador CPU 3.0 GHz o superior y 1 GB de memoria RAM o superior.

2.5. Metáfora

La metáfora para el sistema es la historia que todos pueden contar acerca de cómo este funciona. La tarea de elegir una metáfora para el sistema permite mantener la coherencia de nombres de todo aquello que se va a implementar, es crear una visión global y común del sistema que se pretende desarrollar. El nombre de los objetos o partes del sistema es muy importante.

El sistema para el control de flujo de personas en imágenes de video en tiempo real está compuesto por tres componentes. Este puede integrarse a cualquier sistema de videovigilancia que posea una cámara, un sistema que las manipule y que permita llevar un control del flujo de personas. Este sistema toma los fotogramas del iSpy (encargado de manejar las cámaras disponibles en la institución), mediante el primer componente (Plugin detector de personas) que a su vez se encarga de detectar a las personas que se encuentren en el fotograma y su posición actual. El fotograma adquirido primeramente se le elimina el ruido que pueda tener mediante la aplicación de filtros y se desechan los objetos detectados que no sean de interés. Después de procesado el fotograma, esta información (los blobs, que son las personas

detectadas y su posición actual) será enviada al manejador de colas de mensajes RabbitMQ, encargado de organizar la información que le sea enviada.

El segundo componente (Control de flujo) se encargará de extraer de la cola la información enviada por el Plugin detector de personas. Luego de extraídos de la cola del RabbitMQ los datos correspondientes de la persona, se procede a calcular la dirección de movimiento, para ello se tiene en cuenta la posición de la persona en el fotograma actual y en el anterior. Después de calculada la dirección de movimiento se procede a contar la persona, dicho conteo se lleva a cabo aplicando una función matemática basada en si un rectángulo contiene un punto o no, donde el rectángulo será la zona de conteo definida. Por otra parte el usuario podrá conocer mediante el tercer componente de tipo web los datos estadísticos actuales del flujo, así como un historial del mismo.

2.6. Historias de usuario.

Las historias de usuario (**HU**) son utilizadas en XP para especificar los requisitos del software, las que son escritas por los clientes como las tareas que el sistema debe tener y su construcción depende principalmente de la habilidad que tenga el cliente para definirlas. Estas son descripciones cortas y escritas en el lenguaje del usuario sin terminología técnica y proporcionan los detalles sobre la estimación del riesgo y el tiempo que llevará su implementación (15).

Durante la fase de Planificación se identifican diferentes historias de usuario que se organizan por prioridad, las cuales indican las funcionalidades que el sistema debe cumplir. A continuación se describen algunas historias de usuario (Ver el resto de las HU en Anexo 5).

Tabla 1. Historia de usuario “Obtener los fotogramas del iSpy”.

Historia de Usuario	
Número: HU_1	**Usuario:** Sistema
Nombre historia: Obtener los fotogramas del iSpy.	
Prioridad en negocio: Media	**Riesgo en desarrollo:** Medio
Puntos estimados: 1	**Iteración asignada:** 1
Programador responsable: Aynel Cruz Barrera.	
Descripción: La presente historia de usuario tiene como objetivo obtener los fotogramas del iSpy mediante la selección del Plugin detector de personas, previamente reconocido.	

Observaciones: El iSpy debe contar con el Plugin detector personas.

Tabla 2. Historia de usuario "Procesar imagen".

Historia de Usuario	
Número: HU_2	**Usuario:** Sistema
Nombre historia: Procesar imagen.	
Prioridad en negocio: Alta	**Riesgo en desarrollo:** Alto
Puntos estimados: 3	**Iteración asignada:** 1
Programador responsable: Aynel Cruz Barrera.	
Descripción: La presente historia de usuario tiene como objetivo procesar el fotograma obtenido del iSpy, Donde se procede a la extracción del frente del fotograma mediante la utilización del algoritmo MOG. Con la aplicación de filtros: área del rectángulo englobante y el *Smooth_gaussian* se eliminarán los ruidos que puedan surgir, ya sean introducidos por la cámara o detectados en la aplicación del algoritmo de segmentación.	
Observaciones: Haber obtenido el fotograma previamente.	

Tabla 3. Historia de usuario "Detectar personas".

Historia de Usuario	
Número: HU_3	**Usuario:** Sistema
Nombre historia: Detectar personas.	
Prioridad en negocio: Alta	**Riesgo en desarrollo:** Alto
Puntos estimados: 1	**Iteración asignada:** 1
Programador responsable: Daniel Hernández Díaz.	
Descripción: La presente historia de usuario tiene como objetivo detectar a las personas en los fotogramas. Para la detección se utilizan los filtros de aspecto y tamaño del blob, estos son encargados de diferenciar entre objetos y personas en movimiento. Luego de la obtención de los blob(s) previamente filtrados, se tiene una imagen limpia, lista para realizar la detección de personas y luego realizar su seguimiento en la escena.	
Observaciones: Se le deben haber aplicado los filtros de eliminación de ruido al fotograma previamente. En caso de que no se detecten correctamente a las personas existirán errores estadísticos, afectando los resultados.	

Tabla 4. Historia de usuario "Almacenar datos en la cola".

Historia de Usuario	
Número: HU_4	**Usuario:** Sistema
Nombre historia: Almacenar datos en la cola del RabbitMQ.	
Prioridad en negocio: Alta	**Riesgo en desarrollo:** Alto
Puntos estimados: 1	**Iteración asignada:** 1
Programador responsable: Daniel Hernández Díaz.	
Descripción: La presente historia de usuario tiene como objetivo primeramente establecer la comunicación con el manejador de cola RabbitMQ. Luego de establecida la conexión, se procederá a enviar los datos a la cola.	
Observaciones:	

2.7. Tarjetas CRC.

Las tarjetas CRC (*Class*, *Responsabilities* y *Collaborators*) se refieren a clases, responsabilidades y colaboraciones, son estos precisamente los elementos fundamentales de estas tarjetas, o sea, para su confección se identifican las responsabilidades del software, los objetos que las ejecutan, las clases que los definen, así como aquellos objetos que colaboran en una determinada responsabilidad. Las tarjetas CRC son una técnica utilizada en XP para diseñar la solución informática según el paradigma orientado a objetos. A continuación se hace referencia a las clases identificadas y se muestran las tarjetas CRC de las principales por cada componente (Ver el resto de las tarjetas CRC en Anexo 6).

Para lograr una buena organización durante el desarrollo del componente Plugin detector de personas, se detectó como clase principal la clase Main. Esta clase es la encargada de obtener los fotogramas y procesarlos haciendo uso de la clase DatoPersona, donde permanecen los datos de la persona en tiempo de ejecución para que luego estos sean serializados y enviados a la cola de RabbitMQ mediante el Comunicador.

El componente Control de flujo es el encargado de contar a las personas y guardar dicha información en la base de datos mediante la clase ContarPersona haciendo uso de los diferentes componentes de la librería EmguCV y la inclusión de ComunicadorRabbitMQ.dll para la comunicación con el RabbitMQ.

El componente Aplicación de reportes de tipo web, cuenta con las clases Principal, Historial y EntitiesModel siendo esta última la intermediaria entre las dos primeras y la base de datos, haciendo uso del ORM Telerik OpenAccess.

Tabla 5. Tarjeta CRC Main del componente Plugin detector de personas.

Main	
Responsabilidades	Colaboradores
Obtener los fotogramas del iSpy.	• iSpy
Procesar imagen.	• Emgu.CV.dll • Emgu.GPU.dll • Emgu.UI.dll • Emgu.Util.dll
Detectar persona.	• Emgu.CV.dll • Emgu.UI.dll
Almacenar datos en la cola del RabbitMQ.	• Serializador • Comunicador

Tabla 6. Tarjeta CRC ContarPersona del componente Control de flujo.

ContarPersona	
Responsabilidades	**Colaboradores**
Establecer comunicación con el RabbitMQ.	• ComunicadorRabbitMQ.dll • RabbitMQ.Client.dll
Calcular dirección del movimiento.	–
Contar Persona.	–
Almacenar información en la base de datos.	• AccesoBD.dll

Tabla 7. Tarjeta CRC Principal del componente Aplicación de reportes.

Principal	
Responsabilidades	**Colaboradores**
Graficar datos estadísticos en tiempo real.	• EntitiesModel
Mostrar cantidad de personas que permanezcan dentro del centro como la cantidad de entadas y salidas hasta el momento.	• EntitiesModel

2.8. Diagrama de clases del diseño.

Luego de realizadas las tarjetas CRC se procede a confeccionar el diagrama de clases del diseño. Este diagrama aunque no es un artefacto propio de XP, permitirá representar gráficamente las clases con sus atributos y métodos, así como las relaciones entre clases con sus dependencias y asociaciones, además de la navegabilidad.

En los diagramas de clases del diseño del sistema (Ver Anexo 7) puede observarse la estructura de los componentes Plugin detector de personas, Control de flujo y Aplicación de reportes, los cuales comparten un flujo de datos que utilizan para interactuar entre ellos.

2.9. Estimación de tiempo.

En el período de desarrollo de la fase de planificación se realiza una estimación del esfuerzo que costará implementar las historias de usuario. Este se expresa utilizando como medida puntos estimados, donde un punto representa la semana donde el equipo trabaja el tiempo planeado sin ningún tipo de interrupción. Los resultados en esta estimación de acuerdo a las HU se exponen en el Anexo 8**.**

2.10. Plan de iteraciones.

El plan de iteraciones se realiza a raíz de las HU. Estas HU son traducidas como tareas de programación que van a ser desarrolladas y probadas a través de un ciclo de iteraciones, según el orden establecido.

Las pruebas se van a realizar al final del ciclo en que se esté desarrollando, al igual que en los ciclos siguientes, para comprobar que las siguientes iteraciones no han afectado a las anteriores. De esta manera las pruebas que hayan fallado en el ciclo anterior podrán ser evaluadas para su corrección y prever de esta manera que no vuelvan a ocurrir. Este plan consta de tres iteraciones.

- **Iteración 1**

Se van a implementar las HU que el grupo considera que tenga mayor prioridad, a las cuales se les van a realizar pruebas, dando a la aplicación las primeras funcionalidades, enfocándose en la HU_1, HU_2, HU_3 y HU_4.

- **Iteración 2**

Se van a estar desarrollando las HU_5, HU_6, HU_7 y HU_8 también se corregirán los errores o inconformidades del cliente con las HU implementadas en la iteración anterior.

- **Iteración 3**

Se realiza la implementación de la HU_9, HU_10, HU_11 y HU_12. De esta forma se obtiene la versión 1.0 del producto final.

2.11. Plan de entrega.

Se mostrarán las versiones que se le estarán entregando al cliente al finalizar cada iteración, siendo la versión 1.0 del producto final al concluir la iteración 3 ver Anexo 9.

2.12. Arquitectura de Sistema.

La arquitectura del software es el diseño de más alto nivel de la estructura de un sistema, su selección depende del software a desarrollar, obteniendo de cierta manera la unión de los elementos arquitectónicos de forma adecuada para satisfacer la mayor funcionalidad y requerimientos de desempeño de un sistema, así como requerimientos no funcionales, la confiabilidad, escalabilidad, portabilidad, y disponibilidad. Por ello cada uno de los componentes creados para el desarrollo de este sistema, poseen una arquitectura diferente:

Componente Plugin detector de persona:

Para este primer componente se seleccionó la arquitectura *Pipes and Filters* (tuberías y filtros) ya que es la más adecuada para este tipo de procesos.

Esta arquitectura proporciona una estructura para sistemas que procesan un flujo de datos, dividiendo la tarea de un sistema en varios pasos de procesamiento secuenciales. Cada paso es encapsulado en un componente filtro, los datos son pasados mediante tubos entre filtros adyacentes por lo que los datos de salida de un paso son la entrada del paso siguiente. La recombinación de filtros permite la construcción de sistemas relacionados. Cada paso de procesamiento es implementado por un componente *filter* (29).

El Plugin cuenta con cuatro filtros y el primero se encarga de obtener los fotogramas de iSpy, la **Figura 6** muestra un fotograma de ejemplo.

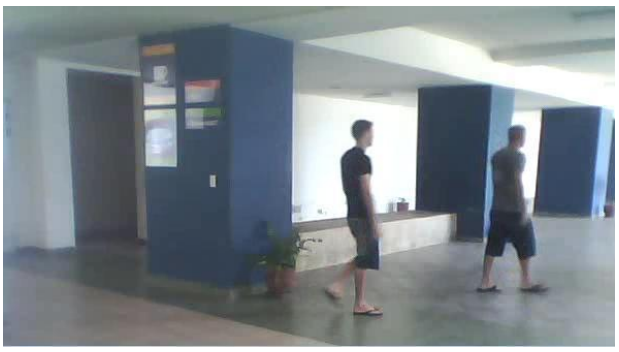

Figura 6. Muestra un fotograma.

El segundo filtro segmenta el fondo del fotograma anterior, obteniendo el frente de mismo, el cual se muestra en la **Figura 7.**

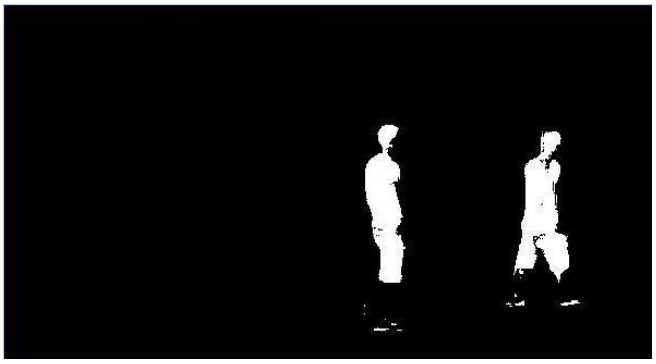

Figura 7. Muestra el frente de un fotograma.

Posteriormente la información pasa al siguiente filtro encargado de eliminar el ruido y los objetos detectados no deseados como se muestra en la **Figura 8.**

Figura 8. Muestra objetos en movimiento en un fotograma.

Finalmente se pasa el filtro responsable de detectar a las personas, mostrándose en la **Figura 9**.

Figura 9. Muestra personas detectadas en un fotograma.

Componente Control de flujo:

Para el desarrollo de éste, se seleccionó una arquitectura n-capas, que separa el código del programa dependiendo de su naturaleza. De forma general el componente se divide en tres capas individuales, las cuales aparecen explicadas de forma independiente a continuación:

Capa de presentación: Es la capa con la que interactúa directamente el usuario y contiene la interfaz visual que posee el componente. Mediante este, el usuario puede ejecutar la acción de: arrancar, detener y salir. Esta capa solo tiene acceso a la capa de negocio.

Capa de negocio: Es donde residen los programas que se ejecutan, se reciben las peticiones del usuario y se envían las respuestas tras el proceso. Se denomina capa de negocio (lógica del negocio) porque es aquí donde se establecen todas las reglas que se deben cumplir. Esta capa se comunica con la capa de presentación, para recibir las solicitudes y presentar los resultados, a su vez, con la capa de acceso a datos, para solicitar al gestor de base de datos almacenar los mismos (30). También se consideran aquí los programas de aplicación.

Capa de acceso a datos: Es donde residen los datos y la encargada de acceder a los mismos. Está formada por uno o más gestores de bases de datos que realizan todo el almacenamiento de datos, reciben solicitudes de almacenamiento o recuperación de información desde la capa de negocio (30).

La separación de capas, lógica de negocio y acceso a datos es fundamental para el desarrollo de arquitecturas consistentes y reutilizables, lo que al final resulta en un ahorro de tiempo en desarrollo y organización en proyectos posteriores.

Al existir dicha separación es más sencillo realizar labores de mejora como:

- Agregar nuevas formas de recolectar las órdenes del usuario.
- Modificar los objetos de negocios, bien sea para mejorar el rendimiento o para migrar a otra tecnología.
- Las labores de mantenimiento también se simplifican y se reduce el tiempo necesario para ellas. Las correcciones solo se deben hacer en un solo lugar y no en varios como sucedería si tuviésemos una mezcla de presentación e implementación de la lógica del negocio.

La **Figura 10** muestra los principales componentes que integran cada una de las capas mencionadas anteriormente.

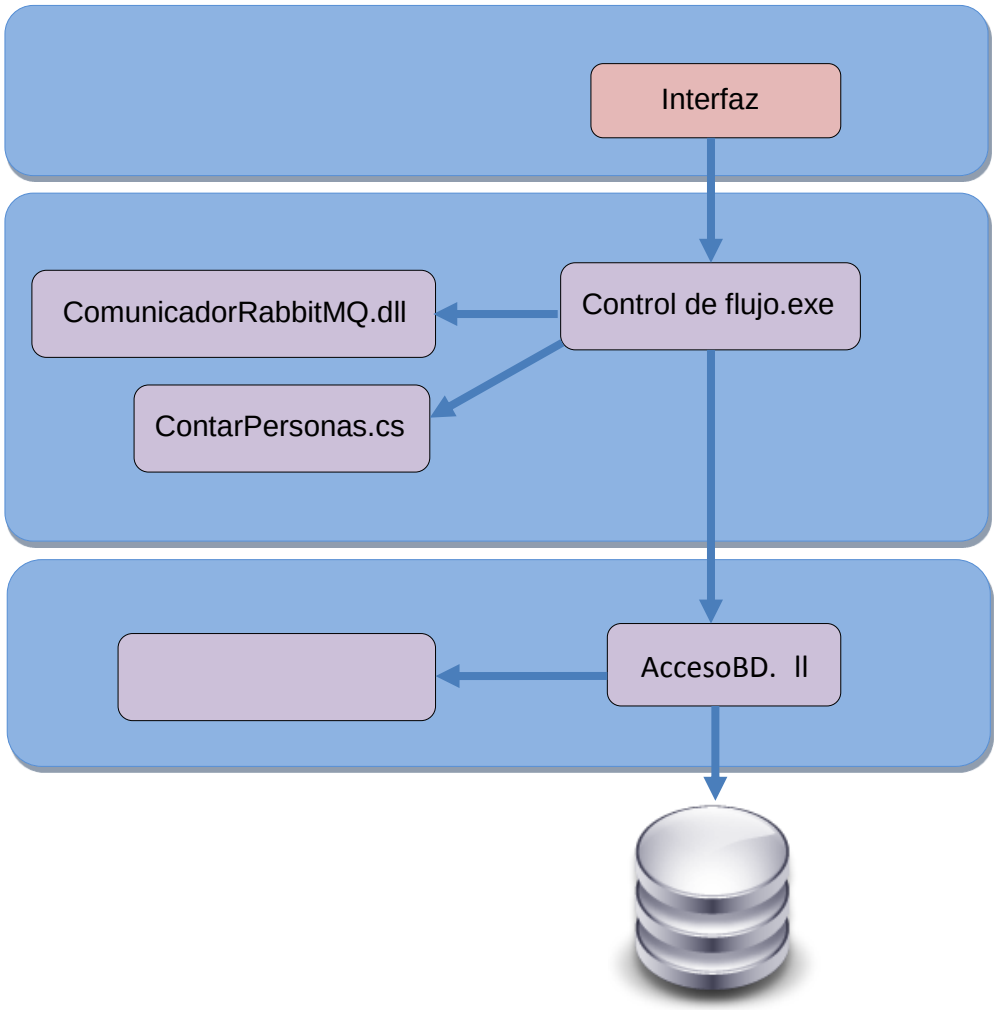

Figura 10. Arquitectura n-capas del componente Control del flujo.

Componente Aplicación de reportes:

Se seleccionó una arquitectura cliente-servidor para el tercer componente, ya que este consiste en mostrarle al usuario mediante una interfaz web el contenido deseado, en este caso los datos estadísticos del flujo en tiempo real y de días anteriores.

Esta arquitectura es también conocida como 2 capas, la primera capa, cliente, contiene los componentes que le permitirán al usuario interactuar con la información. En segundo lugar se tiene la capa servidor,

donde el resto de los componentes del sistema se relacionan entre sí para resolver las solicitudes que vienen desde el cliente.

En la **Figura 11** se muestra lo descrito anteriormente, la arquitectura con sus 2 capas y los diferentes componentes que intervienen en ellas.

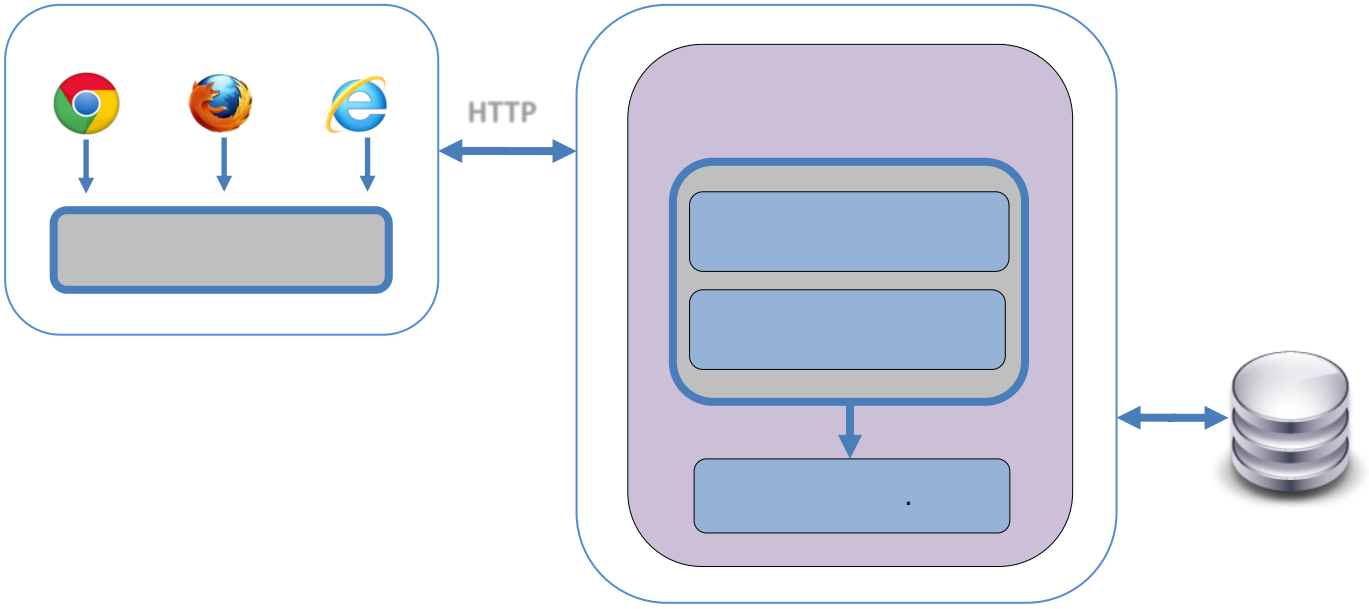

Figura 11. Arquitectura cliente-servidor del componente Aplicación de reportes.

2.13. Patrones de diseño.

Los patrones solucionan un problema en un contexto particular, imponen una regla sobre la arquitectura y son usados junto a un estilo arquitectónico para determinar la forma de la estructura general de un sistema.

Como patrones se utilizaron los patrones generales de software para asignar responsabilidades, GRASP (de sus siglas en inglés, *General Responsibility Asignment Software Patterns*). Estos describen los principios fundamentales de la asignación de responsabilidades a objetos, expresados en forma de patrones con los cuales podemos construir sistemas bien diseñados. Con la utilización del ORM Telerik OpenAccess ya se pone de manifiesto cada uno de estos patrones, brindando un desarrollo bien diseñado.

Dentro de los **GRASP** se usaron:

Experto: Es uno de los más utilizados ya que define quien asumirá la responsabilidad en cada caso, buscando siempre al experto en la información. Permite darle solución al problema el cual sería el principio fundamental para asignar responsabilidades en el diseño orientado a objetos. El cumplimiento de una responsabilidad requiere a menudo información distribuida en varias clases de objetos (31). El uso de este patrón permitió a los objetos valerse de su propia información para hacer lo que se les pide, favoreció la existencia de mínimas relaciones entre las clases, lo que permite contar con un sistema sólido y fácil de mantener. Ejemplo de ello es la clase DatosPersona en el componente Plugin detector de personas, esta clase es experta en la información referente a cada persona.

```
[Serializable]
public class DatosPersona
{
    private int largeF;
    public int LargeF1...

    private int anchoF;
    public int AnchoF1...

    private int id;
    public int Id...

    private PointF centro;
    public PointF Centro...

    private Bitmap imagen;
    public Bitmap Imagen...

    public DatosPersona()...

    public DatosPersona(int id1, PointF centro1, int largo1,int ancho1,Bitmap imagen1)
    {
        id = id1;
        centro = centro1;
        imagen = imagen1;
        largeF = largo1;
        anchoF = ancho1;
    }
}
```

Figura 12. Ejemplo del patrón Experto.

Creador: Se considera para la asignación de responsabilidades a las clases relacionadas con la creación de objetos. Una instancia de un objeto solo pueda ser creada por el objeto que contiene la información

necesaria para ello (31). El uso de este patrón permitió crear las dependencias mínimas necesarias entre las clases, beneficiando el mantenimiento del sistema y brindando mejores oportunidades de reutilización. Este patrón se pone de manifiesto en varias clases, una de ellas es ContarPersona del componente Control de flujo, donde se crea una instancia de la clase DatosPersona que permite la reutilización de los datos de la persona sin necesidad de volver a crear la clase.

```
public class ContarPersonas
{
    List<DatosPersona> listaPersonas = new List<DatosPersona>();

    Rectangle rectangulo;

    private Thread hilo;
    public Thread Hilo...

    private int entrada;
    public int Entrada...

    private int salida;
    public int Salida...

    public ContarPersonas()...

    public void ContarPersona()
    {
        var conexion = new ModeloBD();
        bool entrada;

        Comunicador comunicador = new Comunicador("daniel", "daniel", "daniel", "biometria3.proiden.uci.cu");
        while (true)
        {
            DatosPersona datos = new DatosPersona();
            comunicador.Connect();
```

Figura 13. Ejemplo del patrón Creador.

Controlador: Todas las peticiones son manejadas por un solo controlador frontal, que constituye el punto de entrada único de toda la aplicación en un entorno determinado. Crea una nueva instancia por la clase que tiene la información necesaria para realizar la creación del objeto. Usa directamente las instancias creadas del objeto, almacena o maneja varias instancias de la clase y contiene o agrega la clase (31). Esto se pone de manifiesto en la clase principal del componente Plugin detector de personas donde se crean las instancias necesarias para el manejo de la información.

```
public class Main: IDisposable
{
    private string config="";
    private bool disposed;
    internal int lineWidth = 1;
    public string alert;
    private static MCvFont font = new MCvFont(Emgu.CV.CvEnum.FONT.CV_FONT_HERSHEY_COMPLEX, 1.0, 1.0);
    private static BlobTrackerAuto<Bgr> tracker = new BlobTrackerAuto<Bgr>();
    private static IBGFGDetector<Bgr> detector = new FGDetector<Bgr>(FORGROUND_DETECTOR_TYPE.MOG);

    private static Image<Bgr, Byte> frame1;
    private static Image<Gray, Byte> forgroundMask;

    private DatosPersona datos;
    private int id;
    private PointF centro;
    private Bitmap imagenRecortada;
    private Bitmap imagen;
    private int largeF;
    private int anchoF;

    public Bitmap ProcessFrame(Bitmap frame)
    {
        Comunicador comunicador = new Comunicador("daniel", "daniel", "daniel", "biometria3.proiden.uci.cu");
        frame1 = new Image<Bgr, byte>(frame);
        frame1._SmoothGaussian(3);

        forgroundMask = tracker.ForgroundMask;
```

Figura 14. Ejemplo del patrón Controlador.

2.14. Conclusiones Parciales.

El sistema de control de flujo de personas en imágenes de video en tiempo real le permitirá al CISED conocer la cantidad de personas que se encuentran físicamente en el centro en un determinado momento del día, contarlas y saber con precisión cuáles son los días y horarios en que existe un mayor flujo de personas, así como el nivel de afluencia de personas hacia el centro y viceversa.

- Se definieron el modelo de dominio, los requisitos no funcionales del sistema, la metáfora y la elaboración de historias de usuario permitieron un mejor entendimiento del funcionamiento del sistema y las diferentes funcionalidades con las que este cumple, según las necesidades del cliente.
- La selección de las diferentes arquitecturas correspondientes a cada componente y los patrones de diseño, permitieron una mejor organización de la información.
- El tiempo estimado, el plan de iteraciones y de entrega permitieron fijar metas para el cumplimiento de las diferentes iteraciones.

Capítulo III: Implementación y prueba.

Introducción

En este capítulo se muestra la implementación del sistema dándole cumplimiento a los requerimientos planteados al inicio de la investigación, se refleja el diseño de la base de datos, los diagramas de componentes y el de despliegue. Se visualizan las interfaces de usuario y se da una explicación de cada una de ellas para un mayor entendimiento acerca del funcionamiento de la aplicación. Se realizan las pruebas propuestas por la metodología y de fiabilidad, donde se valida el funcionamiento de los requisitos funcionales.

3.1. Diseño de la Base de datos.

La base de datos desempeña una función importante en el desarrollo de la aplicación. Esta permite que los datos se almacenen de forma coherente y organizada para que no exista pérdida de los mismos. La base de datos que se muestra a continuación, satisface las necesidades de persistencia de los datos que el sistema requiere, de acuerdo a sus requerimientos funcionales. Para su diseño, se utilizan los modelos lógico y físico de datos. La persistencia de los datos no es más que la capacidad de un objeto de mantener su valor en el tiempo y el espacio. La base de datos como se muestra en la **Figura 15** contará con una sola tabla que tendrá como nombre Historial, la cual estará compuesta por cinco campos. La descripción de la tabla y cada uno de sus campos se muestran en el Anexo 10.

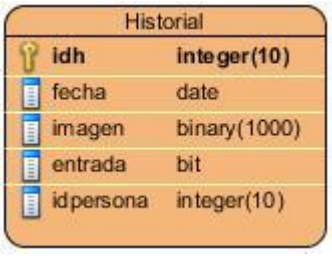

Figura 15. Diagrama de entidad-relación de la base de datos.

3.2. Estándares de codificación

Un estándar de codificación son reglas que se siguen para la escritura del código fuente. Para una mejor comprensión del código se definieron una serie de estándares basados en diversas reglas.

3.2.1. Estilos para la capitalización

Utilización de la convención Pascal: Se define el primer caracter de cada palabra en mayúscula y el resto en minúscula. Ejemplo: ForeColor.

Utilización de la convención Camel: Se define el primer caracter de cada palabra en mayúscula (excepto la primera palabra) y el resto en minúscula. Ejemplo: foreColor.

Convenciones para los nombres

Convención Pascal para el nombre de las clases

Ejemplo:

```
public class ContarPersonas
{
    List<DatosPersona> listaPersonas = new List<DatosPersona>();
```

Convención Pascal para el nombre de los métodos

Ejemplo:

```
public void ContarPersona()...
```

Convención Camel para el nombre de variables.

Ejemplo:

```
private DatosPersona datos;
private int id;
private PointF centro;
private Bitmap imagenRecortada;
```

3.2.2. Reglas de codificación

- No usar caracteres simples para el nombre de las variables i, n, s, etc. Una excepción de esta regla son las variables dentro de los ciclos.
- Todas las variables miembros de las clases deben ser prefijadas con *underscore* (_) para ser diferenciadas de otras variables.
- No usar nombres de variables que coincidan con palabras reservadas.
- Los comentarios deben estar en el mismo nivel del código.
- No escribir comentarios para cada línea de código o para cada variable declarada.
- Las llaves se deben poner al mismo nivel del código que las contiene.

- Usar una línea en blanco para separar agrupaciones lógicas del código.
- Debe dejarse una y solo una línea en blanco entre cada método dentro de las clases.
- Las llaves deben ser utilizadas sobre líneas separadas y no sobre la misma línea como en if, for etc.
- Usar un espacio simple antes y después de cada operador y llave.

3.3. Diagrama de componentes.

Un diagrama de componentes muestra los fragmentos de software por los cuales estará conformado el sistema. Los componentes usualmente se implementan por una o varias clases (u objetos) en tiempo de ejecución a su vez estos pueden comprender una gran porción de un sistema.

En la **Figura 16** se muestra el diagrama de componentes correspondiente al Plugin detector de personas. De los componentes mostrados el fundamental es el proyecto Plugin.dll, el cual utiliza la librería EmguCV para el procesamiento de las imágenes y el proyecto ComunicadorRabbitMQ.dll para establecer la comunicación con el RabbitMQ.

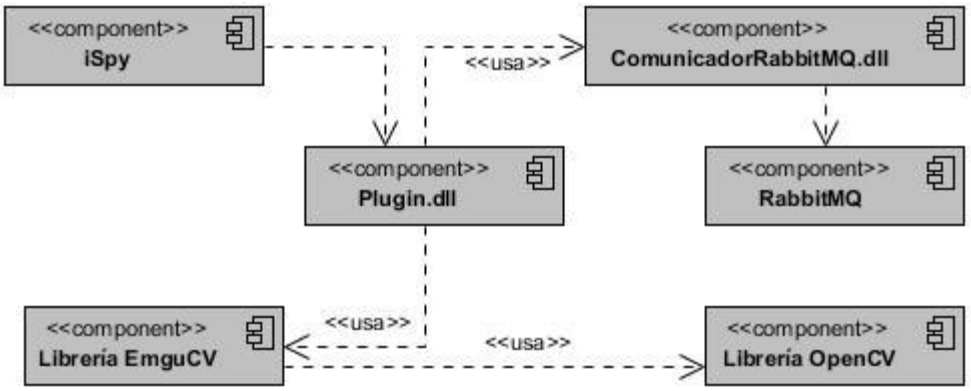

Figura 16. Diagrama de componentes del Plugin detector de persona.

La **Figura 17** muestra el diagrama de componentes del componente Control del flujo, donde aparece la separación de los componentes del negocio y los de acceso a datos para una mejor claridad teniendo en cuenta la complejidad de dicho componente. Como principal componente se muestra Control de flujo que utiliza al ComunicadorRabbitMQ.dll para establecer la comunicación con el RabbitMQ y el Modelo.dll para almacenar la información en la base de datos.

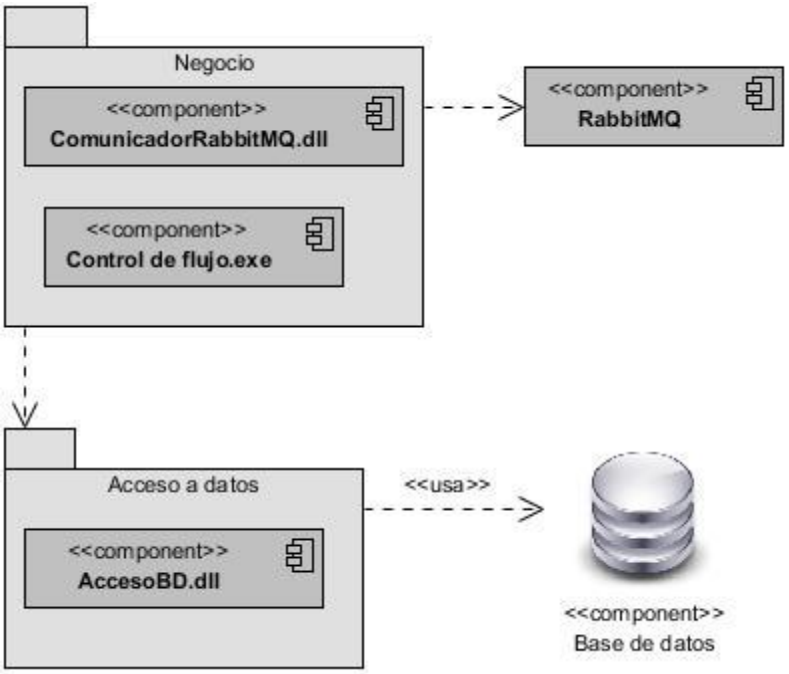

Figura 17. Diagrama de componentes de Control del flujo.

La **Figura 18** muestra el diagrama de componentes correspondiente a la Aplicación de reportes, encargada de mostrar la información procesada en los componentes anteriores mediante las vistas Principal.aspx y Historial.aspx.

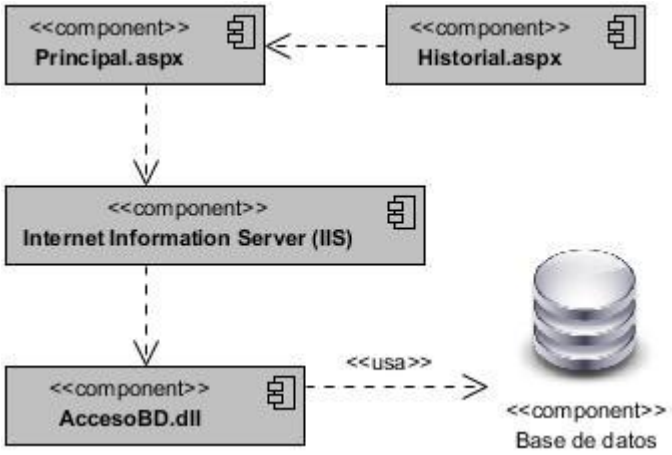

Figura 18. Diagrama de componentes de la Aplicación de reportes.

3.4. Tareas de ingeniería.

Las tareas de ingeniería serán de uso únicamente para el programador, ya que estas son creadas para ayudar a organizar la implementación exitosa de las historias de usuario. Las historias de usuarios no ofrecen el nivel de detalle requerido para llevar a cabo esta tarea, es por eso que son divididas en tareas de la ingeniería. Una historia de usuario se puede dividir en una o más tareas de ingeniería y a partir de estas se comienza el ciclo de la fase de iteración. Estas tareas de ingeniería pueden ser escritas en lenguaje técnico y no necesariamente entendible por el cliente. Según el plan de iteraciones trazado en este trabajo las historias de usuario se agruparon en tres ciclos de iteraciones:

Tabla 8. Muestra las iteraciones en las que está dividida la aplicación con sus historias de usuarios y sus tareas de ingeniería correspondientes.

ITERACIÓN	HISTORIA DE USUARIO	TAREA
1	Obtener los fotogramas del iSpy.	Cargar el Plugin detector de personas al iSpy. Obtener los fotogramas del iSpy.
	Procesar imagen.	Segmentar el fondo del fotograma. Eliminar ruido en fotograma.
	Detectar personas.	Obtener blob. Aplicar filtros. Seguir del blob.
	Almacenar datos en la cola del RabbitMQ.	Establecer la comunicación con el RabbitMQ. Guardar datos en la cola del RabbitMQ.
2	Extraer información de la cola del RabbitMQ.	Establecer la comunicación con el RabbitMQ. Extraer datos de la cola del RabbitMQ.
	Calcular dirección del movimiento.	Comparar posición anterior con la actual.
	Contar personas.	Comprobar si la posición de la persona pertenece o no a la zona de conteo.
	Almacenar información en la base de datos.	Guardar la información correspondiente a cada persona en la base de datos mediante el ORM Telerik OpenAccess.

3	Extraer información de la base de datos.	Mediante el ORM Telerik OpenAccess obtener la información requerida.
	Consultar datos estadísticos en tiempo real.	Implementar métodos correspondientes a la historia de usuario. Implementar interfaz.
	Consultar datos estadísticos de un día determinado.	Implementar métodos correspondientes a la historia de usuario. Implementar interfaz.
	Comparar datos estadísticos de dos días.	Implementar métodos correspondientes a la historia de usuario. Implementar interfaz.

3.5. Interfaz de usuario.

La interfaz de usuario es la parte visual del sistema con la que el usuario va a interactuar, es aquí donde se muestran los resultados que devuelven las funcionalidades implementadas. En la **Figura 19** se muestra la interfaz principal donde el usuario puede observar los datos estadísticos de las entradas y salidas al CISED en tiempo real. En la **Figura 20** se muestra la interfaz de usuario historial, donde el usuario puede seleccionar la forma de búsqueda del historial, ya sea por un día específico o comparar dos días. La **Figura 21** muestra la interfaz de usuario correspondiente al componente de Control de flujo, en esta el usuario puede seleccionar la acción que desea realizar correspondiente al componente.

Figura 19. Interfaz de usuario principal.

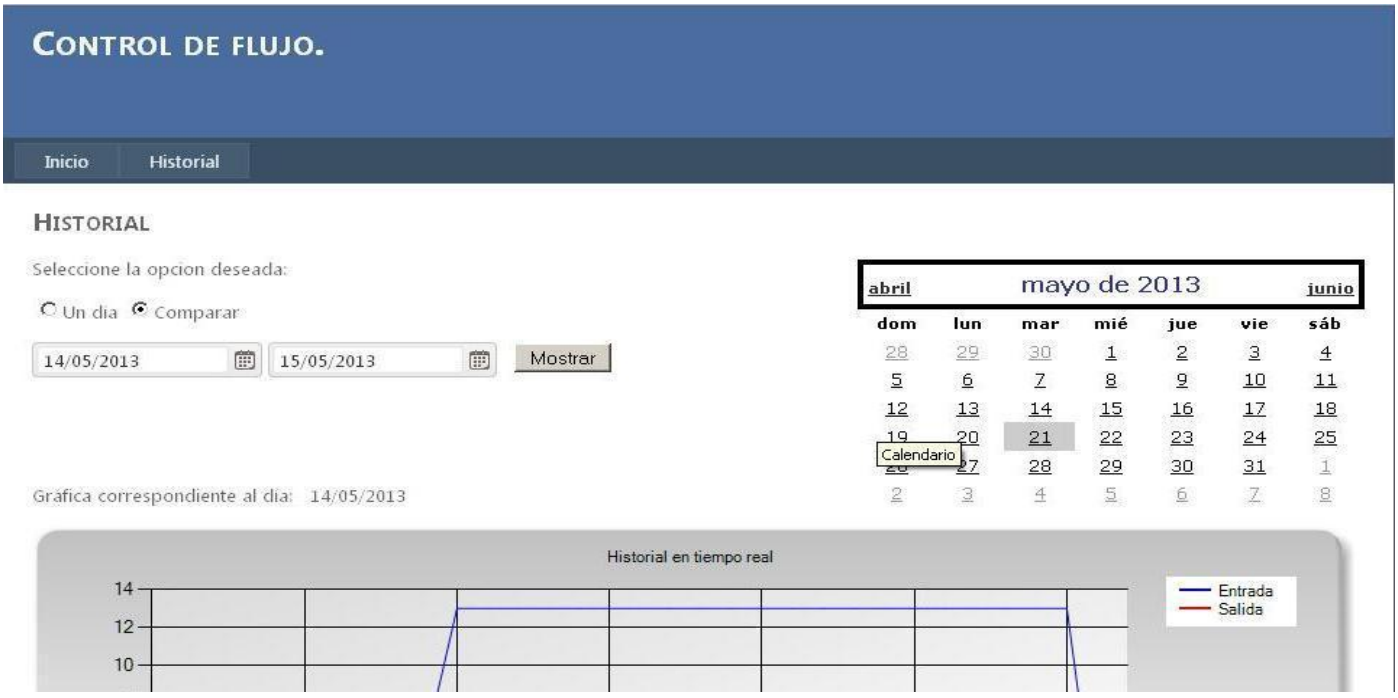

Figura 20. Interfaz de usuario para comparar datos estadísticos de dos días.

Figura 21. Interfaz de usuario del componente Control de flujo.

3.6. Diagrama de despliegue.

El diagrama de despliegue es un tipo de diagrama del Lenguaje Unificado de Modelado que se utiliza para modelar el hardware utilizado en las implementaciones de sistemas y las relaciones entre sus componentes. Los diagramas de despliegue muestran las relaciones físicas de los distintos nodos que componen un sistema y el reparto de los componentes sobre dichos nodos. La vista de despliegue representa la disposición de las instancias de componentes de ejecución en instancias de nodos conectados por enlaces de comunicación. La **Figura 22** muestra la distribución física del sistema, la cual está compuesta por 4 servidores, un dispositivo cámara y una PC cliente. El primero de los servidores contendrá al Plugin detector de personas y el software iSpy, En el segundo servidor será colocado el manejador de colas de mensajes RabbitMQ. Se tendrá un tercer servidor en el cual serán ubicados el componente Control de flujo y la base de datos. Por último se tiene un servidor web donde se ejecutaría el componente Aplicación de reportes.

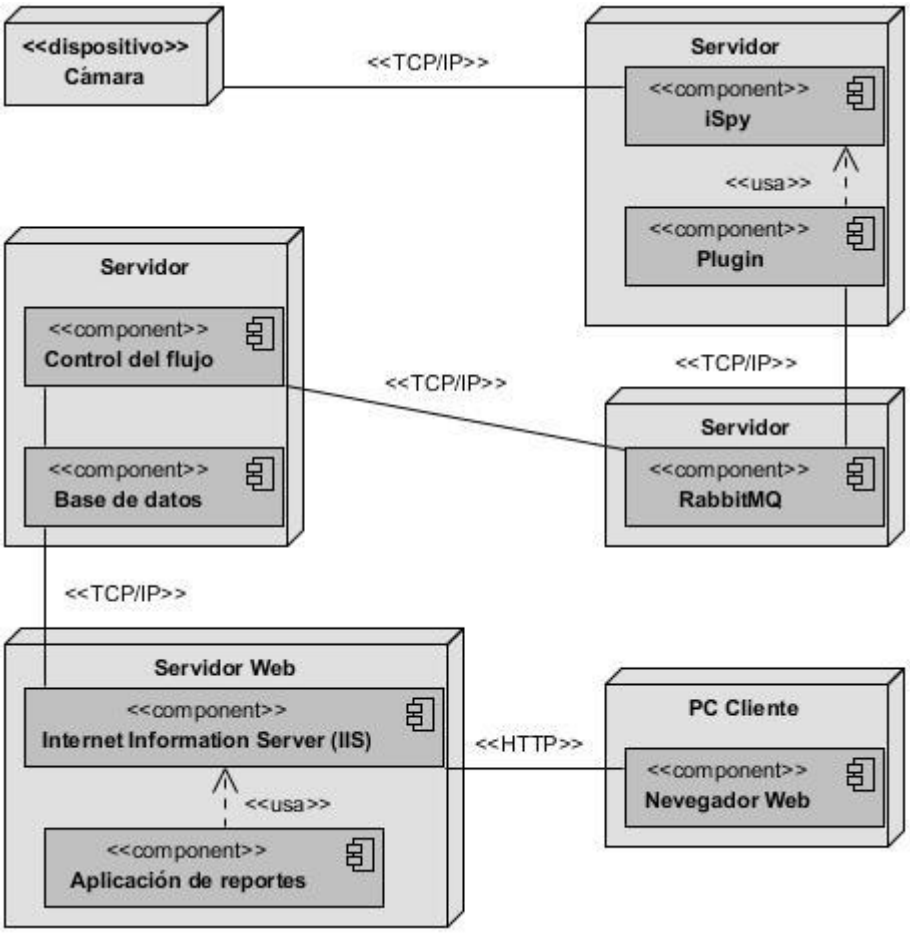

Figura 22. Diagrama de despliegue.

3.7. finalFase de Pruebas.

Para el desarrollo de esta fase, en esta investigación el equipo de desarrollo se centró en los pedidos del cliente. El perfeccionamiento de código sólo fue posible a través de un grupo de pruebas unitarias automatizadas que aseguraron la ejecución correcta del sistema en todo el período de desarrollo. Con esta estrategia se reduce el número de errores no detectados así como el tiempo entre la introducción de éste en el sistema y su detección, lo que contribuye a elevar la calidad de los productos desarrollados y a la seguridad de los programadores cuando es necesario introducir cambios o modificaciones.

Cuando se desarrolla el software, es necesario realizar una gran cantidad de pruebas para verificar que el código esté correcto. Estas pruebas normalmente se repiten varias veces y se ven afectadas por los cambios que se introducen conforme se va desarrollando.

La metodología usada (XP) divide las pruebas del sistema en dos grupos: pruebas unitarias, encargadas de verificar el código y diseñada por los programadores y pruebas de aceptación o pruebas funcionales destinadas a evaluar si al final de una iteración se consiguió la funcionalidad requerida diseñada por el cliente.

3.7.1. Pruebas unitarias.

Las pruebas unitarias son creadas por los desarrolladores antes de comenzar a codificar, lo cual las hará más sencillas y efectivas. Se realizan a lo largo de todo el proyecto asegurando el correcto funcionamiento evitando las ambigüedades y permiten que los requerimientos queden refinados en la prueba.

Para la realización de estas pruebas se utilizó un *Add-in* de Visual Studio llamado ReSharper que permite detectar errores y brinda soluciones instantáneas para corregirlos. Los resultados de estas pruebas pueden consultarse en el Anexo 11.

3.7.2. Pruebas de aceptación.

Las pruebas de aceptación son más importantes que las pruebas unitarias, estas significan la satisfacción del cliente con el producto desarrollado, marcan el final de una iteración y el comienzo de la siguiente, es por ello que el cliente es la persona adecuada para diseñar estas pruebas, siempre y cuando forme parte del equipo de desarrollo.

Las pruebas de aceptación son creadas a partir de las historias de usuario. Durante una iteración la historia de usuario seleccionada en la planificación de iteraciones se convertirá en una prueba de aceptación. El cliente o usuario especifica los aspectos a probar en una historia de usuario que ha sido correctamente implementada. (**Tabla 9-17**).

Una historia de usuario puede tener más de una prueba de aceptación, tantas como sean necesarias para garantizar su correcto funcionamiento y no se considera completa hasta que no supera sus pruebas de aceptación. Es responsabilidad del cliente verificar la corrección de las pruebas y tomar decisiones acerca de las mismas.

Tabla 9. HU1_CP1 Prueba de funcionalidad para obtener los fotogramas.

CASO DE PRUEBA DE ACEPTACIÓN.	
Código de caso de prueba: HU1_CP1	**Nombre de la HU:** Obtener los fotogramas del iSpy.
Responsable de la prueba: Aynel Cruz Barrera	
Descripción de la prueba: Prueba de funcionalidad para obtener los fotogramas.	

Condiciones de ejecución: El iSpy debe de estar activado.
Entrada/Pasos de ejecución: • Seleccionar cámara. • Aplicar *plugin* a la cámara seleccionada. • Se comienza a obtener los fotogramas de la cámara seleccionada.
Resultado esperado: Se obtienen los fotogramas de la cámara seleccionada.
Evaluación de la prueba: Prueba Satisfactoria.

Tabla 10. HU2_CP2 Prueba de funcionalidad para obtener los objetos de interés.

CASO DE PRUEBA DE ACEPTACIÓN.	
Código de caso de prueba: HU2_CP2	**Nombre de la HU:** Procesar imagen.
Responsable de la prueba: Aynel Cruz Barrera	
Descripción de la prueba: Prueba de funcionalidad para obtener los objetos de interés.	
Condiciones de ejecución:	
Entrada/Pasos de ejecución: • Se define que es fondo y que es frente en el fotograma mediante el algoritmo usado. • Se obtiene el frente. • Se eliminan los ruidos que pueden existir, mediante la aplicación de los filtros: tamaño del blob, área del rectángulo englobante, aspecto del blob y el *Smooth_gaussian*.	
Resultado esperado: Se obtiene los objetos de interés.	
Evaluación de la prueba: Prueba Satisfactoria.	

Tabla 11. HU3_CP3 Prueba de funcionalidad para detectar a las personas en los fotogramas.

CASO DE PRUEBA DE ACEPTACIÓN.	
Código de caso de prueba: HU3_CP3	**Nombre de la HU:** Detectar personas.
Responsable de la prueba: Aynel Cruz Barrera	
Descripción de la prueba: Prueba de funcionalidad para detectar a las personas en los fotogramas.	
Condiciones de ejecución: Se debe tener un fotograma libre de ruido.	
Entrada/Pasos de ejecución: • Se detecta a las personas mediante la técnica implementada.	
Resultado esperado: Se obtienen los blog.	
Evaluación de la prueba: Prueba Satisfactoria.	

Tabla 12. HU4_CP4 Prueba de funcionalidad para almacenar datos en la cola del RabbitMQ.

CASO DE PRUEBA DE ACEPTACIÓN.	
Código de caso de prueba: HU4_CP4	**Nombre de la HU:** Almacenar datos en la cola del RabbitMQ.
Responsable de la prueba: Aynel Cruz Barrera	
Descripción de la prueba: Prueba de funcionalidad para almacenar datos en la cola del RabbitMQ.	
Condiciones de ejecución:	
Entrada/Pasos de ejecución: • Se debe establecer conexión con el RabbitMQ. • Guardar datos en la cola del RabbitMQ.	
Resultado esperado: Almacenen datos.	
Evaluación de la prueba: Prueba Satisfactoria.	

Tabla 13. HU5_CP5 Prueba de funcionalidad para extraer información de la cola del RabbitMQ.

CASO DE PRUEBA DE ACEPTACIÓN.	
Código de caso de prueba: HU5_CP5	**Nombre de la HU:** Extraer información de la cola del RabbitMQ.
Responsable de la prueba: Aynel Cruz Barrera	
Descripción de la prueba: Prueba de funcionalidad para extraer información de la cola del RabbitMQ.	
Condiciones de ejecución:	
Entrada/Pasos de ejecución: • Se debe establecer conexión con el RabbitMQ. • Extraer datos en la cola del RabbitMQ.	
Resultado esperado: Extraer datos.	
Evaluación de la prueba: Prueba Satisfactoria.	

Tabla 14. HU6_CP6 Prueba de funcionalidad para calcular la dirección en que se mueve la persona.

CASO DE PRUEBA DE ACEPTACIÓN.	
Código de caso de prueba: HU6_CP6	**Nombre de la HU:** Calcular dirección del movimiento.
Responsable de la prueba: Aynel Cruz Barrera	
Descripción de la prueba: Prueba de funcionalidad para calcular la dirección en que se mueve la persona.	
Condiciones de ejecución:	

Entrada/Pasos de ejecución: • Se obtiene la posición anterior de la persona. • Se compara la posición anterior con la actual para identificar el sentido del movimiento.
Resultado esperado: Se obtiene el sentido del movimiento.
Evaluación de la prueba: Prueba Satisfactoria.

Tabla 15. HU7_CP7 Prueba de funcionalidad para saber la cantidad de personas que han entrado y salido de un local.

CASO DE PRUEBA DE ACEPTACIÓN.	
Código de caso de prueba: HU7_CP7	**Nombre de la HU:** Contar persona.
Responsable de la prueba: Aynel Cruz Barrera	
Descripción de la prueba: Prueba de funcionalidad para saber la cantidad de personas que han entrado y salido de un local.	
Condiciones de ejecución:	
Entrada/Pasos de ejecución: • Se define área o región que se desee controlar. • Se cuenta o descuenta a las personas que salen o entran de dicha área.	
Resultado esperado: Conocer la cantidad de personas	
Evaluación de la prueba: Prueba Satisfactoria.	

Tabla 16. HU8_CP8 Prueba de funcionalidad para almacenar información en la base de datos.

CASO DE PRUEBA DE ACEPTACIÓN.	
Código de caso de prueba: HU8_CP8	**Nombre de la HU:** Almacenar información en la base de datos.
Responsable de la prueba: Aynel Cruz Barrera	
Descripción de la prueba: Prueba de funcionalidad para almacenar información en la base de datos.	
Condiciones de ejecución:	
Entrada/Pasos de ejecución: • Se debe establecer conexión con el RabbitMQ. • Guardar datos en la cola del RabbitMQ.	
Resultado esperado: Almacenar datos.	
Evaluación de la prueba: Prueba Satisfactoria.	

Tabla 17. HU9_CP9 Prueba de funcionalidad para consultar el historial.

CASO DE PRUEBA DE ACEPTACIÓN.	
Código de caso de prueba: HU9_CP9	**Nombre de la HU:** Consultar datos estadísticos.
Responsable de la prueba: Aynel Cruz Barrera	
Descripción de la prueba: Prueba de funcionalidad para consultar datos estadísticos.	
Condiciones de ejecución: El usuario debe solicitar la consulta de los datos.	
Entrada/Pasos de ejecución: • Se ofrecen datos estadísticos en tiempo real. • Se puede consultar datos estadísticos de un día determinado. • Se puede comparar datos estadísticos de dos días.	
Resultado esperado: Se muestran los datos solicitados.	
Evaluación de la prueba: Prueba Satisfactoria.	

3.7.3. Pruebas de fiabilidad.

Con el fin de desarrollar un sistema más eficiente en la detección y conteo de las personas se han realizado pruebas de fiabilidad, ya que las pruebas propuestas por la metodología XP no garantizan del todo, esta variable. En este estudio se decidió realizar pruebas de fiabilidad para medir la efectividad con la que opera el sistema, ofreciendo más detalles sobre el proceso. Los resultados obtenidos aparecen en las **Tabla 18-20**. Las pruebas se realizaron teniendo en cuenta un flujo de personas bajo, medio y alto respectivamente.

Tabla 18. Muestra los resultados obtenidos de la prueba con un flujo bajo.

Variable	**Valor**
Total de personas	33
Personas detectadas el 100 % del tiempo	10
Total de personas detectadas	30
Total de personas no detectadas	3
Total de personas que entraron al local	15
Total de personas que entraron y no fueron contadas	0
Total de personas que salieron del local	18
Total de personas que salieron y no fueron contadas	3
Fiabilidad	90,90%

Tabla 19. Muestra los resultados obtenidos de la prueba con un flujo medio.

Variable	Valor
Total de personas	46
Personas detectadas el 100 % del tiempo	18
Total de personas detectadas	43
Total de personas no detectadas	3
Total de personas que entraron al local	23
Total de personas que entraron y no fueron contadas	0
Total de personas que salieron del local	20
Total de personas que salieron y no fueron contadas	3
Fiabilidad	93,40%

Tabla 20. Muestra los resultados obtenidos de la prueba con un flujo alto.

Variable	Valor
Total de personas	70
Personas detectadas el 100 % del tiempo	24
Total de personas detectadas	63
Total de personas no detectadas	7
Total de personas que entraron al local	39
Total de personas que entraron y no fueron contadas	4
Total de personas que salieron del local	31
Total de personas que salieron y no fueron contadas	2
Fiabilidad	90,00%

Luego de analizar los resultados obtenidos referentes a las pruebas de fiabilidad realizadas, se puede llegar a la conclusión que el sistema implementado ostenta con un 91% de fiabilidad, teniendo en cuenta algunas deficiencias respecto al funcionamiento del algoritmo, así como la tecnología utilizada. En este caso las cámaras con las que fueron filmadas las imágenes de video no son apropiadas para este trabajo. Respecto a las deficiencias que presenta el algoritmo, se detectan problemas en la segmentación de fondo en caso que las prendas de vestir de las personas y el fondo de la imagen sean del mismo color, lo que conlleva a que la persona sea identificada como fondo de la escena y no sea detectada.

Comparando los resultados alcanzados con la empresa Visual Tools, que desarrolló el sistema Análisis de video: conteo de personas, valorando que utilizan tecnologías de última generación, en este caso equipos PeCO con un 95% de fiabilidad, se puede afirmar que el sistema implementado es fiable en cierta medida.

3.8. Conclusiones Parciales.

- El uso de los diagramas de componentes, de despliegue y las tareas de ingeniería facilitó la implementación del sistema.
- La realización de pruebas al sistema posibilitó la detección de diversas no conformidades que se corrigieron tras varias iteraciones.
- Los resultados obtenidos en las pruebas de fiabilidad demostraron que el sistema desarrollado es fiable en un 90%.

Conclusiones

Se diseñó un sistema utilizando imágenes de video en tiempo real que permite el conteo de personas y muestra información gráfica, aplicable para el control del flujo de personas en el Centro de Identificación y Seguridad Digital, de la Universidad de las Ciencias informáticas.

- El estudio de las soluciones existentes y de los conceptos básicos para el desarrollo de un sistema de control de flujo de personas, mediante el análisis de video en tiempo real, permitió hacer un análisis integral del objeto de estudio.
- El análisis de las tecnologías y herramientas relacionadas con la solución propuesta, permitió una correcta elección de las mismas así como la selección de la metodología de desarrollo.
- El análisis y diseño realizado al sistema propuesto, permitió generar los artefactos propuestos por la metodología seleccionada generando así un mejor flujo, entendimiento y organización durante el desarrollo del sistema.
- La implementación e integración de los diferentes componentes en la solución final permitieron una mejor eficiencia en el proceso de la obtención de la información requerida para el cliente y la realización de las pruebas conllevó a una mejora de sus funcionalidades, permitiendo, en algunos casos, la corrección de errores.

Recomendaciones

- Aplicar el sistema utilizando imágenes de video en tiempo real para el control del flujo de personas en el Centro de Identificación y Seguridad Digital de la Universidad de las Ciencias informáticas.
- Impulsar la mejora de los algoritmos para la detección de los objetos en la escena, ya que no presentan un 100% de efectividad.
- Mejorar el seguimiento de los objetos para manejar eficientemente las oclusiones parciales o totales que puedan ocurrir.
- Analizar estadísticamente la eliminación de sombras para ajustar los parámetros a la mayor cantidad de ambientes posibles.

Referencias Bibliográficas

1. **Cruz Domínguez, Idalmys.** CiberSociedad. [En línea] 2007. [Citado el: 3 de diciembre de 2012.] hhtp://www.cibersociedad.net/archivo/articulo.php?art=229.

2. **Roberto Hernández Sampieri, Carlos Hernández Collado, Pilar Baptista Lucio.** *Metodología de la Investigación.* México D.F : McGraw-Hill, 2006. ISBN 970-10-5753-8.

3. **Jain, Anil K., Ross, Arun y Pankanti, Sharath.** Biometrics: A Tool for Information Security. *IEEE TRANSACTIONS ON INFORMATION FORENSICS AND SECURITY.* junio de 2006. Vol. Vol. 1, No. 2.

4. **Morales, Domingo: Ruiz del Solar, Javier.** Sistemas Biométricos: Matching de huellas dactilares mediante transformada de Hough generalizada. [En línea] http://www2.ing.puc.cl/~iing/ed429/sistemas_biometricos.htm. .

5. **González, R.C., Wintz, P.** *Procesamiento digital de imágenes.* s.l. : Addison-Wesley., 1996.

6. **Montagu, Arturo F.** *CULTURA DIGITAL, COMUNICACIÓN Y SOCIEDAD.* Buenos Aires : Paidos Iberica, 2001.

7. **Albusac, Javier, y otros, y otros.** European Centre for Soft Computing. *Aprendizaje de reglas difusas para la clasificación de comportamientos en un sistema de videovigilancia cognitiva.* [En línea] 2005. [Citado el: 9 de mayo de 2013.] http://www.softcomputing.es/estylf08/es/082-5-165.pdf.

8. Real Academia Española. [En línea] [Citado el: 25 de enero de 2013.] http://www.rae.es/RAE.

9. **López, Luis Rodríguez.** Análisis y evaluación de algoritmos de detección de movimiento y su aplicación a sistemas de seguimiento en video. Madrid : UNIVERSIDAD CARLOS III DE MADRID, ESCUELA POLITÉCNICA SUPERIOR, 2009.

10. SenSource. [En línea] 2002. [Citado el: 31 de enero de 2013.] http://www.sensourceinc.com.

11. Visual Tools. [En línea] 1995. [Citado el: 31 de enero de 2013.] http://www.visual-tools.com/productos/analisis-de-video-conteo-de-personas-y-control-pos.

12. DATYS. *Tecnología y sistemas.* [En línea] DATYS, 2011. [Citado el: 17 de mayo de 2013.] http://www.datys.cu/wpinfproducto.aspx?83.

13. **Rodríguez, Lisandra Michelena.** Desarrollo de un video sensor para el conteo de personas. La Habana : Universidad de las Ciencias Informáticas, 2011.

14. **Kruchten, Philippe.** *The Rational Unified Process an Introduction.* Boston : Addison-Wesley, 2004. ISBN 0321197704.

15. **Beck, Kent.** *Extreme Programming Explained. Embrace Change.* s.l. : Pearson Education, 1999.

16. **Klein, Scott.** *Pro Entity Framework 4.0.* New York : Jonathan Gennick, 2010. ISBN-13 978-1-4302-0648-4.

17. **Burrows, Chris.** MSDN Magazine. *Nuevas características de C# en .NET Framework 4.* [En línea] [Citado el: 4 de febrero de 2013.] Burrows, Chris. MSDN Magazine. Nuevas http://msdn.microsoft.com/es-es/magazine/ff796223.aspx..

18. **Serrano Pérez, Jorge.** *Programación con ASP.NET.* Madrid : ANAYA MULTIMEDIA, 2002. ISBN: 978-84-415-1342-6 84-415-1342-2.

19. **Valdez, Joel.** Blog Corporativo. *CAPACITY.* [En línea] [Citado el: 25 de abril de 2013.] http://blog.capacityacademy.com/2013/03/16/jquery-que-es-origenes-ventajas-desventajas/.

20. OpenCV. [En línea] [Citado el: 19 de febrero de 2013.] http://docs.opencv.org/.

21. Emgu CV. [En línea] [Citado el: 4 de febrero de 2013.] http://www.emgu.com/wiki/index.php/Main_Page. GNU Free License 1.2.

22. Microsoft. *Microsoft Visual Studio 2010 Trial Ultimate – ISO Microsoft Download Center .* [En línea] [Citado el: 5 de febrero de 2013.] http://www.microsoft.com/downloads/en/details.aspx?FamilyID=06a32b1c-..

23. **James.** Visual Paradigm. *Boost productivity with innovative and intuitive technologies.* [En línea] Visual Paradigm, 1999. [Citado el: 6 de febrero de 2013.] http://www.visual-paradigm.com/.

24. MySQL. [En línea] [Citado el: 29 de marzo de 2013.] http://www.mysql.com.

25. **Lockhart, Thomas.** IRIS-Libre. *Guia del Programador de PostgreSQL.* [En línea] [Citado el: 5 de febrero de 2013.] https://forja.rediris.es/docman/view.php/312/461/PostgresProgrammer.pdf..

26. RabbitMQ. [En línea] [Citado el: 4 de abril de 2013.] http://www.rabbitmq.com/.

27. Blog. *iSpy Connect.* [En línea] [Citado el: 2 de mayo de 2013.] http://www.ispyconnect.com/.

28. Telerik Open Access. [En línea] [Citado el: 30 de abril de 2013.] http://www.telerik.com/.

29. **Avila Mojica, Jamair Antonio y BustacaraMedina, Cesar Julio.** Pattern Oriented Software Architecture Pipes and Filters. Bogotá : Pontificia Universidad Javeriana, 2000.

30. **Moquillaza Henríquez,, Santiago Domingo, Vega Huerta, Hugo y Guerra Grados, Luis.** *Programación en N capas.* Lima : Revista de Investigación de Sistemas e Informática, 2010. ISSN 1816-3823.

31. *UML y Patrones, Introducción al análisis y diseño orientado a objetos.* Mexico D.F : PRENTICE HALL, 1999. ISBN 970-1 7-0261-1.

Bibliografía Consultada

Roberto Hernández Sampieri, Carlos Hernández Collado, Pilar Baptista Lucio. *Metodología de la Investigación.* México D.F : McGraw-Hill, 2006. ISBN 970-10-5753-8.

Pressman, Roger S. *Software Engineering.* New York : McGraw-Hill, 2005. ISBN 978-0-07-337 597 -7.

Garlan , David y Shaw, Mary. *Introducción a la Arquitectura de Software.* New Jersey : Publishing Company, New Jersey, 1994. MDA972-92-J-1002.

González, R.C., Wintz, P. *Procesamiento digital de imágenes.* s.l. : Addison-Wesley., 1996.

Rodríguez, Lisandra Michelena. Desarrollo de un video sensor para el conteo de personas. La Habana : Universidad de las Ciencias Informáticas, 2011.

López, Luis Rodríguez. Análisis y evaluación de algoritmos de detección de movimiento y su aplicación a sistemas de seguimiento en video. Madrid : UNIVERSIDAD CARLOS III DE MADRID, ESCUELA POLITÉCNICA SUPERIOR, 2009.

González Benítez, Ubaldo. Detección de personas a partir de visión artificial. Madrid : s.n., 2010. Proyecto de fin de carrera.

Yáñez Garcia, Javier. Tracking de personas a partir de visión artificial. Madrid : s.n., 2010. Proyecto de fin de carrera.

Beck, Kent. *Extreme Programming Explained. Embrace Change.* s.l. : Pearson Education, 1999.

Bradski, Gary y Kaehler, Adrian. *"Learning OpenCV".* s.l. : O'Reilly 1ª edición , 2008.

Lockhart, Thomas. IRIS-Libre. *Guia del Programador de PostgreSQL.* [En línea] [Citado el: 5 de febrero de 2013.] https://forja.rediris.es/docman/view.php/312/461/PostgresProgrammer.pdf.

Experts, jQuery Community. *jQuery Cookbook.* Sebastopol : O'Reilly Media, Inc., 2010. ISBN: 978-0-596-15977-1.

Parihar, Mridula. *ASP.NET Bible.* New York : Hungry Minds Inc., 2002. ISBN: 0764548166.

Avila Mojica, Jamair Antonio y BustacaraMedina, Cesar Julio. Pattern Oriented Software Architecture Pipes and Filters. Bogotá : Pontificia Universidad Javeriana, 2000.

Sinc. *Servicio de información y noticias científicas.* [En línea] Sinc, 2008. [Citado el: 12 de abril de 2013.] http://www.agenciasinc.es/Multimedia/Videos/Analisis-de-las-camaras-de-vigilancia-en-tiempo-real.

OpenCV. [En línea] [Citado el: 19 de febrero de 2013.] http://docs.opencv.org/.

Emgu CV. [En línea] [Citado el: 4 de febrero de 2013.] http://www.emgu.com/wiki/index.php GNU Free License 1.2.

Glosario de Términos

Afluencia: Llegada de personas a un lugar determinado. Acción y efecto de afluir.

Flujo: Nivel de tráfico de personas en sentido bidireccional.

***Framework*:** Es una estructura de soporte definida en la cual un proyecto de *software* puede ser organizado y desarrollado.

Http: Protocolo de Transferencia de Hipertexto. Modo de comunicación para solicitar páginas web.

Multiplataforma: Es un término usado para referirse a los programas, sistemas operativos, lenguajes de programación, u otra clase de *software*, que puedan funcionar en diversas plataformas.

Oclusiones: Por oclusión nos referimos a la situación en la que el objeto móvil desaparece durante un cierto número de fotogramas para, transcurrido este tiempo, reaparecer en la imagen.

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

***Plugin*:** Aplicación que se relaciona con otra para aportarle una funcionalidad nueva y generalmente específica.

Anexo

Anexo 1.

Fases y flujos de trabajo de RUP.

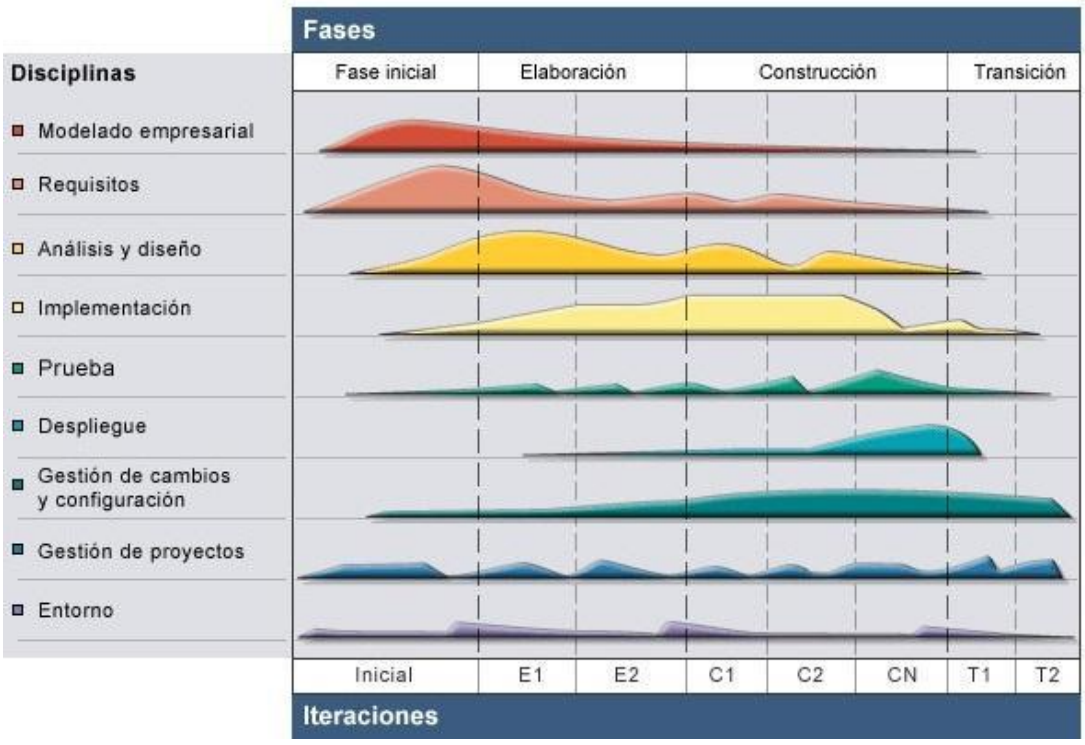

Figura 23. Fases y flujos de trabajo de RUP (14).

Anexo 2.

Combinación de prácticas de XP.

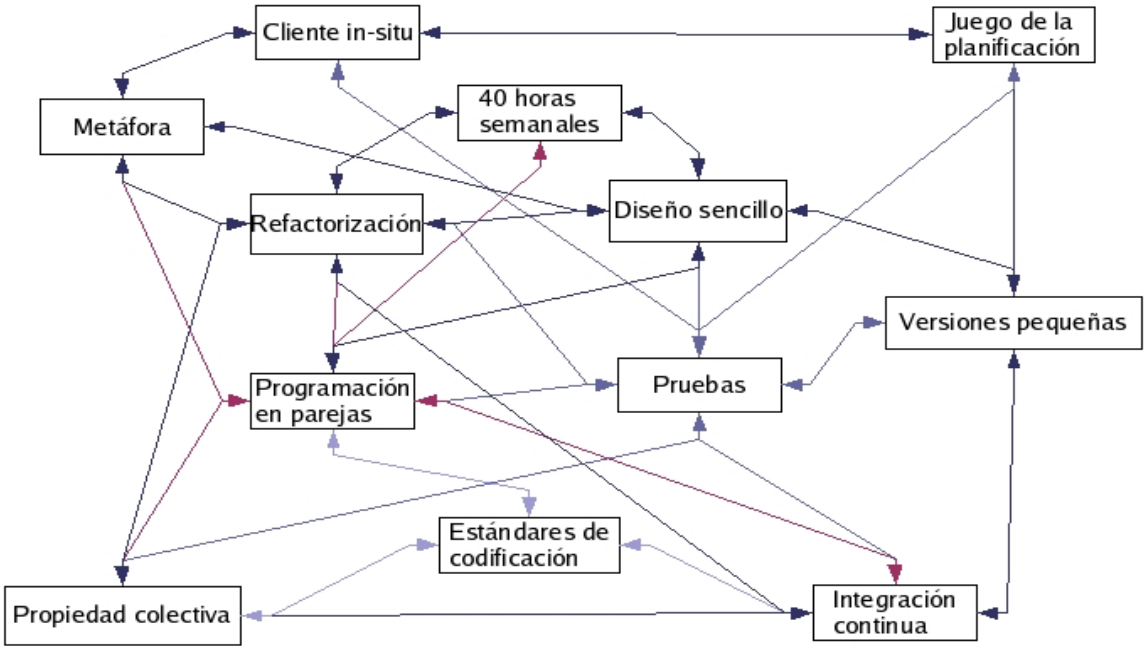

Figura 24. Diagrama que muestra la combinación de todas las prácticas de XP (15).

Anexo 3.

Arquitectura de EmguCV.

La capa de base (*layer* 1) contiene las asignaciones de función, estructura y enumeración que reflejen directamente en OpenCV. La segunda capa (*layer* 2) contiene las clases que combinan las ventajas del entorno .NET.

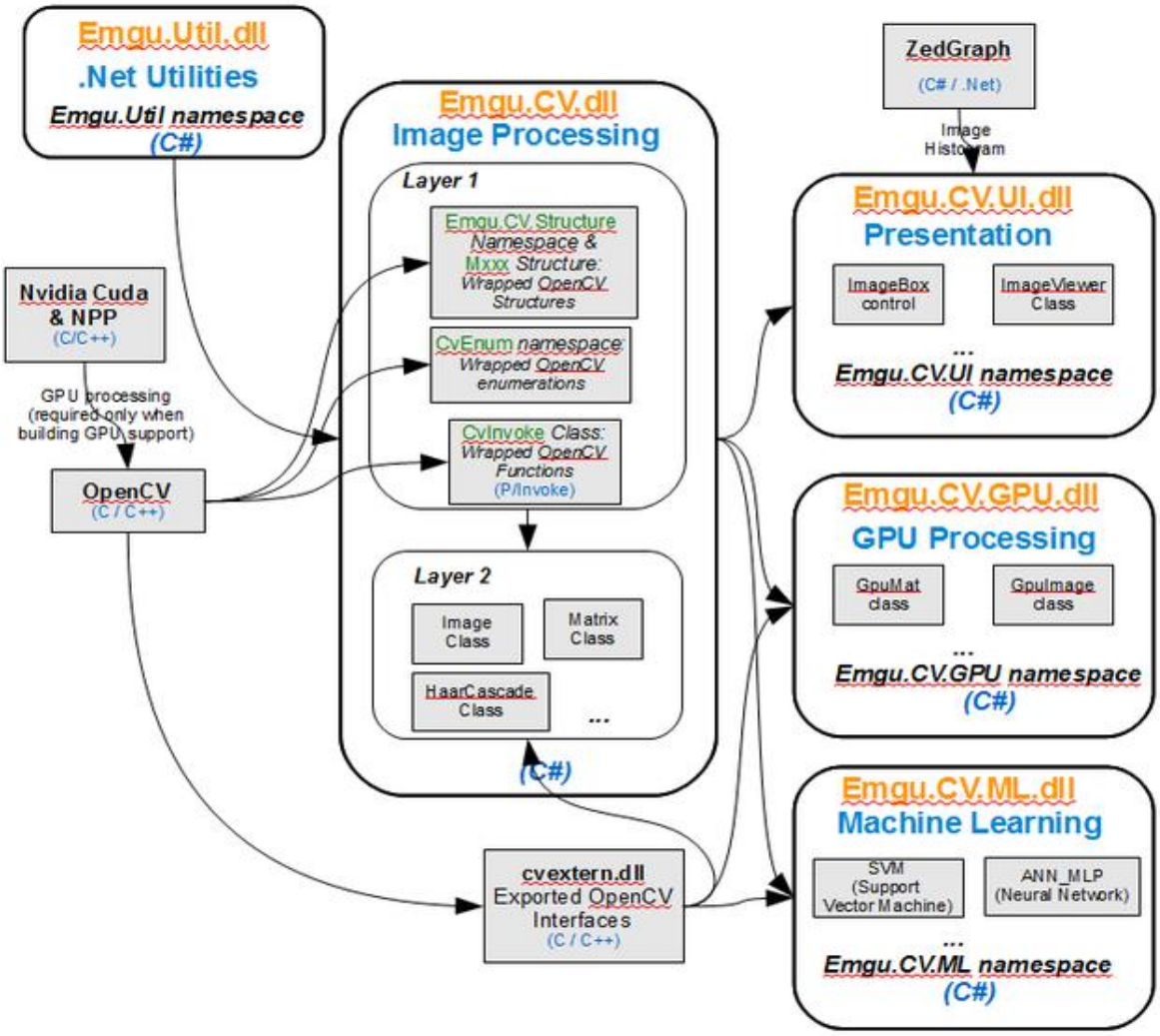

Figura 25. Diagrama que muestra la descripción de la arquitectura de EmguCV (21).

Anexo 4.

Fotograma de ejemplo del proceso de conteo.

Figura 26. Muestra un fotograma donde se aplica el proceso conteo de persona.

Anexo 5.

Historias de usuarios.

Tabla 21. Historia de usuario “Extraer información de la cola del RabbitMQ”.

Historia de Usuario	
Número: HU_5	**Usuario:** Sistema
Nombre historia: Extraer información de la cola del RabbitMQ.	
Prioridad en negocio: Alta	**Riesgo en desarrollo:** Alto
Puntos estimados: 1	**Iteración asignada:** 1
Programador responsable: Daniel Hernández Díaz.	
Descripción: La presente historia de usuario tiene como objetivo primeramente establecer la comunicación con el manejador de cola RabbitMQ. Luego de establecida la conexión, se procederá a la extracción de los datos de la cola.	

Observaciones:

Tabla 22. Historia de usuario “Calcular dirección del movimiento”.

Historia de Usuario	
Número: HU_6	**Usuario:** Sistema
Nombre historia: Calcular dirección del movimiento.	
Prioridad en negocio: Alta	**Riesgo en desarrollo:** Alto
Puntos estimados: 2	**Iteración asignada:** 2
Programador responsable: Daniel Hernández Díaz.	
Descripción: La presente historia de usuario tiene como objetivo calcular la dirección del movimiento de la persona en la escena, para conocer si entra o sale de la zona definida. Para ello se tiene en cuenta la posición de la persona en el fotograma anterior y la posición en el fotograma actual.	
Observaciones:	

Tabla 23. Historia de usuario “Contar personas”.

Historia de Usuario	
Número: HU_7	**Usuario:** Sistema
Nombre historia: Contar personas.	
Prioridad en negocio: Alta	**Riesgo en desarrollo:** Alto
Puntos estimados: 1	**Iteración asignada:** 2
Programador responsable: Daniel Hernández Díaz.	
Descripción: La presente historia de usuario tiene como objetivo contar a las personas que entren o salgan de la zona definida.	
Observaciones: Se debe tener una zona predefinida, a la cual se desee conocer el control de flujo. Se debe conocer de donde viene y hacia dónde va la persona en la escena.	

Tabla 24. Historia de usuario “Almacenar información en la base de datos”.

Historia de Usuario	
Número: HU_8	**Usuario:** Sistema.
Nombre historia: Almacenar información en la base de datos.	
Prioridad en negocio: Media	**Riesgo en desarrollo:** Medio
Puntos estimados: 1	**Iteración asignada:** 3
Programador responsable: Aynel Cruz Barrera.	
Descripción: La presente historia de usuario tiene como objetivo almacenar los datos obtenidos, en los procesos realizados al fotograma.	
Observaciones:	

Tabla 25. Historia de usuario “Extraer información de la base de datos”.

Historia de Usuario	
Número: HU_9	**Usuario:** Sistema.
Nombre historia: Extraer información de la base de datos.	
Prioridad en negocio: Media	**Riesgo en desarrollo:** Medio
Puntos estimados: 1	**Iteración asignada:** 3
Programador responsable: Aynel Cruz Barrera.	
Descripción: La presente historia de usuario tiene como objetivo extraer la información de la base de datos, permitiendo mostrar dicha información en la web.	
Observaciones:	

Tabla 26. Historia de usuario “Consultar datos estadísticos”.

Historia de Usuario	
Número: HU_10	**Usuario:** Administrador.
Nombre historia: Consultar datos estadísticos en tiempo real.	
Prioridad en negocio: Media	**Riesgo en desarrollo:** Medio
Puntos estimados: 1	**Iteración asignada:** 3
Programador responsable: Aynel Cruz Barrera.	

Descripción: La presente historia de usuario tiene como objetivo consultar los datos estadísticos del control de flujo en tiempo real. Consiste en mostrar la cantidad de personas que han entrado y salido hasta el momento de la consulta mediante una gráfica.
Observaciones:

Tabla 27. Historia de usuario "Consultar datos estadísticos de un día determinado".

Historia de Usuario	
Número: HU_11	**Usuario:** Administrador.
Nombre historia: Consultar datos estadísticos de un día determinado.	
Prioridad en negocio: Media	**Riesgo en desarrollo:** Medio
Puntos estimados: 1	**Iteración asignada:** 3
Programador responsable: Aynel Cruz Barrera.	
Descripción: La presente historia de usuario tiene como objetivo consultar los datos estadísticos del control de flujo de un día determinado. Consiste en mostrar mediante una gráfica la cantidad de personas que han entrado y salido en ese día.	
Observaciones:	

Tabla 28. Historia de usuario "Comparar datos estadísticos entre dos días".

Historia de Usuario	
Número: HU_12	**Usuario:** Administrador.
Nombre historia: Comparar datos estadísticos entre dos días.	
Prioridad en negocio: Media	**Riesgo en desarrollo:** Medio
Puntos estimados: 1	**Iteración asignada:** 3
Programador responsable: Aynel Cruz Barrera.	
Descripción: La presente historia de usuario tiene como objetivo comparar los datos estadísticos del control de flujo entre dos días. Consiste en mostrar mediante dos gráficas correspondientes a esos días, la cantidad de personas que han entrado y salido.	
Observaciones:	

Anexo 6.

Tarjetas CRC.

Tabla 29. Tarjeta CRC Comunicador del componente Plugin detector de persona.

Comunicador	
Responsabilidades	**Colaboradores**
Abrir un canal de comunicación.	• RabbitMQ.Client.dll
Cerrar un canal de comunicación.	• RabbitMQ.Client.dll
Enviar un mensaje (debe ser definido según el tipo de comunicador).	• Mensaje
Chequear una cola de mensajes.	• Mensaje
Recibir un mensaje.	• Mensaje
Cerrar una cola de mensajes.	–

Tabla 30. Tarjeta CRC DatoPersona del componente Plugin detector de persona.

DatoPersona	
Responsabilidades	**Colaboradores**
Contener los datos de cada persona detectada en el fotograma.	–

Tabla 31. Tarjeta CRC Serializador del componente Plugin detector de persona.

Serializador	
Responsabilidades	**Colaboradores**
Serializar los datos para enviar a la cola del RabbitMQ.	• DatoPersona

Tabla 32. Tarjeta CRC Historial del componente Aplicación de reportes.

Historial	
Responsabilidades	**Colaboradores**
Graficar datos estadísticos de un día determinado.	• Principal • EntitiesModel.dll
Graficar comparación entre dos días determinados.	• Principal • EntitiesModel.dll

Tabla 33. Tarjeta CRC EntitiesModel del componente Aplicación de reportes.

ENTITIESMODEL	
Responsabilidades	**Colaboradores**
Establecer conexión con la base de datos.	• Telerik.OpenAccess.35.Extensions.dll • Telerik.OpenAccess.dll

Anexo 7.

Diagramas de clase del diseño.

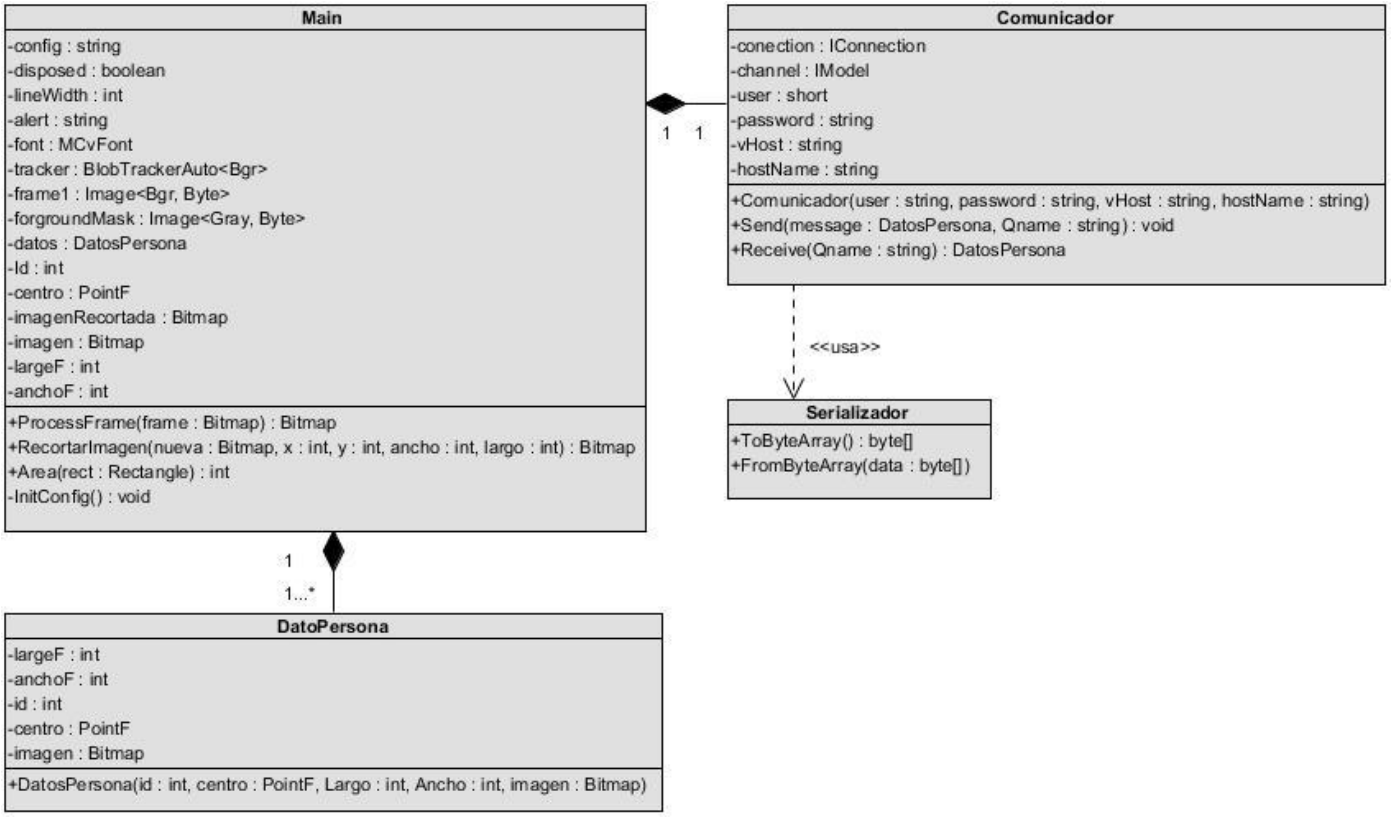

Figura 27. Diagrama de clases del diseño del componente Plugin detector de personas.

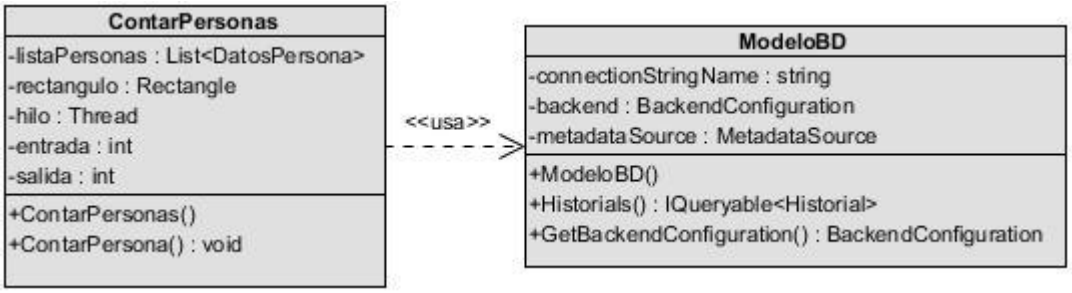

Figura 28. Diagrama de clases del diseño del componente Control de flujo.

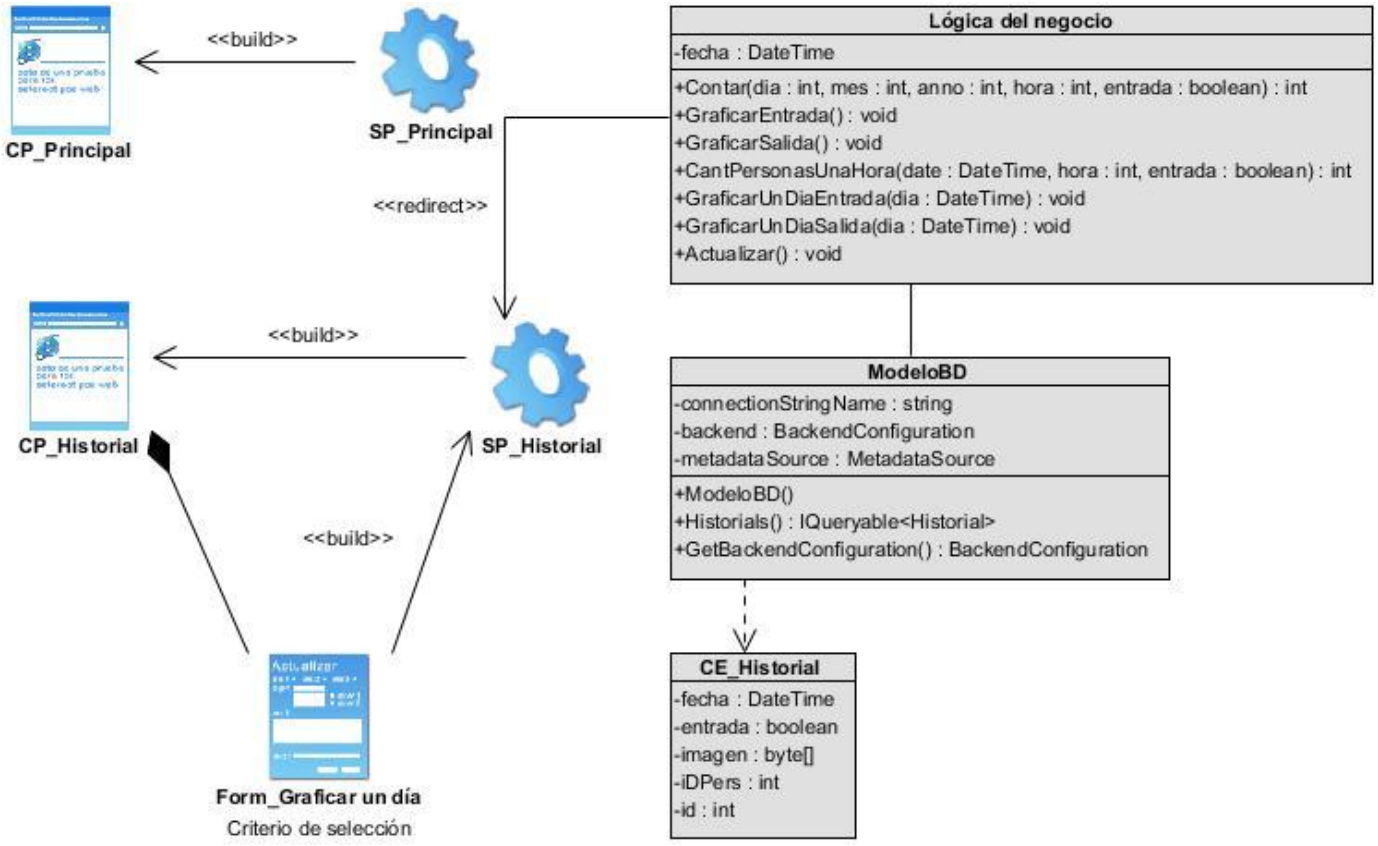

Figura 29. Diagrama de clases del diseño del componente Aplicación de reportes.

Anexo 8.

Estimación de tiempo.

Tabla 34. Historias de usuario con su estimación de tiempo.

Historias de usuario	Estimación (en semana)
Obtener los fotogramas del iSpy.	1
Procesar imagen.	3
Detectar personas.	1
Almacenar datos en la cola del RabbitMQ.	1
Extraer información de la cola del RabbitMQ.	1

Calcular dirección del movimiento	2
Contar personas	1
Almacenar información en la base de datos.	1
Consultar datos estadísticos.	1
Total	12

Anexo 9.

Plan de entrega.

Tabla 35. Muestra la entrega del producto al final de cada iteración.

Entregable	Fin iteración 1 10 de marzo de 2013	Fin iteración 2 10 de abril de 2013	Fin iteración 3 16 de mayo de 2013
Cliente	0.1	0.2	1.0

Anexo 10.

Descripción de la base de datos.

Tabla 36. Muestra la descripción de la base de datos.

NOMBRE	HISTORIAL	
Descripción	Almacena los datos correspondientes a los historiales de las personas.	
Atributo	**Tipo**	**Descripción**
idh	integer	Identificador del historial, único e incremental.
fecha	date	Fecha y hora exacta en que la persona realiza la acción de entrada o salida.
imagen	binary	Imagen recortada del rectángulo englobante de la persona en el instante de la acción realizada.
entrada	bit	Tipo de acción realizada, true en

		caso de entrada, false en caso de salida.
idpersona	integer	Identificador de la persona en la escena, utilizado para realizar su seguimiento.

Anexo 11.

Pruebas unitarias.

En la **Figura 30-31** respectivamente, se muestran como el método *cantPersonaUnDIA()* y *cantPersonasUnaHora()* correspondientes al componente Aplicación de reporte pasaron la prueba satisfactoriamente.

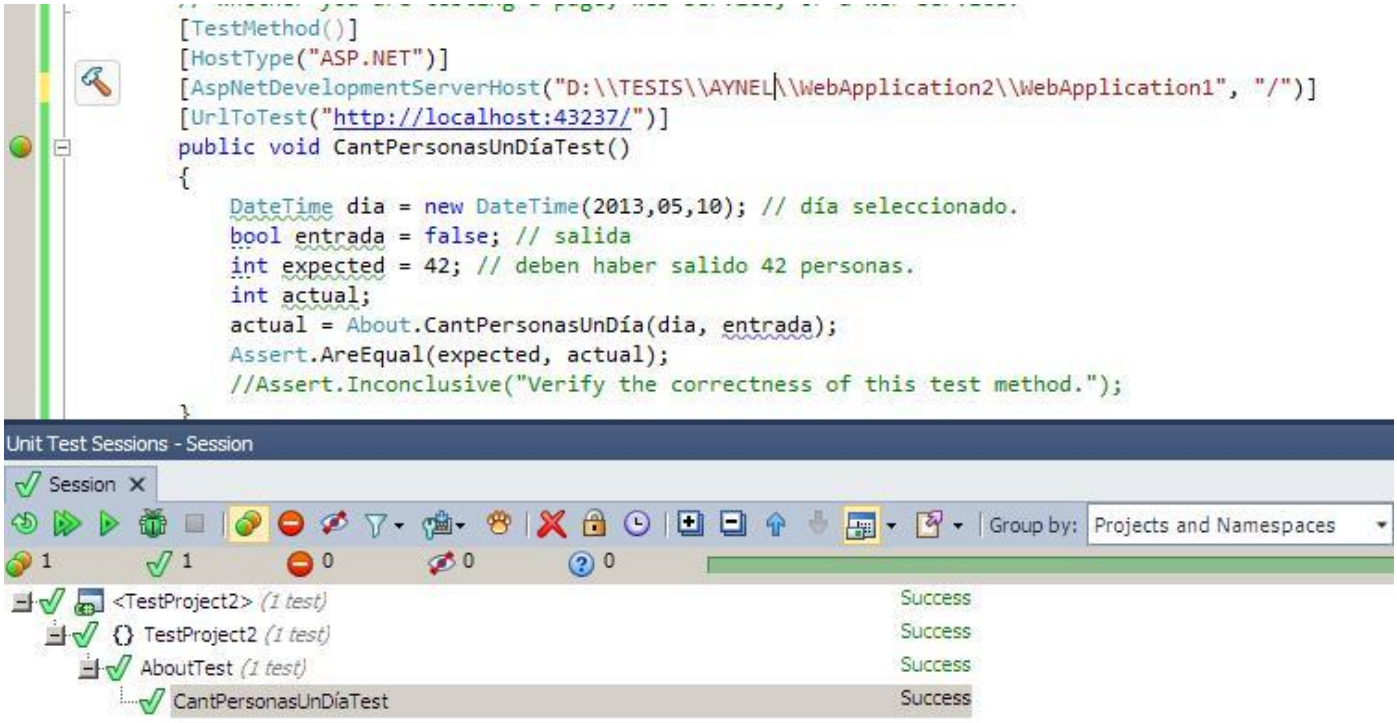

Figura 30. Prueba unitaria al método CantPersonasUnDíadel componente Aplicación de reportes.

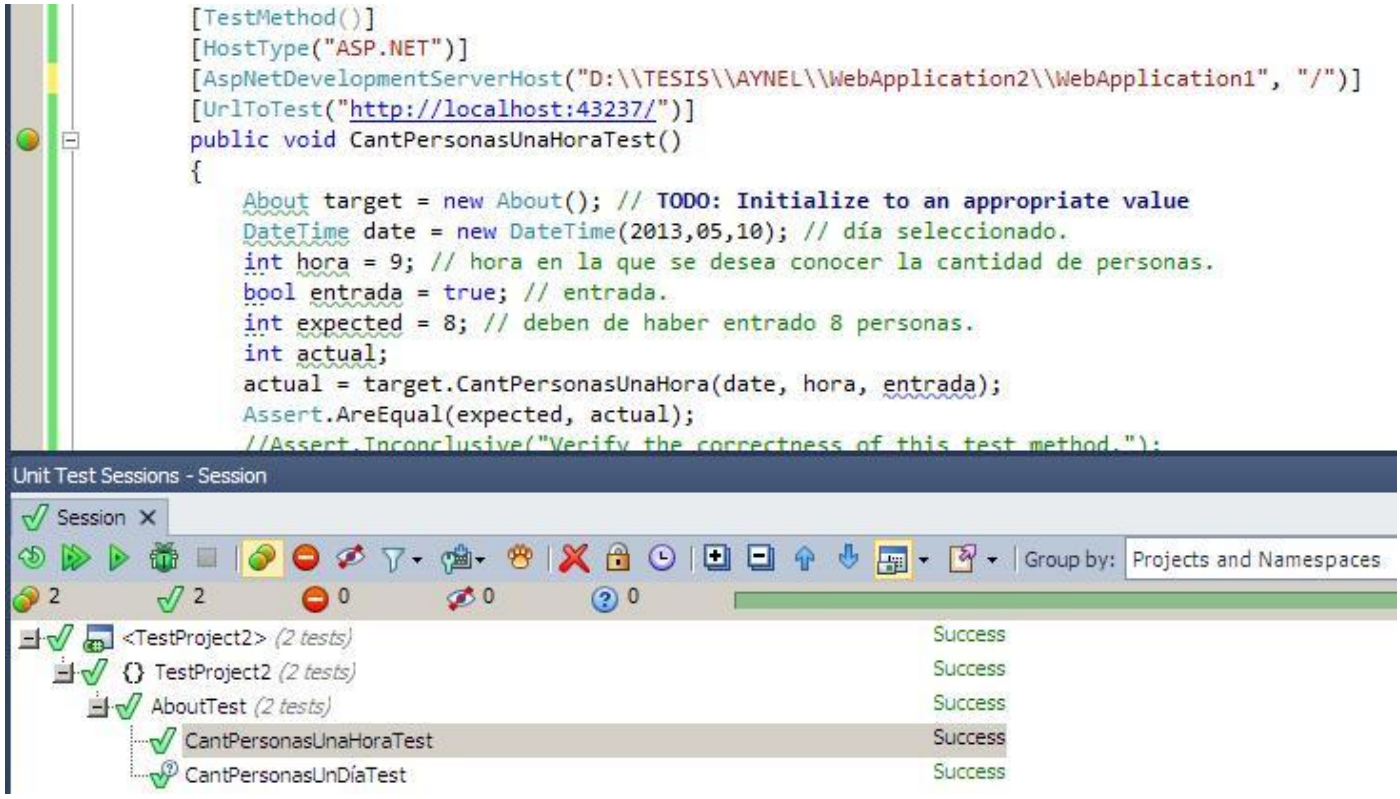

Figura 31. Prueba unitaria al método CantPersonasUnaHora componente Aplicación de reportes.